Neural-Enhanced Live Streaming: Improving Live Video Ingest via Online Learning

Jaehong Kim* Youngmok Jung* Hyunho Yeo Juncheol Ye Dongsu Han
KAIST

ABSTRACT

Live video accounts for a significant volume of today's Internet video. Despite a large number of efforts to enhance user quality of experience (QoE) both at the ingest and distribution side of live video, the fundamental limitations are that streamer's upstream bandwidth and computational capacity limit the quality of experience of thousands of viewers.

To overcome this limitation, we design LiveNAS, a new live video ingest framework that enhances the origin stream's quality by leveraging computation at ingest servers. Our ingest server applies neural super-resolution on the original stream, while imposing minimal overhead on ingest clients. LiveNAS employs online learning to maximize the quality gain and dynamically adjusts the resource use to the real-time quality improvement. LiveNAS delivers high-quality live streams up to 4K resolution, outperforming WebRTC by 1.96 dB on average in Peak-Signal-to-Noise-Ratio on real video streams and network traces, which leads to 12%-69% QoE improvement for live stream viewers.

CCS CONCEPTS

• **Information systems → Multimedia streaming**; • **Networks → Network resources allocation**;

KEYWORDS

video delivery, live streaming, super-resolution, deep neural networks, online learning

ACM Reference Format:
Jaehong Kim, Youngmok Jung, Hyunho Yeo, Juncheol Ye, and Dongsu Han. 2020. Neural-Enhanced Live Streaming: Improving Live Video Ingest via Online Learning. In *Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication (SIGCOMM '20), August 10–14, 2020, Virtual Event, NY, USA.* ACM, New York, NY, USA, 19 pages. https://doi.org/10.1145/3387514.3405856

1 INTRODUCTION

Live video traffic has experienced a rapid growth [10] with the rise of live streaming services, such as YouTube Live [31] and Twitch.tv [17]. Today, live video accounts for a significant volume of Internet video traffic. With its steady growth, market reports predict that live video will take up 17 percent of Internet video traffic by 2022 [5]. With streamers on mobile on the fast rise (more than doubling in 2019 [14]), supporting high-quality (e.g., full HD to 4K) streaming on diverse devices has also become evermore critical.

Live streaming systems consist of two main components. First, the ingest side concerns the delivery of live video from the original streamer to a media server. It uses low-latency streaming protocols [12, 13] to enable timely delivery to dedicated streaming servers [74]. Second, at the distribution side, content distribution servers use adaptive bitrate streaming [2, 6] to optimize the quality experience and scale across thousands to millions of concurrent viewers per live stream [19, 40, 51].

However, the key limitations in an end-to-end live video delivery are that the stream quality is fundamentally constrained by the streamer's uplink bandwidth and its computational capacity. For example, if the network bandwidth between the streamer and the media server is scarce, the ingest stream quality suffers [15, 30, 74], which restricts the quality of the entire delivery downstream. In addition, supporting high-quality, 4K live requires the client to perform real-time 4K encoding, which is infeasible without a high-end CPU [7]/GPU [37] or hardware support and may be too power-hungry for mobile devices [67]. These factors potentially deprive thousands of viewers of their opportunity to enjoy high-quality live stream even if they have ample bandwidth [41, 59].

Recent advances in neural-enhanced video delivery [71, 72] present a unique opportunity to enhance the video quality by applying neural computation on video frames. In particular, NAS [72] integrates neural super-resolution into adaptive streaming using a per-video deep neural network (DNN) and achieves dramatic quality improvement. We expect live streaming can greatly benefit if similar benefits can be translated into its context. However, existing work [39, 72] mainly targets on-demand video delivery and requires pre-training a DNN model for each video. Thus, the approach cannot be easily adopted to live video with stringent delay requirement.

In this paper, we demonstrate neural-enhancement on live video is feasible and effective when combined with online training. To this end, we propose LiveNAS, a new live ingest framework that employs online training and utilizes freshly trained results for super-resolution in the context of live video. The resulting system is one that effectively breaks the strong dependency between the quality of live video and the ingest client's bandwidth, offering high-quality live streams to viewers even when the ingest network is congested. In addition, leveraging the computational power at ingest servers, LiveNAS enables bandwidth- and energy-efficient 4K live ingest for resource-constrained mobile devices.

LiveNAS consists of two ingest components. First, at the media server, online training and inference operate in parallel. It learns

*The first two authors contributed equally to the paper.

SIGCOMM '20, August 10–14, 2020, Virtual Event, NY, USA
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-7955-7/20/08…$15.00
https://doi.org/10.1145/3387514.3405856

new features of a video stream online to update the mapping from low-quality to a high-quality version, while simultaneously enhancing the ingest stream using super-resolution DNNs. Second, the ingest client transmits small patches of high-quality raw frames captured by its camera along with the encoded video. These high-quality patches are used as ground truth labels for training the super-resolution DNN at the media server.

Applying online training and real-time super-resolution to live video ingest, however, involves solving a number of new and non-trivial challenges:

- First, the high-quality patches that a client transmits share the ingest-side bandwidth with live video. Allocating large bandwidth for training patches can improve super-resolution. However, it leaves less bandwidth for live video, which can cause significant quality degradation.
- Second, to benefit a large number of streams, efficient resource use is required. Our observation is that for videos with minimal scene change the online training gain eventually saturates to the point where further training may not provide benefit. However, video segments with frequent scene change benefit from continuous training. Thus, real-time adaptation to content can improve resource-efficiency.
- Finally, minimizing the degradation in quality of experience (QoE) due to the latency introduced by super-resolution is important. In addition, in contrast to offline training where maximizing the final accuracy is more important than the training time, we want the training gain to appear as fast as possible.

LiveNAS addresses the challenges by introducing new components at the ingest client and the server. First, our ingest client incorporates a quality-optimizing scheduler that effectively balances the allocation of uplink bandwidth between training patches and real-time video. Second, at the ingest server, our content-adaptive trainer adapts the amount of training to the scene changes in live video to maximize the resource efficiency. Finally, to benefit from the training as fast as possible, LiveNAS client selects patches that will lead to higher quality gains from the most recent video frames. The ingest server, then, utilizes GPUs in a way that produces training gain as fast as possible, reduces inference latency, and enables real-time super-resolution up to 4K. This work does not raise any ethical issues.

We evaluate LiveNAS using a full system implementation on top of a commercial state-of-the-art live ingest system, WebRTC [26]. We use nine categories of recorded live streams from Twitch and YouTube and measure user QoE against hours of real-network traces. Our evaluation, consisting over 300+ hours of streaming, shows that LiveNAS improves video quality by 1.96 dB over WebRTC and by 1.19 dB over generic super-resolution in Peak-Signal-to-Noise-Ratio (PSNR). This improves the QoE of end viewers watching live streams via HTTP adaptive streaming by 12% to 69%.

In summary, we make four key contributions:

- **Video quality improvement for end-users:** By improving ingest quality, LiveNAS brings significant QoE improvements to end viewers of live streams.

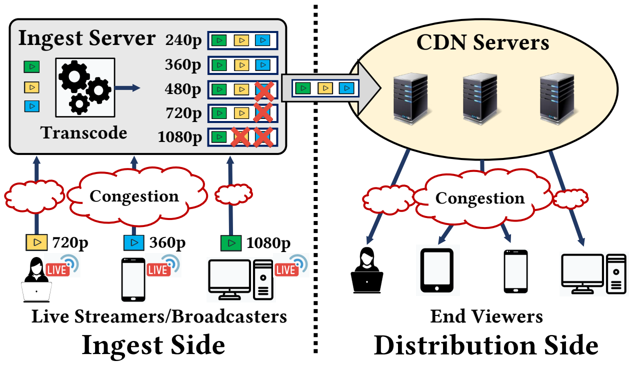


Figure 1: Live Streaming End-to-End Workflow

- **4K live in constrained environments:** LiveNAS enables 4K streaming without requiring real-time 4K encoding or imposing a high-bandwidth requirement on ingest clients.
- **Quality optimization with online training:** LiveNAS clients maximize the net video quality by balancing the bandwidth allocation between high-quality training labels and real-time video in an online learning context.
- **Resource optimization for online training:** We propose a new online training method that adapts to the content of live video by detecting training gain saturation and scene transitions.

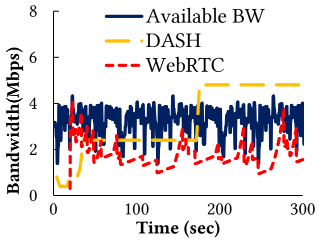
2 BACKGROUND

Live video delivery. Figure 1 shows the end-to-end workflow of live streaming divided into ingest and distribution. The *ingest* side is responsible for sourcing the origin stream from ingest clients. A live video captured at the ingest client is streamed to a media server via streaming protocol, such as RTSP/RTMP [11, 12]. The media server, often located in a relatively well-provisioned facility, such as the cloud, transcodes the stream into 2-10 second chunks with multiple quality levels for HTTP adaptive streaming [66] and publishes the new chunk to content delivery networks (CDNs). This paper focuses on the first hop delivery between origin streamer and media server, namely the ingest.

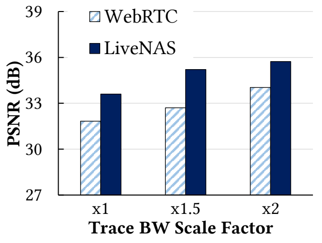
Super-resolution up-samples low resolution image/video to produce higher resolutions. It is often used in contexts where a high-quality version is not available. Recent studies find deep neural networks [42, 52, 55, 64, 75] are effective in learning the mapping from a low resolution to higher resolution. Many algorithmic advances [54, 65, 76] have been made to make inference and training more effective.

Content-aware neural adaptive streaming. NAS [72] applies super-resolution on top of adaptive streaming. It generates a super-resolution DNN for each video offline. When a client requests a video from CDN server, the server provides a deep neural network (DNN) corresponding to the video. The client then applies the DNN to the received low-quality video chunks by utilizing its own computing power. NAS significantly improves user QoE by utilizing super-resolution and client computation. However, live video cannot benefit from NAS because it requires a pre-trained DNN model for each video and ten minutes of DNN training is required on GPU per minute of video [72].

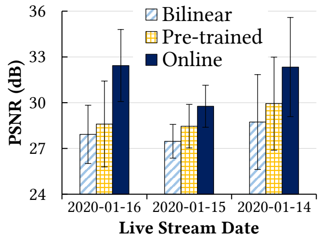
WebRTC [38] ingest framework. WebRTC is a de facto standard for live video ingest designed to deliver live video with highest



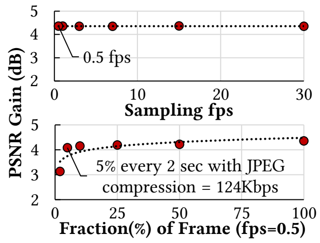
(a) Live streaming is sensitive to bandwidth variation.



(b) Super-resolution provides significant gains.



(c) Online learning with fresh data has clear advantage.



(d) Training with fractional high-quality labels.

Figure 2: Motivation of LiveNAS

quality and minimal latency. Its client (streamer-side) consists of a transport layer that uses google congestion controller (GCC) on top of Real-Time Protocol (RTP) and a video engine that uses the VP8/VP9 video codec. The RTP implementations are built on both TCP and UDP, where UDP is preferred over TCP due to latencies inflated by TCP packet retransmission. At runtime, GCC estimates the available network throughput, similar to adaptive streaming, and outputs the target bitrate. Then the video engine encodes video with the given target bitrate to match the available network bandwidth.

3 MOTIVATION AND APPROACH

Case for super-resolution at ingest. The quality of a live video stream is sensitive to the variation in available network bandwidth. A live ingest client cannot use bandwidth aggressively because 1) it must avoid packet loss to minimize latency due to retransmission; and 2) the ingest server cannot use much buffer to absorb the bandwidth variations. Figure 2a shows the available bandwidth and video bitrate for live video using WebRTC compared with on-demand video using adaptive streaming [57] given a FCC broadband network trace [8]. The result is obtained by using the Mahimahi [58] network emulation tool. WebRTC uses bandwidth much more conservatively than adaptive streaming [47, 57] whose large buffer effectively absorbs bandwidth variations. As a result, WebRTC requires much larger bandwidth than adaptive streaming clients for the same quality.

In contrast, super-resolution can significantly improve video quality without requiring more bandwidth. NAS [72] shows one can improve video quality by 1-5 dB, giving benefit comparable to using 17% more bandwidth. The conservative bandwidth use of live streaming makes super-resolution even more attractive in its context. To exemplify this, Figure 2b compares the video quality with and without super-resolution in average Peak-Signal-to-Noise-Ratio (PSNR) while scaling the bandwidth of the earlier trace by a factor between 1 and 2 using a video [35] from YouTube. We observe that super-resolution can provide benefit comparable to having 1.5x to 2x the bandwidth! Further analyzing the cause, we find that WebRTC on average uses only 55-64% bandwidth for live video out of what the network actually allows due to its conservative behavior.

Ingest-side super-resolution looks even more attractive considering its deployment model. Super-resolution can be applied at the media ingest server, which is typically in a cloud environment where computation can be more easily provisioned. In addition, ingest servers are less resource-constrained than client devices which can be mobile. This enables LiveNAS to support energy-efficient 4K live streaming, even with a client that cannot support real-time 4K encoding, as we demonstrate in §8.

