Deep Neural Network-based Enhancement for Image and Video Streaming Systems: A Survey and Future Directions

ROYSON LEE*, University of Cambridge, UK
STYLIANOS I. VENIERIS*, Samsung AI Center, Cambridge
NICHOLAS D. LANE, Samsung AI Center, Cambridge & University of Cambridge, UK

Internet-enabled smartphones and ultra-wide displays are transforming a variety of visual apps spanning from on-demand movies and 360° videos to video-conferencing and live streaming. However, robustly delivering visual content under fluctuating networking conditions on devices of diverse capabilities remains an open problem. In recent years, advances in the field of deep learning on tasks such as super-resolution and image enhancement have led to unprecedented performance in generating high-quality images from low-quality ones, a process we refer to as neural enhancement. In this paper, we survey state-of-the-art content delivery systems that employ neural enhancement as a key component in achieving both fast response time and high visual quality. We first present the components and architecture of existing content delivery systems, highlighting their challenges and motivating the use of neural enhancement models as a countermeasure. We then cover the deployment challenges of these models and analyze existing systems and their design decisions in efficiently overcoming these technical challenges. Additionally, we underline the key trends and common approaches across systems that target diverse use-cases. Finally, we present promising future directions based on the latest insights from deep learning research to further boost the quality of experience of content delivery systems.

CCS Concepts: • **General and reference** → *Surveys and overviews*; • **Computing methodologies** → *Computer vision tasks*; *Neural networks*; *Distributed computing methodologies.*

Additional Key Words and Phrases: Deep Learning, Content Delivery Networks, Distributed Systems

ACM Reference Format:
Royson Lee, Stylianos I. Venieris, and Nicholas D. Lane. 2021. Deep Neural Network-based Enhancement for Image and Video Streaming Systems: A Survey and Future Directions. *ACM Comput. Surv.* 1, 1, Article 1 (January 2021), 31 pages. https://doi.org/10.1145/3469094

1 INTRODUCTION

Internet content delivery has seen a tremendous growth over the past few years. Specifically, video traffic is estimated to account for 82% of global Internet traffic by 2022 – up from 75% in 2017 [23, 124]. This growth is attributed to not only the rapid increase of Internet-enabled devices, but also the support for higher-resolution content. For instance, an estimated 66% of TV sets will support Ultra-High-Definition (4K) videos by 2023 as compared to 33% in 2018 [22]. Most importantly, content traffic such as live streaming, video-conferencing, video surveillance, and both short- and long-form video-on-demand, are expected to rise very quickly. To meet these demands,

*Both authors contributed equally to this research.

Authors' addresses: Royson Lee, dsrl2@cam.ac.uk, University of Cambridge, UK; Stylianos I. Venieris, s.venieris@samsung.com, Samsung AI Center, Cambridge; Nicholas D. Lane, Samsung AI Center, Cambridge & University of Cambridge, UK, nic.lane@samsung.com.

0360-0300/2021/1-ART1
https://doi.org/10.1145/3469094

a new class of distributed systems has emerged. Such systems span from content delivery systems (CDS) [114, 166] that aim to maximize the user satisfaction and quality of experience (QoE), to video analytics frameworks that provide support for augmented and virtual reality apps by co-optimizing latency and accuracy [36, 152].

One of the primary challenges of distributed systems for content delivery is their reliance on networking conditions. Currently, the quality of the communication channel between client and server plays a key role in meeting the application-level performance needs due to the significant amount of transferred data and the tight latency targets. Nevertheless, in real-life mobile networks, the communication speed fluctuates and poor network conditions lead to excessive response times, dropped frames or video stalling, that rapidly degrade the user experience. This phenomenon is further amplified by the increasing number of users which compete for the same pool of network resources.

For many years, bitrate adaptation has been the dominant approach for counteracting the networking dynamicity and the main driver behind adaptive video streaming [11], with already deployed solutions by industry content providers such as iQiyi [140], Netflix [62] and YouTube [103]. Under this scheme, each video is broken down into segments that are encoded using multiple bitrates. Upon client's request, the segments are streamed over HTTP, with the client selecting the bitrate in a per-segment manner. Adaptive bitrate (ABR) algorithms typically consider either the connection speed [71, 100, 172] or the playback buffer state [63, 137], to dynamically select the bitrate of each video segment. Although ABR technology has achieved significant gains in sustaining high QoE across diverse settings, existing approaches still leave substantial portion of the available bandwidth underutilized [61, 103] either due to the ABR algorithm's suboptimality or due to the service provider's policy of minimizing bandwidth usage. With ABR leaving substantial room for further optimization, there is an emerging need for novel techniques to further boost the performance of content delivery systems and ensure high QoE across various visual content delivery applications, network conditions and device capabilities.

One of these techniques includes the use of newly emerging technologies such as telepresence [170]. For instance, the recently released Nvidia Maxine [120] can be used to drastically reduce the network load by allowing each user to only transmit at least one key frame, which is used to initialize the model at the receiver's end. During video-conferencing, facial landmarks, which use considerably less bandwidth than frames, are exchanged and used by their respective models to synthesize the video and mimic the other user's facial expressions, hence resulting in a fast and accurate video-conferencing experience even under extremely low-bandwidth settings. However, the usage of such techniques is limited and solely for video-conferencing as working in dynamic environments or with multiple users is still an unsolved challenge.

Another recent key method that enables tackling this challenge in general is *neural enhancement* through super-resolution (SR) and image enhancement models. These models are capable of processing a low-resolution or low-quality image and generating a high-quality output. With the unprecedented performance of convolutional neural networks (CNNs), content delivery systems have begun integrating neural enhancement models as a core component. The primary paradigm of using neural enhancement models in content delivery systems comprises the transmission of compact low-resolution or low-quality content, often along with the associated neural model, followed by its subsequent quality enhancement on the receiver side through a enhance-capable model [165]. In this manner, the transfer load is minimized, drastically reducing the network footprint and the corresponding bandwidth requirements, with the visual quality recovered on the client side.

Despite their benefits, integrating state-of-the-art neural enhancement models into visual content delivery systems poses significant challenges. First, these models, especially SR models, have

Table 1. Overview of Visual Content Delivery Systems

System	Task	Year
MobiSR [92]	On-demand image delivery	October 2019
Yeo *et al.* [165]	On-demand video delivery	November 2017
NAS [166]	On-demand video delivery	October 2018
NEMO [164]	On-demand video delivery	September 2020
PARSEC [28]	360° video delivery	August 2020
Supremo [167]	On-demand image delivery	September 2020
CloudSeg [152]	Video segmentation	July 2019
Dejavu [54]	Video-conferencing	February 2019
LiveNAS [79]	Live streaming	July 2020
SplitSR [108]	Digital zoom	March 2021

excessive computational demands that are measured up to hundreds of TFLOPs per frame in order to achieve an upscaling of up to 4K or 8K. With client platforms typically involving devices with strict resource and battery constraints [6, 85, 146], clients are still struggling to execute neural enhancement models on-device while meeting the target quality [92]. This fact is aggravated by the stringent latency and throughput requirements that are imposed in order to sustain high QoE. Finally, enabling the deployment of such systems requires overcoming unique technical challenges stemming from the diversity of use-cases, spanning from on-demand video streaming [166] to video-conferencing [54].

In this paper, we provide a timely and up-to-date overview of the growing area of visual content delivery systems that employ neural enhancement. Through this survey, we aim to equip researchers new to the field with a comprehensive grounding of next-generation content delivery system design, revealing how neural enhancement techniques can lead to greater performance than status quo methods. More specifically, we make the following novel contributions:

- We motivate neural enhancement for CDS by describing the architecture, major components, performance metrics and open challenges of conventional content delivery systems.
- We survey the state-of-the-art existing systems that utilize neural enhancement (Table 1) across diverse content delivery applications including on-demand video and image services, visual analytics, video-conferencing, live streaming and 360° videos. We detail their neural model design and system optimization strategies, assessing their strengths and limitations.
- Drawing from the latest progress by the computer vision community, we propose several promising future directions for future research and describe how they can be integrated to value-add existing and inspire future content delivery systems.

2 VISUAL CONTENT DELIVERY SYSTEMS

Content delivery systems are systems that aim to deliver image or video content to the users with minimal latency and high visual quality, while supporting a diverse set of client platforms. Across the years, several different aspects of CDS have been studied, from optimized networking [81, 101] and system design [10, 26, 33, 43, 114] to user behavior [98, 169], cost minimization [56, 94] and performance evaluation [2, 47, 97, 99]. Common underlying challenges for sustaining high performance and meeting the agreed-upon quality of service (QoS) are the client device heterogeneity – from powerful desktops to diverse mobile devices [157] – and the reliance on networking conditions. In this section, we present *1)* the common *performance metrics*, *2)* the typical

architecture of CDS and *3) the adaptive bitrate methods* that are conventionally used to counteract bandwidth fluctuations. Then, we introduce *4)* the recent technique of *neural enhancement* in terms of its principle of operation and the associated technical challenges.

2.1 Performance Evaluation Metrics

The performance evaluation of CDS is complex, comprising multiple metrics with often competing dynamics. The primary metrics of interest include essential attributes such as visual quality, frame rate, response time, rebuffering time, accuracy and quality of experience. Based on the characteristics of the target task, the design of CDS often prioritizes a subset of metrics and aims to reach a task-specific balance among them. Therefore, measuring and reporting all these criteria across diverse scenarios plays an important role in determining the strategic trade-offs made by CDS.

Visual Quality. Traditionally, the bitrate dictates the maximum resolution of the content without rebuffering; the higher the bitrate, the higher the supported resolution and thus the better the visual quality. With the introduction of neural enhancement models in CDS, the maximum resolution is determined by both the bitrate and the amount of computational resources available to execute these models. Additionally, the visual quality of the content is influenced not only by its maximum resolution, but also by the performance of the model used to enhance the content, *i.e.* the model's enhancement capability. The dependency on the bitrate for visual quality is therefore relaxed and the extent of this dependency is determined by the available compute, upscale factor and degree of compression. In other words, given sufficient computational resources, the impact on the content's visual quality is shifted towards the performance of the neural model. For instance, higher upscale factors and more computational resources enable the streaming of high-definition resolution at lower bitrates and thus rely on the performance of the model for high visual quality.

The visual performance of a neural enhancement model is often measured using either *i)* a distortion-based metric, such as Peak Signal-to-Noise Ratio (PSNR) [44] or Structural Similarity Index Measure (SSIM) [155], or *ii)* a perceptual-based metric, such as Naturalness Image Quality Evaluator (NIQE) [117] or Learned Perceptual Image Patch Similarity (LPIPS) [174]. Although optimizing using a perceptual-based metric will lead to more natural-looking images, existing CDS adopt *distortion-based metrics*, which aim to maximize image fidelity. These metrics are resolution-agnostic and the visual quality is determined solely by the absolute pixel-to-pixel error or their inter-dependencies between the original frame and its reconstructed version at the receiver. In order to accommodate these model performance metrics into traditional QoE metrics that utilized solely the bitrates, some existing neural enhancement-based CDS directly map these model performance metrics into bitrates as detailed below.

Accuracy. In video analytics use-cases [36, 152], such as security surveillance, traffic monitoring and face recognition, the most commonly used performance metric is accuracy, which captures the proportion of correct classifications over the input samples processed upon deployment. Accuracy is reported as percentage for classification tasks or in the range 0-1 for the Intersection-over-Union (IoU) metric of semantic segmentation tasks [110].

Response Time. Latency, or *response time*, is the primary performance indicator in latency-sensitive interactive applications, such as virtual reality [18, 30, 107], video analytics services [152] and video-conferencing [54]. Measured in seconds, response time is the end-to-end time between when an input event occurs (*e.g.* new frame in the user's real-time video or caller makes a gesture) and when the output is returned to the user's device (*e.g.* analytics result or enhanced view of caller). In such scenarios, excessive buffering adds a prohibitive latency overhead that degrades the QoE and is often not an option.

Frame Rate. Frame rate is among the primary metrics of interest in throughput-driven applications, such as video-on-demand [7, 90, 164, 166] and live video streaming [79]. Frame rate is measured in frames per second (fps), with the lower bound of real-time frame rate being between 24-30 fps.

Rebuffering. A critical metric for characterizing the user-perceived quality of video applications is *rebuffering* [118]. This phenomenon occurs when the playback buffer is drained due to the slow transmission of video segments, with the video player stalling and the playback pausing. Rebuffering captures the temporal properties of a video playback, independently of its content, and is typically analyzed with respect to: *1) initial buffering time*, the time between the request of a video by the user and the beginning of its playback; *2) mean rebuffering duration*, the average time of a rebuffering event; and *3) rebuffering frequency*, the occurrence frequency of rebuffering events.

Quality of Service (QoS). Network-level QoS captures the performance of a network connection and its capabilities to provide packet transfer with the agreed-upon requirements [39]. QoS is typically quantified using a set of networking-level metrics, including delay, jitter, packet loss rate and bandwidth. By monitoring these quantities, service providers can tune network parameters and apply traffic shaping to meet specific service-level agreements (SLAs) and sustain a high QoS. In this endeavor, the expected outcome is that high network-level QoS corresponds to high user satisfaction for the underlying application.

Quality of Experience (QoE). Despite the merits of network-level QoS, the relation between QoS measurements and user satisfaction is not trivial. With a more user-centric perspective, quality of experience (QoE) aims to capture the quality of an application or service from the point of view of the users. QoE is typically assessed either via subjective or objective methods. To directly measure the subjective perceived quality, a group of actual users are asked to provide a quality score over a sequence of videos in order to obtain a mean opinion score (MOS) [68]. Despite the accuracy of this approach, it relies on significant manpower and has to be conducted offline [5], preventing the automated monitoring of QoE that is required by real-time video applications.

With the subjective approach not deployable, objective methods have emerged. Such approaches consist of continuously collecting network-level QoS measurements together with application performance metrics [118], such as video bitrate, mean rebuffering time, rebuffering frequency and per-frame spatial quality [122], and using analytical formulas to aggregate them in order to automatically estimate QoE. Such an approach is adopted by the majority of the covered content delivery systems to track in real-time the achieved QoE and adapt accordingly.

