MLVC: A Multi-platform Learned Video Codec for Real-World Deployment

Tanel Pärnamaa, Martin Lumiste, Ardi Loot, Evgenii Indenbom, Andrei Znobishchev, and Ando Saabas

Microsoft Corporation
{taparnam, ardiloot, eindenbom, aznobishchev, ansaaba}@microsoft.com

Abstract. Neural video codecs have surpassed classical codecs in coding efficiency but remain impractical for deployment due to cross-platform incompatibility and high computational cost. Existing quantization-based solutions fail to produce deterministic results across diverse hardware platforms, leading to catastrophic decoding failures. We introduce MLVC, a hardware-robust neural video codec designed for practical cross-platform inference. The key idea is to explicitly transmit scale parameters through the hyperprior, which guarantees entropy coding consistency across devices without requiring bit-exact arithmetic. While this increases bitrate overhead, we recover most of the coding efficiency through architectural improvements (gated memory, ReGLU activation), a long-term reference recovery mechanism, and domain-specific perceptual training. On the VCD video conferencing benchmark, MLVC achieves >70% BD-rate (MOS) improvement over hardware HEVC, the strongest deployable baseline, while reaching subjective quality competitive with DCVC-RT, which cannot operate across diverse platforms. Both the encoder and decoder run at 100 FPS on average on commodity NPUs from Apple, Intel, and Qualcomm. MLVC is the first neural video codec to combine competitive compression performance, real-time speed, and cross-platform robustness across diverse consumer devices, making it suitable for widespread deployment. Code will be released.

Keywords: Neural Video Compression · Cross-Platform Robustness · Real-Time Video Processing

1 Introduction

Neural video compression has matured to the point where learned codecs consistently outperform traditional standards. [17, 24, 34, 38] Recent neural codecs achieve 60-70% bitrate savings over H.265 at equivalent perceptual quality and surpass even the latest ECM standard in rate-distortion performance. [24, 34] However, despite all this research progress, neural codecs remain unused in production systems like Zoom, Teams or WebRTC.

Two critical barriers prevent deployment. The first is computational efficiency: most neural codecs are evaluated on high-end datacenter GPUs. For practical impact in applications like video conferencing, neural codecs need to

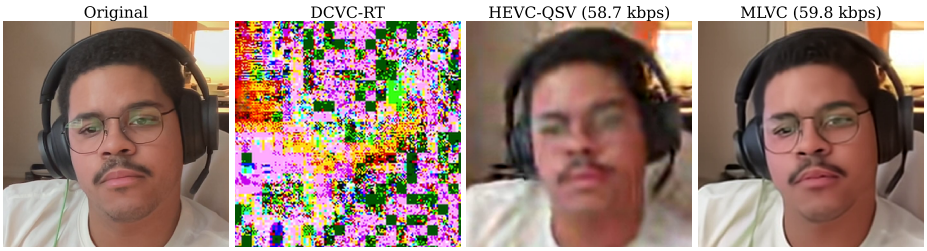


Fig. 1: Neural codecs fail across diverse platforms. A video is encoded on an Apple M3 NPU and then decoded on another device (Intel Lunar Lake NPU). Existing state-of-the-art neural video codecs (e.g., DCVC-RT [24]) suffer from catastrophic decoding failures due to numerical divergence between platforms, even when using quantization or calibration information [50]. Classical codecs like H.265 and the proposed neural codec MLVC both decode reliably across diverse platforms, while MLVC achieves significantly higher visual quality at the same bitrate.

run in real-time on the neural processing units (NPUs) present in commodity consumer devices.

The second and more fundamental barrier is cross-platform compatibility. Many applications, such as video conferencing systems, inherently require encoder and decoder to run on different devices, often with hardware from different vendors. Figure 1 illustrates the severity of this problem: when we encode a video on an Apple M3 NPU and decode on an Intel NPU, the state-of-the-art neural codec DCVC-RT [24], even in its quantized form, produces completely corrupted output.

To address these barriers, we introduce MLVC, a neural video codec built for real world deployment. We specifically focus on model performance on NPUs rather than high-performance GPUs to show the practicality of our approach. Our contributions are as follows:

- We develop a scale sharing mechanism and transmit scale indices within the hyperprior, enabling reliable inference across diverse hardware platforms without requiring bit-exact arithmetic.
- We introduce architectural improvements (gated memory for long-term modeling, hardware-compatible ReGLU activations), training and inference-time strategies (I-frame dropout, long-term reference recovery) that recover the coding efficiency lost to increased bitrate overhead, achieving >70% BD-rate improvement over HEVC-QSV in subjective quality (MOS) on a video conferencing benchmark (VCD).
- We demonstrate real-time performance (100+ FPS average for both encoding and decoding) on commodity NPUs from Apple, Intel, and Qualcomm, making MLVC the first neural video codec practical for widespread deployment on diverse consumer devices.

2 Related Work

2.1 Learned Compression

The hyperprior architecture introduced by Ballé et al. [11] forms the foundation of modern learned compression [13, 19, 34]. This is commonly combined with autoregressive prior [40]. Since fully autoregressive priors are computationally prohibitive, recent work explores efficient approximations including checkerboard patterns [20], channel-wise autoregression [41] and hybrid spatial-channel models [32]. In learned video compression, numerous architectures since [36] have advanced the field by improving motion modeling, temporal hierarchy, and entropy estimation [8, 22, 35, 44, 47, 53]. Video models often employ a third *temporal* prior to exploit inter-frame correlation. Li et al. [31] showed that contextual coding outperforms explicit residual coding, with subsequent advances often related to better context modelling [33, 46]. The current state-of-the-art models [24, 34] achieve improved rate-distortion performance over the reference software codec for ECM [2].

2.2 Mobile and Low-Latency Codecs

While most learned compression research prioritizes improving rate-distortion performance, a parallel direction focuses on computational efficiency for edge development. Codecs based on overfitting such as C3 [26] and Cool-chic video [30] achieve very low decoding complexities, but require costly encoding. MobileCodec [29] achieves 720p video decoding on a mobile phone with a neural accelerator. MobileNVC [51] extends this to 1080p video. DCVC-RT achieves over an 18× FPS speed-up compared to DCVC-FM [34] on NVIDIA A100 GPUs, though edge device performance is not reported.