Case for online learning. In DNN-based super-resolution, it is known that using a neural network trained on the same content, namely a content-aware DNN, provides greater benefit [71, 72]. However, the pre-trained approach does not suit live streaming. Streamers provide a variety of content even within a single live streaming session [40]. Thus, training super-resolution DNNs with videos from previous streaming sessions is less effective and shows significant variations in quality, making the approach unreliable.

Online learning provides two key benefits. First, it provides larger and reliable quality gains than pre-training. To demonstrate the effectiveness of online learning with fresh live data, we emulate pre-trained and online learning approaches using popular "Just Chatting" videos from Twitch. For pre-training, we use recorded live streams from the same steamer [22] between January 13 to 16, 2020. For each live stream, we use its previous stream as training data. For the online approach, we use the same stream for online training and testing. We keep the GPU training time identical for both. Figure 2c shows the resulting video quality in PSNR. The result suggests that online learning with fresh data shows higher quality gain over using a pre-trained model from history. The quality gain of the pre-trained model is only 0.7 dB higher than the naive bilinear up-sampling that does not use DNN, whereas an online learning model delivers at least 2.3 dB gain in PSNR.

Second, online learning allows us to adapt the amount of training to the real-time quality gain that super-resolution delivers, which depends on the actual content of the video. For example, if the super-resolution gain has decreased, it might mean that online training is not keeping up with recent scene changes, in which case one can leverage more computation for online training by using multiple GPUs. On the other hand, the quality gain may saturate over time and further training may not pay off in quality, in which case temporarily suspending online training makes sense. This presents a unique opportunity to online learning in delivering reliable quality with efficient resource use.

LiveNAS challenges. Based on the idea that online learning can effectively enhance the quality of live streams at ingest, this paper explores the design of maximizing its benefit while being cost-effective. However, despite the promises, applying online learning for super-resolution of live video involves solving a series of non-trivial challenges.

First, online learning requires both high-resolution videos (used as ground truth labels for super-resolution training) and powerful

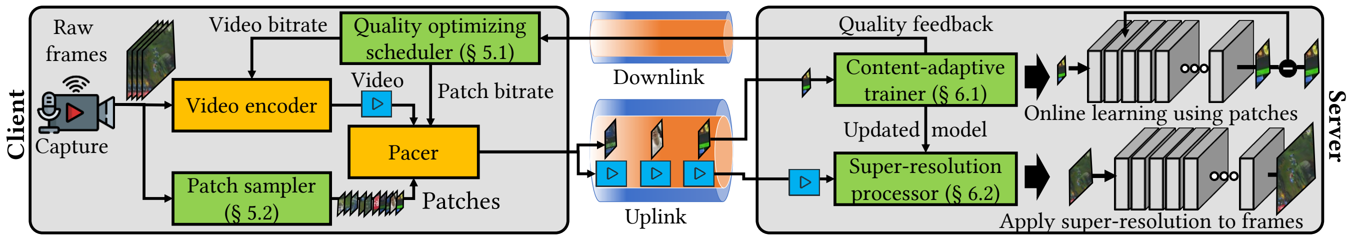

Client
Raw frames
Capture
Video bitrate
Quality optimizing scheduler (§ 5.1)
Video encoder
Video
Patch bitrate
Pacer
Patch sampler (§ 5.2)
Patches
Downlink
Uplink
Quality feedback
Content-adaptive trainer (§ 6.1)
Online learning using patches
Updated model
Super-resolution processor (§ 6.2)
Apply super-resolution to frames
Server

Figure 3: LiveNAS System Overview

computing devices such as server-class GPUs and NPUs [46]. The original streamer can capture raw uncompressed video at high-resolution (e.g., 1080p or 4K), but relying on that one has enough computing power for online training falls short of a practical design. Thus, LiveNAS chooses to provision for and utilize computation at media servers.

Second, utilizing server computation results in a challenge that media servers do not have high-resolution labels required for online training but transmitting high-resolution labels from the original streamer requires too much bandwidth and often infeasible. To address the challenge, we rely on our key observation that even a fraction of ground truth labels can provide substantial training gains, which is demonstrated in Figure 2d. For example, when we sample 1.7% frames (0.5 fps) within a 1080p video and crop 5% of them, the size of resulting frames, referred to as patches, is only 124 kbps. Even training a DNN with these (small) patches, it delivers significant quality improvement (+4 dB PSNR), which is comparable to that of using all frames (-0.27 dB PSNR). This is because video contains a large amount of redundancy that can be eliminated for training a DNN.

Leveraging this, the paper establishes the feasibility of online learning for super-resolution using a fraction of live video frames (patches) as ground-truth labels.

Finally, transmitting the ground-truth labels consumes available bandwidth and leaves less bandwidth for the live video itself. Thus, we must strike a balance between the two to maximize the net video quality. Next, we describe how LiveNAS solves the challenges and address the relevant issues in making LiveNAS more effective.

4 SYSTEM OVERVIEW

The goal of LiveNAS is to enhance the video quality of the origin stream at the ingest server using deep neural network-based super-resolution, while not affecting or minimizing the impact on other QoE metrics for live streaming, such as delay and frame loss [43, 44, 61]. It must be agnostic to video codec, transport and rate control algorithm used for ingest. Finally, we assume the ingest media server is provisioned with GPUs that can be used for online training and inference.

System overview. Figure 3 shows an overview of LiveNAS that consists of an ingest client and an ingest server. A notable feature of LiveNAS ingest client is that it sends patches of high-quality frames for online training along with video. Two design components work coherently at the client to maximize the video quality: the *quality-optimizing scheduler* that allocates bandwidth between the training patches and live video stream (§5.1); and the *patch sampler* (§5.2) that selects high-quality labels to be transmitted.

LiveNAS media server performs online learning and transforms the original video steam into a higher quality using super-resolution DNNs. Two novel components operate here: the *content-adaptive online learning* engine that dynamically adapts the GPU usage for training for resource efficiency (§6.1); and the *super-resolution processor* (§6.2) that enables 4K super-resolution with multiple GPUs.

5 LIVENAS INGEST CLIENT DESIGN

We first detail the two new components at our ingest client.

5.1 Quality-Optimizing Scheduler

Problem and goal. LiveNAS uses an estimate of available bandwidth from the underlying transport layer. Given the available bandwidth C_t at time t, the LiveNAS scheduler splits its use between the high-quality labels, p_t, and the live video, v_t. The goal of this module is to find the bandwidth allocation between high-quality labels and live video, that maximizes the overall quality. This involves balancing the trade-off between the video quality and future expected quality gain from online learning. If we use more bandwidth to encode the video, less bandwidth is available for transmitting high-quality labels that drive online learning. Thus, we express our optimization goal as the sum of video quality and the future discounted expected quality gain due to online training:

$$\max_{v_t,p_t} Q_{video}(v_t) + \gamma \cdot Q_{DNN}(\sum_{k=0}^{t} p_k) \qquad s.t, \forall t, v_t + p_t \le C_t \tag{1}$$

where $\gamma \le 1$ is a discount factor. The first term, $Q_{video}(v_t)$, in our optimization objective is the video stream quality given the encoding rate, v_t, and the second term, $Q_{DNN}(\sum_{k=0}^{t} p_k)$, reflects the future gain due to online training which is a function of the total training patches given so far, $\sum_{k=0}^{t} p_k$.

Approach. We observe that the optimization objective is a concave function. First, Q_{video} is concave as video quality with regard to the bitrate is concave [56]. Second, Q_{DNN} is also concave because it monotonically increases with respect to increase in the training set, p_t, yet the marginal improvement diminishes [70]. By the definition of concavity, any linear combination of concave functions is concave.

The concavity of our objective function allows us to apply gradient ascent to find the global optimum [63], where the update rule is represented as:

$$p_{t+1} = \alpha \cdot \{\gamma \cdot \frac{dQ_{DNN}}{dp_t} + \frac{dQ_{video}}{dp_t}\} + p_t \tag{2}$$

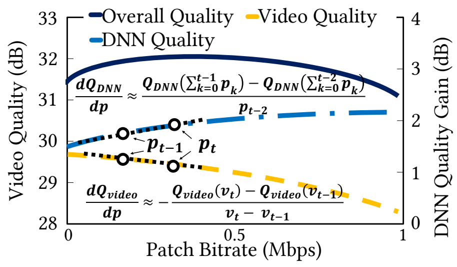


Figure 4: Computing Gradient

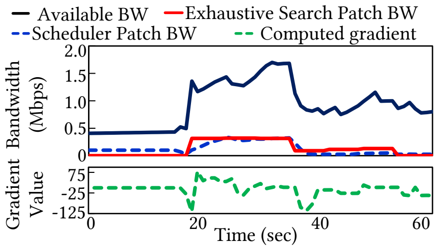


Figure 5: Case study for quality optimization

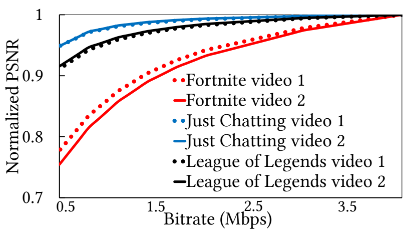


Figure 6: Normalized bitrate-to-quality curve.

where α is the step size. At the end of each time step, LiveNAS client takes the new patch bandwidth allocation, p_{t+1}, and uses the bandwidth to transmit high-quality training patches. The rest, $C_{t+1} - p_{t+1}$, is used for streaming live video, where C_{t+1} is given by the underlying transport layer. As it is difficult to derive an analytical closed-form equation of the quality and its derivative, we use a numerical approach to compute the gradient using values from live measurements. Figure 4 illustrates the process of computing the two gradients. We explain the details below.

Estimating gradient for DNN-enhancement. To obtain the derivative of $Q_{DNN}(\sum_{k=0}^{t} p_k)$, we use two recent DNN quality improvement, $Q_{DNN}(\sum_{k=0}^{t-2} p_k)$ and $Q_{DNN}(\sum_{k=0}^{t-1} p_k)$ and compute the slope between the two. Unfortunately, the ingest client cannot directly compute the DNN quality improvement, but the information is available at the media server. Our media server provides this information through periodic feedback (§6.1). It computes the quality improvement due to super-resolution at time $t-2$ and $t-1$ using the most recent ground-truth label. For this, the media server keeps the two most recent DNNs. We set the time interval between the two to 5 seconds by making each training epoch to be 5-second long. We explore how LiveNAS is sensitive to the length of training epoch (i.e., training window) in our evaluation (§8.4). Any video quality metric can be used, but our implementation uses PSNR because it is less expensive to compute than structural similarity index.

Estimating gradient for video quality. To obtain the derivative term of Q_{video}, we need two video quality points, $Q_{video}(v_{t-1})$ and $Q_{video}(v_t)$ to calculate the linear slope between the two. However, video quality exhibit non-trivial variations across video frames (Figure 2c). Thus, relying on taking quality measurements for a few frames can be misleading. To obtain robust measurements, one needs to take the average quality across multiple group of pictures (GoP).

However, this places a significant burden on clients because it requires them to perform encoding at two different bitrates and video encoding is often expensive and power-hungry especially on a mobile device [37, 45]. To estimate the quality difference, we use two strategies. First, we gather measurements of average PSNR values for different encoded bitrates at the client. However, this can be subject to large variation when we do not have enough measurements. Thus, as a second strategy, we rely on the observation that the normalized bitrate-to-quality curve is very similar across similar types of content. Figure 6 shows the video quality normalized to the highest PSNR while varying the encoding rate by video categories of Twitch. It shows a striking similarity in the normalized bitrate-quality curves for different video streams of the same category. Our media server provides this information as a function that given a stream class, it outputs the normalized quality value, $NQ_{type}(v_t)$ for bitrate v_t. LiveNAS client initially estimates $Q_{video}(v_t)$ by scaling the relative difference in the normalized quality at v_{t-1} and v_t to the observed video quality of the previous epoch $Q_{video}(v_{t-1})$. Over time, the client adjusts it to the current video using exponentially weighted averaging. This provides a robust estimate with reasonable accuracy while avoiding the cost of extra encoding.

Update frequency and step size. The patch bitrate frequency must be long enough to observe the effect of online training. But short enough to respond to changes in network conditions and scene changes that affect video quality. Thus, we set the update frequency on the same order as our online training epoch. We use a 5-second training epoch and 1-second update frequency, which are empirically determined.

Finally, we set the step size as α =100 kbps in Equation 2, and the initial patch rate as 100 kbps. This reflects the minimal amount of training data required to show online training gain for the first few epochs. If the available bandwidth, C_t, falls below the minimum encoding bitrate enforced by WebRTC (which is 200 kbps), we do not transmit any training patches and the system falls back to vanilla WebRTC.

Validation case study. Using a real network trace from a 3G network [62] and a video stream from YouTube, we perform a case study to demonstrate how the algorithm works in practice. Figure 5 (top) shows the available bandwidth by WebRTC and LiveNAS' allocation between the patch and live video bandwidth. The patch bandwidth is updated every second from the gradient, shown in Figure 5 (bottom). On average, LiveNAS allocated 8.9% of available bandwidth for patch bitrate. For comparison, we also show an offline optimal patch bandwidth allocation that we obtained by doing an exhaustive search over a time window of 5 seconds in 25 Kbps increments. We find that LiveNAS closely approximates the offline optimal policy that maximizes the overall video quality.