The most commonly used objective QoE metric (Eq.(1)) [32, 61, 168] to date for neural enhancement-based CDS takes into account the bitrate of each video sequence, R, and rebuffering measures. In this respect, QoE is generally defined as a linear combination of: the bitrate utility, $q(R)$, that maps bitrate R to video quality; the rebuffering time caused by downloading the n-th segment, T_n; the initial buffering time, T_s; and the smoothness of the selected quality, $q(R_{n+1}) - q(R_n)$, which captures the variation in visual quality between subsequent segments (n and $n+1$) in terms of both their number and amplitude.

$$QoE = \frac{\alpha \sum_{n=1}^{N} q(R_n) - \beta \sum_{n=1}^{N} T_n - \gamma \sum_{n=1}^{N-1} |q(R_{n+1}) - q(R_n)|}{N} - \delta T_s \tag{1}$$

where N is the total number of video sequences, and the bitrate utility is modeled as linear [168], $q(R) = R$, logarithmic [137], $q(R) = log(R/R_{\min})$, to diminish improvement at higher bitrates, or fixed [114] to favor higher-definition bitrates. Hyperparameters α, β, γ and δ determine the

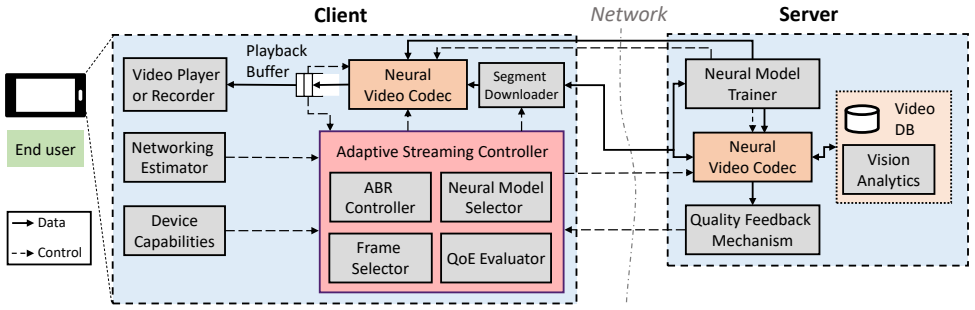

Fig. 1. Architecture of content delivery systems.

importance of the four components, penalizing the average segment quality, rebuffering delay, smoothness of quality variation, and initial buffering time, respectively.

In order to take into account the neurally enhanced quality of the videos, a few works [164, 166] used an inverse mapping from quality to bitrate through interpolation (*e.g.* piecewise linear interpolation). Specifically, $\widehat{R}_n = V_n f^{-1}(f(m(V_n)))$ where V is the video sequence, m is the neural enhancement model and f is the model's performance metric. The QoE is, therefore, determined using the estimated bitrate, $\widehat{R}_n$, given the performance of the model instead of the actual bitrate.

Other considerations. In addition to traditional performance measures, other metrics of interest for CDS are the server-provider-side (SP-side) and user-side costs. On the SP side, costs can be quantified based on bandwidth usage, per user or aggregate, or processing time on the SP's servers. Analytically, these can be captured either as additional hard constraints with specific bandwidth budget and computation time limit, or as additional objectives to be optimized, where the bandwidth and server resource usage are minimized. On the user side, cost can be quantified based on the additional energy overhead or the computational and memory resource usage imposed by running neural enhancement on the client device. Similarly to SP-side costs, these can be modeled either as hard constraints or as metrics in an optimization problem which maximizes energy efficiency and minimizes processing time.

2.2 Architecture

CDS typically adopt *distributed* architectures consisting of one or multiple clients and servers. Figure 1 depicts the typical architecture of such a system. Focusing on video-on-demand (VOD) without loss of generality, the operation starts on the client side with the user selecting an online video to watch and the video player app (*Video Player)* sending a request to the video server. Next, the server fetches the video from its database (*Video DB*) and launches a transmission-preparation stage. In modern adaptive video streaming services [11, 138], instead of sending one large monolithic video file, the video content is divided into a series of seconds-long segments (*e.g.* 4 seconds per segment). This approach enables the client's *Adaptive Streaming Controller* to request different bitrate encoding per segment through the *ABR Controller* in order to dynamically adapt to changing environmental conditions [29]. To this end, each video segment is compressed by the server-side *Video Codec* using a predefined encoding scheme and the client-specified bitrate. Finally, the client's *Segment Downloader* pulls the encoded segments, where they are decoded by the client-side *Video Codec* and passed to the *Playback Buffer* to be concatenated and eventually played. As discussed in Section 4, depending on the target application and adopted design decisions, CDS may optionally integrate additional components such as a *Frame Selector*, *Model Selector* and/or *QoE Evaluator* on

the client side, and a *Quality Feedback Mechanism* and/or a *Vision Analytics* backend on the server side. All processing stages are typically pipelined to enable streaming operations and a higher attainable throughput.

For such systems to meet the performance targets, the communication channel between client and server has to sustain high bandwidth throughout the streaming process. This constitutes a strong assumption that breaks for mobile clients where the connectivity conditions vary continuously. Hence, additional techniques, such as adaptive bitrate and neural enhancement, have been introduced that enable the dynamic adaptation to the instability of the channel.

2.3 Bitrate Adaptation

To remedy the dependence of content delivery systems on network conditions, adaptive bitrate (ABR) algorithms have emerged [11, 61, 63, 71, 100, 114, 137, 168, 172]. The majority of the existing bitrate adaptation methods reside on the client side, complying with the Dynamic Adaptive Streaming over HTTP (DASH) standard [138] and enabling large-scale deployment by applying only client-side modifications [11]. Under this scheme, the client device first monitors its instantaneous bandwidth (*Networking Estimator* in Figure 1) to assess the current network state [71, 100], or the occupancy of its *Playback Buffer* [63, 137], or both [4, 136, 168]. Next, the *ABR Controller* tunes accordingly the per-segment bitrate that it requests from the server and configures the client's codec to decode at the selected rate. On the remote side, the server encodes each video segment with the specified bitrate and responds to the *Segment Downloader*'s pull request. Overall, ABR techniques function as a way to control the network footprint *at run time* and thus help to minimize rebuffering. Although ABR has been significantly improved through deep learning-based algorithms [41, 60, 61, 114, 172] to better perform under unexpected settings, it often fails in scarce network conditions as it relies solely on network resources.

2.4 Neural Enhancement

Neural enhancement aims to restore and recover the quality/resolution of visual input. As this problem is inherently ill-posed (*i.e.* many high-resolution solutions can be downsampled to the same low-resolution image), most works enforce a strong prior to mitigate its ill-posed nature. To this end, most state-of-the-art approaches utilize CNNs to capture the prior as it results in superior visual performance. These methods train a model to either map a low- to a high-quality image using exemplar pairs [34, 78, 104] or exploit the internal recurring statistics of the image to enhance/upscale it [130].

The primary paradigm of using neural enhancement models in content delivery systems comprises the transmission of compact low-resolution/low-quality content followed by its subsequent enhancement on the receiver side through the enhance-capable models [165, 166]. In this manner, the transfer load is tunable by means of the upscaling factor (for SR) and the degree of compression, controlling the system's network footprint and the associated bandwidth requirements. Therefore, neural enhancement opens up a new dimension in the design space by introducing a trade-off between computational and network resources, effectively overcoming existing systems' sole reliance on network resources. To this end, existing systems may choose to independently optimize the utilization of these neural enhancement models [79, 92, 167] or integrate them within existing ABR algorithms [28, 164, 166].

3 NEURAL ENHANCEMENT AND ITS DEPLOYMENT CHALLENGES

So far, a wide range of neural enhancement models and techniques have been integrated in content delivery systems. Although the computer vision field of neural enhancement includes both SR and image enhancement neural networks, CDS generally adopt SR models as reducing the spatial

Table 2. Comparison of neural enhancement models used in CDS (Section 4) for commonly-used scaling factors on standard benchmark datasets

Scale	Model	Params (K)	Mult-Adds (G)	Set5 [12] PSNR/SSIM	Set14 [162] PSNR/SSIM	B100 [115] PSNR/SSIM	Urban100 [59] PSNR/SSIM
2×	VDSR [78]	665	612.6	37.53/0.9587	33.03/0.9124	31.90/0.8960	30.76/0.9140
	CARN [3]	1,592	222.8	37.76/0.9590	33.52/0.9166	32.09/0.8978	31.92/0.9256
	IDN [64]	552	202.8	37.83/0.9600	33.30/0.9148	32.08/0.8985	31.27/0.9196
	EDSR [104]	40,711	9384.7	38.11/0.9601	33.92/0.9195	32.32/0.9013	32.93/0.9351
	MDSR [104]	7,953	1501.5	38.11/0.9602	33.85/0.9198	32.29/0.9007	32.84/0.9347
	RCAN [175]	15,444	3526.8	38.27/0.9614	34.12/0.9216	32.41/0.9027	33.34/0.9384
4×	VDSR [78]	665	612.6	31.35/0.8838	28.01/0.7674	27.29/0.7251	25.18/0.7524
	CARN [3]	1,592	90.9	32.13/0.8937	28.60/0.7806	27.58/0.7349	26.07/0.7837
	IDN [64]	552	89.0	31.82/0.890	28.25/0.7730	27.41/0.7297	25.41/0.7632
	EDSR [104]	43,070	2894.5	32.46/0.8968	28.80/0.7876	27.71/0.7420	26.64/0.8033
	MDSR [104]	7,953	410.6	32.50/0.8973	28.72/0.7857	27.72/0.7418	26.67/0.8041
	RCAN [175]	15,592	916.9	32.63/0.9002	28.87/0.7889	27.77/0.7436	26.82/0.8087

dimensions is more effective at reducing network usage. An exception to this approach is a subset of systems [54] that repurpose SR models for image enhancement by feeding a low-quality input – which is of the *same* resolution as its output – to the SR model. Moreover, existing systems adopt CNNs for *supervised single-image* SR, due to *1)* its efficiency as opposed to multi-frame super-resolution [141, 143, 148] and *2) the availability of the degradation operation as compared to blind* unsupervised or self-supervised methods [9, 46, 130]. In this section, we provide an overview of SR models (Table 2) used in existing content delivery systems (Section 4) and discuss their unique deployment challenges.

3.1 Neural Enhancement Models used in Content Delivery Systems

After the first proposed super-resolution CNN [34] that only used 3 layers, VDSR [78] was the first SR model that used a significantly deep (20-layer) convolutional network to obtain substantial performance gains, and is one of the first SR models adopted for on-demand video streaming [165]. The model takes in an interpolated low-resolution input and passes it through twenty convolutional layers, before merging through addition the result with a final residual connection from the input. Each convolutional layer has 64 feature maps followed by a ReLU [119] activation function. Although VDSR is relatively lightweight in terms of workload and memory requirements, Yeo *et al.* [165] showed that real-time upscaling to 720p is only possible when the number of layers and feature maps are dropped by half and targeting a powerful desktop GPU, highlighting the computational challenges that will be discussed in Section 3.2.

EDSR [104], winner of the NTIRE 2017 Super-Resolution Challenge [144], and its multi-scale variant, MDSR, are both heavily adopted in various content delivery systems such as `NAS`, `LiveNAS`, `PARSEC`, `Dejavu`, `LiveNAS` and `NEMO`. MDSR, in particular, extends EDSR by supporting multiple scales and consists of three stages: *1)* a front-end feature extraction stage with separate feature extractors per scale, *2)* a number of shared intermediate layers, and *3)* independent upsampling layers for each scale. Both models adopt residual blocks from ResNet [49], without the batch normalization [67] layers, as their building block. Specifically, the authors showed empirically that the use of batch normalization [67] led to performance degradation in SR, a phenomenon

that was further studied in [151], leading to the abandonment of batch normalization layers in subsequent SR model designs. Unlike VDSR, EDSR and MDSR employ a variety of compute-efficient techniques to reduce their computational complexity and memory footprint. For instance, they take in a low-resolution input and upscale it only at the end [35] using the more efficient pixel shuffle [129] module as opposed to the more costly deconvolution.

Nevertheless, EDSR and MDSR are still considerably heavier than VDSR, with 43 million and 8 million parameters for EDSR and MDSR respectively as compared to 0.6 million parameters for VDSR. As a result, CDS often introduce further optimization strategies in order to alleviate the excessive workload, allow scalability based on the client's capabilities and achieve the desired performance-latency trade-off. Such techniques include the generation of multiple single-scale variants of EDSR/MDSR with fewer layers and feature maps, in order to support heterogeneous clients with varying computational capabilities [164]. Apart from multiple CNN configurations, early-exit strategies [57, 87, 142, 161] are often utilized to allow the client to adapt to their available resources and execute a partially downloaded model [166]. Other methods to speed up execution include training on the luminance channel as opposed to the RGB channels, deploying patch selection to selectively upscale patches through the models [54], and parallelizing execution across multiple GPUs for higher-end clients [79].

Zhang *et al.* [175] proposed RCAN, a model that achieves better performance at a lower cost as compared to EDSR and is adopted in `MobiSR` [92] and `SplitSR` [108]. RCAN achieves this through channel attention blocks and a residual-in-residual structure as its building block. Specifically, instead of designing residual blocks like in EDSR [104], RCAN consists of residual blocks within residual blocks, named residual groups. Each residual block in each group is followed by channel attention blocks, allowing the model to focus on the more informative components of the image. Similar to EDSR and MDSR, RCAN is computationally heavy and requires smaller variants for it to be deployable. For instance, apart from manually tuning the number of layers and feature maps, `MobiSR` employed a wide range of tensor decomposition and compression techniques to approximate convolutions in RCAN. Similarly, `SplitSR` substitutes the standard convolutions with channel splitting operations and tunes the depth of the RCAN to optimize for either quality or latency.

Towards efficiency, Ahn *et al.* [3] proposed CARN which utilizes concatenated skip connections [58] in a block named cascading block, which consists of residual blocks and convolutions. The authors showed that adding concatenated skip connections at both a block-wise and layer-wise level allowed CARN to be more accurate and efficient compared to VDSR [78]. To further speed up latency, IDN [64] proposed channel splitting [112], thereby only processing a subset of its feature maps for some convolutions in each block. Moreover, IDN utilizes group convolutions [84] which can be executed in parallel. Both CARN and IDN are computationally efficient SR models that are able to run in real-time on a high-end desktop GPU and have thus been adopted by cloud-based solutions such as `CloudSeg` [152] and `Supremo` [167], respectively.