These works employ two main strategies for efficiency. First, integer quantization to improve inference speed and also provide cross-platform consistency through bit-exact arithmetic. However, cross-platform validation is limited: MobileNVC [51] reports no cross-platform testing, and DCVC-RT [24] validates only across NVIDIA GPUs. Since NVIDIA GPUs allow for low level kernel control, their validation demonstrates only single-vendor and not true cross-platform compatibility. We will argue that these models are unlikely to work across heterogeneous NPU platforms.

Secondly, one of the main performance bottlenecks on edge devices has been the expensive pixel or feature space warping operator. As a result, most mobile methods redesign motion compensation to either use convolutions [24, 29], or simplify the warp operation [51].

2.3 Cross-platform Consistency

The cross-platform consistency problem arises because entropy coding requires encoder and decoder to use identical probability distributions. In the case of DCVC-RT, the scale parameters $\boldsymbol{\sigma}$ characterize these distributions. Even tiny

numerical differences, for example those caused by non-deterministic floating-point computations, can cause the entropy decoder to select different lookup-table indices, resulting in incorrect latent values. These errors cascade through temporal prediction, causing catastrophic failures (Fig. 1). We refer readers to Tian et al. [50] for detailed exposition. Prior work addresses this through three main approaches, each with fundamental limitations.

Network Quantization. The most common approach quantizes entropy model computations to integer arithmetic, aiming for bit-exact results across platforms [10,15,18,24,27,48]. In theory, the subnetwork responsible for producing $\boldsymbol{\sigma}$ could run in fixed-point integer arithmetic to guarantee cross-platform consistency, as traditional codecs do. In practice, this is difficult to achieve on commodity NPUs. First, many works adopt nonstandard bit-widths such as INT16 [24, 27, 28] and custom quantization scales that the NPU toolchains don't support – attempting to convert such models typically results in a compiler error, preventing acceleration entirely. Second, even for the relatively standard INT8 convolution with INT32 accumulators, some NPUs (e.g. Apple NE) do not execute true INT8 but simulate it via FP16.[1] Third, across NPUs that do implement true INT8, compiler choices (kernel selection, operator fusion, reduction order), approximations, and rounding modes prevent bit-exact parity. Unlike with traditional video codecs, there is no bit-exact reference specification for NPU vendors and quantized models can diverge more than analogous FP models due to compounding off-by-one errors. For these reasons, we use FP16 for inference, which is broadly supported across NPUs. We present detailed cross-platform divergence measurements for both floating point and integer layers in the Supplementary Material.

Calibration Information. Tian et al. [50] transmit an additional side stream of *calibration information* to decrease the likelihood of cross-platform index mismatches. While this reduces mismatch probability, it cannot provide hard guarantees - catastrophic failures remain possible, just less frequent. This probabilistic approach is insufficient for production deployment where reliability is critical. Moreover, their experiments relied only on a single 96-frame video from the UVG dataset, encoded and decoded in FP32 precision. Neural processing units typically use FP16 rather than FP32. Since FP16 has a unit roundoff about four orders of magnitude larger, it is far more susceptible to rounding-error drift. In our cross-device experiments (see supplementary material), the method fails in FP16: we observe float-index deviations $|\delta| > 0.5$, a situation where calibration cannot prevent index mismatches.

Fixed-prior Approaches. More recently, Tian et al. [49] proposed to transmit discrete codebook indices, eliminating entropy modelling entirely. This effortlessly avoids catastrophic cross-platform failures. However, their experimental results do not effectively validate competitive compression performance. Most critically, their comparison uses mismatched GOP settings: GOP=32 for their

[1] Vendor documentation indicates the int8-int8 compute path is only enabled starting from M4, released in 2024. `https://apple.github.io/coremltools/docs-guides/source/opt-overview.html`

model versus GOP=12 for H.264/H.265, giving their method an unfair advantage since classical codecs must encode expensive I-frames more frequently.

3 Proposed Method

We use the DCVC-RT [24] floating point model as a starting point, as it shows state-of-the-art compression ratio, while remaining lightweight and fast. An overview of the video coding framework is shown in Fig. 2. While our experiments focus on DCVC-RT, the proposed methods rely on components common to most modern video codec. The scale-sending mechanism applies to any codec using adaptive entropy priors (e.g. hyperprior, autoregressive prior, temporal prior), the memory module to any recurrent codec maintaining temporal state, and the LTR mechanism to codecs that use reference features.

3.1 Avoiding Catastrophic Failures

Catastrophic decoding failures in existing neural video codecs are caused by mismatches in entropy model parameters between the encoder and decoder, specifically the scale parameters in DCVC-RT. While eliminating all adaptive priors would trivially eliminate catastrophic decoding failures, this significantly degrades BD-rate performance in practice (see row *0-prior* in Table 1). The alternative, transmitting all scale parameters explicitly, incurs prohibitive overhead, since there are $\frac{H}{16} \times \frac{W}{16} \times C_y$ scale parameters, comparable in magnitude to the encoded frame itself. However, scale parameters exhibit strong spatial and cross-channel correlation that can be exploited for compression. We reduce the parameter count by a factor $s^2 \cdot r$ (typically 128×) through structured parameter sharing, then entropy-code the remaining parameters within the hyperlatent representation. This achieves entropy coding consistency with acceptable bitrate overhead.

Scale Sending Mechanism. We derive scale indices $\boldsymbol{I} \in \mathbb{N}_0^{C_y \times H_y \times W_y}$ from the quantized hyperlatent $\hat{\mathbf{z}} \in \mathbb{Z}^{C_z \times H_z \times W_z}$ through deterministic expansion. Since $\hat{\mathbf{z}}$ is coded with a fixed factorized entropy model, it is identical on both the encoder and decoder sides, guaranteeing that the resulting scale parameters are also identical and thus avoiding catastrophic decoding failures. This design exploits two observations: (1) neighboring spatial locations often require similar scales, and (2) groups of channels often share statistical properties.

