Internet Video Delivery using Neural Video Codecs

Mallesham Dasari, Samir R. Das
Stony Brook University
{mdasari,samir}@cs.stonybrook.edu

ABSTRACT

Source compression plays a crucial role in reducing the bandwidth needs of Internet video applications. Traditional video compression algorithms have served well in meeting the bandwidth requirements of today's applications. However, these compression algorithms are reaching a saturation point in compression efficiency while hitting the complexity wall in terms of computation and power consumption, creating a major bottleneck to stream next generation 4K/8K/360-degree video streaming and AR/VR applications. In recent years, computer vision and deep learning community has started exploring data-driven neural video coding and shown significant promise for compression opportunities. This work is the first attempt in understanding the implications of replacing the traditional coding algorithms with neural video codecs in the Internet video ecosystem and illustrating its opportunities and challenges.

1 BACKGROUND AND MOTIVATION

Internet traffic is dominated by video applications such as video sharing (e.g., YouTube/TikTok), video conferencing (e.g., Zoom), increasingly popular 360-degree video streaming, and Augmented and Virtual Reality (AR/VR) applications. Internet video delivery has seen a tremendous growth in the last few decades and still growing at rapid rate.

One key driver of these video applications is source video coding. Video coding plays an important role at multiple points across the video delivery path between two or more end-points (e.g., users). At the serving-side, the video is encoded to reduce the bandwidth needs of applications. On the client-side, the video decoded in real-time for playback. For several applications, video coding also plays major role to transcode to different rates, resolutions, and formats at the content distribution networks (CDNs), to adapt the video for different users based on their resource capacity.

Technical gaps in algorithmic codecs. These traditional compression algorithms limit the performance of video applications multiple ways. First, the commonly used video coding methods are standards-driven (e.g., H.264/265 [7]), shipped in hardware for computational efficiency. This creates a bottleneck for faster upgrades of codec features and evolves slowly relative to applications, protocols and CDNs. Second, while these coding methods have been well engineered and thoroughly tuned over the past few decades, the fundamentals

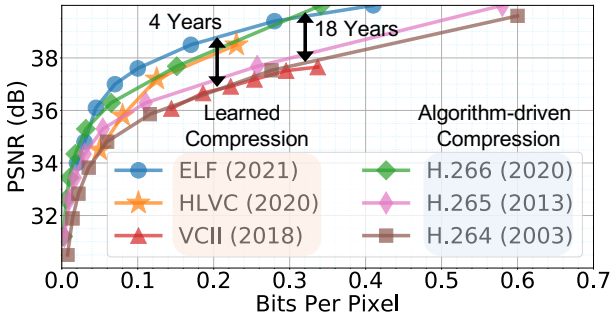

Figure 1: Evolution of neural video codecs and traditional algorithmic codecs.

of coding principles have not changed considerably, witnessing a saturation point in performance improvement. More importantly, the new generation video codecs built on these principles show a very little improvement in compression efficiency while increasing the computational complexity by 6-10×.

Deep neural video codecs. We explore a different, data-driven, deep learning-based approach for video coding for Internet video, that not only envisions a fundamentally new way of coding videos, but also brings radical changes in the entire end-to-end video delivery path addressing the above concerns. Following success in deep learning, the computer vision community has taken a significant interest in this area in recent years [6, 8], showing a tremendous improvement in compression efficient in just a few years. See Figure 1. We find that the traditional coding methods acheive only 2dB in quality improvement for the same bitrate in almost 2 decades, while newer learning methods achieving the same in just 4 years. This trend not only shows great potential to stream future video applications using neural codecs, but also brings opportunities for several other stakeholders in the end-to-end video delivery path (§2).

This work takes a first attempt in answering the following question: *how will advances in deep neural video codecs change Internet video delivery?* We answer this question by illustrating the potential benefits of neural codecs, and illustrate the challenges and its far-reaching implications across the video delivery path.

2 ALGORITHMIC VS. NEURAL CODECS

In the following, we briefly discuss the algorithmic and neural codecs and contrast their design strategies.

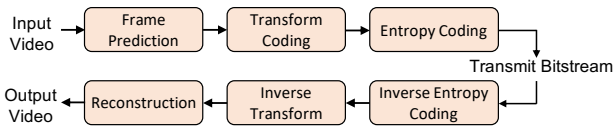

Figure 2: Traditional compression architecture.