Apart from the existing SR models and techniques that have already been adopted in content delivery systems, we discuss other works from the SR literature that can be utilized to further support and optimize the real-world deployment of CDS in Section 5. We refer the reader to existing surveys [105, 156, 163] for a more comprehensive discussion on deep learning-based SR from a computer vision perspective.

3.2 Challenges of Neural Enhancement

Despite their advantages, deep neural enhancement models pose a set of critical challenges. First, neural enhancement CNNs are extremely expensive in terms of both computational and memory burden. Key factors behind the resource-intensive nature of these models are the large spatial

dimensions of feature maps throughout the model's layers together with the excessive number of memory accesses in memory-bound upscaling operations, such as Pixel Shuffle [129]. In this respect, super-resolution CNNs are orders of magnitude larger than image discriminative models, with TFLOP-scale workloads compared to the tens of GFLOPs for classification models [6]. Similarly, the workload of *efficiency-optimized* SR models [35, 91] are measured in GFLOPs, whereas their image classification counterparts [52] are measured in MFLOPs [6]. A common approach to significantly reduce the peak run-time memory footprint is to split the image into patches that are processed sequentially, with the partial results stitched together to produce the final upscaled image. Nevertheless, the computational cost is still a big challenge for real-time applications, especially when targeting mobile platforms [28, 65, 92] where the latency per frame spans from 100s of milliseconds up to seconds depending on the target image resolution.

Furthermore, models trained on standard datasets that aim to generalize across all videos/images result in outputs of varying performance upon deployment and often fail catastrophically on unexpected inputs [92, 165]. On the other hand, tailoring a CNN towards a specific video/image helps to mitigate this drop in performance at the cost of additional training per video/image [53, 74, 128]. This is typically achieved by first pretraining a model on standard datasets and then producing multiple specialized models by fine-tuning different model variants through additional training iterations either per video category [165], per video [164, 166], per video segment [28], per video-conference caller [54] or on a live video stream [79]. In this respect, a *generalization-specialization trade-off* is exposed which system designers need to decide how to control based on the target use-case.

4 THE LANDSCAPE OF CNN-DRIVEN VISUAL CONTENT DELIVERY

Despite their deployment barriers, several recent frameworks have incorporated neural enhancement methods into their pipelines and introduced novel techniques for overcoming their challenges. In this context, we survey the state-of-the-art visual content delivery systems that leverage neural enhancement models, taxonomizing them based on the type of content (on-demand or live, video or image), and provide an analysis of how they counteract ⓐ the excessive computational requirements and ⓑ the performance variability across different content.

4.1 On-demand Content Delivery Systems

4.1.1 On-demand Image Delivery. Image delivery systems consume significantly fewer network resources than their video counterparts, therefore requiring considerably less bandwidth. Nevertheless, apart from reducing the load in a shared communication channel, these systems can help users with a limited mobile data plan to save data. For instance, these systems can be deployed in *data-saving* mobile app alternatives, such as Facebook Lite and Messenger Lite. In this scenario, instead of disabling the downloading of content when the user is not on Wi-Fi, visual content can be downloaded in low resolution to save data, and then be locally enhanced to high quality. In this manner, users can continue to scroll through their feed or send picture messages at ease. This applies to any image-centric application including news apps, dating apps, gallery apps, and many others.

Additionally, as chipsets on commodity devices are gradually getting more powerful [6, 65], this enables many of these applications to run fully on-device, avoiding the latency and privacy issues of cloud or edge offloading. In this direction, Lee *et al.* [92] proposed `MobiSR` (Figure 2a), a system that capitalizes over the heterogeneous compute engines of modern smartphones, *e.g.* CPU, GPU and NPU, through a model selection mechanism to deliver rapid image super-resolution.

As a first step, MobiSR derives two model variants by applying a wide range of compression techniques on a user-provided reference model; the authors use a smaller variant of RCAN [175]. These compression techniques mainly consist of low-rank tensor decompositions, such as depthwise separable convolutions [131], and other efficient model designs, such as channel splitting [112], that have been successful in high-level vision tasks. The resulting Pareto-optimal models are then assigned to the different available compute engines and a hardware-aware scheduler (*Difficulty Evaluation Unit* (DEU)) is deployed during inference to rapidly process the patches of each input image. Specifically, the DEU processes hard-to-upscale patches using a more compact model (m_1) while feeding the easier patches to a larger model (m_2) to obtain higher quality, leveraging on an insight that both large and small models perform similarly on hard-to-upscale patches, with difficulty quantified using the total-variation metric [126]. Hence, the image quality is maximized while meeting the applications' latency constraints (challenge ⓐ). With respect to challenge ⓑ, MobiSR is optimized to achieve higher overall performance through a generic model and does not employ model specialization.

MobiSR, however, requires scheduling using multiple processors to leverage its benefits, and may not be suitable in certain deployment scenarios. In this regard, Liu *et al.* [108] proposed SplitSR (Figure 2f), a system that focuses on optimizing either quality or latency on a single CPU and, thus, making it accessible to a wider range of smartphones and computational settings. Similar to MobiSR, SplitSR mitigates challenge ⓐ through an efficient variant of RCAN [175]. The proposed model, however, emphasizes the use of channel splitting operations, instead of depthwise separable convolutions, and tuning the model's depth in order to reduce the DNN workload. The final DNN model is optimized for either quality or latency. Furthermore, SplitSR extends the TVM deep learning compiler [20] to support operations required in their model and generate highly optimized implementations for mobile CPUs. Finally, although the models in SplitSR achieve higher image fidelity when compared to the models proposed in MobiSR, both frameworks do not specifically handle the variability in upscaling quality across images (challenge ⓑ). Despite being the state-of-the-art for image-centric on-device use-cases, the processing rates achieved by both MobiSR and SplitSR are still below 24-30 fps and hence not yet suitable for real-time video applications.

4.1.2 On-demand Video Streaming. Video-on-demand (VOD) services allow users to watch content at their suitable time from any Internet-enabled device. The user selects a video which is then fetched and streamed by a video server to the user device. With the majority of VOD services being interactive, on-demand video streaming systems have to yield low response time and minimal rebuffering while not compromising visual quality in order to maximize the QoE. In achieving these targets, the bottleneck lies in the link between the video server and the client with the bandwidth of the connection directly affecting the end performance.

Yeo *et al.* [165] presented one of the first works that employed neural enhancement to overcome this limitation and offered a way to utilize the clients' computational power. Specifically, the authors first proposed grouping videos into clusters according to their category (basketball, athletics, *etc.*) through a clustering module. This classification can be performed by either utilizing image classification models or using the video's metadata and predefined categories provided by content platform providers, such as YouTube. A specialized SR model, VDSR [78], is then trained for each cluster, reducing the performance variation (challenge ⓑ) as compared to using a single generic model. Besides RGB frames, they also proposed the use of more compact representations such as their edges or the luminance channel, on top of tuning the spatial size of the frames, to further reduce both the bandwidth and computational resource usage. As these alternative representations contain less information than their corresponding RGB frames, the authors utilized Generative Adversarial Network-based [45] (GAN-based) training to learn the distribution of natural textures

in order to synthesize natural-looking frames. Although the authors showed that these compact representations did not work well in practice with the H.264 codec, they were adopted in later works, such as Dejavu [54], to tackle challenge (a) in different use-cases.

To handle the excessive computational needs of neural enhancement models (challenge (a)), the authors constrained their system to work with up to 720p videos and targeted homogeneous client platforms hosting powerful desktop-grade GPUs. This limitation was subsequently addressed to accommodate clients with heterogeneous computational capabilities in their extended proposed framework - NAS [166].

In NAS [166] (Figure 2b), the authors addressed the problem of heterogeneous clients through the use of early-exit SR models of varying sizes and computational workload, allowing each client to select the appropriate model segments (challenge (a)) based on both their computational capabilities and run-time device load. To this end, they extended previous reinforcement-learning-based (RL-based) ABR algorithms [114] to decide not only the bitrate (*ABR*), but also the *fraction* of the SR model (*Model Selector*) to be transmitted for each video segment. Specifically, the RL-based network is optimized on a modified QoE metric to take into account the enhancement quality of the content, as detailed in Section 2.1, in order to make the aforementioned decisions. These model segments are, therefore, progressively sent, incrementally updating the model at the client until the full model is delivered. Further mitigating challenge (a), the authors based their early-exit SR model on a smaller variant of MDSR [104], which is quantized at 16-bit half-precision floating-point format and executed on a desktop-grade GPU on the client side in order to hit the real-time requirement. Finally, instead of categorizing videos into coarse clusters as in [165], NAS tackles challenge (b) by first pre-training a generic SR model and then fine-tuning a specialized model *for each video*. Overall, as shown in Figure 2b, the client selects the bitrate b for the i-th video segment and the fraction j of the SR model for the particular video and receives the i-th low-resolution segment s_i^v and the associated fraction of the specialized model m_j^v for video v that are used by the *Super-resolution Processor* (SRP) – which is part of the *Neural Video Codec* in Figure 1 – to locally produce a high-resolution output.

Although NAS is able to support heterogeneous clients, these clients are assumed to have *at least* the computational power of a desktop-class GPU. To support lower-end commodity devices, existing on-device SR systems like MobiSR are inadequate at upscaling in real-time. To this end, Yeo *et al.* [164] proposed NEMO, a framework that leverages frame dependencies using information from the video codec VP9, trading quality degradation for on-device real-time video streaming and reduced energy consumption. Instead of running SR per frame, NEMO applies SR to selected frames called anchor frames, mitigating challenge (a), and uses the cached super-resolved anchor frames and the frame dependencies defined in the codec to upscale the remaining non-anchor frames. Specifically, the client runs a SR-integrated codec that refers to a list of anchor frames sent by the server in order to decide whether to run the model or reuse previously cached super-resolved frames for anchor and non-anchor frames respectively. For a non-anchor frame, following the VP9 codec, the SR-integrated codec first uses a reference index to select a previously upscaled frame from the *Frame Cache*. After this step, the provided motion vector is upscaled via bilinear interpolation and motion-compensated before it is used to warp the selected frame. Lastly, the residual block of the compressed frame is decoded and upscaled via bilinear interpolation before adding to the resulting warped frame.

Although the VP9 codec provides key frames, which have a high-degree of reference from other frames, NEMO deploys its own anchor frame selection in order to adhere to a given performance threshold. The algorithm first computes the video quality from all possible anchor point sets in a video and then, following a greedy approach, iteratively selects an anchor point that results in the maximum video quality gain until it reaches the quality requirement. Running the quality

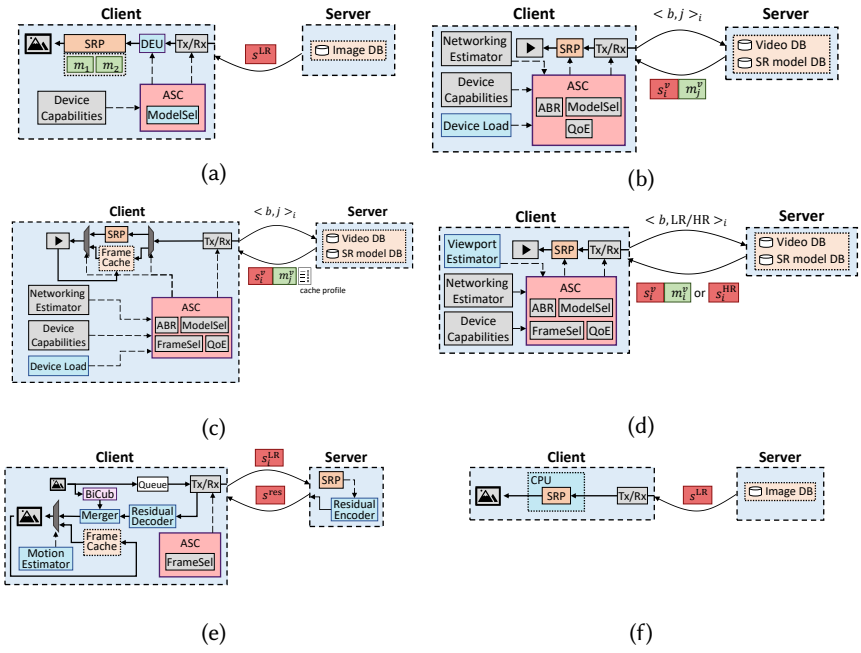

Fig. 2. Overview of on-demand delivery systems: 2a) MobiSR [92], 2b) NAS [166], 2c) NEMO [164], 2d) PARSEC [28], 2e) Supremo [167], 2f) SplitSR [108]. Cyan blocks indicate novel components introduced by each system.

measurements over all anchor point sets, however, is infeasible. Therefore, in order to speed-up the algorithm, the authors selected the most impactful anchor point in each frame as a proxy, effectively reducing the number of anchor point sets to the number of frames. The estimated video quality can thus be quickly calculated as an average of each estimated frame quality, which is computed from a single anchor point, rendering the algorithm computationally feasible.

Following NAS, NEMO alleviates challenges ⓐ and ⓑ and supports heterogeneous clients by preparing a list of models of various sizes and computational demands, along with a list of their anchor frames generated offline by its anchor frame selection algorithm, for each video and their bitrate version. To guarantee real-time processing, each compute unit in the client processes an anchor and a non-anchor frame once upfront in order to estimate the overall processing time required for other videos. Given this estimation, the client then selects the available configuration (*Model Selector*) that maximizes quality and meets the real-time constraint.

4.1.3 360° Video Streaming. Compared to regular videos, streaming 360° videos has significantly elevated bandwidth requirements. To alleviate this, existing systems employ viewport prediction techniques [37] which estimate which part of the video the user will look towards and only download this spatial content. However, accurate viewpoint prediction is still difficult to achieve, exacerbating the problem as missing patches of the current viewpoint need to be fetched at the time of viewing. Therefore, existing solutions [123] send additional patches in the neighborhood of the predicted viewport patches. Although neural enhancement models can be utilized to mitigate this challenge, the larger spatial dimensions of 360° content further aggravates challenge ⓐ and calls for dedicated deployment solutions.