Let $r \in \mathbb{N}$ denote the channel repetition factor and $s = H_y/H_z$ the spatial expansion factor. The generation process comprises three steps:

$$\boldsymbol{I}^{\text{base}} = |\hat{\mathbf{z}}_{1:C_y/r}| \in \mathbb{N}_0^{C_y/r \times H_z \times W_z} \tag{1}$$

$$\boldsymbol{I}_{c,h,w} = \boldsymbol{I}^{\text{base}}_{\lfloor c/r \rfloor, \lfloor h/s \rfloor, \lfloor w/s \rfloor} \tag{2}$$

$$\boldsymbol{\sigma}_{c,h,w} = \text{lookup}(\boldsymbol{I}_{c,h,w}) \tag{3}$$

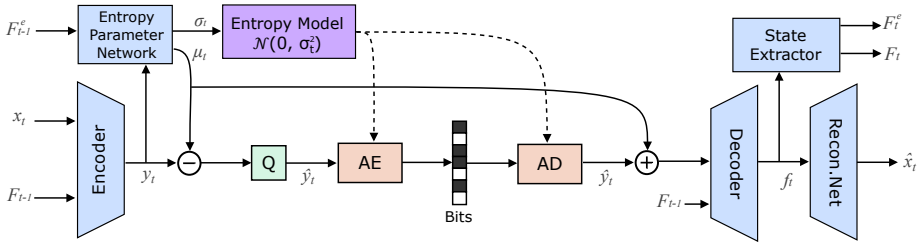


Fig. 2: Overview of the video coding framework adopted in this work, following modern learned codecs (e.g., DCVC-RT). The encoder maps the current frame x_t and temporal context F_{t-1} to the latent representation y_t. The Entropy Parameter Network, conditioned on y_t and the temporal prior context F^e_{t-1}, produces the mean μ_t and scale σ_t parameters. The mean is subtracted from y_t, and the residual is quantized (Q) and arithmetic encoded (AE) under a zero-mean Gaussian entropy model $\mathcal{N}(0, \sigma_t^2)$ to obtain the bitstream. On the decoder side, arithmetic decoding (AD) recovers $\hat{y}_t$, and the mean μ_t is added back. The decoder, further conditioned on temporal context F_{t-1}, produces the decoded feature f_t, from which the reconstruction network (Recon. Net) yields the reconstructed frame $\hat{x}_t$. Finally, the state extractor derives the updated temporal contexts F_t and F^e_t from f_t for the next time step.

where $c \in [0, C_y)$, $h \in [0, H_y)$, $w \in [0, W_y)$. The absolute value ensures non-negative indices, the expansion maps each hyperlatent position to an $s \times s$ spatial block and replicates each channel r times, and the final lookup retrieves quantized scale values from a fixed codebook. Notably, the same scale parameters are shared across all groups in the autoregressive prior, so all groups can be arithmetic-decoded in a single pass without interleaving arithmetic decoding and neural network calls. Figure 3 shows the standard entropy model of DCVC-RT, Fig. 4 shows the proposed scale-sending variant, and Fig. 5a details the individual steps.

3.2 Reducing Divergence

While our scale-sending approach eliminates catastrophic entropy decoding failures, floating-point differences still cause gradual divergence in the features and reconstructed frames. Since encoder and decoder maintain separate feature buffers that are updated recursively, even small per-frame differences compound over long prediction chains. Without a synchronization mechanism, the divergence can grow arbitrarily large. We address this through three strategies: a novel prediction chain control via long-term reference recovery inspired by traditional codecs, hardware-compatible activation functions, and periodic synchronization via I-frames.

Long-Term Reference Recovery. The most straightforward method for mitigating divergence over long prediction chains is to reduce the chain length by employing a small I-frame period (e.g., 64 frames). Although this approach can effectively

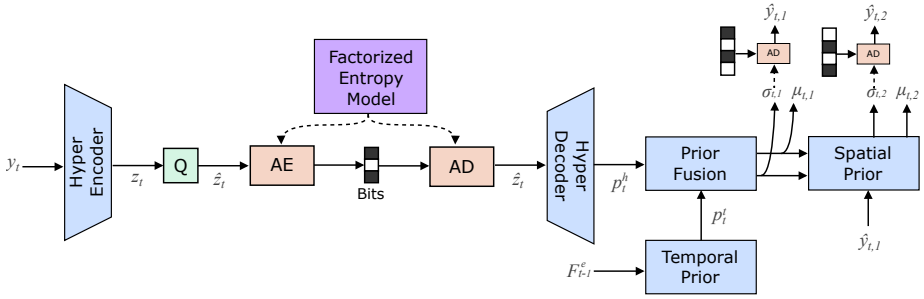


Fig. 3: Entropy Parameter Network in modern learned codecs (e.g., DCVC-RT) incorporating hyper, temporal and spatial prior. The hyper encoder transforms y_t into z_t, which is quantized and arithmetic coded using a factorized entropy model to yield $\hat{z}_t$. The hyper decoder then produces the hyperprior p_t^h. Because the factorized entropy model is identical and fixed on both the encoder and decoder sides, coding of $\hat{z}_t$ does not cause catastrophic decoding failures. The temporal prior maps the temporal context F_{t-1}^e to p_t^t, and the Prior Fusion module merges p_t^h and p_t^t to produce the entropy parameters $(\sigma_{t,1}, \mu_{t,1})$ used to arithmetic code the first latent group $\hat{y}_{t,1}$. The Spatial Prior then conditions autoregressively on the already decoded $\hat{y}_{t,1}$ to produce $(\sigma_{t,2}, \mu_{t,2})$ for the second group $\hat{y}_{t,2}$. However, the $\sigma_{t,i}$ can diverge between the encoding and decoding side, leading to catastrophic decoding failures.

address divergence, it incurs a substantial cost in terms of rate-distortion performance due to the use of expensive intra frames. As an alternative, a less costly mechanism is proposed, inspired by long-term reference (LTR) recovery techniques in traditional codecs. Figure 5b illustrates the core concept of this approach: rather than relying exclusively on I-frames, proactive LTR recovery frames are periodically introduced to manage the prediction chain. This strategy reduces the frequency of expensive I-frames while maintaining the same number of frames in the prediction chain, resulting in an overall improvement in BD-rate.

Beyond reducing the prediction chain, proactive LTR recovery also preserves its original advantage of enhancing codec robustness against packet loss by providing periodic recovery frames without explicit requests, as in traditional codecs. It should be noted that, on a single platform with a fixed intra period, the LTR mechanism would strictly regress BD-rate due to the introduction of additional redundancy. However, in cross-platform scenarios, this reduction in prediction chain length can improve compression efficiency, as reference divergences may otherwise lead to significant quality degradation in the absence of LTR.