5.2 Patch Selection

Problem and goal. LiveNAS client sends training patches of size 120x120 pixels to match the dimension of our patch-based super-resolution DNN. Each patch is a small fraction of an entire frame. For example, the area of a 1080p frame is 144 times that of a patch. However, the bandwidth constraint from the quality-optimizing scheduler typically allows only a few training patches to be transmitted per training epoch. But the patch selection actually affects the training gain because some patches provide higher training gains than others. Thus, the goal of patch selection is to select

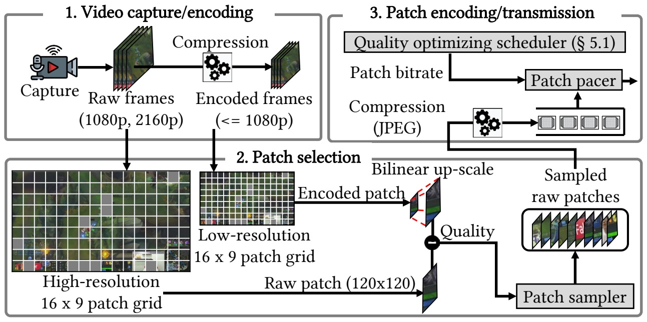

1. Video capture/encoding
Capture
Raw frames (1080p, 2160p)
Compression
Encoded frames (<= 1080p)
3. Patch encoding/transmission
Quality optimizing scheduler (§ 5.1)
Patch bitrate
Patch pacer
Compression (JPEG)
2. Patch selection
Encoded patch
Bilinear up-scale
Sampled raw patches
Low-resolution 16 x 9 patch grid
Quality
High-resolution 16 x 9 patch grid
Raw patch (120x120)
Patch sampler

Figure 7: Patch Selection Overview

training patches to make the online training gain appear as large and quickly as possible given the patch bandwidth computed in §5.1. Figure 7 illustrates the process of patch selection, which we explain in more detail.

Patch selection. To achieve our goal, we establish two criteria for patch selection. First, we prioritize the ones that give the most benefit. The training gain a patch may provide varies as the scene complexity varies over patches. We would like to send patches that provide larger gains without introducing a large bias. Second, in contrast to local training, we avoid sampling redundant data. In general, super-resolution trainer samples a fixed-size patch from a random location given a frame [42, 52, 55, 64, 75]. However, using the same method will result in selecting and transmitting overlapping regions, which we want to avoid.

LiveNAS uses a light-weight sampling method that satisfies the criteria as illustrated in Figure 7. First, we obtain the most recent raw frame and its decoded version to calculate the frame's encoded quality (e.g., in PSNR). Next, our client randomly samples a patch in the frame from a grid of non-overlapping patches; e.g., a 1080p frame is divided into 16x9 grid, where each cell is a 120x120 patch. Finally, it includes the patch for transmission only if the patch's encoded quality is lower than the entire frame's. Otherwise, it discards the patch. The rationale is that including patches that are more difficult to encode as a training set will provide greater benefit. The process iterates until we select a small number of patches (around 10) for transmission. Our sampling method improves the average PSNR by 0.1 to 0.3 dB over random sampling within five minutes of online training.

Patch encoding and transmission. The sampled patch has to be sent at high resolution, but the raw RGB patch is very large (43 KB). Thus, we need to balance the quality and the size using compression. Lossless PNG compression reduces the size by 50% on average. LiveNAS makes a better trade-off by using lossy compression. Our implementation uses JPEG with a quality level of 95, where 100 provides the highest quality. This reduces a patch size to 1/10 on average without causing significant degradation in training quality (i.e., less than 0.1 dB in PSNR).

When transmitting a patch, we include its timestamp and its location within the corresponding frame. This enables the media server to assign a larger weight to the latest patch for online training and find the low resolution counterpart from the encoded video stream, as we will describe in §6.2. Finally, LiveNAS transmits the content of the patch transmission buffer according to the allocated patch bandwidth. When the patch transmission buffer is empty, we invoke the patch sampler which refills the buffer.

6 LIVENAS MEDIA SERVER DESIGN

This section describes the design of our ingest server that performs online learning and inference for super-resolution.

6.1 Content-Adaptive Online Learning

Problem and goal. Generally speaking, the online learning approach exhibits a trade-off between GPU training cost and video quality gain. But, the actual tradeoff depends on the content of live video. Specifically, for a video with infrequent scene changes, due to the large amount of redundancy across frames, the online training gain shows diminishing return over time and eventually saturates. In contrast, video with frequent scene changes consistently benefit from online learning because the content changes over time. The online learning approach allows us to take advantage of this and increase resource efficiency (or reduce the GPU usage) by dynamically adapting to the real-time super-resolution gain on recent frames. Note, this is a unique benefit of online learning because a pre-trained model cannot adapt the GPU usage in training to future streams.

To enhance resource efficiency, we carefully monitor the real-time super-resolution (SR) gain and detect gain saturation and scene changes to adapt the GPU usage for training. When we observe the saturation of SR gain, we suspend online training to save the resource. Online training resumes, when the current model no longer delivers significant quality improvement for recent frames, signaling the need for training. Algorithm 1 in Appendix A summarizes our design that results in 65% savings in the GPU used for training.

Measuring quality. Measuring the quality of a video requires a reference. The reference is readily available at the ingest client, but at the media server, we do not have the full reference. Instead, we use the high-quality training patches as a reference at the media server. This allows us to compute the quality of a patch from a video stream that corresponds to the high-quality training patch. Next, we show how video quality measurements at the media server are used to detect training gain saturation and scene changes.

Detecting gain saturation. At the end of a training epoch, the trainer estimates the quality gain due to online training by computing the quality difference after applying the most two recent DNNs, DNN_{t-1} and DNN_t (Algorithm 1, line 6) using a recent high-quality patch. If the quality difference is smaller than $thresh_{sat}$, the trainer increments the $patience$ value (line 8). If the $patience$ value exceeds a threshold, $count_{sat}$, LiveNAS deems saturation has occurred, resets the $patience$ value, and suspends the training. It then signals the ingest client which in turn sets the patch bitrate to a minimum value, p_{min}. In our evaluation, we use p_{min} = 25 kbps, sending a patch approximately every two seconds.

Adapting to scene changes. LiveNAS performs quality validation of super-resolution when a recent high-quality patch arrives during the period of suspended training. The media server computes the quality difference between the latest online-trained DNN, DNN_t, and the initial DNN, $DNN_{t=0}$ (line 14), for which we use a DNN trained using a standard benchmark dataset, such as NTIRE 2017 dataset [36]. $DNN_{t=0}$ reflects the state before any online training. If the difference between the two is smaller than $thresh_{online}$, it means the current model no longer reflects the recent content. The trainer then increments the $patience$ value (line 17). If the

Component	Lines of Code (LoC)	Changed
LiveNAS Client	1489 lines of Python	- (1489)
LiveNAS Server	1633 lines of Python	- (1633)
WebRTC library	524k lines of C++	0.21% (1102)

Table 1: LiveNAS implementation (Lines of Code)

patience value exceeds a threshold, $count_{online}$, LiveNAS deems scene changes has occurred, notifies the ingest client, and resumes online training. When the ingest client is notified, it sets the patch bitrate to initial value (§5.1), and the online training resumes with recent patches. This re-bootstraps the feedback process between online training and quality-optimizing scheduler as explained in §5.1.

Persistent online learning. So far, we assumed online training starts from a generic super-resolution DNN trained using a standard benchmark dataset. However, operators can choose to keep and reuse the result of online learning for future streams for popular streamers. This reuses learned results from previous sessions, similar to pre-training, but delivers further quality gain while keeping the two main benefits of online learning: 1) it still benefits from resource-efficient content-adaptive training; and 2) guarantees significant quality gains even when dramatic scene changes occur. We explore this option in our evaluation (§8).

6.2 Super-resolution Processor

Low latency and 4K support. The super-resolution DNN [72] we use delivers real-time super-resolution up to 1080p resolution and is done frame-by-frame in a serial manner. LiveNAS further enables real-time 4K super-resolution using multiple GPUs. When multi-GPU inference is enabled, LiveNAS splits up a frame into multiple pieces in equal size and performs super-resolution for each piece on different GPUs, enabling intra-frame parallelism. After super-resolution, the CPU gathers results from each GPU to put together the pieces. This reduces latency and enables 4K super-resolution.

For example, we use three GPUs for super-resolution from 720p (1080p) to 4K to enable real-time processing in our evaluation. Each frame is split up into three pieces of size 1280x240 (1920x360 for 1080p), which are then individually super-resoluted by a factor of three (two for 1080p) and stitched back to produce 3840x2160, which is 4K. This takes 23 msec for 720p and 29 msec for 1080p for each frame, which translates to 43 and 34 frames per second respectively.

Online learning with live data. In contrast to the offline training, our training data get constantly updated. This leads to bias in training; the earlier the data is added to the dataset, the more it gets exposed to training. This is unfavorable, as the latest received patches often better reflect the current status of the live video. To mitigate this unbalance, our online trainer gives a larger weight to recent K patches when composing a mini-batch for training. For our evaluation, we use $K = 150$ and give four times the weight to them than remaining older patches. This results in a fairly modest improvement of 0.07-0.28 dB in PSNR.

Support for multi-GPU training. LiveNAS supports multi-GPU training to speed up online learning and thus further improves the resulting video quality. For multi-GPU training, we partition the data across multiple GPUs such that patches are grouped by their arrival sequence. Each GPU then computes the gradient using the same loss function. However, when aggregating multiple gradients to synchronize the models, we give a larger weight to the gradient computed with more recent patches. Our multi-GPU learning gives an additional improvement of 0.77 dB to 1.1 dB in PSNR when using three GPUs on top of the improvement from a single GPU (see §8).

7 IMPLEMENTATION

LiveNAS is implemented on top of the state-of-the-art open-sourced ingest framework, WebRTC [26]. LiveNAS consists of ~4K lines of new or modified code. Table 1 shows the lines of code (LoC) for each component.

WebRTC integration. To integrate LiveNAS with WebRTC, we add new APIs to libWebRTC [9]. We implement LiveNAS server and client in python to utilize Pytorch framework and image processing modules. Since libWebRTC is implemented in C++, we use a python C++ wrapper to integrate LiveNAS and WebRTC. We add custom APIs by modifying `call.cc` in libWebRTC to obtain estimated network bandwidth, timestamp for each frame, raw decoded frames, and encoded frames. More specifically, we add an observer [48] inside decoder and encoder callback function, which returns the timestamp of each frame, decoded frame, and encoded frame from VP8 encoder. Finally, to retrieve the estimated network bandwidth from WebRTC [38], we extend the callback triggered upon network bandwidth change.

Training and Inference. We implement the online training process and inference process as a separate process in Pytorch. We use the "ultra-high" model from NAS [72] for super-resolution. The output of super-resolution is 1080p or 4K, and input can be either 270p, 360p, 540p, 720p or 1080p. When the training patches arrive at LiveNAS server, they are fed into the online training process along with the decoded frames from the video retrieved from libWebRTC. The online trainer utilizes the ADAM optimizer [53] to optimize DNN parameters. The number of iteration in one epoch, minibatch size, output patch size, and learning rate are set to 50, 64, 120, and 10^{-4}, respectively. The training uses single-precision (32-bit), but the inference is done with half-precision (16-bit) for speed. At the end of every training epoch, the inference process is synchronized. The final output is passed to multi-bitrate encoders at the distribution side.

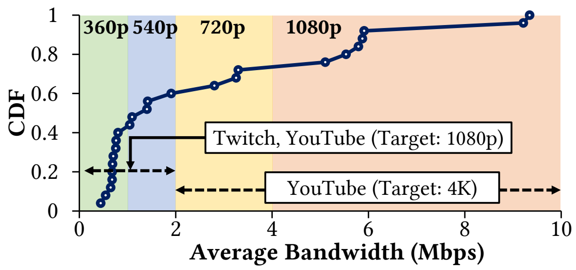
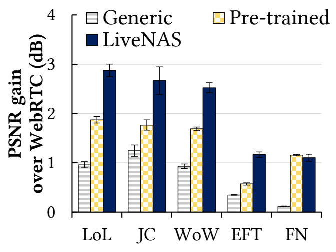


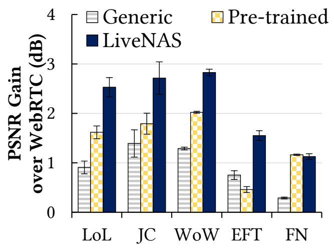
Figure 8: CDF of FCC traces (BW< 10 Mbps) and our 25 samples with ingest/target resolutions.