In this context, Dasari *et al.* [28] proposed a 360° video streaming framework named PARSEC (Figure 2d). Unlike previous works, the authors extended the *Adaptive Streaming Controller* (Figure 1) logic to decide on the low-resolution (LR) patches to be upscaled (*Frame Selector*) and the bitrate of high-resolution (HR) patches to be downloaded (*ABR*) based on 1) the networking conditions, 2) the client's computational resources, 3) the viewport prediction (*Viewport Estimator*) and 4) the quality (PSNR) of both the HR and the upsampled patches. Since the proposed ABR algorithm is designed to maximize QoE and can thus selectively decide which patches are upscaled or downloaded, challenge (b) is mitigated. Moreover, due to the substantial increase in spatial content of 360° over conventional videos, each manually tuned efficient SR model, similar to that used by NAS, is fine-tuned for each video segment (*Model Selector*) as opposed to each video.

To overcome challenge (a), PARSEC uses an extreme upscaling factor of ×64 on top of the H.265 compression that allows all ultra-LR patches to be transmitted, alleviating the limitations of viewpoint prediction. Additionally, each SR model is varied depending on the targeted quality and the length of the video segment, resulting in shorter inference times for shorter video segments.

4.1.4 Cloud-assisted Content Delivery. In most content delivery scenarios, the client that is requesting the content adaptively allocates the compute needed for enhancement between the server and the client through its ABR algorithm as seen in PARSEC, NAS, and many others. However, for cases in which the ABR algorithm fails, the user's QoE is heavily impacted as the client struggles to handle its workload. In order to accommodate such cases, computation offloading can be deployed, resulting in a trade-off between performance and network utilization. Specifically, offloading content to be enhanced offers faster processing, but imposes an extra cost in the bandwidth, along with additional privacy issues, due to the uploading and downloading of low-quality and high-quality content, respectively. Furthermore, this scheme is mainly applicable in use-cases where the low-resolution image already resides in the client device, *e.g.* pre-downloaded images or videos or zooming in camera-captured images. In such cases, there is no transmission cost to obtain the LR image and hence the additional bandwidth overhead due to offloading might be justified.

To this end, Yi *et al.* [167] proposed Supremo (Figure 2e), a framework that enables real-time mobile SR by selectively offloading computation to the cloud. In order to mitigate challenge (a), Supremo only runs bicubic interpolation on-device and uses a lightweight variant of the IDN [64] SR model to be run on the resource-rich server, while performing patch selection to only transmit key patches. Specifically, Supremo's patch selection mechanism starts by extracting the edges, using the Canny edge detector, from each image, dividing the image into blocks, and sorting the blocks according to edge intensity; the highest edge intensity has the highest priority for offloading, as edges are degraded the most when upscaled using a neural-based model compared to using plain bicubic interpolation. Next, depending on the networking conditions, latency requirements and their ranking, these patches are sent in parallel to the cloud to be upsampled through the SR model. To further reduce the network footprint required to download the super-resolved patches, Supremo exploits the sparsity of the difference between the super-resolved patches and bicubic-upsampled patches. As these differences are often very sparse, encoding them through the *Residual Encoder*, which resembles JPEG encoding, results in a heavily compressed signal, thus minimizing bandwidth. For video-centric use-cases, Supremo also deploys a caching mechanism to exploit temporal redundancy by reusing super-resolved frames on matching blocks (*Frame Cache*), matched using motion estimation [177]. Similar to MobiSR, Supremo handles challenge (b) by employing a generic model that aims to maximize the average upscaling performance across all processed images.

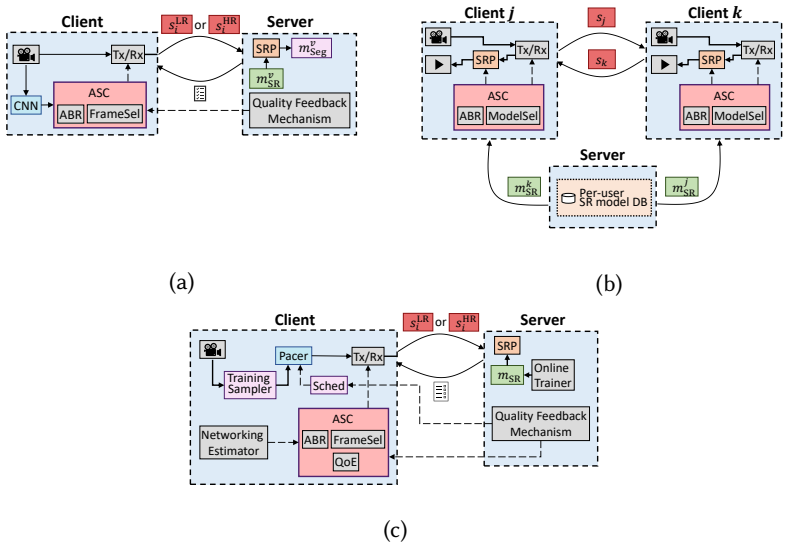

Fig. 3. Overview of live streaming systems: 3a) `CloudSeg` [152], 3b) `Dejavu` [54], 3c) `LiveNAS` [79]. Cyan blocks indicate novel components introduced by each system.

4.2 Live Content Streaming Systems

4.2.1 Streaming for Video Analytics. Pipelines for video analytics [25, 36, 53, 171] perform real-time intelligent tasks over user inputs in order to enable the development of novel applications such as telepresence [170] and augmented reality apps [106]. Such tasks span from scene labeling and object detection to face recognition. To meet the real-time performance requirements across diverse hardware platforms, such systems often rely on cloud-centric solutions. In this setup, the client device transmits the input frames to a powerful server for analysis and collects back only the result.

Naturally, these video analytics frameworks can benefit from the transmission of lower-resolution or lower-quality images under bandwidth-constrained settings. However, the use of low-quality images is known to reduce accuracy [15, 21] as some of these degraded images do not contain the sufficient information for the task at hand. For instance, high-frequency details, such as the texture of an object or the facial identity of distant persons, during downsampling [14]. Therefore, these frameworks can use additional server-side computation to deploy neural enhancement models, recover the quality of the images and thus minimize the accuracy loss of the target task. To achieve this goal, Wang *et al.* [152] proposed `CloudSeg` (Figure 3a) that jointly trains an SR model (CARN [3]) together with its target analytics task, *i.e.* semantic segmentation (ICNet [176]). Specifically, they trained CARN using gradients computed from both the *content loss* between the HR and the super-resolved image and the *accuracy difference* between using both images in ICNet.

During inference, the SRP feeds both the LR image and super-resolved image to the pyramid (also known as multi-scale) segmentation model, avoiding any redundant downsampling process. Towards efficiency (challenge ⓐ), `CloudSeg` employs frame selection at the client side by deploying a small convolutional neural network (*CNN*) that estimates pixel deviation in segmentation maps from low-level features of consecutive frames [96] in order to skip redundant stale frames (*Frame Selector)*. Finally, `CloudSeg` overcomes challenge ⓑ by training both the SR and segmentation model on the same dataset and falling back to streaming in high resolution whenever the analytics accuracy falls below a threshold, *i.e.* the *Quality Feedback Mechanism* guides the *ABR* controller.

4.2.2 Video-Conferencing. To sustain the interactive communication between callers, video-conferencing requires low response time. In an effort to achieve that, existing services [40] often adopt conservative strategies that relax the bandwidth requirements, but also compromise visual quality.

In this context, Hu *et al.* [54] observed that in contrast to generic live streaming, video-conferencing has the property of visual similarity between *recurring* sessions and designed `Dejavu` (Figure 3b) to exploit these unique computational and specialization opportunities; video calls that take place periodically and in the same or similar rooms have inherent redundancy. The developed system starts with the *offline* training of image enhancement models that are specialized *per caller* (challenge ⓑ). In this process, the model learns to improve the video quality by increasing its *encoding rate* without changing the spatial resolution.

Upon deployment, when a video-conference session is established, the associated caller-specific enhancement model is transferred from a client to the other and vice versa ($m_{\text{SR}}^{\{k,j\}}$ in Figure 3b). During the call, the frames from the caller are re-encoded to a lower quality at the same resolution prior to transmission, reducing the bandwidth usage, and the quality is enhanced on the receiver side through the caller-specific enhancement model.

To address challenge ⓐ, `Dejavu` uses several techniques. Apart from designing an efficient variant of EDSR that is trained only on the luminance channel (Section 3), a powerful GPU is assumed to be available on each calling side. Additionally, `Dejavu` introduces a patch-scoring CNN that predicts the expected quality gain for each image patch. In this manner, only the top-k patches (*Frame Selector*), which consist of mainly edges and complex textures, that are expected to yield the highest quality improvement are processed by the quality-enhancing neural network to lower the run-time resource usage.

4.2.3 Live Video Streaming. In contrast to VOD services that focus on stored content, live streaming targets videos produced in real-time. In this case, an additional bottleneck is introduced on the upstream client-to-server channel: a degraded quality from the streaming user would propagate to the end users that watch the video, leading to additional challenges in sustaining high QoE. Moreover, while stored content in VOD or the recurrence of video-conferencing allows for offline specialization of the enhancement models, the real-time dynamic nature of live streaming requires methods to tailor the models online to the incoming video. Online learning, however, requires extra bandwidth on top of the stream as the parts of the high-quality stream often needs to be transmitted for model fine-tuning.

To tackle these challenges, Kim *et al.* [79] proposed the `LiveNAS` system (Figure 3c) that focuses on optimizing the upstream transmission from streamer to server. In this system, a generic SR model, pre-trained on previously played similar streams, resides on the server side. Upon deployment, the streamer uploads a series of low-resolution frames which are then enhanced at the server side by the *Super-resolution Processor* (*SRP* in Figure 3c).

To counteract the performance variability across diverse content (challenge ⓑ), `LiveNAS` introduces an online learning scheme that tailors the model to the particular unseen video. This scheme consists of selectively picking high-quality patches using the *Training Sampler* and transmitting them from the streamer to the server using the *Pacer* (Figure 3c). Since the patches needed for online training share bandwidth with the video, it is crucial to send only the patches with the highest expected impact. Hence, the *Training Sampler* detects patches that are hard to compress with high quality, by calculating the PSNR between HR patches and their bilinearly interpolated LR encoding, and selecting the lowest-PSNR patches. The *Pacer*, on the other hand, is responsible for allocating the available bandwidth between the low-resolution patches to be upscaled and the high-resolution training patches by adaptively tuning the respective bitrate. This is achieved

Table 3. Comparison of Visual Content Delivery Systems

System	Ref. CNN Model	Computational Optimizations	Model Specialization	CNN Execution
MobiSR [92]	RCAN [175]	1) Difficulty-aware model selection 2) Parallel execution 3) Efficient convolutions and upsampling	Generic model	Client
Yeo *et al.* [165]	VDSR [78]	1) Powerful desktop clients 2) Homogeneous clients 3) Up to 720p	Offline per video category	Client
NAS [166]	MDSR [104]	1) Powerful desktop clients 2) Early-exit models 3) FP16 quantization	Offline per video	Client
NEMO [164]	MDSR [104]	1) Key-frame selection 2) Early-exit models	Offline per video	Client
PARSEC [28]	MDSR [104]	1) Ultra low-resolution input 2) Manually-tuned models	Offline per video segment	Client
Supremo [167]	IDN [64]	1) Patch selection 2) Lightweight model IDN-lite	Generic model	Server
CloudSeg [152]	CARN [3]	1) Frame selection	Generic model	Server
Dejavu [54]	EDSR [104]	1) Patch selection 2) Luminance-only training 3) Powerful desktop client	Offline per caller	Client
LiveNAS [79]	MDSR [104]	1) Patch Selection 2) Parallel execution	Online per video	Server
SplitSR [108]	RCAN [175]	1) Efficient convolutions and model depth 2) Compiler optimizations for CPU	Quality- or latency-optimized	Client

through a scheduling algorithm (*Sched* in Figure 3c), which aims to optimize the video quality together with the expected online quality gain given the total training patches thus far.

On the server side, the transferred high-resolution patches are used by the *Online Trainer* to fine-tune the SR model. Further mitigating challenge ⓑ, recent patches are weighted more than older patches, better reflecting the current content of the stream. The invocation frequency of the *Online Trainer* is controlled based on an adaptive mechanism that detects training saturation and scene changes, periodically sending feedback to the client (*Quality Feedback Mechanism*). Training saturation, in particular, is tracked by measuring the performance gain from the two most recent models, suspending training and sending a signal to the client which sets the patch bitrate to a minimum value if the gain falls under a threshold consecutively. Scene change, on the other hand, is identified by comparing the performance between the initial and latest model, resuming online training with recent patches and prompting the client to reset the patch bitrate to its initial value. Thus, the amount of training for each live stream is adapted to maximize performance without excessive resource usage. Finally, to alleviate challenge ⓐ, LiveNAS supports scale-out execution by parallelizing the SR computations across multiple GPUs (*e.g.* three GPUs for 1080p to 4K real-time enhancement), if available, on the server.

4.3 Discussion: Dataset Collection

A challenging issue of neural enhancement-based CDS systems is the collection of data for the training of the employed DNNs and the end-to-end evaluation of the whole system. Depending on whether the system is image-centric, video-centric or targets a more specialized application (*e.g.* 360°videos, semantic segmentation or video-conferencing), each CDS employs a different strategy for the data collection stage. *Image-centric systems*, such as MobiSR, Supremo and SplitSR, employ broadly used super-resolution datasets, including the DIV2K dataset for the training of their DNN models and Set5, Set14, B100 and Urban100 for the evaluation stage. *Video-oriented systems*, such as NAS, NEMO and LiveNAS, rely on ad-hoc methodologies for the collection of datasets from

existing service providers, such as YouTube. Such methodologies typically comprise a selection of the top-N most popular videos from the top-K most popular categories or channels on the service-provider's platform. For instance, NEMO, which focuses on on-demand video delivery, selected three 4K videos from the top-10 most popular categories on YouTube, while LiveNAS, which targets live streaming, used the most recent streams from the top streamer in Twitch's five most popular categories and one video for each of the top-4 most popular live categories on YouTube. On the other hand, PARSEC used the most broadly used 360°head movement dataset [109] that contains head movement from 50 users on 10 360°videos from YouTube. CloudSeg employed the widely used Cityscapes [24] which is a semantic segmentation dataset with urban scenery. Finally, for Dejavu, the authors constructed a proprietary dataset by conducting five mock interviews under the same conditions, *i.e.* same meeting room on the same day, and recording them using a commodity smartphone on a tripod. Regardless of the dataset used, the respective low-quality counterpart can be synthesized from the original high-resolution image using a uniform degradation operation, such as the widely used bicubic interpolation, to form input-output pairs for supervised training. Although simple, these degradation operations suffice as the same degradation operations are used during inference. Nonetheless, the choice of dataset would directly affect the visual quality and the upscaling process will be considerably impacted if the data distribution encountered at run time greatly differs from the training data distribution, resulting in fail-safe mechanisms such as directly sending the HR patches [152]. To counteract this, we provide directions to improve the robustness of these models in Section 5.4.