Hardware-Compatible Activations. Many NPUs implement nonlinear activation functions through piecewise approximations rather than exact computation. Since these approximations are vendor-specific and unstandardized, even small differences compound through the deep network, causing cross-platform divergence. Furthermore, complex activations such as WSiLU often lack optimized NPU kernels, resulting in reduced inference speed. We perform a sweep over

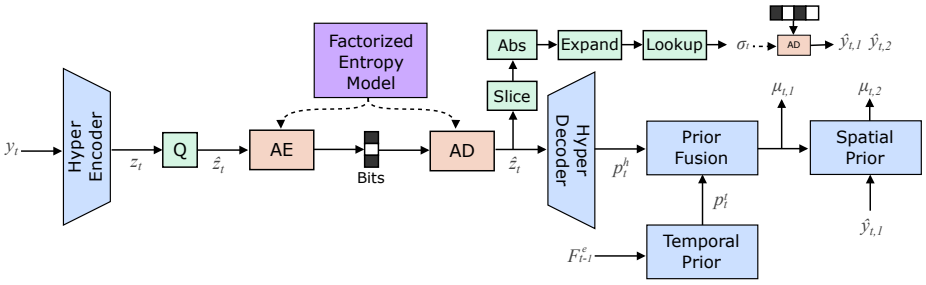


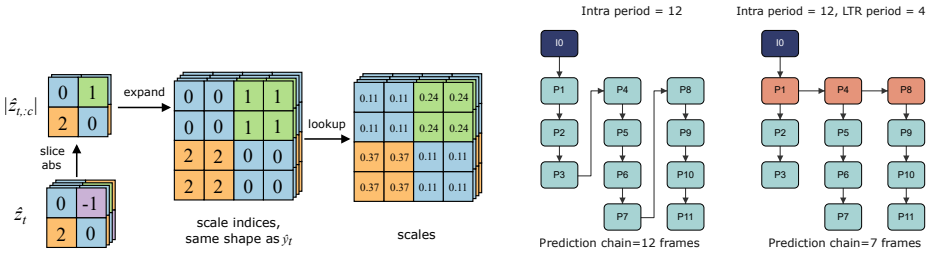
Fig. 4: Entropy Parameter Network using the proposed scale-sending mechanism. A slice of the hyper decoder output is passed through Abs, Expand, and Lookup operations (detailed in Fig. 5a) to deterministically obtain the scale parameters σ_t from the hyperlatent $\hat{z}_t$. Since $\hat{z}_t$ is coded with a fixed factorized entropy model, σ_t is identical on both the encoder and decoder sides, avoiding catastrophic decoding failures. The mean parameters $\mu_{t,i}$ are still modeled autoregressively: Prior Fusion merges the hyper prior p_t^h and temporal prior p_t^t to produce $\mu_{t,1}$, and the Spatial Prior conditions on the already-decoded $\hat{y}_{t,1}$ to produce $\mu_{t,2}$. A further benefit over the standard approach (Fig. 3) is decoding speed: because σ_t is available before decoding any latent group, all groups can be arithmetic-decoded in a single AD pass. In contrast, the standard approach requires interleaving arithmetic decoding and neural network calls for each group.

common activation functions supported by Apple Neural Engine [6] and find that only ReLU and LeakyReLU incur no error compared to the true value, we therefore restrict our architecture to use these. While simpler activations incur a BD-rate penalty, they are necessary for reliable cross-platform operation.

Regular I-Frame Period. Even with LTR recovery and simple activations, feature buffers still diverge over extended sequences due to fundamental floating-point precision limits. We therefore employ periodic I-frames to fully resynchronize encoder and decoder feature states. While our model is able to handle very long prediction chains on a single platform, regular I-frames are necessary for practical deployment across multiple hardware platforms. This periodic synchronization incurs BD-rate overhead compared to single-platform operation but remains necessary for reliable cross-platform decoding. This differs from traditional codecs, which could run cross-platform without periodic I-frames.

3.3 Unified I-Frame and P-Frame Model

Most learned video codecs employ separate models for I-frames and P-frames, even though these architectures share substantial structural overlap. Maintaining two distinct models increases both storage requirements and deployment complexity. For instance, in DCVC-RT [24], the I-frame and P-frame models require 174 MB and 79 MB respectively in FP32. To reduce this overhead, we simplify to a single unified model by defining I-frames as P-frames with a uniform gray



(a) Scale sending architecture. **(b)** LTR prediction structure.

Fig. 5: (a) A visualization of the steps of the scale-sending mechanism described in Sec. 3.1 (b) Comparison of prediction structures with and without long-term reference (LTR) frames. (b, left) Standard encoding with I-frame period of 12 creates prediction chains up to 12 frames long. (b, right) Using LTR period of 4 (starting from frame 1) reduces maximum chain length to 7 frames and improves robustness to packet loss. In practice, LTR period is another inference parameter, similar to intra and reset period.

reference image (0.5 in YUV colorspace). This eliminates the need for a separate I-frame model while maintaining the ability to encode frames without temporal dependencies. Although this simplification introduces a BD-rate degradation, it reduces engineering and deployment complexity. Given that practical systems prioritize simplicity and maintainability, this trade-off is well justified.

3.4 Improving Quality

Memory. While DCVC-RT employs a recurrent architecture with features serving as temporal state, this representation has limited capacity for preserving long-term information, which is important for reconstructing temporarily occluded objects or maintaining consistency across frames.

Drawing inspiration from gated recurrent architectures [21], we augment the decoder with an explicit long-term memory state that enables the model to selectively retain and retrieve information over longer temporal horizons. The memory-enhanced decoder operates as follows:

$$\begin{aligned} \mathbf{f}_{in} &= \text{Decoder}(\hat{\mathbf{y}}_t, \mathbf{F}_{t-1}) \\ [\mathbf{s}_t, \mathbf{g}_f, \mathbf{g}_o] &= \text{Split}_3(\mathbf{f}_{in}) \\ \mathbf{m}_t &= \sigma(\mathbf{g}_f) \odot \mathbf{m}_{t-1} + (1 - \sigma(\mathbf{g}_f)) \odot \tau(\mathbf{s}_t) \\ \mathbf{f}_{out} &= \sigma(\mathbf{g}_o) \odot \tau(\mathbf{m}_t) \cdot \mathbf{q}_{dec} \end{aligned} \tag{4}$$

where $\mathbf{F}_{t-1}$ is the temporal context, $\mathbf{q}_{dec}$ is a learnable decoder quantization step, $\mathbf{m}_t$ is the memory state, $\sigma(\cdot)$ and $\tau(\cdot)$ denote the gate and memory activation functions, and $\odot$ denotes element-wise multiplication. The forget gate $\mathbf{g}_f$ controls information retention from the previous memory state, while the output gate $\mathbf{g}_o$ modulates the contribution to the output features.