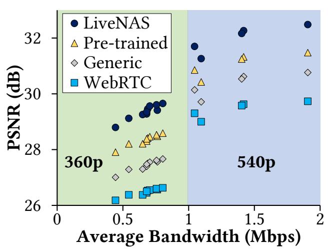
8 EVALUATION

We evaluate LiveNAS to answer the following questions:

- Does LiveNAS effectively enhance the video quality and enable 4K streaming under a constrained environment? (§8.1)


(a) Ingest: 360p, Target: 1080p


(b) Ingest: 540p, Target: 1080p


(c) Result for each trace: LoL

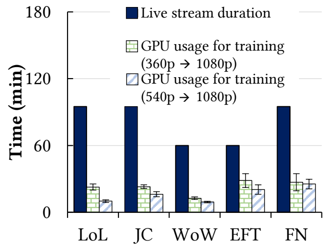
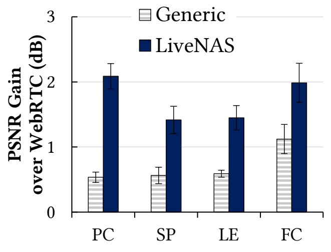
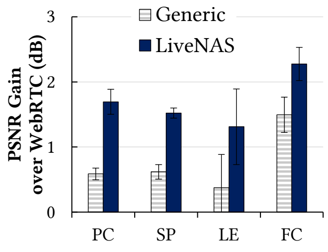
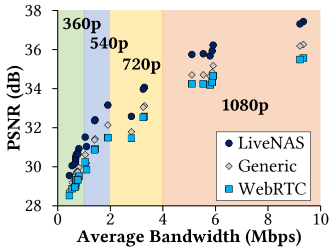

(d) Average GPU usage time.

Figure 9: LiveNAS end-to-end quality improvement and GPU usage time of Twitch top 5 video contents. (LoL: League of Legends, JC: Just Chatting, WoW: World of Warcraft, EFT: Escape from Tarkov, FN: Fortnite)


(a) Ingest: 720p, Target: 4K


(b) Ingest: 1080p, Target: 4K


(c) Result for each trace: Sports

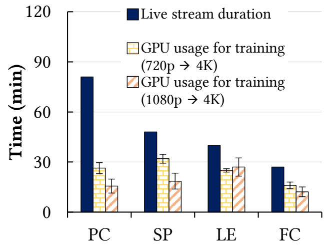

(d) Average GPU usage time.

Figure 10: LiveNAS end-to-end quality improvement and GPU usage time of YouTube 4K video contents. (PC: Podcast, SP: Sports, LE: Live Event, FC: Food/Cooking)

- How effective is content-adaptive online learning? (§8.2)
- What is the impact of the ingest improvement of LiveNAS on the distribution side? (§8.3)
- Why online learning is effective and how sensitive is LiveNAS to the training window? (§8.4)

Evaluation Setup. We use two Geforce RTX 2080 Ti GPUs at the ingest server, one for inference and the other for training, unless otherwise noted. We use 25 real-world network traces from 2019 FCC U.S. broadband uplink measurements [8]. The traces were sampled according to the bandwidth distribution of the entire dataset, excluding the top 38% whose average uplink bandwidth exceeded 10 Mbps to model a bandwidth-constrained environment. When streaming, we choose the original ingest resolution according to the average bandwidth of each trace, following the YouTube Live settings [30], as shown in Figure 8. When the original resolution is 360p or 540p (720p or 1080p) LiveNAS upscales it to 1080p (4K). We use Mahimahi's network emulation [58] to apply the network trace to LiveNAS and WebRTC streams.

We use streams from Twitch and YouTube. For Twitch, from the top five categories [16], we choose the most recent stream from the top streamer of each category who makes their history publicly available [20–24].[1] Thus, earlier streaming history was available, but the original quality was 1080p, and 4K was not available. To obtain 4K videos, we search YouTube for four popular types of live video (Podcast, Sports, Live event, Food/cooking) [32–35] and pick a 4K video with high quality (bitrate >16.8 Mbps) whose length is close to the average Twitch stream. However, for the YouTube videos, prior streaming sessions were not available. We cut the length to the average live stream of Twitch (95 mins [51]) if it is longer. With 9 videos streamed across 25 traces, the total streaming time amounts to 366 hours. We use WebRTC's default codec, VP8 [25], unless otherwise noted.

Metrics. We use a widely used peak-signal-to-noise ratio (PSNR) [49] and structural similarity index (SSIM) [69] metrics for video quality and also measure latency.

8.1 Video Quality Improvement

To demonstrate LiveNAS delivers significant quality improvement, we compare LiveNAS with three alternative designs:

- **WebRTC** does not utilize DNNs but uses bilinear interpolation to scale up to target resolution.
- **Generic super-resolution** uses a super-resolution DNN trained on DIV2K benchmark dataset [36].
- **Super-resolution with pre-trained model** uses a prior stream, if available, from the same streamer to pre-train the model. For Twitch streams, we hand-pick the best prior stream that closely resembles the current stream within five days prior to its air time. For a fair comparison, we use the same amount of GPU for training as LiveNAS.

Figure 9 (a) and (b) show the quality improvement over vanilla WebRTC in PSNR for each scheme by their original ingest resolution for Twitch streams. Figure 10 (a) and (b) show the same for YouTube video. We make LiveNAS and WebRTC video samples available at http://ina.kaist.ac.kr/~livenas/. (Figure 25 in Appendix B shows the quality improvement in SSIM.) The error bars represent the standard deviation of average video quality. Figures 9 (c) and 10 (c) present the absolute quality of a video for each network trace; Figures 26-29 in Appendix show the absolute quality and snippets of video frames for other streams. LiveNAS significantly outperforms vanilla WebRTC and generic super-resolution by at least 0.81 dB up to

[1]Streams are aired between Dec. 7, 2019 and Feb 7, 2020 depending on when experiments were conducted.

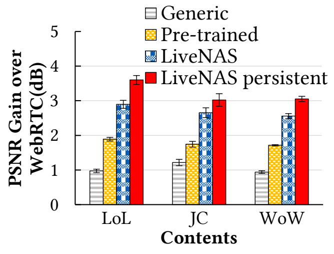


Figure 11: LiveNAS persistent training.

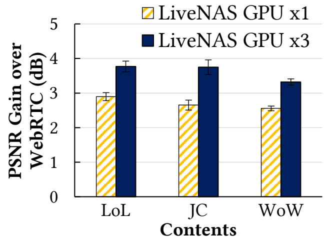


Figure 12: LiveNAS multi-GPU training.

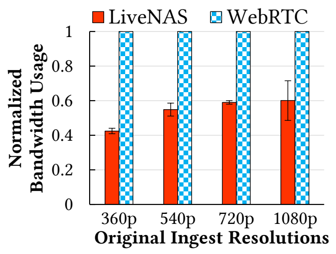


Figure 13: LiveNAS bandwidth savings.

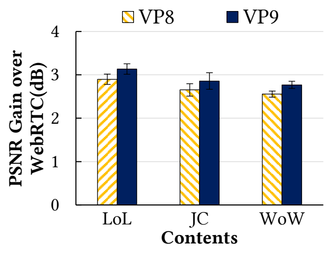


Figure 14: LiveNAS is codec-agnostic.

3.04 dB in PSNR. Generic super-resolution provides a much smaller improvement of 0.11 dB to 1.93 dB. In SSIM, the quality sometimes even is worse than vanilla WebRTC (Figure 25 in Appendix).

Compared to pre-training, online training delivers robust quality because the improvement does not depend on the training set. A pre-trained model can only be applied when similar content can be easily identified. LiveNAS outperforms the pre-trained model by 0.6 to 1.14 dB in PSNR, except for Fortnite whose quality was nearly the same as pre-trained. However, this result only appears when the best train data is carefully selected. As shown in §3, using streams from the previous day results in much poorer quality than LiveNAS (Figure 2c). This suggests, while pre-training might be useful in some context, online learning provides greater benefit without having to rely on choosing a good training set.

Note Fortnite's quality improvement is smaller than other streams for all schemes. This is because the nature of the game (first person shooter) involves highly dynamic movement, which is also reflected in the original encoded stream quality being the lowest, as shown in Figure 15.

Achieving higher quality. Persistent online learning and multi-GPU learning can further enhance the ingest quality. First, persistent online learning stores the DNN after a stream has ended and reuses it as a starting point for the next streaming event of the same streamer. This combines the benefit of pre-training and online learning in that it uses pre-trained knowledge while the enhancement of online learning is applied to the current stream. Figure 11 shows persistent online learning adds 0.37-0.7 dB gain over online learning for the top three Twitch streams on our traces. We project this is because it gets a larger set of training data from both earlier and current streams, enough to produce additional benefit, from Figure 2d.

Second, multi-GPU training gives an orthogonal opportunity for further improvement. Figure 12 shows using more GPUs improve the quality, but marginal quality gain shows a diminishing return. When multiple GPU is used, LiveNAS quality optimizer transmits more training data because the online learning gain becomes greater as learning is accelerated; a training epoch finishes faster, and the gain accumulated over multiple, shortened epochs becomes greater.

Implications on streamer bandwidth. LiveNAS provides much higher quality compared to WebRTC at the same bandwidth. We quantify how much more bandwidth is required for WebRTC to deliver the same quality as LiveNAS. For this, we increase the bandwidth of network traces by a factor greater than 1 and repeat the experiment. Figure 13 shows the bandwidth usage of LiveNAS normalized to WebRTC's bandwidth usage for each original ingest resolution when delivering the same quality. LiveNAS achieves the same quality as WebRTC using only 45.9% bandwidth on average.

Latency and ingest QoE implications. Table 2 in Appendix shows the inference delay at each resolution. LiveNAS adds 10-29 ms of latency depending on ingest resolution to the end-to-end average delay of 209.6 ms of WebRTC in our experiment with 10 ms network latency emulated by Mahimahi. The largest latency comes from processing and queuing delay inside the WebRTC ingest client.

According to the QoE model derived from a user study [44] using a video call, a latency increase of 100 msec is equivalent to 1.0 dB degradation in visual quality in SSIM. Note, LiveNAS produces a quality gain of 1.21 dB in SSIM on average with our traces, which results in a net gain in the ingest QoE. This is looking at the ingest side in isolation. In §8.3, we show the ingest enhancement produces a dramatic improvement on QoE for end viewers at the distribution side.

LiveNAS is codec-agnostic. We repeat the experiment of Figure 9 (a) with VP9 instead of VP8 with the top three Twitch videos. The result in Figure 14 shows the quality improvement is almost the same. This is because content-aware super-resolution [72] takes advantage of long-term redundancy that appears across multiple group of pictures (GoPs), while video codecs only concern redundancy within a GoP.

8.2 Resource Efficiency of LiveNAS

This section evaluates the resource efficiency of LiveNAS in various aspects. In particular, we compare LiveNAS with three baseline training methods:

- **One-time Customization** trains DNN upfront on the first few seconds of the stream. We use the first 60 seconds of the stream unless otherwise noted.
- **Online learning with early-stop** trains DNN upfront once for the time dynamically chosen by our gain saturation detection. After that, it never resumes training throughout the stream even when a scene change is detected.
- **Continuous online learning** trains DNN continuously throughout the stream without any suspension. The training time is identical to the streaming time.

Efficiency of online learning. Figures 9 (d) and 10 (d) show the GPU usage for online training compared to stream length. On average, LiveNAS performs online learning for only 35% of streaming time. The savings from content-adaptive training actually reflect the learning difficulty which depends on multiple factors: First, it depends on how dynamic the video is. Second, observe when the stream lasts longer, the savings generally become more significant.

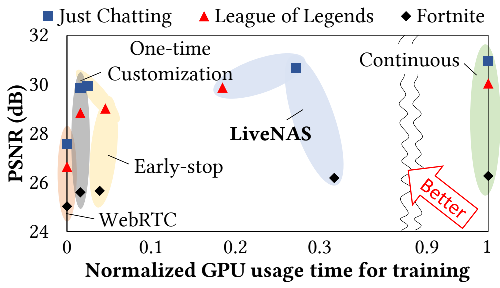


Figure 15: LiveNAS saves GPU usage.

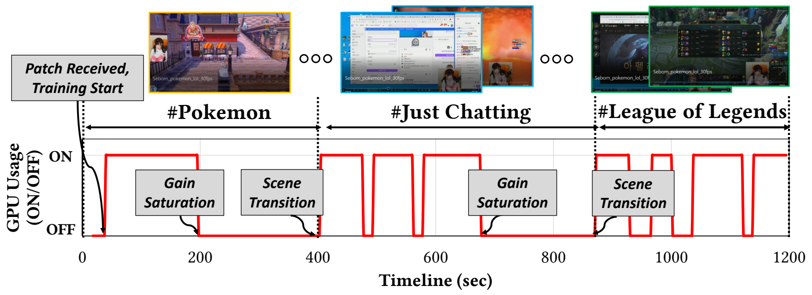


Figure 16: Case Study: Content-adaptive trainer in operation.

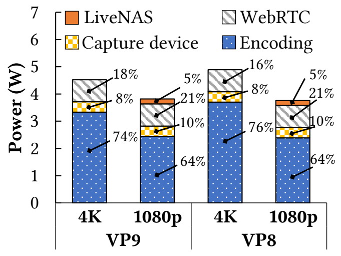


Figure 17: LiveNAS power savings.