4.4 Discussion: Trends in Neural Enhancement-based Content Delivery Systems

State-of-the-art neural enhancement-based content delivery systems continue to face tougher deployment challenges as compared to earlier systems due to the naturally evolving landscape of various applications and devices. These recent systems have to take into account of a wider range of heterogeneous clients and thus ensure backward compatibility with older devices. On top of that, existing mainstream applications, such as live streaming and video calls, involve dynamic content, exacerbating challenge ⓑ. Newly emerging applications such as 360° videos, in contrast, have unprecedented computational demand in order to support new immersive experiences at higher resolutions (challenge ⓐ). In response to neural enhancement deployment challenges, existing CDS have deployed numerous mitigation techniques as summarized in Table 3. Although the diversity of target applications, with their unique requirement and challenges, does not allow us to draw conclusive and meaningful results on the performance comparison between the existing neural enhancement-based CDS, the following observations are made.

Frameworks that combine both lightweight and adaptive/scalable neural networks demonstrate higher adaptability. This property can be observed in CDS such as NAS, NEMO, MobiSR and SplitSR. On the one hand, NAS and NEMO employ: *1)* a lightweight model design through a compact variant of MDSR [104] with 16-bit quantization, together with *2.1)* a per-video catalog of models with diverse complexity-performance trade-offs that enables the *static* model selection based on the client device nominal capabilities, and *2.2)* a scalable design of individual models that enables the *dynamic* adaptation of model complexity to the run-time device load. Adopting an alternative approach, MobiSR combines: *1)* a lightweight compressed version of RCAN [175], together with *2.1)* a *static* model selection based on the target device capabilities and model performance and *2.2)* a *run-time* model selection scheduler that pins a distinct model to each compute engine of the target platform and dispatches each patch to the suitable model-engine pair. Similarly, SplitSR employs a customized version of RCAN, optimized for either quality or latency. Such approaches provide greater flexibility at both design and run time. In this manner, CDS have more room to adapt to

static (*e.g.* device capabilities) and dynamic constraints (*e.g.* device load, networking conditions), meet the latency and visual quality requirements, and maximize QoE.

Applying informed frame/patch-selective enhancement through a quality-aware mechanism tends to radically improve QoE with minimal visual quality degradation. NEMO and Supremo's caches that store previously super-resolved frames enables the reuse of already computed results, thus skipping the costly processing overhead of neural enhancement. With the goal to sustain high visual quality in the challenging task of 360° video delivery, PARSEC selectively chooses which frame patches to enhance and which to directly download in high resolution. A similar strategy is adopted by CloudSeg with the aim to maximize the accuracy of the target semantic segmentation task. In another direction, LiveNAS employs frame selection to determine which high-resolution frames should be transmitted to the server side for further neural model fine-tuning on the current live stream. Across these CDS, the frame/patch selection criteria aim to either select content based on their upscaling or encoding difficulty or the degree of temporal redundancy. For instance, patches with complex textures and edges are often prioritized to be enhanced in order to maximize performance, or frames with high correlations with subsequent frames are cached after enhancement in order to be reused.

Specializing models to specific content tends to reduce the complexity of neural model design and rely more on system-level solutions. General mitigations to challenge ⓑ involve fine-tuning from a pre-trained generic model to the targeted content per scale for each client's computational regime. With the exemption of MobiSR, Supremo, and CloudSeg, all existing CDS adopt a variant of this model specialization approach. These systems require $C*S*\epsilon$ trained models where C is the number of unique content (*i.e.* entire video, video cluster or video segment), S is the number of available scales for each content, and ϵ is the number of supported computational regimes across its clients (*e.g.* low-end, mid-tier and high-end platforms). This computational load is a one-time cost for offline applications, but imposes additional challenges for their online counterparts, such as LiveNAS, as discussed in Section 4.2. This approach relieves CDS designers from designing sophisticated neural enhancement models by deriving multiple specialized and scale-specific variants from a single reference model, with the CDS deploying the suitable one as requested by the client. Nevertheless, it leaves space for further model-level exploration as discussed in Section 5.2.

5 FUTURE DIRECTIONS

In this section, we propose various approaches that are drawn from the latest computer vision research and provide insights on how neural enhancement can further benefit content delivery systems. Specifically, we highlight promising directions to further address the challenges of neural enhancement.

5.1 Improving Visual Quality

One of the main open challenges in neural enhancement algorithms is the design of a metric that will correspond well with human raters. As mentioned in Section 2.1, distortion-based metrics, such as PSNR [44] or SSIM [155], which aim to minimize the per-pixel error between two images, have been extensively shown to improve image fidelity at the cost of perceptual quality, leading to blurry and unnatural outcomes [88]. On the other hand, optimizing only for a perceptual-based metric such as Naturalness Image Quality Evaluator (NIQE) [117] and Learned Perceptual Image Patch Similarity (LPIPS) [174] will lead to more natural-looking images at the expense of fidelity and therefore the occasional occurrence of image artefacts. Mathematically, there is a trade-off between fidelity and perceptual quality [13].

As all the existing neural enhancement frameworks (Section 4) train their models using a distortion-based metric, the outputs of these models are accurate, but may look unnatural. Although this will benefit video analytics frameworks, such as CloudSeg, having blurry outputs will undermine the goal of other content streaming systems, such as Dejavu and LiveNAS. To close this gap, these systems can benefit further by utilizing recent methods proposed in computer vision to train and optimize their neural enhancement models. Some of these works focus on striking an optimal point between image fidelity and perceptual quality by optimizing for both distortion-based and perceptual-based metrics [91, 151]. Specifically, these works often optimize their models jointly on L1/Mean-Squared-Error (MSE) losses for image fidelity and a variety of losses, including perceptual loss [73], adversarial loss [45], and contextual loss [116], for perceptual quality. Additionally, other methods such as interpolating between a distortion-based and perceptual-based output image [31] or model [150, 151], or introducing additional priors [111, 133], can also be used to alleviate undesirable artefacts caused by optimizing only for perception-based metrics. For instance, utilizing rich texture priors found in pre-trained generative models has been shown to result in images that are high in both fidelity and perceptual quality [16]. Although the aforementioned works have found success in achieving high visual quality content, most propose techniques and models that are too computationally demanding to be deployed in existing CDS. Hence, there are potential gaps in the literature to realize the benefits of utilizing these approaches.

For video-based solutions, the majority of existing vision works utilize temporal information by aligning and fusing spatial information from multiple frames to further boost image fidelity [17, 75, 102, 127]. Specifically, these CNN- or RNN-based solutions align adjacent frames by warping each supporting frame to a reference frame using the respective optical flow. These optical flows are usually estimated using either traditional motion estimation algorithms [8] or CNN-based approaches, such as spatial transformer networks [69] and task-specific motion estimation networks [66, 125, 139]. Instead of utilizing an explicit component for motion estimation, another line of work [72, 143, 148] has proposed performing alignment implicitly through deformable convolutions [27, 178] or dynamically-generated filters.

As a result of effectively utilizing multiple frames to upscale each frame, these multi-frame approaches are able to achieve better restoration performance as compared to their single-image counterparts. Besides improving the visual quality spatially, temporal interpolation methods [158, 180] can enable a system to increase the achieved frame rate, and hence the QoE, by estimating intermediate frames rather than transferring them and applying super-resolution. In this manner, the bandwidth requirements are reduced by cutting both the content's spatial and temporal resolution. Nonetheless, video-based enhancement solutions are more computationally demanding than image-based solutions due to additional processing in the temporal domain. Therefore, these techniques can be utilized more effectively in CDS such as NAS and CloudSeg, which perform their computations on powerful desktop clients and on server-grade processors, respectively.

With regards to QoE, the actual bitrate or an estimated bitrate, given the performance of the model, is used to quantify the content's visual quality in relations to the user's experience (Section 2.1). As the amount of computational resources continue to grow in commodity hardware, the content's spatial quality will be largely dependent on the performance of the model. Therefore, there is a need for better designs of the QoE metric to reflect these changes for neural enhancement-based CDS.

5.2 Utilizing Efficiency-optimized Models

Most neural enhancement frameworks generally adopt popular full-blown SR models (Table 3) and revise them in order to speed up training and inference or fit into the limiting constraints for client-side computation. However, these revisions are usually suboptimal or, in some cases,

detrimental. For instance, PARSEC uses of batch normalization [67] to speed up training, reducing image fidelity [104] and introducing image artefacts [151]. Therefore, instead of naively scaling down and modifying full-blown SR models, these systems can leverage existing off-the-shelf efficient models that are specifically optimized for both performance and efficiency in order to deliver higher-quality enhancement at a lower computational cost (challenge ⓐ). These models include manually designed variants such as IDN [64] – already used by Supremo, automatically-designed variants such as ESRN [134] and TPSR [91] through neural architecture search [179], or even quantized [93] and binarized SR models [113, 160] if the hardware supports their efficient execution.

5.3 Faster Model Specialization through Meta-learning

To mitigate challenge ⓑ, some systems specialize a model for each specific image/video through overfitting. This approach results in a computationally expensive process (challenge ⓐ) as each model requires thousands of gradient updates in order to be adequately specialized. In response, later systems introduce a two-stage approach. First, a generic neural enhancement model is pre-trained offline, and then fine-tuning is applied either offline or online. For instance, NAS first trains a generic model on an external dataset before using its weights to fine-tune a separate model for each video, amortizing in this manner the one-time offline training cost.

To further speed up and improve the performance during the fine-tuning step, these works can adopt a *meta-learning* approach [51] in order to find a more optimal set of initialization parameters for fine-tuning. In other words, pre-training a neural enhancement model via meta-learning on an external dataset will require fewer gradient updates during the fine-tuning stage, therefore requiring fewer computational resources and leading to higher performance compared to brute-force fine-tuning [121, 132].

This fast adaptation is achieved by training the model so that it is easy to fine-tune. Specifically, besides having a regular training loop that optimizes the model on an external dataset, there is an additional inner loop that computes the adapted parameters given a different set of paired images. The main set of parameters is then iteratively updated based on the derivative of the loss with respect to the adapted parameters, resulting in a set of parameters that can quickly adapt to a test image during inference. Therefore, the quick adaption of the meta-trained parameters can further save the computations required to fine-tune in both offline (*e.g.* on-demand streaming) and online cases (*e.g.* live streaming).

5.4 Improving Robustness through Conditional Enhancement

Originally, the usage of SR was to improve the quality of a low-quality image under the assumption that the high-quality version was not available. As such, one of the key benefits of deploying an SR model as a neural enhancement unit is its ability to work without the high-resolution *ground-truth*. However, in the context of many content delivery settings, the ground-truth *is* available and the service provider scales it down to a lower quality in order to minimize the transmission cost. Therefore, instead of discarding the high-frequency information during downsampling, this information can be encoded in a latent variable and transmitted, along with the low-quality frame, to the receiver in CDS. The upsampling process can then be conditioned on that variable, therefore counteracting performance variability (challenge ⓑ) by allowing the same model to better handle differing content instead of having to overfit one model for each content. To this end, several computer vision works leverage the downscaling process during the upscaling process using neural image rescaling techniques to further boost image reconstruction. For instance, a downscaling CNN can be trained jointly with an existing SR model as shown in [77] and techniques such as

encoder-decoder frameworks and invertible neural networks can also be utilized as shown in [95] and [159] respectively.

Despite its benefits, neural-based image rescaling incurs an additional cost of executing a downscaling neural network as compared to that of interpolation methods, utilizing additional computational resources for a more robust improvement in visual quality. Therefore, image rescaling techniques may be more suitable for on-demand video systems, such as NAS and PARSEC, in which the downscaling cost is an offline one-time cost across videos. Additionally, although these approaches have not been shown to fully mitigate challenge ⓑ, they can significantly reduce the dependency of having to frequently transmit model segments for each content. Ultimately, as models get more robust to varying inputs, they can be deployed once across varying content and even across various applications.

5.5 Dynamic Deep Neural Networks

A growing body of work in the computer vision literature is investigating the design and deployment of dynamic deep neural networks. Such models employ conditional execution mechanisms in order to provide a run-time tunable accuracy-complexity trade-off, by scaling up and down their computational complexity. Such mechanisms would allow CDS to dynamically adapt the execution of neural enhancement models based on the client device capabilities and compute load, and provide another configurable dimension for CDS to ensure high QoE. Nevertheless, the majority of existing work focuses on classification models [38, 42, 48, 53, 55, 74, 76, 82, 83, 86, 87, 128, 142, 149, 154], rather than super-resolution and image enhancement.

Although NAS [166] and MobiSR [92] explored this direction by introducing the scalable NAS-MDSR model and a difficulty-aware model selection scheme, respectively, the potential of other dynamic methods has remained unexplored. So far, the computer vision community has proposed a broader range of dynamic neural network techniques, spanning from *dynamic pruning* schemes that skip either layers [149] or channels [38, 42, 55, 153] in an input-dependent manner, *model cascades* [48, 53, 74, 82, 83, 128], and *early-exit models* (either hand-crafted [57, 173], hardware-aware [87], distributed [86, 154] or generic [76, 142]). By adapting and modifying such techniques, system designers can develop dynamic neural enhancement models [161] that can be highly optimized specifically for the deployment use-cases of content delivery systems.

5.6 Hardware Acceleration through Neural Processing Units

Another promising approach to alleviate the high computational demands of neural enhancement models is targeting the problem from a hardware perspective. At the moment, device vendors have integrated specialized hardware units – often named *neural processing units* – that are optimized for fast deep neural network processing in both smartphones [65] and embedded platforms [6]. To surpass the limitations of CPUs and GPUs and further boost the processing speed of neural models, several works have proposed custom hardware accelerators [135, 145].