Compared to the baseline decoder, the memory variant only triples the output channels of the final 1×1 convolution layer. The activation functions can be

standard (σ=sigmoid, τ=tanh) or piecewise-linear approximations using ReLU for hardware-friendly deployment.

Concurrent to our work, a recent study also introduced a memory mechanism for neural video compression [17]. In contrast, our design focuses on real-time, multi-platform deployment, emphasizing hardware compatibility and temporal robustness.

ReGLU. The model performance is highly sensitive to activation function choice. Moreover, optimal activation functions differ across architectures. While cross-platform requirements restrict us to simple functions (ReLU, LeakyReLU), we can improve expressiveness through gating mechanisms that use only cross-platform-compatible operations. Inspired by [45], we find that a channel split gating variant of ReGLU improves performance for the MLVC architecture without increasing encoder–decoder divergence:

$$\mathrm{ReGLU}(\mathbf{x}) = \mathbf{x}_{:C/2} \odot \mathrm{ReLU1}(\mathbf{x}_{C/2:}) \tag{5}$$

where $\mathbf{x}_{:C/2}$ and $\mathbf{x}_{C/2:}$ denote the first and second halves of channels, and $\mathrm{ReLU1}(\cdot) = \min(\mathrm{ReLU}(\cdot), 1)$ caps activations at 1. The capping stabilizes training by preventing large activations from the multiplicative gating.

We apply ReGLU in DCVC-RT's depth-wise convolution blocks, replacing the channel-wise addition with multiplicative gating, while using LeakyReLU elsewhere.

I-Frame Dropout. To train a single model to handle both standard P-frames and I-frames without requiring a separate intra codec, we introduce I-frame dropout during training: with probability p, the model receives a constant gray frame in place of a true I-frame. We use $p = 0.5$ in our experiments. The effectiveness of this approach depends on the I-frame period: shorter periods benefit more, as I-frame conditions are encountered more frequently during inference.

Perceptual Losses. Cross-platform architectural constraints create a cumulative BD-rate penalty, and even with the above improvements, a gap remains compared to unconstrained architectures when measured by PSNR. However, PSNR does not reflect subjective quality. For real-world applications, perceptual optimization can matter more than pixel-wise accuracy. Moreover, different applications have different perceptual priorities - video conferencing prioritizes facial detail, while screen sharing prioritizes text clarity. We can narrow the perceptual quality gap by tailoring the codec to specific use cases through perceptual losses and region-of-interest (ROI) weighting. Our subjective evaluation demonstrates that by adapting the codec to video conferencing, we achieve comparable results to reference DCVC-RT checkpoint despite lower PSNR.

Specifically, we use a perceptual loss with two components. First, we use the common LPIPS [54] loss to improve texture retention. In addition, inspired by the success of [25, 37] in the CLIC [1] challenge, we use a region of interest (ROI) mask to weight the loss pixels. We use a face-segmentation model [14, 55]

Table 1: Ablation study results showing PSNR-based BD-rate (%) in YUV420 colorspace relative to HEVC-QSV anchor in same- and cross-platform scenarios.

Arch	I	IP	RP	Act	SS	Mem	I-D	LTR	Avg (SP)	TH (SP)	TH (XP)
DCVC-RT	✓	-1	64	WSiLU	×	×	×	×	-69.6	-	∞
DCVC-RT	×	-1	64	WSiLU	×	×	×	×	-68.6	-	∞
DCVC-RT	×	64	0	WSiLU	×	×	×	×	-59.4	-55.9	∞
0-prior	×	-1	64	WSiLU	×	×	×	×	20.5	9.6	718.1
MLVC	×	-1	64	WSiLU	✓	×	×	×	-54.5	-56.1	101.8
MLVC	×	-1	64	LReLU	✓	×	×	×	-49.2	-53.5	6.7
MLVC	×	-1	64	LReLU	✓	✓	×	×	-52.1	-58.4	-21.9
MLVC	×	-1	64	ReGLU	✓	✓	×	×	-56.6	-61.9	-39.8
MLVC	×	64	0	ReGLU	✓	✓	×	×	-41.1	-42.7	-41.0
MLVC	×	64	0	ReGLU	✓	✓	✓	×	-46.1	-46.7	-44.9
MLVC	×	1024	0	ReGLU	✓	✓	✓	✓	-52.0	-60.2	**-58.7**
MLVC-S	×	1024	0	ReGLU	✓	✓	✓	✓	-36.5	-48.6	-47.3

Legend: I: separate I-frame model, IP: intra period (−1: single intra frame), RP: reset period (0: no reset), SS: scale sending, Mem: memory module, I-D: I-frame dropout, LTR: long-term reference. SP: same-platform, XP: cross-platform, Avg: average BD-rate over HEVC B-E and 720p VCD sequences (NVIDIA GPU), TH: VCD TH subset at 360p (Apple M3), TH (SP): NPU encoder, NPU decoder, TH (XP): NPU encoder, GPU decoder, 0-prior: no adaptive priors, MLVC-S: small variant of MLVC.

to derive the ROI mask during training. We do not explicitly modulate latents with the mask like [37], therefore avoiding dependence on an external model in deployment. We also refrain from using a GAN loss as in [9, 17, 39], since it can lead to temporal flickering and may introduce artifacts that perturb identity.

4 Experiments

4.1 Experimental Setup

Datasets. We employ the same training schedule as DCVC-RT, with Vimeo-90K [52] septuplets used for first stage training, and longer Vimeo sequences up to 64 frames used for fine-tuning the model. To assess the real-world deployability of neural codecs in video conferencing applications, we conduct a subjective test on the Video Conferencing Dataset (VCD) [43]. Objective metric results are also presented on the more common HEVC B-E [16] datasets.

Training. We use the same variable rate control method as [24] and train the model for the YUV colorspace. When training on long sequences, we use gradient and frame gradient clipping for improved stability. Additionally, we use gradient checkpointing in the fine-tuning stage. Training with sequences of up to 32 frames takes 5 days on 8 NVIDIA Tesla V100s. For perceptual models, we train the first stage using standard PSNR loss, but switch to the perceptual loss in fine-tuning.