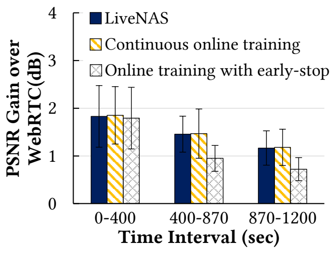
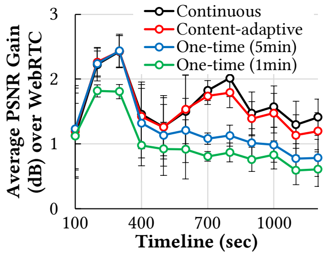
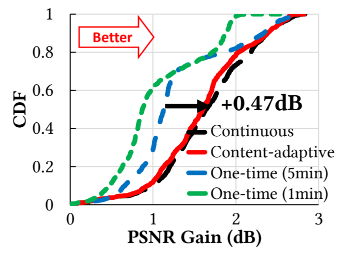


Figure 18: Case Study: adaptive learning.



(a) PSNR gain over Time.



(b) CDF of PSNR gain.

Figure 19: Content-adaptive vs One-time Customization.

For YouTube videos of different lengths, we see this trend. Second, the larger the scaling factor in super-resolution, it is more difficult to learn. For example, in 720p to 4K super-resolution, the scaling factor is x3, and 1080p to 4K is x2. Thus, the former requires more GPU usage, as shown in Figure 10 (d).

Figure 15 compares the GPU usage time in training and the resulting video quality for each video and scheme. LiveNAS delivers almost the same quality while using 25% of GPU resources compared to continuous training. Early-stop and one-time customization, however, come at a significant decrease in quality because it does not keep up with scene changes.

Finally, despite the complex relationship that determines the learning difficulty and video quality, we confirm our quality-optimizing scheduler outperforms any scheme that allocates fixed bandwidth for transmitting high-quality training labels in terms of video quality (not shown in figures).

Cost. We quantify the computation cost assuming the ingest bandwidth follows the distribution used in §8.1, the ingest server is located at a public cloud, the cost is amortized over a large number of live streams, and overhead of model update across distributed GPU is disregardable. One GPU is used for training and multiple GPUs are used to ensure real-time inference for 4K (as shown in Appendix B Table 2). The average cost is $4.69 per hour per stream using a 16 vCPU machine with four Nvidia Tesla V100 GPUs on Google Cloud.

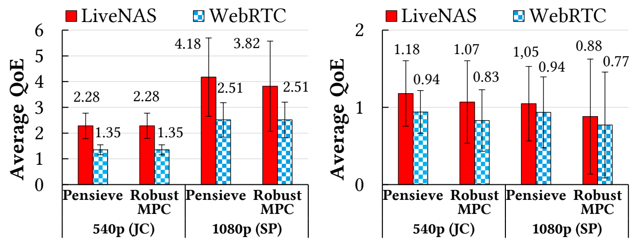
Energy-efficiency of LiveNAS client. LiveNAS consumes less energy due to the fact that clients can perform encoding at lower resolution thanks to the super-resolution at the ingest server. This can be crucial for mobile clients that operate on battery (e.g., for outdoor streaming).

We compare the power consumption of a client against WebRTC using NVIDIA's Jetson TX2, an embedded system whose main chip has modules for video encoding/decoding for major codecs, including VP8 and VP9. We used TX2's internal power monitor and measure the power consumption of the TX2 chip and the board with its peripherals. We compare 4K stream on WebRTC at 9.5 Mbps versus LiveNAS that provide the same quality with 1080p live ingest at 7 Mbps. Figure 17 shows the power consumption. LiveNAS saves 16% and 23% power for VP9 and VP8 encoding respectively, while providing the same quality. This is primarily due to 4K encoding consuming 36.3% and 54.7% more power than 1080p encoding for VP9 and VP8 respectively.

Case study. We perform a case study to show how content-adaptive training works with an actual live stream from one of the top streamers on Twitch.tv [18]. Figure 16 shows the timeline of the stream, which shows multiple transitions within the first 20 minutes. As shown, it is not uncommon for streamers to change content during a live stream.

When the stream goes live, the trainer performs online training within 12 seconds, until it detects gain saturation (at $t = 203$ sec). However, when new content appears (at $t = 400$), the trainer detects this and resumes online training. This is repeated multiple times throughout. Content-adaptive learning reduces the GPU usage by 54% compared to continuous training but still shows comparable quality to that of continuous training. In contrast, the early-stop scheme's super-resolution gain diminishes over scene transitions as shown in Figure 18. The result shows LiveNAS' online learning effectively adapts to scene transitions to produce high-quality live stream, while delivering resource efficiency.

To better understand where the effectiveness of content-adaptive learning stems from, we measure the quality gain over streaming time in this scenario. As shown in Figure 19a, the quality gain from one-time customization diminishes over time (>= 62sec) especially when video content changes (e.g., Just Chatting from Pokemon). In contrast, content-adaptive learning consistently achieves the quality gain comparable to that of continuous training. Overall, as



(a) FCC broadband traces (b) Pensieve traces

Figure 20: Impact on viewer QoE at distribution-side.

shown in Figure 19b, it improves the quality by 0.43 dB on average over all frames and median quality by 0.47 dB compared to the one-time customization baselines.

8.3 QoE Improvement on Distribution Side

LiveNAS improves the quality of experience (QoE) of live stream viewers because it enhances the ingest stream quality. To quantify this, we measure the *QoE of viewers* at the distribution side by creating a video distribution setting that provides adaptive streaming to end viewers. We use "Just Chatting" video [22] from Twitch and "Sports" [35] from YouTube for adaptive streaming on two traces: 1) 100 downlink traces randomly sampled from 2019 FCC U.S. broadband traces [8] (avg. bandwidth=72 Mbps); and 2) 3G and broadband traces used in Pensieve (average is 1.48 Mbps). We use Pensieve's adaptive streaming setting and simulator for measuring the linear QoE [57]. We use Pensieve and robustMPC [73] as the adaptive bitrate (ABR) algorithm. The ingest resolution for the Twitch stream is 540p, and 1080p for the YouTube video. LiveNAS then upscales the ingest stream into 1080p and 4K for Twitch and YouTube videos respectively. For YouTube, we add 2K and 4K chunk options.

The widely-used QoE metric [57, 72, 73] for adaptive streaming takes in the bitrate of each chunk. To quantify the visual enhancement of LiveNAS in bitrate, we created an inverse mapping from video quality to the corresponding bitrate, similar to how it is done in NAS [72], using WebRTC's encoding as the baseline. This allows us to obtain the "effective bitrate" of video chunks sourced from LiveNAS and reflect the effect of super-resolution on the QoE metric.

Figure 20 shows LiveNAS ingest delivers 27%-69% improvement in average QoE for the Twitch stream and 12%-66% improvement for the YouTube video over WebRTC ingest. The improvement comes from two reasons: 1) LiveNAS provides higher resolution chunks, benefiting viewers who have ample bandwidth. 2) Each encoded chunk is of better quality because the ingest video quality improves with LiveNAS. The improvement for the FCC broadband traces (Figure 20a) is much larger than that for Pensieve traces (Figure 20b) because the former has higher bandwidth to receive higher resolution chunks with better quality (Figure 10c).

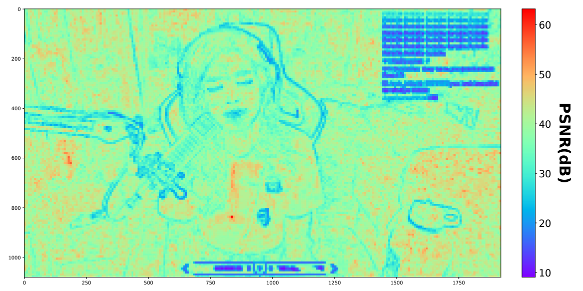
In many cases, the improvement LiveNAS gives is much larger than what the advanced ABR algorithm produces over its predecessor. For example, Pensieve's improvement over robustMPC is at most 13% for Twitch video. But LiveNAS gives 27% improvement over WebRTC ingest on the same traces. This implies that in live streaming, ingest-side improvement is crucial and it is at least as important as ABR algorithms for enhancing the QoE of end viewers.

8.4 LiveNAS Deep Dive

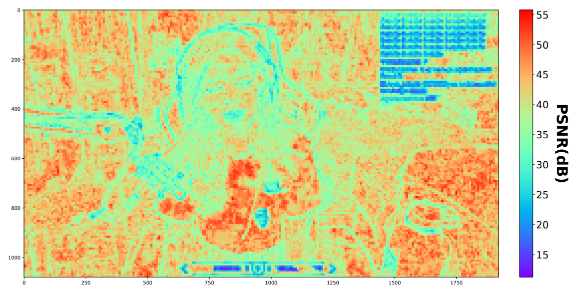
Why online learning is effective in live video streaming? There are two key observations that make online learning feasible for live video streaming. First, as illustrated in Figure 22, while training a super-resolution DNN, the majority of quality gain is achieved in the first few epochs because the DNN delivers steep diminishing gain over training time. Thus, even though the DNN is trained only during a live streaming session, it still provides large quality improvement (e.g., 1.83-2 dB). Next, a small portion of patches (2.5% per frame) can effectively benefit the entire video streaming. This is because subsequent video frames bear large temporal redundancy, and therefore, attaining better super-resolution for even a part of a frame has a lasting benefit for multiple subsequent frames as shown in Figure 2d and Figure 24. Moreover, due to the compression artifacts and patterns in the high-frequency regions specific to the video, the quality of other areas, which are not included in the training patches, also improves (refer Figure 21).
How sensitive is LiveNAS to the training window? To select patch bitrate at the beginning of each training epoch, LiveNAS 's quality-optimizing scheduler (§5.1) uses two recent DNNs to predict the quality gain of a DNN trained for the next training epoch. As demonstrated in Figure 23a, the length of each training epoch (i.e., training window) has a large impact on the prediction accuracy. The result shows that the prediction error is minimal at 5 seconds, which is our default setting, while increasing epoch length to 20-80 seconds produces 31-93% large error. This is because as the time distance between two DNNs becomes closer, the DNN quality improvement comes closer to resembling the actual value at the current time period. However, further shortening epoch results in 37% higher error as training with shorter epoch is not long enough to observe the effect of online training. Finally, the more accurate prediction leads to larger overall quality gain as demonstrated in Figure 23b. On average, 5-second epoch achieves 0.3 dB higher quality compared to the other settings; here, both the accurate prediction and the frequent update of DNN increase the quality.

9 RELATED WORK ON LIVE STREAMING

Low-latency streaming protocols. Among many industry-led designs [27, 28], WebRTC [26] is a free open source framework that enables near-simultaneous communications by direct peer-to-peer communications. Commercial media servers provide live broadcasting by combining WebRTC ingest with delivery via HTTP adaptive streaming [1, 29]. The new standards [3, 4] break video into smaller chunks to minimize the latency due to chunk-based encoding and publishing. LiveNAS focuses on improving the video quality on top of existing low-latency protocols for live video.
Integrating codec and transport. Salsify [44] integrates video codec and network protocol to match and respond quickly to the available bandwidth in real-time video streaming. As a result, Salsify improves user QoE in terms of delay and video quality. However, its optimization strategy requires a "purely functional video codec" that allows applications to explore alternative encodings of different quality levels [44, 45]. Unlike Salsify, LiveNAS is agnostic to codec, but focuses on enhancing video quality by applying neural computation. Alternatively, LiveNAS can also employ functional



(a) PSNR heatmap of the original ingest stream. It shows the PSNR between high resolution (1080p) and low resolution WebRTC (360p, 800kbps) with bilinear upsampling. Closer to red indicates high PSNR. (i.e., the pixel is similar to the original ground truth value).



(b) PSNR heatmap after online training. It shows the PSNR between high resolution (1080p) and LiveNAS (360p→1080p, 800kbps). The quality of other areas—not included in the training patches—also improves. Figure24a in Appendix shows the transmitted patches.

Figure 21: Small fraction of data is sufficient for online training. (Twitch content [22])

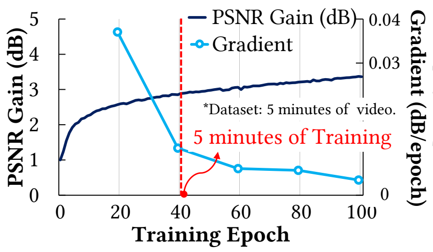
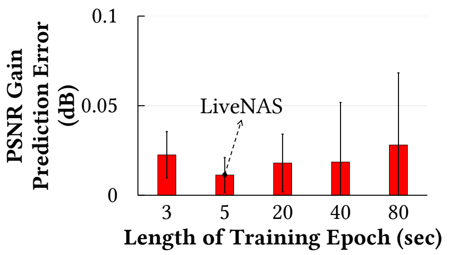
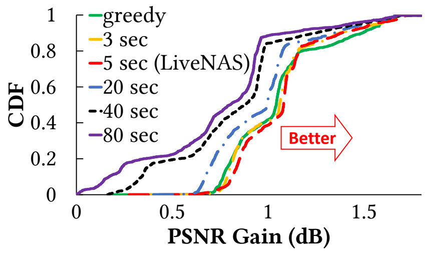


Figure 22: The majority of quality gain is achieved in the first few epochs.



(a) Prediction error of DNN quality improvement with various training windows.



(b) LiveNAS PSNR gain over WebRTC with various training windows.

Figure 23: LiveNAS sensitivity to training window.

codecs, which will allow us to determine the quality of encoding at different bitrates more accurately in §5.1.