In this context, by considering the unique needs of content delivery systems and their use-cases, highly customized hardware accelerators can be designed. This approach would tailor the underlying processing platform to the CDS' performance requirements, overcoming the performance and energy-efficiency bottlenecks of conventional CPUs and GPUs, and thus ensure high QoE at a lower overall cost. Such custom hardware solutions can be obtained either through hand-crafted designs [50] and super-resolution neural processing units [89], automated accelerator generation tools [147], or sophisticated model-hardware co-design frameworks [1, 19, 80] for the joint development of the neural enhancement model and its efficient hardware accelerator.

6 CONCLUSION

This paper presents a survey of a new class of visual content delivery systems that employ neural enhancement techniques to boost their performance. Through a detailed analysis of how they tackle the deployment challenges of deep neural enhancement models, we highlight their design choices in terms of system architecture, novel components and neural model design, and indicate their strengths and weaknesses. Despite the rapid progress of recent systems, the demand for visual content traffic will grow over the coming years due to both existing and emerging technologies, such as augmented and virtual reality [70, 106] and telepresence [170]. To this end, based on recent developments from both the computer vision and systems communities, we identify key optimization opportunities and propose promising research directions to address emerging challenges of the field, enhancing their performance and enabling a wider large-scale deployment of high-QoE content delivery services.

REFERENCES

[1] Mohamed S. Abdelfattah, Łukasz Dudziak, Thomas Chau, Royson Lee, Hyeji Kim, and Nicholas D. Lane. 2020. Best of Both Worlds: AutoML Codesign of a CNN and its Hardware Accelerator. In *Design Automation Conference (DAC)*.

[2] Adnan Ahmed, Zubair Shafiq, and Amir Khakpour. 2016. QoE Analysis of a Large-Scale Live Video Streaming Event. In *Proceedings of the 2016 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Science (SIGMETRICS '16)*. 395–396.

[3] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. 2018. Fast, Accurate, and Lightweight Super-Resolution with Cascading Residual Network. In *European Conference on Computer Vision (ECCV)*.

[4] Zahaib Akhtar, Yun Seong Nam, Ramesh Govindan, Sanjay Rao, Jessica Chen, Ethan Katz-Bassett, Bruno Ribeiro, Jibin Zhan, and Hui Zhang. 2018. Oboe: Auto-Tuning Video ABR Algorithms to Network Conditions. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication (SIGCOMM)*. 44–58.

[5] Mario Almeida, Muhammad Bilal, Alessandro Finamore, Ilias Leontiadis, Yan Grunenberger, Matteo Varvello, and Jeremy Blackburn. 2018. CHIMP: Crowdsourcing Human Inputs for Mobile Phones. In *Proceedings of the 2018 World Wide Web Conference (WWW)*. International World Wide Web Conferences Steering Committee, 45–54.

[6] Mario Almeida, Stefanos Laskaridis, Ilias Leontiadis, Stylianos I. Venieris, and Nicholas D. Lane. 2019. EmBench: Quantifying Performance Variations of Deep Neural Networks Across Modern Commodity Devices. In *The 3rd International Workshop on Deep Learning for Mobile Systems and Applications (EMDL)*. 6.

[7] 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 (MobiCom)*.

[8] S. Baker, D. Scharstein, J. Lewis, S. Roth, Michael J. Black, and R. Szeliski. 2007. A Database and Evaluation Methodology for Optical Flow. *International Journal of Computer Vision* (2007).

[9] Sefi Bell-Kligler, Assaf Shocher, and Michal Irani. 2019. Blind Super-Resolution Kernel Estimation using an Internal-GAN. In *Advances in Neural Information Processing Systems*.

[10] Ibrahim Ben Mustafa, Tamer Nadeem, and Emir Halepovic. 2018. FlexStream: Towards Flexible Adaptive Video Streaming on End Devices Using Extreme SDN. In *Proceedings of the 26th ACM International Conference on Multimedia (MM)*. 555–563.

[11] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer, and R. Zimmermann. 2019. A Survey on Bitrate Adaptation Schemes for Streaming Media Over HTTP. *IEEE Communications Surveys Tutorials* 21, 1 (2019), 562–585.

[12] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie line Alberi Morel. 2012. Low-Complexity Single-Image Super-Resolution based on Nonnegative Neighbor Embedding. In *Proceedings of the British Machine Vision Conference*.

[13] Yochai Blau and Tomer Michaeli. 2018. The Perception-Distortion Tradeoff. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[14] Dingding Cai, Ke Chen, Y. Qian, and J. Kämäräinen. 2019. Convolutional Low-Resolution Fine-Grained Classification. *Pattern Recognition Letters* 119 (2019), 166–171.

[15] Christopher Canel, Thomas Kim, Giulio Zhou, Conglong Li, Hyeontaek Lim, David G. Andersen, Michael Kaminsky, and Subramanya R. Dulloor. 2019. Scaling Video Analytics on Constrained Edge Nodes. In *Conference on Machine Learning and Systems (MLSys)*.

[16] Kelvin CK Chan, Xintao Wang, Xiangyu Xu, Jinwei Gu, and Chen Change Loy. 2021. GLEAN: Generative Latent Bank for Large-Factor Image Super-Resolution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[17] Kelvin CK Chan, Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. 2021. BasicVSR: The Search for Essential Components in Video Super-Resolution and Beyond. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[18] Chun-Ming Chang, Cheng-Hsin Hsu, Chih-Fan Hsu, and Kuan-Ta Chen. 2016. Performance Measurements of Virtual Reality Systems: Quantifying the Timing and Positioning Accuracy. In *Proceedings of the 24th ACM International Conference on Multimedia (MM)*. 655–659.

[19] T. Chau, L. Dudziak, M. Abdelfattah, Royson Lee, H. Kim, and N. Lane. 2020. BRP-NAS: Prediction-based NAS using GCNs. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[20] Tianqi Chen, Thierry Moreau, Ziheng Jiang, Lianmin Zheng, Eddie Yan, Haichen Shen, Meghan Cowan, Leyuan Wang, Yuwei Hu, Luis Ceze, Carlos Guestrin, and Arvind Krishnamurthy. 2018. TVM: An Automated End-to-End Optimizing Compiler for Deep Learning. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*. 578–594.

[21] T. Chen, L. Ravindranath, Shuo Deng, P. Bahl, and H. Balakrishnan. 2015. Glimpse: Continuous, Real-Time Object Recognition on Mobile Devices. In *SenSys '15*.

[22] Cisco. 2020. *Cisco Annual Internet Report (2018–2023) White Paper*. Technical Report. Cisco Systems, Inc. https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html [Retrieved: June 7, 2021].

[23] Cisco. 2020. *Cisco Visual Networking Index (VNI) Complete Forecast Update, 2017 - 2022*. Technical Report. Cisco Systems, Inc. https://www.cisco.com/c/dam/m/en_us/network-intelligence/service-provider/digital-transformation/knowledge-network-webinars/pdfs/1213-business-services-ckn.pdf [Retrieved: June 7, 2021].

[24] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. 2016. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[25] Daniel Crankshaw, Gur-Eyal Sela, Xiangxi Mo, Corey Zumar, Ion Stoica, Joseph Gonzalez, and Alexey Tumanov. 2020. InferLine: Latency-Aware Provisioning and Scaling for Prediction Serving Pipelines. In *Proceedings of the 11th ACM Symposium on Cloud Computing (SoCC)*. 477–491.

[26] Simon Da Silva, Sonia Ben Mokhtar, Stefan Contiu, Daniel Négru, Laurent Réveillère, and Etienne Rivière. 2019. PrivaTube: Privacy-Preserving Edge-Assisted Video Streaming. In *Proceedings of the 20th International Middleware Conference (Middleware)*. 189–201.

[27] Jifeng Dai, Haozhi Qi, Y. Xiong, Y. Li, Guodong Zhang, H. Hu, and Y. Wei. 2017. Deformable Convolutional Networks. *IEEE International Conference on Computer Vision (ICCV)* (2017).

[28] Mallesham Dasari, A. Bhattacharya, Santiago Vargas, Pranjal Sahu, A. Balasubramanian, and S. Das. 2020. Streaming 360-Degree Videos Using Super-Resolution. In *IEEE Conference on Computer Communications (INFOCOM)*.

[29] L. De Cicco, V. Caldaralo, V. Palmisano, and S. Mascolo. 2013. ELASTIC: A Client-Side Controller for Dynamic Adaptive Streaming over HTTP (DASH). In *2013 20th International Packet Video Workshop*. 1–8.

[30] Jonathan Deber, Ricardo Jota, Clifton Forlines, and Daniel Wigdor. 2015. How Much Faster is Fast Enough? User Perception of Latency & Latency Improvements in Direct and Indirect Touch. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI) (CHI '15)*. 1827–1836.

[31] Xin Deng. 2018. Enhancing Image Quality via Style Transfer for Single Image Super-Resolution. *IEEE Signal Processing Letters* (2018).

[32] Giorgos Dimopoulos, Ilias Leontiadis, Pere Barlet-Ros, and Konstantina Papagiannaki. 2016. Measuring Video QoE from Encrypted Traffic. In *Proceedings of the 2016 Internet Measurement Conference (IMC)*. 513–526.

[33] Pradeep Dogga, Sandip Chakraborty, Subrata Mitra, and Ravi Netravali. 2019. Edge-based Transcoding for Adaptive Live Video Streaming. In *2nd USENIX Workshop on Hot Topics in Edge Computing (HotEdge 19)*. USENIX Association.

[34] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. 2016. Image Super-Resolution Using Deep Convolutional Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (2016).

[35] Chao Dong, Chen Change Loy, and Xiaoou Tang. 2016. Accelerating the Super-Resolution Convolutional Neural Network. In *European Computer on Computer Vision (ECCV)*.

[36] Kuntai Du, Ahsan Pervaiz, Xin Yuan, Aakanksha Chowdhery, Qizheng Zhang, Henry Hoffmann, and J. Jiang. 2020. Server-Driven Video Streaming for Deep Learning Inference. *SIGCOMM* (2020).

[37] Ching-Ling Fan, J. Lee, Wen-Chih Lo, C. Huang, Kuan-Ta Chen, and C. Hsu. 2017. Fixation Prediction for 360 Video Streaming in Head-Mounted Virtual Reality. In *Network and Operating System Support for Digital Audio and Video*.

[38] Biyi Fang, Xiao Zeng, and Mi Zhang. 2018. NestDNN: Resource-Aware Multi-Tenant On-Device Deep Learning for Continuous Mobile Vision. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking (MobiCom)*. 115–127.

[39] M. Fiedler, T. Hossfeld, and P. Tran-Gia. 2010. A Generic Quantitative Relationship between Quality of Experience and Quality of Service. *IEEE Network* 24, 2 (2010), 36–41.

[40] Sadjad Fouladi, John Emmons, Emre Orbay, C. 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)*.

[41] M. Gadaleta, F. Chiariotti, M. Rossi, and A. Zanella. 2017. D-DASH: A Deep Q-Learning Framework for DASH Video Streaming. *IEEE Transactions on Cognitive Communications and Networking (TCCN)* 3, 4 (2017), 703–718.

[42] Xitong Gao, Yiren Zhao, Łukasz Dudziak, Robert Mullins, and Cheng zhong Xu. 2019. Dynamic Channel Pruning: Feature Boosting and Suppression. In *International Conference on Learning Representations*.

[43] C. Ge, N. Wang, G. Foster, and M. Wilson. 2017. Toward QoE-Assured 4K Video-on-Demand Delivery Through Mobile Edge Virtualization With Adaptive Prefetching. *IEEE Transactions on Multimedia (TMM)* 19, 10 (2017), 2222–2237.

[44] Rafael C. Gonzalez and Richard E. Woods. 2008. *Digital Image Processing*. Prentice Hall, Upper Saddle River, N.J.

[45] Ian Goodfellow et al. 2014. Generative Adversarial Nets. In *NeurIPS*.

[46] Jinjin Gu, Hannan Lu, W. Zuo, and C. Dong. 2019. Blind Super-Resolution With Iterative Kernel Correction. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019).

[47] T. Guarnieri, I. Drago, A. B. Vieira, I. Cunha, and J. Almeida. 2017. Characterizing QoE in Large-Scale Live Streaming. In *IEEE Global Communications Conference (GLOBECOM)*. 1–7.

[48] Seungyeop Han, Haichen Shen, Matthai Philipose, Sharad Agarwal, Alec Wolman, and Arvind Krishnamurthy. 2016. MCDNN: An Approximation-Based Execution Framework for Deep Stream Processing Under Resource Constraints. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*.

[49] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016).

[50] Z. He, H. Huang, M. Jiang, Y. Bai, and G. Luo. 2018. FPGA-Based Real-Time Super-Resolution System for Ultra High Definition Videos. In *2018 IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*. 181–188.

[51] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. 2020. Meta-Learning in Neural Networks: A Survey. arXiv:2004.05439 [cs.LG]

[52] A. Howard, Mark Sandler, G. Chu, Liang-Chieh Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Quoc V. Le, and H. Adam. 2019. Searching for MobileNetV3. *International Conference on Computer Vision (ICCV)* (2019).

[53] Kevin Hsieh, Ganesh Ananthanarayanan, Peter Bodik, Shivaram Venkataraman, Paramvir Bahl, Matthai Philipose, Phillip B. Gibbons, and Onur Mutlu. 2018. Focus: Querying Large Video Datasets with Low Latency and Low Cost. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation (OSDI)*. 269–286.

[54] Pan Hu, Rakesh Misra, and Sachin Katti. 2019. Dejavu: Enhancing Videoconferencing with Prior Knowledge. In *HotMobile*.

[55] Weizhe Hua, Yuan Zhou, Christopher M De Sa, Zhiru Zhang, and G. Edward Suh. 2019. Channel Gating Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*. 1886–1896.

[56] Cheng Huang, Jin Li, and Keith W. Ross. 2007. Can Internet Video-on-Demand Be Profitable?. In *Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*. 133–144.

[57] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Weinberger. 2018. Multi-Scale Dense Networks for Resource Efficient Image Classification. In *International Conference on Learning Representations (ICLR)*.

[58] Gao Huang, Zhuang Liu, and K. Weinberger. 2017. Densely Connected Convolutional Networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

[59] J. Huang, A. Singh, and N. Ahuja. 2015. Single image super-resolution from transformed self-exemplars. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[60] Tianchi Huang, Rui-Xiao Zhang, Chao Zhou, and Lifeng Sun. 2018. QARC: Video Quality Aware Rate Control for Real-Time Video Streaming Based on Deep Reinforcement Learning. In *Proceedings of the 26th ACM International Conference on Multimedia (MM)*. 1208–1216.