Evaluation Metrics We measure the compression quality using BD-rate [12], computed from the actual encoded bitrate rather than the rate estimate from the entropy model. The distortion metric is PSNR in YUV420 colorspace or mean opinion score (MOS) in the subjective evaluation. Additional metrics are provided in the supplementary material. We primarily target 360p to 720p, as these resolutions are typical in real-time video conferencing. For the HEVC datasets, metrics are reported at native resolution.

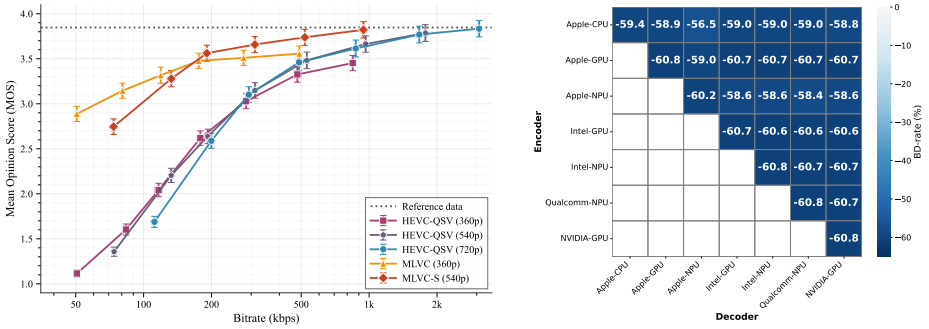
Cross-Platform Evaluation We report results in two settings: same-platform (SP), where encoder and decoder run on the same platform, and cross-platform (XP), where they run on different platforms (e.g., GPU encoder, NPU decoder). When cross-platform mismatch causes catastrophic decoding failure, we denote the BD-rate as ∞.

Anchor. We use Intel Quick Sync Video (HEVC-QSV) [5] as the single anchor for all BD-rate calculations throughout the paper, as it represents a widely deployed hardware codec comparable to what neural codecs must match in practice. More details on the anchor configuration can be found in the supplementary material. While we provide rate-distortion results for the reference implementation HM [4] in the appendix, we omit VTM [7] and ECM [2], as they lack commodity hardware implementations. We do compare against DCVC-RT, which outperforms HM, VTM, and ECM in PSNR.

P.910 Testing. We use the P.910 software package [42] to conduct subjective evaluations, running a 5-point ACR test as defined in ITU-T Recommendation P.910 [23]. We collect 15 votes for each encoded clip in the VCD-TH subset. All videos are displayed at 720p resolution to raters. For lower processing resolutions (360p, 540p), the input is downscaled before encoding and upscaled back to 720p after decoding using ffmpeg's [3] Lanczos algorithm.

4.2 Experimental Results

Ablation Study. Table 1 presents an ablation study of architectural changes from DCVC-RT to MLVC. The original DCVC-RT achieves -69.6% average BD-rate in the same-platform scenario, while running at 102 FPS at 360p resolution (Table 2). Our full model reaches -52% BD-rate at 130 FPS at 360p, a gap of roughly 18 percentage points. This gap is the direct cost of cross-platform constraints: scale sending, hardware-compatible activations, periodic I-frames, and a unified I/P-frame model each contribute (Tab. 1). Importantly, DCVC-RT produces catastrophic decoding failures on heterogeneous hardware and thus cannot be deployed across diverse platforms (BD-rate = ∞ in the XP column). Among neural codecs that successfully decode across platforms, MLVC achieves the best BD-rate by a large margin (-45.2% vs. $+15.0\%$ for [49] on HEVC B, see supplementary material for details). LTR frames enable operation at much larger intra periods (IP=1024 vs. IP=64), providing additional BD-rate gains and improving robustness to packet loss. For higher resolutions, MLVC-S achieves 1080p encoding at 30 FPS while still reaching -37% BD-rate against the anchor.



(a) Rate–MOS curves. **(b)** Cross-platform BD-rate matrix.

Fig. 6: (a) Rate-MOS curves on VCD-TH, comparing hardware H.265 with the MLVC models at multiple processing resolutions. MLVC-S is a smaller model tuned for 150 FPS at 540p. (b) Cross-platform BD-rate on VCD-TH 360p relative to HEVC-QSV. Rows indicate encoder platform, columns indicate decoder platform. Diagonal entries are same-platform; off-diagonal entries are cross-platform. Results demonstrate cross-platform compatibility.

Subjective Test. We conduct a P.910 [42] subjective evaluation on VCD-TH to assess MLVC in a video conferencing setting. All codecs are tested at 360p and 540p processing resolutions, the H.265 hardware codec (HEVC-QSV) additionally at native 720p, and displayed at 720p. Results are shown in Fig. 6a and Tab. 2.

MLVC fine-tuned with perceptual losses achieves $-$**75.5%** BD-rate (MOS) at 360p relative to the hardware HEVC-QSV anchor. The lightweight MLVC-S achieves -65.4% at 540p while matching HEVC-QSV in encoding speed. Combining both via a bitrate ladder (MLVC at 360p for low bitrates, MLVC-S at 540p for high bitrates) yields -71.8% overall.

The reference DCVC-RT is MSE-trained. To obtain a controlled perceptual baseline, we fine-tuned it with the same ROI and perceptual losses used for MLVC and evaluated the model at 360p. Perceptual training improves DCVC-RT from -76.4% to -81.9% BD-rate (MOS), placing the cost of cross-platform constraints at roughly 6 percentage points relative to MLVC's -75.5% — a modest price given that DCVC-RT cannot decode across platforms.

Cross-Platform Validation. Figure 6b shows BD-rate for encoder-decoder platform pairs across Apple, Intel, Qualcomm, and NVIDIA hardware. Most pairs achieve BD-rate within 2 points, confirming cross-platform compatibility. Importantly, all configurations avoid catastrophic failures.