Time-shifted viewing. Vantage [61] observes live streams are watched by viewers at different delays. Considering the time-shifted viewers, it improves the overall quality by selectively re-transmitting the previously delivered frames with better quality. The objective is to optimize the overall user QoE by balancing between QoE improvement for delayed viewers and potential QoE loss of real-time viewers. LiveNAS on the other hand enhances the quality of the original live stream directly, benefiting all viewers. We believe the approaches of Vantage and LiveNAS are orthogonal and further improvements can be made by combining the two.

Use of pre-trained DNNs. Inspired by the fact that recurring video conferencing sessions bear significant similarity, Dejavu [50] presents preliminary work that explores the possibility of applying quality-enhancing DNNs using a pre-trained model from prior video conferencing sessions. Yet, this limits its content to video conferencing with high similarities and cannot fully cover ever-growing user generated live video [51, 60]. LiveNAS demonstrates online training is not only feasible but benefits a wide range of live streams.

Use of super-resolution on ingest. Edge-to-cloud video analytics frameworks require high-quality video input at the cloud servers for high accuracy, but the bandwidth between edge and cloud can be scarce. To address the challenge, CloudSeg [68] presents preliminary work that maximizes analytics accuracy by applying super-resolution on ingest video streams at the cloud server. Both LiveNAS and CloudSeg use super-resolution on ingest stream, however, LiveNAS differs from CloudSeg in two aspects. First, LiveNAS maximizes the QoE of live stream viewers while CloudSeg maximizes the accuracy of analytics tasks. Second, to maximize the quality gain, LiveNAS sends training data along with ingest stream and uses DNNs that is freshly updated online while CloudSeg uses DNNs trained offline. We believe our online learning approach can be also applied to the video analytics frameworks to improve performance.

10 CONCLUSION

We present LiveNAS, a new live video ingest framework that utilizes super-resolution deep neural networks to enhance the live video quality independent of the ingest-side network bandwidth. LiveNAS applies online training, a new approach that addresses the core challenge of utilizing super-resolution in the context of live video delivery. LiveNAS introduces novel design components, including quality-optimizing scheduler and content-adaptive trainer, to fully realize the benefit of online training. LiveNAS achieves an average of 1.96 dB overall video quality improvement in PSNR over WebRTC across various real-world network traces and produces significant (12%-69%) QoE improvement for live stream viewers.

ACKNOWLEDGMENTS

We thank our shepherd Ganesh Ananthanarayanan and anonymous reviewers for their valuable feedback. We also thank Chan Ju Chong for his fruitful feedback. This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (Ministry of Science and ICT) [No.2018-0-00693].

REFERENCES

[1] Ant Media Server Official Website. https://antmedia.io/.

[2] Apple HTTP Live Streaming Official Website. https://developer.apple.com/streaming/.

[3] Apple Low-Latency HLS Specification. https://developer.apple.com/documentation/http_live_streaming/protocol_extension_for_low-latency_hls_preliminary_specification.

[4] Chunked-encoded and Chunked-transferred CMAF Specification. https://www.akamai.com/us/en/multimedia/documents/white-paper/low-latency-streaming-cmaf-whitepaper.pdf.

[5] Cisco Visual Networking Index Report. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.pdf.

[6] DASH Industry Forum Official Website. https://dashif.org/.

[7] Encoding Video at the Edge with Intel® Xeon® Processors. https://builders.intel.com/docs/networkbuilders/encoding-video-at-the-edge-with-intel-xeon-processors.pdf.

[8] FCC Broadband Bandwidth Measurement. https://www.fcc.gov/reports-research/reports/measuring-broadband-america/raw-data-measuring-broadband-america-eighth.

[9] libWebRTC Official Github Repository. https://github.com/aisouard/libwebrtc.

[10] Live Streaming Statistics about Twitch and Facebook. https://www.theverge.com/2020/1/9/21058907/twitch-youtube-mixer-facebook-live-streaming-numbers-growth-q4.

[11] Real-Time Messaging Protocol (RTMP) Specification. https://wwwimages2.adobe.com/content/dam/acom/en/devnet/rtmp/pdf/rtmp_specification_license_1.0.pdf.

[12] Real Time Streaming Protocol (RTSP) Specification. https://tools.ietf.org/html/rfc2326.

[13] Real-time Transport Protocol (RTP) Specification. https://tools.ietf.org/html/rfc3550.

[14] Streamlabs Q2 2019 Live Streaming Industry Report. https://blog.streamlabs.com/q2-2019-7e8039277b11.

[15] Twitch Broadcasting Guidelines. https://stream.twitch.tv/encoding/.

[16] Twitch Most Popular Game Category on February 2020. https://twitchtracker.com/games/rating. Last accessed Feb. 6, 2020.

[17] Twitch Official Website. https://www.twitch.tv/.

[18] Twitch Statistics about Streamer Channel (Saddummy). https://www.twitch.tv/saddummy.

[19] Twitch Statistics Report by Influencer MarketingHub. https://influencermarketinghub.com/twitch-statistics/.

[20] Twitch Stream Dataset (Escape from Tarkov). https://www.twitch.tv/stylishnoob4/.

[21] Twitch Stream Dataset (Fornite). https://www.twitch.tv/nickmercs/videos.

[22] Twitch Stream Dataset (Just Chatting). https://www.twitch.tv/pokimane/videos.

[23] Twitch Stream Dataset (League of Legends). https://www.twitch.tv/riotgames/videos.

[24] Twitch Stream Dataset (World of Warcraft). https://www.twitch.tv/method/.

[25] WebM Official Website. https://www.webmproject.org/.

[26] WebRTC Official Website. https://webrtc.org/.

[27] Wowza 2019 Live-Streaming Advancements Report. https://www.wowza.com/blog/streaming-advancements-2019.

[28] Wowza Live Streaming Guidebook. https://www.wowza.com/uploads/images/The-Complete-Guide-to-Live-Streaming-Wowza.pdf.

[29] Wowza WebRTC Streaming Guidebook. https://www.wowza.com/low-latency/webrtc-streaming-software.

[30] Youtube Live Encoding Guidelines. https://support.google.com/youtube/answer/2853702?hl=en.

[31] Youtube Live Streaming Official Website. https://www.youtube.com/live.

[32] Youtube Video Dataset (Food). https://www.youtube.com/watch?v=oLbRhd7p0Ts.

[33] Youtube Video Dataset (Live Event). https://www.youtube.com/watch?v=aVZngNyEsjg.

[34] Youtube Video Dataset (Pod Cast). https://www.youtube.com/watch?v=CpQZRLxJE5M&amp=&index=14.

[35] Youtube Video Dataset (Sport). https://www.youtube.com/watch?v=ppb7hmQA6sU.

[36] Eirikur Agustsson and Radu Timofte. 2017. NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

[37] Ghufran Baig, Jian He, Mubashir Adnan Qureshi, Lili Qiu, Guohai Chen, Peng Chen, and Yinliang Hu. 2019. Jigsaw: Robust live 4k video streaming. In *The 25th Annual International Conference on Mobile Computing and Networking*. 1–16.

[38] Gaetano Carlucci, Luca De Cicco, Stefan Holmer, and Saverio Mascolo. 2016. Analysis and Design of the Google Congestion Control for Web Real-Time Communication (WebRTC). In *Proceedings of the 7th International Conference on Multimedia Systems (MMSys '16)*. Article Article 13, 12 pages.

[39] M. Dasari, A. Bhattacharya, S. Vargas, P. Sahu, A. Balasubramanian, and S. R. Das. 2020. Streaming 360◦ Videos using Super-resolution. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*.

[40] J. Deng, F. Cuadrado, G. Tyson, and S. Uhlig. 2015. Behind the game: Exploring the twitch streaming platform. In *2015 International Workshop on Network and Systems Support for Games (NetGames)*. 1–6.

[41] Florin Dobrian, Vyas Sekar, Asad Awan, Ion Stoica, Dilip Joseph, Aditya Ganjam, Jibin Zhan, and Hui Zhang. 2011. Understanding the Impact of Video Quality on User Engagement. *SIGCOMM Comput. Commun. Rev.* 41, 4 (Aug. 2011), 362–373.

[42] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. 2014. Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*. Springer, 184–199.

[43] Zhengfang Duanmu, Kai Zeng, Kede Ma, Abdul Rehman, and Zhou Wang. 2016. A quality-of-experience index for streaming video. *IEEE Journal of Selected Topics in Signal Processing* 11, 1 (2016), 154–166.

[44] Sadjad Fouladi, John Emmons, Emre Orbay, Catherine Wu, Riad S. Wahby, and Keith Winstein. 2018. Salsify: Low-Latency Network Video through Tighter Integration between a Video Codec and a Transport Protocol. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. Renton, WA, 267–282.

[45] Sadjad Fouladi, Riad S. Wahby, Brennan Shacklett, Karthikeyan Vasuki Balasubramaniam, William Zeng, Rahul Bhalerao, Anirudh Sivaraman, George Porter, and Keith Winstein. 2017. Encoding, Fast and Slow: Low-Latency Video Processing Using Thousands of Tiny Threads. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*. Boston, MA, 363–376.

[46] Jeremy Fowers, Kalin Ovtcharov, Michael Papamichael, Todd Massengill, Ming Liu, Daniel Lo, Shlomi Alkalay, Michael Haselman, Logan Adams, Mahdi Ghandi, et al. 2018. A configurable cloud-scale DNN processor for real-time AI. In *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 1–14.

[47] F. Fund, C. Wang, Y. Liu, T. Korakis, M. Zink, and S. S. Panwar. 2013. Performance of DASH and WebRTC Video Services for Mobile Users. In *2013 20th International Packet Video Workshop*. 1–8.

[48] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. 1994. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading, Mass. http://www.worldcat.org/search?qt=worldcat_org_all&q=0201633612

[49] Alain Hore and Djemel Ziou. 2010. Image quality metrics: PSNR vs. SSIM. In *2010 20th International Conference on Pattern Recognition*. IEEE, 2366–2369.

[50] Pan Hu, Rakesh Misra, and Sachin Katti. 2019. Dejavu: Enhancing Videoconferencing with Prior Knowledge. In *Proceedings of the 20th International Workshop on Mobile Computing Systems and Applications (HotMobile '19)*. 63–68.

[51] Mehdi Kaytoue, Arlei Silva, Loïc Cerf, Wagner Meira, and Chedy Raïssi. 2012. Watch Me Playing, I am a Professional: A First Study on Video Game Live Streaming. In *Proceedings of the 21st International Conference on World Wide Web (WWW '12 Companion)*. 1181–1188.

[52] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. 2016. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1646–1654.

[53] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[54] Royson Lee, Stylianos I Venieris, Lukasz Dudziak, Sourav Bhattacharya, and Nicholas D Lane. 2019. MobiSR: Efficient On-Device Super-Resolution through Heterogeneous Mobile Processors. In *The 25th Annual International Conference on Mobile Computing and Networking*. 1–16.

[55] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. 2017. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 136–144.

[56] Siwei Ma, Wen Gao, and Yan Lu. 2005. Rate-distortion analysis for H. 264/AVC video coding and its application to rate control. *IEEE transactions on circuits and systems for video technology* 15, 12 (2005), 1533–1544.

[57] Hongzi Mao, Ravi Netravali, and Mohammad Alizadeh. 2017. Neural Adaptive Video Streaming with Pensieve. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '17)*. Association for Computing Machinery, 197–210.

[58] Ravi Netravali, Anirudh Sivaraman, Somak Das, Ameesh Goyal, Keith Winstein, James Mickens, and Hari Balakrishnan. 2015. Mahimahi: Accurate Record-and-Replay for HTTP. In *Proceedings of the USENIX Annual Technical Conference (ATC)*. 417–429.

[59] Karine Pires and Gwendal Simon. 2014. Dash in twitch: Adaptive bitrate streaming in live game streaming platforms. In *Proceedings of the 2014 Workshop on Design, Quality and Deployment of Adaptive Video Streaming*. 13–18.

[60] Karine Pires and Gwendal Simon. 2015. YouTube Live and Twitch: A Tour of User-Generated Live Streaming Systems. In *Proceedings of the 6th ACM Multimedia Systems Conference (MMSys '15)*. 225–230.

[61] Devdeep Ray, Jack Kosaian, K. V. Rashmi, and Srinivasan Seshan. 2019. Vantage: Optimizing Video Upload for Time-Shifted Viewing of Social Live Streams. In *Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM*

'19). 380–393.

[62] Haakon Riiser, Paul Vigmostad, Carsten Griwodz, and Pål Halvorsen. 2013. Commute path bandwidth traces from 3G networks: analysis and applications. In *Proceedings of the 4th ACM Multimedia Systems Conference*. ACM, 114–118.

[63] Shai Shalev-Shwartz et al. 2012. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning* 4, 2 (2012), 107–194.

[64] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. 2016. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1874–1883.

[65] Assaf Shocher, Nadav Cohen, and Michal Irani. 2018. "zero-shot" super-resolution using deep internal learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3118–3126.

[66] Iraj Sodagar. 2011. The mpeg-dash standard for multimedia streaming over the internet. *IEEE multimedia* 18, 4 (2011), 62–67.

[67] Mikko Uitto. 2016. Energy consumption evaluation of H. 264 and HEVC video encoders in high-resolution live streaming. In *2016 IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE, 1–7.