[61] T. Huang, C. Zhou, X. Yao, R. X. Zhang, C. Wu, B. Yu, and L. Sun. 2020. Quality-Aware Neural Adaptive Video Streaming With Lifelong Imitation Learning. *IEEE Journal on Selected Areas in Communications (JSAC)* 38, 10 (2020), 2324–2342.

[62] Te-Yuan Huang, Chaitanya Ekanadham, Andrew J. Berglund, and Zhi Li. 2019. Hindsight: Evaluate Video Bitrate Adaptation at Scale. In *Proceedings of the 10th ACM Multimedia Systems Conference (MMSys)* (Amherst, Massachusetts) *(MMSys '19)*. Association for Computing Machinery, New York, NY, USA, 86–97. https://doi.org/10.1145/3304109.3306219

[63] Te-Yuan Huang, Ramesh Johari, Nick McKeown, Matthew Trunnell, and Mark Watson. 2014. A Buffer-Based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service. In *SIGCOMM*.

[64] Zheng Hui, Xiumei Wang, and Xinbo Gao. 2018. Fast and Accurate Single Image Super-Resolution via Information Distillation Network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[65] Andrey Ignatov, Radu Timofte, Andrei Kulik, Seungsoo Yang, Ke Wang, Felix Baum, Max Wu, Lirong Xu, and Luc Van Gool. 2019. AI Benchmark: All About Deep Learning on Smartphones in 2019. In *International Conference on Computer Vision (ICCV) Workshops*.

[66] Eddy Ilg, N. Mayer, Tonmoy Saikia, Margret Keuper, A. Dosovitskiy, and T. Brox. 2017. FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

[67] S. Ioffe and Christian Szegedy. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *International Conference on Machine Learning (ICML)*.

[68] ITU. 2008. ITU-T Recommendation P.910. Subjective video quality assessment methods for multimedia applications. https://www.itu.int/rec/T-REC-P.910-200804-I. [Retrieved: June 7, 2021].

[69] Max Jaderberg, K. Simonyan, Andrew Zisserman, and K. Kavukcuoglu. 2015. Spatial Transformer Networks. In *Advances in Neural Information Processing Systems*.

[70] Puneet Jain, Justin Manweiler, and Romit Roy Choudhury. 2015. OverLay: Practical Mobile Augmented Reality. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*.

[71] J. Jiang, V. Sekar, and H. Zhang. 2014. Improving Fairness, Efficiency, and Stability in HTTP-Based Adaptive Video Streaming With Festive. *IEEE/ACM Transactions on Networking* 22, 1 (2014), 326–340.

[72] Younghyun Jo, S. Oh, Jaeyeon Kang, and S. Kim. 2018. Deep Video Super-Resolution Network Using Dynamic Upsampling Filters Without Explicit Motion Compensation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018).

[73] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 2016. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. In *European Conference on Computer Vision (ECCV)*.

[74] Daniel Kang, John Emmons, Firas Abuzaid, Peter Bailis, and Matei Zaharia. 2017. NoScope: Optimizing Neural Network Queries over Video at Scale. *VLDB* (2017).

[75] Armin Kappeler, Seunghwan Yoo, Qiqin Dai, and A. Katsaggelos. 2016. Video Super-Resolution With Convolutional Neural Networks. *IEEE Transactions on Computational Imaging* (2016).

[76] Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. 2019. Shallow-Deep Networks: Understanding and Mitigating Network Overthinking. 3301–3310.

[77] Heewon Kim, Myungsub Choi, Bee Lim, and Kyoung Mu Lee. 2018. Task-Aware Image Downscaling. In *European Conference on Computer Vision (ECCV)*.

[78] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. 2016. Accurate Image Super-Resolution using Very Deep Convolutional Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[79] Jae-Hong Kim, Youngmok Jung, H. Yeo, Juncheol Ye, and D. Han. 2020. Neural-Enhanced Live Streaming: Improving Live Video Ingest via Online Learning. In *SIGCOMM*.

[80] Y. Kim, J. Choi, and M. Kim. 2018. A Real-Time Convolutional Neural Network for Super-Resolution on FPGA with Applications to 4K UHD 60 fps Video Services. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)* (2018).

[81] G. T. Kolyvas, S. E. Polykalas, and I. S. Venieris. 1997. Performance Evaluation of Intelligent Signaling Servers for Broadband Multimedia Networks. In *Proceedings Second IEEE Symposium on Computer and Communications (ICC)*. 96–103.

[82] A. Kouris, S. I. Venieris, and C. Bouganis. 2020. A Throughput-Latency Co-Optimised Cascade of Convolutional Neural Network Classifiers. In *2020 Design, Automation Test in Europe Conference Exhibition (DATE)*. 1656–1661.

[83] A. Kouris, S. I. Venieris, and C. S. Bouganis. 2018. CascadeCNN: Pushing the Performance Limits of Quantisation in Convolutional Neural Networks. In *28th International Conference on Field Programmable Logic and Applications (FPL)*.

[84] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Processing Systems (NeurIPS)*.

[85] N. D. Lane, S. Bhattacharya, A. Mathur, P. Georgiev, C. Forlivesi, and F. Kawsar. 2017. Squeezing Deep Learning into Mobile and Embedded Devices. *IEEE Pervasive Computing* 16, 3 (2017), 82–88.

[86] Stefanos Laskaridis, Stylianos I. Venieris, Mario Almeida, Ilias Leontiadis, and Nicholas D. Lane. 2020. SPINN: Synergistic Progressive Inference of Neural Networks over Device and Cloud. In *The 26th Annual International Conference on Mobile Computing and Networking (MobiCom)*.

[87] Stefanos Laskaridis, Stylianos I. Venieris, Hyeji Kim, and Nicholas D. Lane. 2020. HAPI: Hardware-Aware Progressive Inference. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*.

[88] C. Ledig, L. Theis, Ferenc Huszár, J. Caballero, Andrew Aitken, Alykhan Tejani, J. Totz, Zehan Wang, and W. Shi. 2017. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[89] J. Lee, J. Lee, and H. J. Yoo. 2020. SRNPU: An Energy-Efficient CNN-Based Super-Resolution Processor With Tile-Based Selective Super-Resolution in Mobile Devices. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems (JECAS)* 10, 3 (2020), 320–334.

[90] Kyungjin Lee, Juheon Yi, Youngki Lee, Sunghyun Choi, and Young Min Kim. 2020. GROOT: A Real-Time Streaming System of High-Fidelity Volumetric Videos. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking (MobiCom)*.

[91] Royson Lee, L. Dudziak, M. Abdelfattah, Stylianos I. Venieris, H. Kim, Hongkai Wen, and N. Lane. 2020. Journey Towards Tiny Perceptual Super-Resolution. In *European Conference on Computer Vision (ECCV)*.

[92] Royson Lee, Stylianos I. Venieris, L. Dudziak, S. Bhattacharya, and N. Lane. 2019. MobiSR: Efficient On-Device Super-Resolution through Heterogeneous Mobile Processors. In *The 25th Annual International Conference on Mobile Computing and Networking (MobiCom)*.

[93] Huixia Li, Chenqian Yan, Shaohui Lin, Xiawu Zheng, B. Zhang, Fan Yang, and Rongrong Ji. 2020. PAMS: Quantized Super-Resolution via Parameterized Max Scale. In *ECCV*.

[94] X. Li, M. A. Salehi, M. Bayoumi, N. Tzeng, and R. Buyya. 2018. Cost-Efficient and Robust On-Demand Video Transcoding Using Heterogeneous Cloud Services. *IEEE Transactions on Parallel and Distributed Systems (TPDS)* 29, 3 (2018), 556–571.

[95] Y. Li, D. Liu, H. Li, Lianghuan Li, Z. Li, and F. Wu. 2019. Learning a Convolutional Neural Network for Image Compact-Resolution. *IEEE Transactions on Image Processing (TIP)* (2019).

[96] Yule Li, J. Shi, and D. Lin. 2018. Low-Latency Video Semantic Segmentation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2018).

[97] Yuheng Li, Yiping Zhang, and Ruixi Yuan. 2011. Measurement and Analysis of a Large Scale Commercial Mobile Internet TV System. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference (IMC)*. 209–224.

[98] Z. Li, M. A. Kaafar, K. Salamatian, and G. Xie. 2017. Characterizing and Modeling User Behavior in a Large-Scale Mobile Live Streaming System. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)* 27, 12 (2017), 2675–2686.

[99] Zhenyu Li, Jiali Lin, Marc-Ismael Akodjenou, Gaogang Xie, Mohamed Ali Kaafar, Yun Jin, and Gang Peng. 2012. Watching Videos from Everywhere: A Study of the PPTV Mobile VoD System. In *Proceedings of the 2012 Internet Measurement Conference (IMC)*. 185–198.

[100] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen, and D. Oran. 2014. Probe and Adapt: Rate Adaptation for HTTP Video Streaming At Scale. *IEEE Journal on Selected Areas in Communications (JSAC)* 32, 4 (2014), 719–733.

[101] Li-Shen Juhn and Li-Ming Tseng. 1997. Harmonic Broadcasting for Video-on-Demand Service. *IEEE Transactions on Broadcasting (TBC)* 43, 3 (1997), 268–271.

[102] Renjie Liao, X. Tao, R. Li, Z. Ma, and J. Jia. 2015. Video Super-Resolution via Deep Draft-Ensemble Learning. *IEEE International Conference on Computer Vision (ICCV)* (2015).

[103] Melissa Licciardello, Maximilian Grüner, and Ankit Singla. 2020. Understanding Video Streaming Algorithms in the Wild. In *Passive and Active Measurement (PAM)*, Anna Sperotto, Alberto Dainotti, and Burkhard Stiller (Eds.). Springer International Publishing, Cham, 298–313.

[104] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. 2017. Enhanced Deep Residual Networks for Single Image Super-Resolution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*.

[105] H. Liu, Zhubo Ruan, Peng Zhao, F. Shang, Linlin Yang, and Yuanyuan Liu. 2020. Video Super Resolution Based on Deep Learning: A comprehensive survey. *ArXiv* abs/2007.12928 (2020).

[106] Luyang Liu, Hongyu Li, and Marco Gruteser. 2019. Edge Assisted Real-Time Object Detection for Mobile Augmented Reality. In *The 25th Annual International Conference on Mobile Computing and Networking (MobiCom)*.

[107] Luyang Liu, Ruiguang Zhong, Wuyang Zhang, Yunxin Liu, Jiansong Zhang, Lintao Zhang, and Marco Gruteser. 2018. Cutting the Cord: Designing a High-Quality Untethered VR System with Low Latency Remote Rendering. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*. 68–80.

[108] Xin Liu, Yuang Li, Josh Fromm, Yuntao Wang, Ziheng Jiang, Alex Mariakakis, and Shwetak Patel. 2021. SplitSR: An End-to-End Approach to Super-Resolution on Mobile Devices. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. (IMWUT)* (2021).

[109] Wen-Chih Lo, Ching-Ling Fan, Jean Lee, Chun-Ying Huang, Kuan-Ta Chen, and Cheng-Hsin Hsu. 2017. 360° Video Viewing Dataset in Head-Mounted Virtual Reality. In *Proceedings of the 8th ACM on Multimedia Systems Conference (MMSys)*. 211–216.

[110] Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully Convolutional Networks for Semantic Segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 3431–3440.

[111] Cheng Ma, Yongming Rao, Yean Cheng, Ce Chen, Jiwen Lu, and J. Zhou. 2020. Structure-Preserving Super Resolution With Gradient Guidance. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[112] Ningning Ma, X. Zhang, Hai-Tao Zheng, and Jian Sun. 2018. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In *European Conference on Computer Vision (ECCV)*.

[113] Y. Ma, Hongyu Xiong, Zhe Hu, and L. Ma. 2019. Efficient Super Resolution Using Binarized Neural Network. *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2019).

[114] Hongzi Mao, R. Netravali, and M. Alizadeh. 2017. Neural Adaptive Video Streaming with Pensieve. *SIGCOMM* (2017).

[115] D. Martin, C. Fowlkes, D. Tal, and J. Malik. 2001. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *IEEE International Conference on Computer Vision (ICCV)*.

[116] Roey Mechrez, I. Talmi, Firas Shama, and L. Zelnik-Manor. 2018. Maintaining Natural Image Statistics with the Contextual Loss. In *Asian Conference on Computer Vision*.

[117] Anish Mittal, Rajiv Soundararajan, and Alan C. Bovik. 2013. Making a “Completely Blind" Image Quality Analyzer. *IEEE Signal Processing Letters* (2013).

[118] R. K. P. Mok, E. W. W. Chan, and R. K. C. Chang. 2011. Measuring the Quality of Experience of HTTP Video Streaming. In *12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops*. 485–492.

[119] Vinod Nair and Geoffrey E. Hinton. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning (ICML)* (Haifa, Israel) *(ICML'10)*. Omnipress, Madison, WI, USA, 807–814.

[120] NVIDIA. 2020. NVIDIA Maxine - Cloud-AI Video-Streaming Platform. https://developer.nvidia.com/maxine. [Retrieved: June 7, 2021].

[121] Seobin Park, Jinsu Yoo, Donghyeon Cho, Jiwon Kim, and Tae Hyun Kim. 2020. Fast Adaptation to Super-Resolution Networks via Meta-Learning. In *European Conference on Computer Vision (ECCV)*.

[122] K. Piamrat, C. Viho, J. Bonnin, and A. Ksentini. 2009. Quality of Experience Measurements for Video Streaming over Wireless Networks. In *2009 Sixth International Conference on Information Technology: New Generations (ITNG)*. 1184–1189.

[123] F. Qian, B. Han, Qingyang Xiao, and V. Gopalakrishnan. 2018. Flare: Practical Viewport-Adaptive 360-Degree Video Streaming for Mobile Devices. *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking (MobiCom)* (2018).