2.1 Traditional Video Codecs

The key idea behind traditional algorithmic codecs is that the consecutive frames in video are similar, and so it is useful, for each pixel, to encode its motion vector: that is, where its relative location was in the previous frame. Given these motion vectors, we do not need to encode the color of each pixel— we can simply copy it from the right location in the previous frame, obtaining what is known as the motion-compensated frame. The difference between the motion-compensated frame and the original, called the residual, is often compressed using discrete cosine transform and entropy coding techniques [7]. Figure 2 shows a typical architecture of current generation algorithmic codecs used by media industry.

Limitations. The computational complexity of these codecs mainly comes from the motion estimation process, as the motion vectors are computed for hundreds of thousands of blocks of pixels for each frame by referencing other frames. As the resolution increasing (e.g., 4K/8K), this process becomes more complex and becomes prohibitive to run on commodity devices. Figure 4 shows one such example. Note that the figure only shows decoding; the encoder is much more slower than decoder. Such complexity makes these codecs to be operated in hardware. But the problem with the shipping in hardware is that the design, standardization, development and deployment cycle of these codecs take decades (see Figure 1), making it difficult to meet the demands of applications. This motivates us to revisit the coding fundamentals and explore new ways that scale well for next generation video applications.

2.2 Neural Video Codecs

Neural networks have been applied to learn image/ video coding for a long time [3–5]. While computing was a major bottleneck for using neural networks in the past, the idea is getting increasing traction with recent advances in GPUs/data parallel accelerations and the field has made strides in the last several years.

The key idea behind much of the neural coding is AutoEncoder, originally used for dimentionality reduction using neural networks [5]. Figure 3 shows an example of AutoEncoder, consisting of an encoding part and a decoding part. The encoder converts an input image to a *code* vector that has lower dimension than the input size, and the decoder reconstructs the original image from the low-dimension *code vector*. The neural network weight parameters (W_i for encoder and W_i' for decoder) are trained by minimizing the *reconstruction error*. The smaller the code, the larger the compression factor but higher the reconstruction error. While this is a general idea, the neural network structure consists of a series of CNN and RNNs to capture spatial and temporal correlations across the sequence of images, and the internal design of the codec can be much more complex [6, 8].

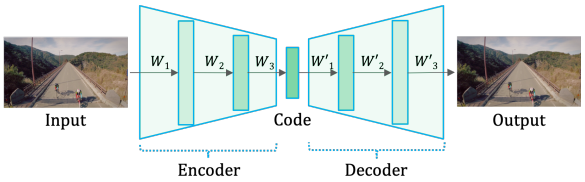

Figure 3: AutoEncoder architecture. The code is a compact version of an input image which is then reconstructed by the decoder from the code.

The advances now has reached to a point where deep learning-based codecs perform on par with standards-based codecs (Figure 1). As seen from the trend, we speculate that the compression efficiency of these codecs will overtake the current known standards soon as the research in this field reaches some level of maturity while bringing in several other benefits. Based on this premise, we take an attempt to answer the questions that may arise in order to replace current coding methods with neural video codecs.

2.3 Advantages of Neural Codecs for Internet Video Delivery

Learning the compression can eliminate the limitations of the conventional coding approaches (as discussed in 1) and brings new benefits as discussed below.

1) Software-defined video coding. Unlike the existing codecs, neural codecs do not need to be baked into a fixed hardware. Thus, it is more easily upgradable, as the models can run efficiently in software (e.g., GPUs/NPUs [1, 2]). Given the growth of ML-based applications, it is expected that this capability will only increase in future. Bringing compression development into software gives the content providers the flexibility to integrate codec features on-demand, supports agile codec development, provides royalty-free codecs, and eliminates cross-platform compatibility issues.

2) Flexible data-driven approach. A key advantage of neural codecs is elimination of a fixed, handcrafted approach to video coding and allowing the learning to automatically optimize the coding for specific video characteristics. For example, it now enables the *compressed codes* to capture motion more effectively or optimize division of code bits between motion and residuals that is not possible using traditional approaches [6]. By exposing to thousands of videos while training, these codecs can extract the prior knowledge of the video content and its inherent features (in terms of W_i' in Figure 3).

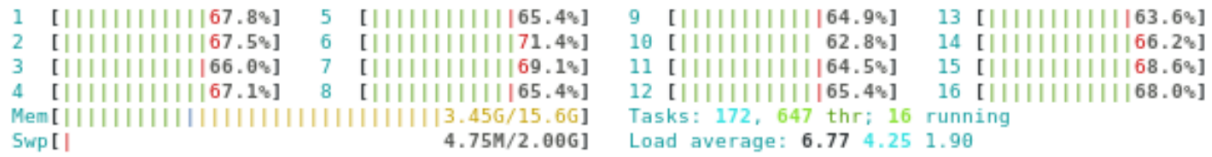

Figure 4: Computational complexity of H.264 decoding a 8K video in a Chrome browser on an Intel i9-9900K CPU with 3.60GHz and 16 cores. Even with 800% CPU usage, Chrome is not able to render the video.