[68] Yiding Wang, Weiyan Wang, Junxue Zhang, Junchen Jiang, and Kai Chen. 2019. Bridging the Edge-Cloud Barrier for Real-time Advanced Vision Analytics. In *11th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 19)*. USENIX Association, Renton, WA. https://www.usenix.org/conference/hotcloud19/presentation/wang

[69] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing* 13, 4 (2004), 600–612.

[70] David H Wolpert. 1996. The lack of a priori distinctions between learning algorithms. *Neural computation* 8, 7 (1996), 1341–1390.

[71] Hyunho Yeo, Sunghyun Do, and Dongsu Han. 2017. How will Deep Learning Change Internet Video Delivery?. In *Proceedings of the 16th ACM Workshop on Hot Topics in Networks*. ACM, 57–64.

[72] Hyunho Yeo, Youngmok Jung, Jaehong Kim, Jinwoo Shin, and Dongsu Han. 2018. Neural Adaptive Content-aware Internet Video Delivery. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*. Carlsbad, CA, 645–661.

[73] Xiaoqi Yin, Abhishek Jindal, Vyas Sekar, and Bruno Sinopoli. 2015. A control-theoretic approach for dynamic adaptive video streaming over HTTP. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*. 325–338.

[74] Cong Zhang and Jiangchuan Liu. 2015. On Crowdsourced Interactive Live Streaming: A Twitch.Tv-Based Measurement Study. In *Proceedings of the 25th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV '15)*. 55–60.

[75] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. 2018. Residual dense network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2472–2481.

[76] Zhengdong Zhang and Vivienne Sze. 2017. FAST: A framework to accelerate super-resolution processing on compressed videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 19–28.

Appendices are supporting material that has not been peer-reviewed.

APPENDIX A ALGORITHM OF LIVENAS

Algorithm 1: Content-Adaptive Online Training

```
set initial STATE as Training;
repeat
    switch STATE
        case Training
            for each training epoch
                Compute diff of $Q_{DNN_t}$ and $Q_{DNN_{t-1}}$;
                if diff < $thresh_{sat}$
                    patience += 1;
                    if patience > $count_{sat}$
                        Suspend online training;
                        STATE = Suspended;
                        patience = 0;
                else
                    patience = 0;
        case Suspended
            for each validation period
                Compute diff of $Q_{DNN_t}$ and $Q_{DNN_{t=0}}$;
                if diff < $thresh_{online}$
                    patience += 1;
                    if patience > $count_{online}$
                        resume online training;
                        STATE = Training;
                        patience = 0;
                else
                    patience = 0;
until stream ends;
```

APPENDIX B ADDITIONAL RESULTS

Quality improvement in SSIM. Figure 25 shows the quality improvement of LiveNAS in structrual similarity index (SSIM) [69] over vanilla WebRTC for top three category videos from Twitch and four Youtube videos. LiveNAS outperforms the pretrained model except for Fortnite. This is similar to the PSNR result in §8.1. However, the quality of generic super-resolution often even worse than vanilla WebRTC. This shows generic super-resolution does not cover the diversity that video content exhibits.

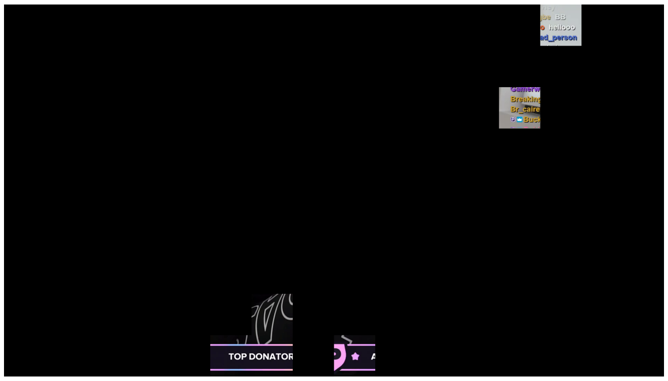
Quality improvement for all videos. Figures 26 and 27 show the absolute quality of LiveNAS compared to WebRTC and generic super-resolution in PSNR for top three Twitch and Youtube videos respectively. LiveNAS delivers consistent benefit over WebRTC and generic super-resolution as summarized in §8.1.

Screenshot examples. Figures 28 and 29 respectively show snippets of video frames from Twitch and Youtube video we used in the evaluation. The snippets of vanilla WebRTC and LiveNAS highlights a clear improvement in visual quality that LiveNAS provides.

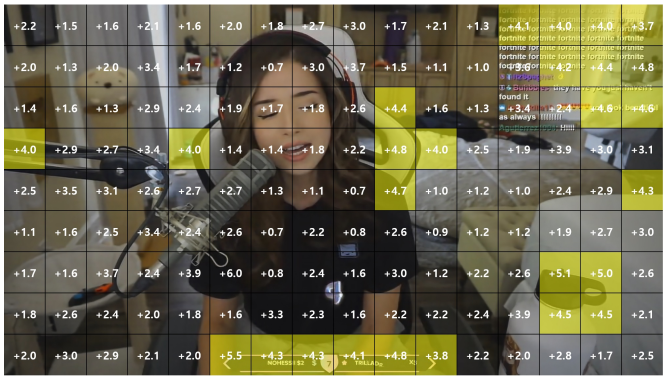
Super-resolution Inference Delay. Table 2 shows the delay introduced by super-resolution. It includes the time to transmit a frame to GPU, perform DNN inference, and copy the result back to the CPU. When multiple GPU is used, we include the preprocessing and postprocessing time as well. We show six different ingest resolutions with two target resolutions. Note LiveNAS can support any ingest resolutions. For example, LiveNAS can also support 240p, 480p resolutions by performing bilinear upsampling followed by super-resolution with scale factor of x4 and x2 to obtain 1080p target resolution [72]. For simplicity however, we evaluated with 540p instead of 480p which enables super-resolution with integer up-scale factor.

Ingest	Upscale	Target	fps	Delay	# of GPU
270p	x4	**1080p**	46	21ms	x1
360p	x3	**1080p**	39	25ms	x1
540p	x2	**1080p**	41	24ms	x1
720p	x1(w/ bilinear)	**1080p**	99	10ms	x1
720p	x3	**4K**	43	23ms	x3
1080p	x2	**4K**	34	29ms	x3

Table 2: LiveNAS super-resolution inference delay.

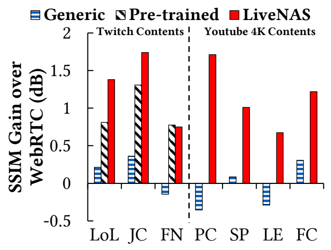


(a) Example of transmitted patches selected by our patch selection algorithm. On average, 6 patches are transmitted every 2 seconds which translates to 103.2 Kbps. Our patch selection algorithm preferentially selects such areas that have room for improvement.

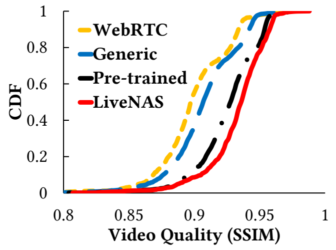


(b) Patch-wise psnr improvement after online training. Yellow patches indicate significant PSNR improvement (over 85% percentile). Notice the quality of every patch improves, particularly the ones that show similar patterns with the patches that LiveNAS sends.

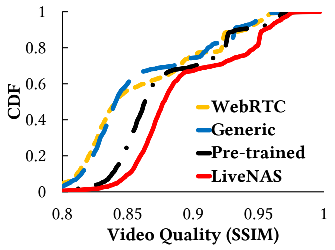
Figure 24: Selected patches and resulting patch-wise PSNR improvement. (Twitch content [22])



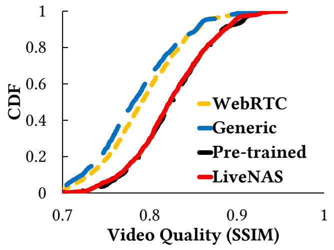
(a) Average SSIM gain of (b-h).



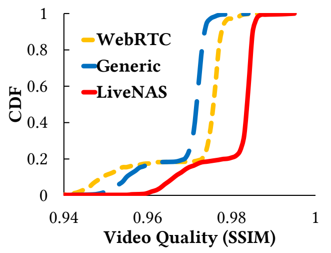
(b) CDF of Just Chatting



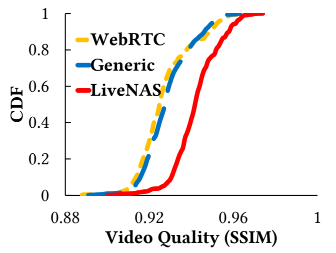
(c) CDF of League of Legends



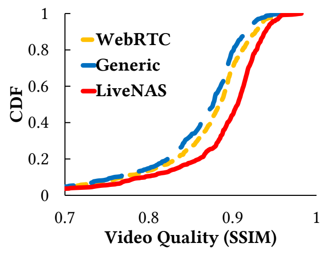
(d) CDF of Fortnite



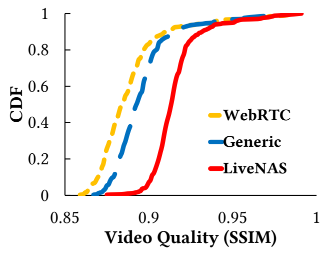
(e) CDF of Podcast



(f) CDF of Sports

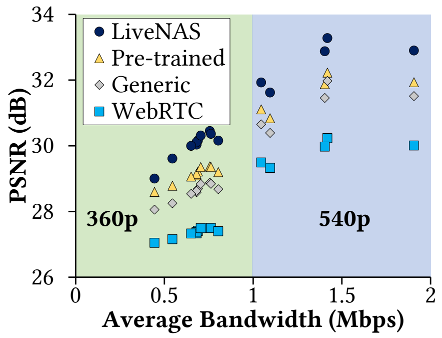


(g) CDF of Live Event

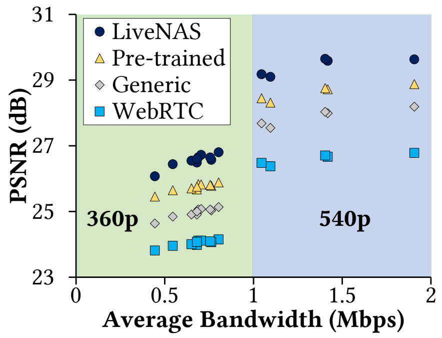


(h) CDF of Food/Cooking

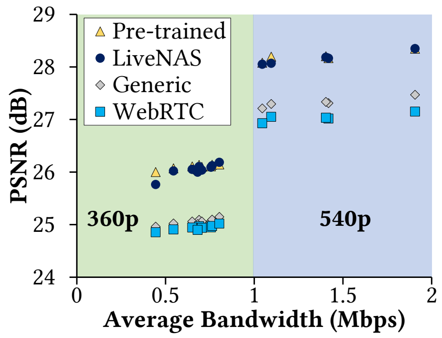
Figure 25: Cumulative distribution of video frames and video quality improvement in SSIM.



(a) Content: Just Chatting

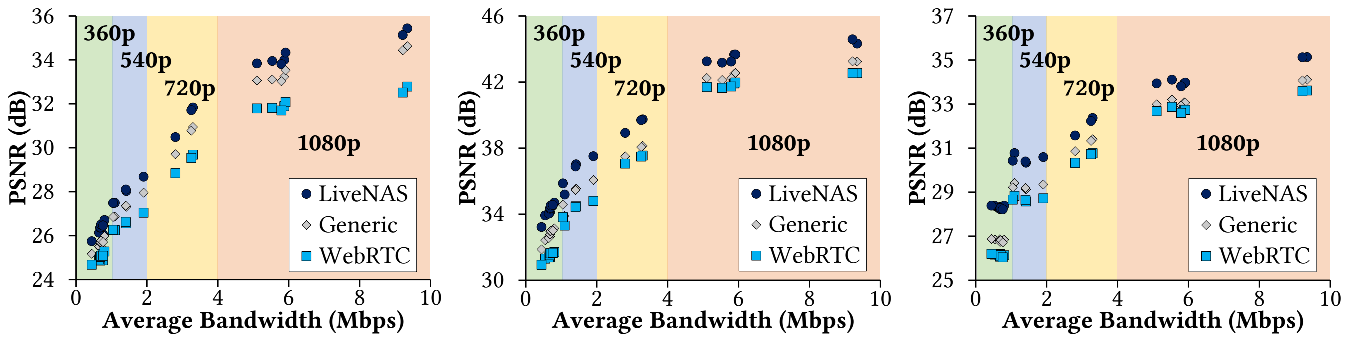


(b) Content: World of Warcraft



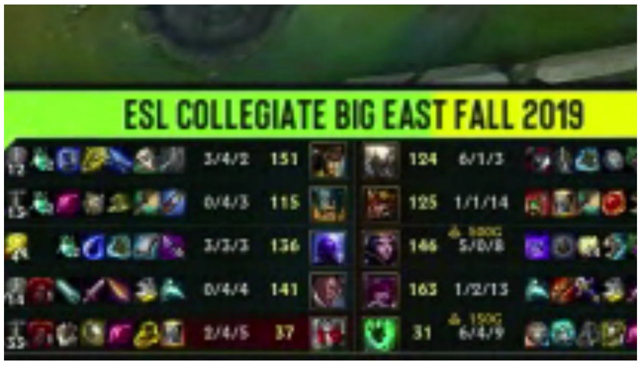
(c) Content: Fortnite

Figure 26: LiveNAS end-to-end quality improvement of Twitch video contents for each trace (Target: 1080p).