[124] Parthasarathy Ranganathan, Daniel Stodolsky, Jeff Calow, Jeremy Dorfman, Marisabel Guevara, Clinton Wills Smullen IV, Aki Kuusela, Raghu Balasubramanian, Sandeep Bhatia, Prakash Chauhan, Anna Cheung, In Suk Chong, Niranjani Dasharathi, Jia Feng, Brian Fosco, Samuel Foss, Ben Gelb, Sara J. Gwin, Yoshiaki Hase, Da-ke He, C. Richard Ho, Roy W. Huffman Jr., Elisha Indupalli, Indira Jayaram, Poonacha Kongetira, Cho Mon Kyaw, Aaron Laursen, Yuan Li, Fong Lou, Kyle A. Lucke, JP Maaninen, Ramon Macias, Maire Mahony, David Alexander Munday, Srikanth Muroor, Narayana Penukonda, Eric Perkins-Argueta, Devin Persaud, Alex Ramirez, Ville-Mikko Rautio, Yolanda Ripley, Amir Salek, Sathish Sekar, Sergey N. Sokolov, Rob Springer, Don Stark, Mercedes Tan, Mark S. Wachsler, Andrew C. Walton, David A. Wickeraad, Alvin Wijaya, and Hon Kwan Wu. 2021. Warehouse-Scale Video Acceleration: Co-Design and Deployment in the Wild. In *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. 600–615.

[125] A. Ranjan and Michael J. Black. 2017. Optical Flow Estimation Using a Spatial Pyramid Network. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

[126] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. 1992. Nonlinear Total Variation Based Noise Removal Algorithms. *Phys. D* 60, 1-4 (Nov. 1992), 259–268.

[127] Mehdi S. M. Sajjadi, Raviteja Vemulapalli, and M. Brown. 2018. Frame-Recurrent Video Super-Resolution. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018).

[128] Haichen Shen, Seungyeop Han, Matthai Philipose, and Arvind Krishnamurthy. 2017. Fast Video Classification via Adaptive Cascading of Deep Models. In *CVPR*.

[129] W. Shi, J. Caballero, Ferenc Huszár, J. Totz, A. Aitken, R. Bishop, D. Rueckert, and Zehan Wang. 2016. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016).

[130] Assaf Shocher, N. Cohen, and M. Irani. 2018. "Zero-Shot" Super-Resolution Using Deep Internal Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[131] Laurent Sifre and Stéphane Mallat. 2014. Rigid-motion scattering for image classification. *PhD thesis, Ph. D. thesis* (2014).

[132] Jae Woong Soh, Sunwoo Cho, and N. I. Cho. 2020. Meta-Transfer Learning for Zero-Shot Super-Resolution. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[133] Jae Woong Soh, G. Y. Park, Junho Jo, and N. I. Cho. 2019. Natural and Realistic Single Image Super-Resolution With Explicit Natural Manifold Discrimination. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[134] Dehua Song, Chang Xu, Xu Jia, Yiyi Chen, Chunjing Xu, and Yunhe Wang. 2020. Efficient Residual Dense Block Search for Image Super-Resolution. In *Association for the Advancement of Artificial Intelligence (AAAI)*.

[135] J. Song, Y. Cho, J. Park, J. Jang, S. Lee, J. Song, J. Lee, and I. Kang. 2019. 7.1 An 11.5TOPS/W 1024-MAC Butterfly Structure Dual-Core Sparsity-Aware Neural Processing Unit in 8nm Flagship Mobile SoC. In *IEEE International Solid-State Circuits Conference (ISSCC)*. 130–132.

[136] Kevin Spiteri, Ramesh Sitaraman, and Daniel Sparacio. 2018. From Theory to Practice: Improving Bitrate Adaptation in the DASH Reference Player. In *Proceedings of the 9th ACM Multimedia Systems Conference (MMSys)*. 123–137.

[137] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman. 2016. BOLA: Near-Optimal Bitrate Adaptation for Online Videos. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*. 1–9.

[138] Thomas Stockhammer. 2011. Dynamic Adaptive Streaming over HTTP –: Standards and Design Principles. In *Proceedings of the Second Annual ACM Conference on Multimedia Systems (MMSys)*. 133–144.

[139] Deqing Sun, X. Yang, Ming-Yu Liu, and J. Kautz. 2018. PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2018).

[140] Yi Sun, Xiaoqi Yin, Junchen Jiang, Vyas Sekar, Fuyuan Lin, Nanshu Wang, Tao Liu, and Bruno Sinopoli. 2016. CS2P: Improving Video Bitrate Selection and Adaptation with Data-Driven Throughput Prediction. In *Proceedings of the 2016 ACM SIGCOMM Conference (SIGCOMM '16)*. 272–285.

[141] X. Tao, H. Gao, Renjie Liao, J. Wang, and J. Jia. 2017. Detail-Revealing Deep Video Super-Resolution. *IEEE International Conference on Computer Vision (ICCV)* (2017).

[142] Surat Teerapittayanon, Bradley McDanel, and HT Kung. 2016. BranchyNet: Fast Inference via Early Exiting from Deep Neural Networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*. 2464–2469.

[143] Yapeng Tian, Yulun Zhang, Yun Fu, and Chenliang Xu. 2020. TDAN: Temporally-Deformable Alignment Network for Video Super-Resolution. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).

[144] R. Timofte, Eirikur Agustsson, L. Gool, M. Yang, Lei Zhang, Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, K. Lee, Xintao Wang, Yapeng Tian, K. Yu, Yulun Zhang, Shixiang Wu, C. Dong, L. Lin, Y. Qiao, C. C. Loy, W. Bae, Jae Jun Yoo, Yoseob Han, J. C. Ye, Jae-Seok Choi, M. Kim, Yuchen Fan, J. Yu, Wei Han, Ding Liu, Haichao Yu, Zhangyang Wang, Humphrey Shi, X. Wang, T. Huang, Yunjin Chen, Kai Zhang, W. Zuo, Z. Tang, Linkai Luo, S. Li, M. Fu, L. Cao, Wen Heng, G. Bui, Truc Le, Ye Duan, D. Tao, Ruxin Wang, Xu Lin, Jianxin Pang, Jinchang Xu, Y. Zhao, Xiangyu Xu, Jin shan Pan, Deqing Sun, Y. Zhang, X. Song, Yuchao Dai, X. Qin, X. Huynh, Tiantong Guo, H. Mousavi, T. Vu, V. Monga, C. Cruz, K. Egiazarian, V. Katkovnik, Rakesh Mehta, A. Jain, Abhinav Agarwalla, Ch V. Sai Praveen, Ruofan Zhou, Hongdiao Wen, C. Zhu, Zhiqiang Xia, Z. Wang, and Q. Guo. 2017. NTIRE 2017 Challenge on Single Image Super-Resolution: Methods and Results. *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2017).

[145] S. I. Venieris and C. S. Bouganis. 2019. fpgaConvNet: Mapping Regular and Irregular Convolutional Neural Networks on FPGAs. *IEEE Transactions on Neural Networks and Learning Systems* 30, 2 (2019), 326–342.

[146] Stylianos I Venieris, Alexandros Kouris, and Christos-Savvas Bouganis. 2018. Deploying Deep Neural Networks in the Embedded Space. In *2nd International Workshop on Embedded and Mobile Deep Learning (EMDL)*.

[147] Stylianos I. Venieris, Alexandros Kouris, and Christos-Savvas Bouganis. 2018. Toolflows for Mapping Convolutional Neural Networks on FPGAs: A Survey and Future Directions. *ACM Comput. Surv. (CSUR)* 51, 3, Article 56 (June 2018), 39 pages. https://doi.org/10.1145/3186332

[148] Xintao Wang, Kelvin C. K. Chan, K. Yu, C. Dong, and Chen Change Loy. 2019. EDVR: Video Restoration With Enhanced Deformable Convolutional Networks. *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2019).

[149] Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. 2018. SkipNet: Learning Dynamic Routing in Convolutional Networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 409–424.

[150] Xintao Wang, K. Yu, C. Dong, X. Tang, and Chen Change Loy. 2019. Deep Network Interpolation for Continuous Imagery Effect Transition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[151] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Chen Change Loy, Yu Qiao, and Xiaoou Tang. 2018. ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks. In *European Conference on Computer Vision Workshops (ECCVW)*.

[152] Yiding Wang, Weiyan Wang, Junxue Zhang, J. Jiang, and K. Chen. 2019. Bridging the Edge-Cloud Barrier for Real-time Advanced Vision Analytics. In *HotCloud*.

[153] Yulong Wang, Xiaolu Zhang, Xiaolin Hu, Bo Zhang, and Hang Su. 2020. Dynamic Network Pruning with Interpretable Layerwise Channel Selection. In *AAAI*. 6299–6306.

[154] Zizhao Wang, Wei Bao, Dong Yuan, Liming Ge, Nguyen H. Tran, and Albert Y. Zomaya. 2019. SEE: Scheduling Early Exit for Mobile DNN Inference during Service Outage. In *Proceedings of the 22nd International ACM Conference on*

Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM). 279–288.

[155] 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 (TIP)* (2004).

[156] Zhihao Wang, Jian Chen, and S. Hoi. 2020. Deep Learning for Image Super-resolution: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (2020).

[157] C. Wu, D. Brooks, K. Chen, D. Chen, S. Choudhury, M. Dukhan, K. Hazelwood, E. Isaac, Y. Jia, B. Jia, T. Leyvand, H. Lu, Y. Lu, L. Qiao, B. Reagen, J. Spisak, F. Sun, A. Tulloch, P. Vajda, X. Wang, Y. Wang, B. Wasti, Y. Wu, R. Xian, S. Yoo, and P. Zhang. 2019. Machine Learning at Facebook: Understanding Inference at the Edge. In *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. 331–344.

[158] X. Xiang, Yapeng Tian, Yulun Zhang, Y. Fu, J. Allebach, and Chenliang Xu. 2020. Zooming Slow-Mo: Fast and Accurate One-Stage Space-Time Video Super-Resolution. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).

[159] Mingqing Xiao, Shuxin Zheng, Chang Liu, Yaolong Wang, Di He, Guolin Ke, Jiang Bian, Zhouchen Lin, and Tie-Yan Liu. 2020. Invertible Image Rescaling. In *European Conference on Computer Vision (ECCV)*.

[160] Jingwei Xin, Nannan Wang, Xinrui Jiang, Jie Li, Heng Huang, and Xinbo Gao. 2020. Binarized Neural Network for Single Image Super Resolution. In *European Conference on Computer Vision (ECCV)*.

[161] Qunliang Xing, Mai Xu, Tianyi Li, and Zhenyu Guan. 2020. Early Exit Or Not: Resource-Efficient Blind Quality Enhancement for Compressed Images. In *European Conference on Computer Vision (ECCV)*. Springer.

[162] Jianchao Yang, John Wright, Thomas S. Huang, and Yi Ma. 2010. Image Super-resolution via Sparse Representation. *IEEE Transactions on Image Processing (TIP)* (2010).

[163] Wenming Yang, X. Zhang, Yapeng Tian, W. Wang, Jing-Hao Xue, and Qingmin Liao. 2019. Deep Learning for Single Image Super-Resolution: A Brief Review. *IEEE Transactions on Multimedia (TMM)* (2019).

[164] Hyunho Yeo, Chan Ju Chong, Youngmok Jung, Juncheol Ye, and Dongsu Han. 2020. NEMO: Enabling Neural-enhanced Video Streaming on Commodity Mobile Devices. In *The 26th Annual International Conference on Mobile Computing and Networking (MobiCom)*.

[165] H. Yeo, Sunghyun Do, and D. Han. 2017. How will Deep Learning Change Internet Video Delivery?. In *HotNets*.

[166] H. Yeo, Youngmok Jung, Jaehong Kim, Jinwoo Shin, and D. Han. 2018. Neural Adaptive Content-aware Internet Video Delivery. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*.

[167] Junheon Yi, Seongwon Kim, Joongheon Kim, and Sunghyun Choi. 2020. Supremo: Cloud-Assisted Low-Latency Super-Resolution in Mobile Devices. *IEEE Transactions on Mobile Computing (TMC)* (2020).

[168] Xiaoqi Yin, A. Jindal, V. Sekar, and B. Sinopoli. 2015. A Control-Theoretic Approach for Dynamic Adaptive Video Streaming over HTTP. In *SIGCOMM*.

[169] Hongliang Yu, Dongdong Zheng, Ben Y. Zhao, and Weimin Zheng. 2006. Understanding User Behavior in Large-Scale Video-on-Demand Systems. In *Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems (EuroSys)*. 333–344.

[170] E. Zakharov, Aleksei Ivakhnenko, Aliaksandra Shysheya, and V. Lempitsky. 2020. Fast Bi-layer Neural Synthesis of One-Shot Realistic Head Avatars. In *European Conference on Computer Vision (ECCV)*.

[171] Haoyu Zhang, Ganesh Ananthanarayanan, Peter Bodik, Matthai Philipose, Paramvir Bahl, and Michael J. Freedman. 2017. Live Video Analytics at Scale with Approximation and Delay-tolerance. In *Proceedings of the 14th USENIX Conference on Networked Systems Design and Implementation (NSDI)*. 377–392.

[172] Huanhuan Zhang, Anfu Zhou, Jiamin Lu, Ruoxuan Ma, Yuhan Hu, Cong Li, Xinyu Zhang, Huadong Ma, and Xiaojiang Chen. 2020. OnRL: Improving Mobile Video Telephony via Online Reinforcement Learning. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking (MobiCom)*. Article 29, 14 pages.

[173] Linfeng Zhang, Zhanhong Tan, Jiebo Song, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. 2019. SCAN: A Scalable Neural Networks Framework Towards Compact and Efficient Models. In *Advances in Neural Information Processing Systems (NeurIPS)*. 4027–4036.

[174] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[175] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. 2018. Image Super-Resolution Using Very Deep Residual Channel Attention Networks. In *European Conference on Computer Vision (ECCV)*.

[176] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, J. Shi, and J. Jia. 2018. ICNet for Real-Time Semantic Segmentation on High-Resolution Images. In *European Conference on Computer Vision (ECCV)*.

[177] C. Zhu, X. Lin, and Lap-Pui Chau. 2002. Hexagon-based search pattern for fast block motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)* (2002).

[178] X. Zhu, H. Hu, Stephen Lin, and Jifeng Dai. 2019. Deformable ConvNets V2: More Deformable, Better Results. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019).

[179] Barret Zoph and Quoc V. Le. 2017. Neural Architecture Search with Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*.

[180] Liad Pollak Zuckerman, S. Bagon, Eyal Naor, George Pisha, and M. Irani. 2020. Across Scales & Across Dimensions: Temporal Super-Resolution using Deep Internal Learning. In *European Conference on Computer Vision (ECCV)*.