2.4 Challenges of Neural Video Codecs

Despite the compression benefits, realizing the neural video codecs in practice encounters several challenges across the video delivery path. We discuss a few of them below.

1) Generalizability concerns. A key problem with deep learning solutions in general is the performance on unseen datasets. In this case, the neural codecs are trained well on hundreds of thousands of videos and architecting the models to be as resilient as possible when used for unseen video contents. However, unlike traditional algorithmic solutions, it difficult to diagnose the compression performance if it produces compression artifacts on unseen video content. For storing the videos, the encoder and decoder can be adapted (i.e., re-learning the weights) on-demand, but in a streaming scenario, the decoder is shipped to client offline, and impractical adapt the codec on-demand. One solution to address this challenge is an incremental upgrades to the decoder offline (e.g., Netflix can update the decoder in the app, whenever a new movie uploaded to their CDN), but needs further exploration for fresh unseen content streaming. Another solution can be to enable a hybrid algorithmic and data-driven approach to exploit the efficiency and robustness strengths of each methods.

2) Application/End-point related challenges. Video compression is used by plethora of Internet applications. Notable applications include: 1) video on-demand streaming (e.g., Netflix), where the content is transcoded offline and users are served with the content on-demand, 2) video conferencing (e.g., Zoom) and live broadcasters (e.g., Twitch), where a live video feed exchanged multiple users. A practical challenge with such applications is the heterogeneity of end-point devices ranging from high end TVs, Desktops, Laptops, Tablets, to diverse mobile phones. Training a single codec (like the hardware traditional codec) for such diverse environments may not lead to optimal performance and also encounters scalability issues in terms of both encoding and decoding latency. Moreover, the neural codecs need to be run power-hungry accelerators such as GPUs which consume significant power. Figure 5 shows an example of a neural codec [8] run on Jetson platform in real-time. The compression models need to be auto-scaled to meet the dynamic requirements of power sensitive mobile devices.

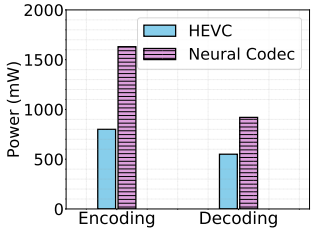

Figure 5: Power consumption of traditional vs. neural codec on Nvidia Jetson TX2 (a mobile-grade GPU).

3) Transcoding challenges on CDNs. Video coding also plays an important role in adapting the content to diverse users, networks, resolutions, formats, platforms etc by transcoding each video into thousands of video files on CDNs (e.g., Netflix encodes each video into ≈1200 files to support its diverse customers). Using hardware accelerators to such extensive transcoding is not cost-effective and also do not scale well across multiple data-centers. Instead, current media industry uses server-grade multi-core CPUs with thousands of threads for parallel transcoding. However, CPUs are extremely inefficient to run neural codecs, requiring significant changes to computing capacity on video servers. The servers need to be equipped with co-processor clusters of GPUs, DSPs, and other neural accelerators. While some cloud operators already offer such accelerators as a service (e.g., Google cloud GPUs), transcoding at warehouse-scale video data requires a clean-slate change to computing requirements.

REFERENCES

[1] Apple a14 processor. https://www.cultofmac.com/640134/tsmc-apple-a14-2020-iphone-12/, 2019.
[2] Qualcomm snapdradon. https://www.androidauthority.com/qualcomm-snapdragon-mobile-npu-896223/, 2019.
[3] C. e. Cramer. Low bit-rate video compression with neural networks and temporal subsampling. *IEEE*, 84(10):1529–1543, 1996.
[4] M. Dasari, A. Bhattacharya, S. Vargas, P. Sahu, A. Balasubramanian, and S. R. Das. Streaming 360-degree videos using super-resolution. *IEEE INFOCOM*, 2020.
[5] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.
[6] O. Rippel and et.al. Elf-vc: Efficient learned flexible-rate video coding. *arXiv preprint arXiv:2104.14335*, 2021.
[7] G. J. Sullivan and et.al. Overview of the high efficiency video coding (hevc) standard. *IEEE TCSVT*, 22(12):1649–1668, 2012.
[8] C.-Y. Wu, N. Singhal, and P. Krahenbuhl. Video compression through image interpolation. In *ECCV*, pages 416–431, 2018.