Latency Evaluation. Table 3 shows compute complexity results across resolutions on Apple M3 Pro ANE, Intel Lunar Lake NPU 4 i7/i9 version, and Qualcomm Snapdragon X Elite Hexagon NPU. MLVC achieves real-time encoding and decoding (each >30 FPS) up to 540p on all three platforms. MLVC-S ex-

Table 2: BD-rate (MOS) relative to HEVC-QSV on VCD-TH at different processing resolutions: BD-360 and BD-540 denote BD-rate at 360p and 540p respectively, BD-All combines both via bitrate ladder. FPS indicates encoding speed on Apple M3 Pro NPU for neural codecs and Intel hardware for HEVC-QSV. MOS is always evaluated at 720p. CP denotes cross-platform capability.

Codec	IP	RP	BD-360	BD-540	BD-All	FPS (360/540)	CP
HEVC-QSV	-1	–	0.0	0.0	0.0	221/170	✓
DCVC-RT	-1	64	–76.4	–80.3	–77.9	102/46	×
DCVC-RT (perceptual)	-1	64	–81.9	–	–	102/46	×
MLVC	64	32	–75.5	–78.8	–77.4	130/66	✓
MLVC-S	64	32	–	–65.4	–	300/152	✓
MLVC-multi	64	32	–75.5	–65.4	–71.8	130/152	✓

Table 3: Speed comparison of MLVC models across different platforms and resolutions.

Model	Res.	Apple (FPS)		Intel (FPS)		Qualcomm		Avg (FPS)		Params	GFLOPs	
		Enc	Dec	Enc	Dec	Enc	Dec	Enc	Dec	(M)	Enc	Dec
MLVC	360p	129.5	122.9	98.0	98.0	80.0	77.5	103	99	18.3	63.9	66.2
MLVC	540p	65.7	62.3	45.7	45.7	37.0	35.1	49	48	18.3	142.2	147.0
MLVC	720p	33.8	33.1	24.4	26.5	21.4	20.4	27	27	18.3	253.8	262.5
MLVC	1080p	15.6	15.2	10.5	10.7	10.6	9.5	12	12	18.3	567.3	587.1
MLVC-S	360p	300.3	277.8	263.2	270.3	193.4	173.9	252	241	5.4	21.6	23.2
MLVC-S	540p	152.0	137.6	120.5	125.0	89.6	77.9	121	114	5.4	48.0	51.4
MLVC-S	720p	83.8	73.5	62.1	70.4	51.3	44.1	66	63	5.4	85.7	91.9
MLVC-S	1080p	39.5	34.6	24.0	25.8	26.5	19.6	30	27	5.4	191.9	205.7

tends real-time capability to 720p on all platforms and reaches 1080p at 30 FPS on Apple M3 Pro.

5 Conclusion

We present MLVC, the first cross-platform neural video codec that achieves robust decoding across diverse consumer NPUs while maintaining competitive perceptual quality and real-time performance. Our scale-sending mechanism eliminates catastrophic cross-platform decoding failures, while the memory module, ReGLU activations, I-frame dropout, and LTR frames reduce the compression cost of cross-platform constraints. MLVC averages 100 FPS at 360p across three major NPU platforms and achieves perceptual quality on par with original DCVC-RT on the video conferencing benchmark, while being 30% faster.

Several directions remain for future work. Power consumption is likely to become the next optimization frontier now that real-time performance has been achieved. Additionally, simultaneous encoding and decoding on a single device at high resolutions (1080p and above) remains challenging for symmetric architectures and warrants further investigation.

References

1. CLIC challenge 2024. https://archive.compression.cc/2024/
2. ECM reference software. https://vcgit.hhi.fraunhofer.de/ecm/ECM
3. FFmpeg. https://ffmpeg.org/
4. HM reference software for HEVC. https://vcgit.hhi.fraunhofer.de/jvet/HM
5. Intel Quick Sync Video. https://en.wikipedia.org/wiki/Intel_Quick_Sync_Video
6. Operators supported by the model intermediate language (MIL). https://apple.github.io/coremltools/source/coremltools.converters.mil.mil.ops.defs.html
7. VTM reference software for VVC. https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM
8. Agustsson, E., Minnen, D., Johnston, N., Balle, J., Hwang, S.J., Toderici, G.: Scale-space flow for end-to-end optimized video compression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8503–8512 (2020)
9. Agustsson, E., Tschannen, M., Mentzer, F., Timofte, R., Gool, L.V.: Generative adversarial networks for extreme learned image compression. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 221–231 (2019)
10. Ballé, J., Johnston, N., Minnen, D.: Integer networks for data compression with latent-variable models. In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net (2019), https://openreview.net/forum?id=S1zz2i0cY7
11. Ballé, J., Minnen, D., Singh, S., Hwang, S.J., Johnston, N.: Variational image compression with a scale hyperprior. In: International Conference on Learning Representations (2018)
12. Bjontegaard, G.: Calculation of average PSNR differences between RD-curves. ITU-T SG16, Doc. VCEG-M33 (2001)
13. Cheng, Z., Sun, H., Takeuchi, M., Katto, J.: Learned image compression with discretized Gaussian mixture likelihoods and attention modules. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 7939–7948 (2020)
14. Deng, J., Guo, J., Ververas, E., Kotsia, I., Zafeiriou, S.: RetinaFace: Single-shot multi-level face localisation in the wild. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5203–5212 (2020)
15. Duan, W., Chang, Z., Jia, C., Wang, S., Ma, S., Song, L., Gao, W.: Learned image compression using cross-component attention mechanism. IEEE Transactions on Image Processing **32**, 5478–5493 (2023)
16. Flynn, D., Sharman, K., Rosewarne, C.: Common test conditions and software reference configurations for HEVC range extensions, document JCTVC-N1006
17. Guo, Z., Jia, Z., Li, J., Zhang, X., Li, B., Lu, Y.: Generative latent video compression. arXiv preprint arXiv:2510.09987 (2025)
18. He, D., Yang, Z., Chen, Y., Zhang, Q., Qin, H., Wang, Y.: Post-training quantization for cross-platform learned image compression. arXiv preprint arXiv:2202.07513 (2022)
19. He, D., Yang, Z., Peng, W., Ma, R., Qin, H., Wang, Y.: ELIC: Efficient learned image compression with unevenly grouped space-channel contextual adaptive coding. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5718–5727 (2022)