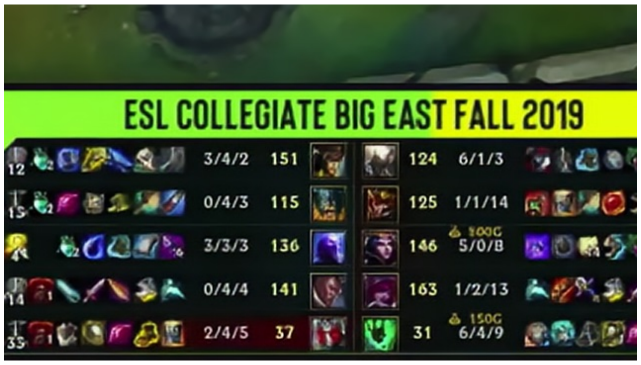


(a) Content: Food/Cooking (b) Content: Podcast (c) Content: Live Event

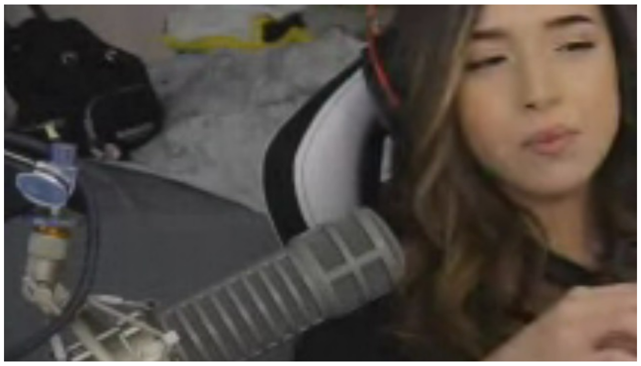
Figure 27: LiveNAS end-to-end quality improvement of Youtube video contents for each trace (Target: 4K).



(a) League of Legends (WebRTC) / PSNR: 25.5 dB, SSIM: 0.85

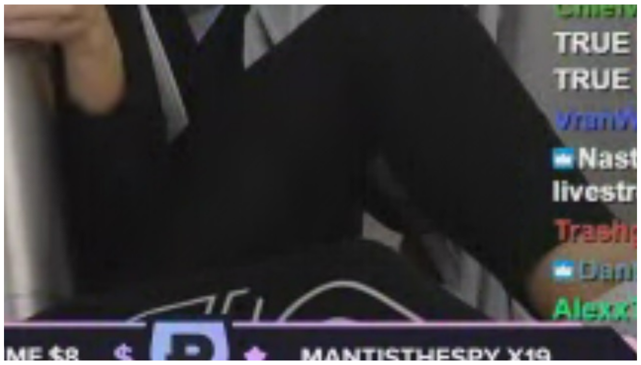


(b) League of Legends (LiveNAS) / PSNR: 30 dB, SSIM: 0.9

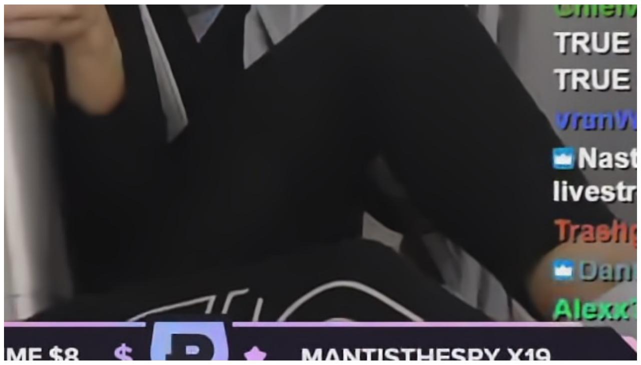


(c) Just Chatting (WebRTC) / PSNR: 27.1 dB, SSIM: 0.9

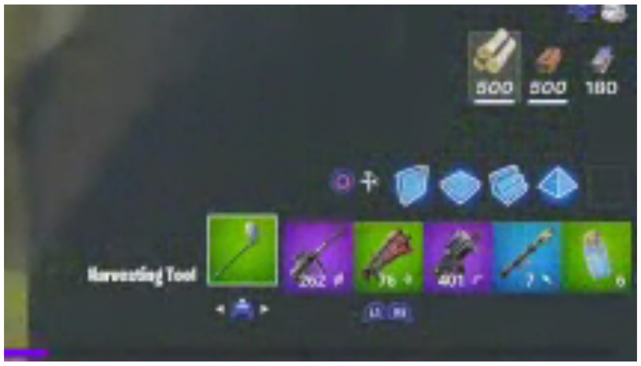
(d) Just Chatting (LiveNAS) / PSNR: 33.2 dB, SSIM: 0.96



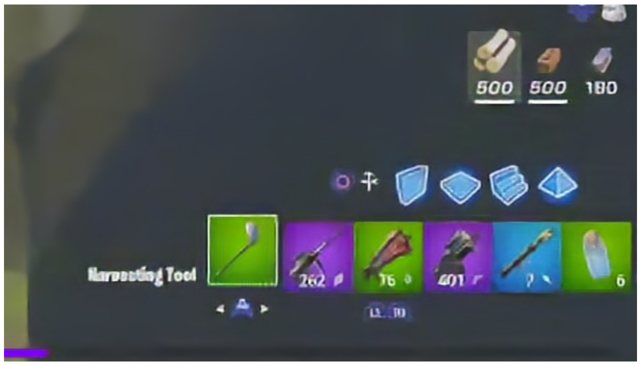
(e) Just Chatting (WebRTC) / PSNR: 27.1 dB, SSIM: 0.9



(f) Just Chatting (LiveNAS) / PSNR: 33.2 dB, SSIM: 0.96

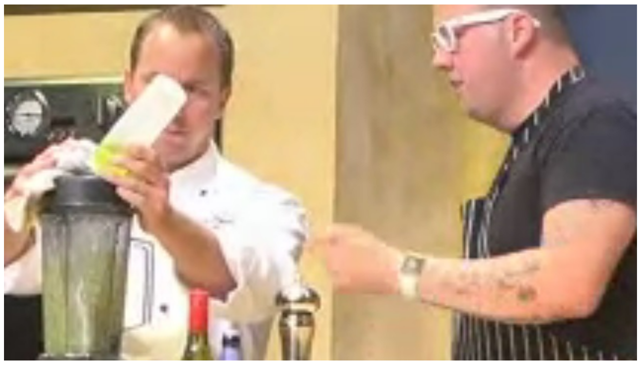


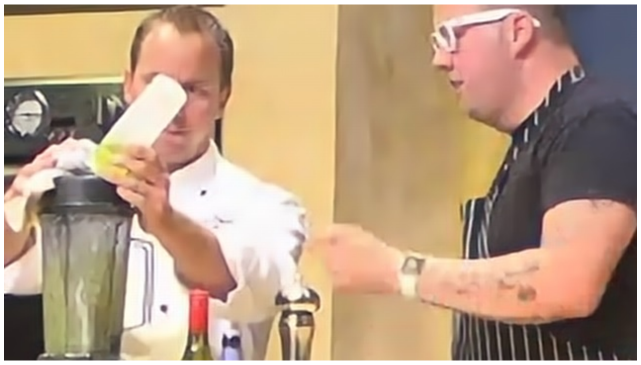
(g) Fortnite (WebRTC) / PSNR: 26.3 dB, SSIM: 0.87

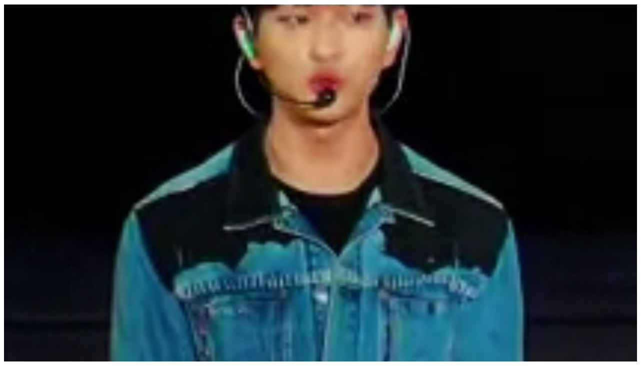


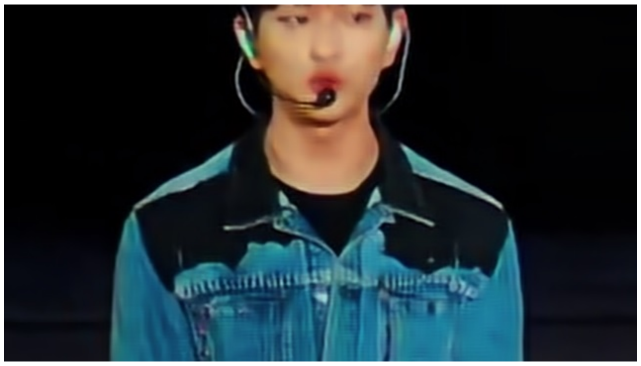
(h) Fortnite (LiveNAS) / PSNR: 28.7 dB, SSIM: 0.91

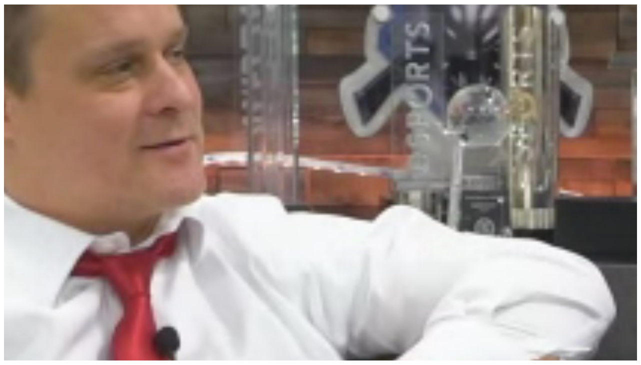
Figure 28: LiveNAS live stream results: 360p to 1080p (Twitch contents [21–23])

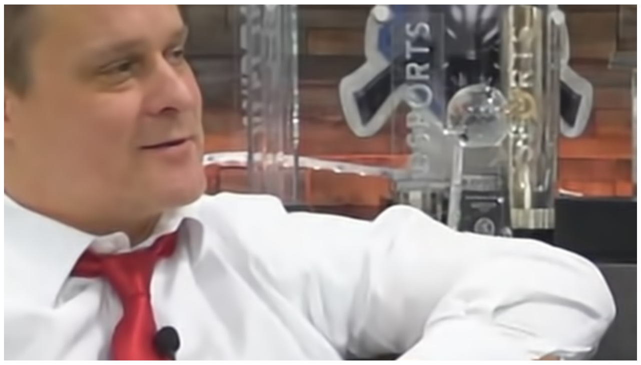

(a) Food/Cooking (WebRTC) / PSNR: 29.1 dB, SSIM: 0.88

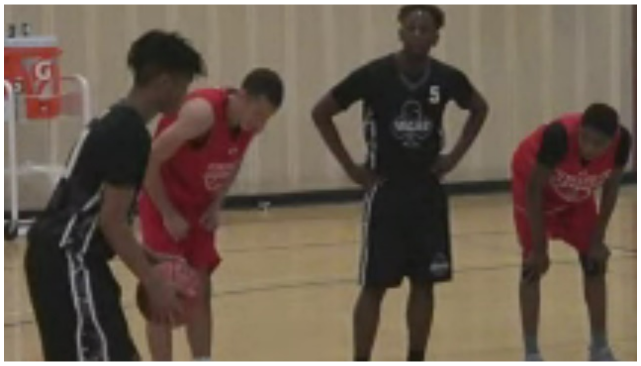

(b) Food/Cooking (LiveNAS) / PSNR: 32.2 dB, SSIM: 0.92


(c) Live Event (WebRTC) / PSNR: 31.8 dB, SSIM: 0.89


(d) Live Event (LiveNAS) / PSNR: 34.2 dB, SSIM: 0.93


(e) Podcast (WebRTC) / PSNR: 34.8 dB, SSIM: 0.95


(f) Podcast (LiveNAS) / PSNR: 37.9 dB, SSIM: 0.97


(g) Sports (WebRTC) / PSNR: 34.2 dB, SSIM: 0.93

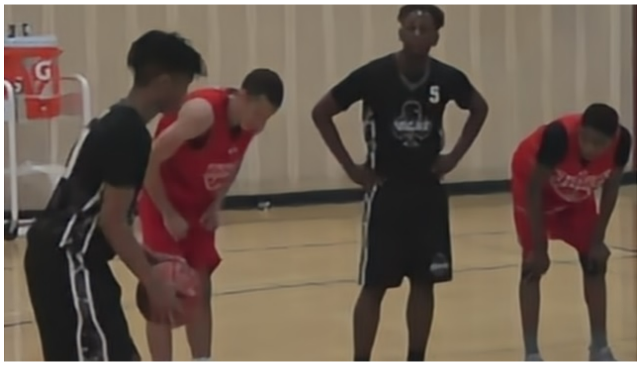

(h) Sports (LiveNAS) / PSNR: 36.6 dB, SSIM: 0.95

Figure 29: LiveNAS live stream results: 720p to 4K (Youtube 4K contents [32–35])