20. He, D., Zheng, Y., Sun, B., Wang, Y., Qin, H.: Checkerboard context model for efficient learned image compression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14771–14780 (2021)
21. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
22. Hu, Z., Lu, G., Xu, D.: FVC: A new framework towards deep video compression in feature space. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1502–1511 (2021)
23. ITU-T: ITU-T recommendation P.910: Subjective video quality assessment methods for multimedia applications. International Telecommunication Union, Geneva (2021)
24. Jia, Z., Li, B., Li, J., Xie, W., Qi, L., Li, H., Lu, Y.: Towards practical real-time neural video compression. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-25, 2024 (2025)
25. Jiang, W., Zhai, Y., Li, H., Wang, R.: TLIC: Learned image compression with ROI-weighted distortion and bit allocation. arXiv preprint arXiv:2401.08154 (2024)
26. Kim, H., Bauer, M., Theis, L., Schwarz, J.R., Dupont, E.: C3: High-performance and low-complexity neural compression from a single image or video. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9347–9358 (2024)
27. Koyuncu, E., Solovyev, T., Alshina, E., Kaup, A.: Device interoperability for learned image compression with weights and activations quantization. In: 2022 Picture Coding Symposium (PCS). pp. 151–155. IEEE (2022)
28. Koyuncu, E., Solovyev, T., Sauer, J., Alshina, E., Kaup, A.: Quantized decoder in learned image compression for deterministic reconstruction. In: ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 3985–3989. IEEE (2024)
29. Le, H., Zhang, L., Said, A., Sautière, G., Yang, Y., Shrestha, P., Yin, F., Pourreza, R., Wiggers, A.J.: MobileCodec: Neural inter-frame video compression on mobile devices. In: Murray, N., Simon, G., Farias, M.C.Q., Viola, I., Montagud, M. (eds.) MMSys '22: 13th ACM Multimedia Systems Conference, Athlone, Ireland, June 14 - 17, 2022. pp. 324–330. ACM (2022). https://doi.org/10.1145/3524273.3532906, https://doi.org/10.1145/3524273.3532906
30. Leguay, T., Ladune, T., Philippe, P., Déforges, O.: Cool-Chic video: Learned video coding with 800 parameters. In: 2024 Data Compression Conference (DCC). pp. 23–32. IEEE (2024)
31. Li, J., Li, B., Lu, Y.: Deep contextual video compression. Advances in Neural Information Processing Systems **34**, 18114–18125 (2021)
32. Li, J., Li, B., Lu, Y.: Hybrid spatial-temporal entropy modelling for neural video compression. In: Proceedings of the 30th ACM International Conference on Multimedia. pp. 1503–1511 (2022)
33. Li, J., Li, B., Lu, Y.: Neural video compression with diverse contexts. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 22616–22626 (2023)
34. Li, J., Li, B., Lu, Y.: Neural video compression with feature modulation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 26099–26108 (2024)
35. Lin, J., Liu, D., Li, H., Wu, F.: M-LVC: Multiple frames prediction for learned video compression. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 3546–3554 (2020)

36. Lu, G., Ouyang, W., Xu, D., Zhang, X., Cai, C., Gao, Z.: DVC: An end-to-end deep video compression framework. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11006–11015 (2019)
37. Ma, Y., Zhai, Y., Yang, C., Yang, J., Wang, R., Zhou, J., Li, K., Chen, Y., Wang, R.: Variable rate ROI image compression optimized for visual quality. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1936–1940 (2021)
38. Mentzer, F., Agustsson, E., Ballé, J., Minnen, D., Johnston, N., Toderici, G.: Neural video compression using GANs for detail synthesis and propagation. In: European Conference on Computer Vision. pp. 562–578. Springer (2022)
39. Mentzer, F., Toderici, G.D., Tschannen, M., Agustsson, E.: High-fidelity generative image compression. Advances in neural information processing systems **33**, 11913–11924 (2020)
40. Minnen, D., Ballé, J., Toderici, G.D.: Joint autoregressive and hierarchical priors for learned image compression. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 31. Curran Associates, Inc. (2018), https://proceedings.neurips.cc/paper_files/paper/2018/file/53edebc543333dfbf7c5933af792c9c4-Paper.pdf
41. Minnen, D., Singh, S.: Channel-wise autoregressive entropy models for learned image compression. In: 2020 IEEE International Conference on Image Processing (ICIP). pp. 3339–3343. IEEE (2020)
42. Naderi, B., Cutler, R.: A crowdsourcing approach to video quality assessment (2024)
43. Naderi, B., Cutler, R., Khongbantabam, N.S., Hosseinkashi, Y., Turbell, H., Sadovnikov, A., Zou, Q.: VCD: A video conferencing dataset for video compression. In: ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 3970–3974. IEEE (2024)
44. Qi, L., Li, J., Li, B., Li, H., Lu, Y.: Motion information propagation for neural video compression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6111–6120 (2023)
45. Shazeer, N.: GLU variants improve transformer. arXiv preprint arXiv:2002.05202 (2020)
46. Sheng, X., Li, J., Li, B., Li, L., Liu, D., Lu, Y.: Temporal context mining for learned video compression. IEEE Transactions on Multimedia **25**, 7311–7322 (2022)
47. Shi, Y., Ge, Y., Wang, J., Mao, J.: AlphaVC: High-performance and efficient learned video compression. In: European Conference on Computer Vision. pp. 616–631. Springer (2022)
48. Sun, H., Yu, L., Katto, J.: Learned image compression with fixed-point arithmetic. In: 2021 Picture Coding Symposium (PCS). pp. 1–5. IEEE (2021)
49. Tian, K., Guan, Y., Xiang, J., Zhang, J., Han, X., Wei, Y.: Effortless cross-platform video codec: A codebook-based method (2024), https://openreview.net/forum?id=mRw9BuNO9i
50. Tian, K., Guan, Y., Xiang, J., Zhang, J., Han, X., Yang, W.: Towards real-time neural video codec for cross-platform application using calibration information. In: Proceedings of the 31st ACM International Conference on Multimedia. pp. 7961–7970 (2023)
51. Van Rozendaal, T., Singhal, T., Le, H., Sautiere, G., Said, A., Buska, K., Raha, A., Kalatzis, D., Mehta, H., Mayer, F., et al.: MobileNVC: Real-time 1080p neural video compression on a mobile device. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 4323–4333 (2024)

52. Xue, T., Chen, B., Wu, J., Wei, D., Freeman, W.T.: Video enhancement with task-oriented flow. International Journal of Computer Vision (IJCV) **127**(8), 1106–1125 (2019)
53. Yang, R., Mentzer, F., Gool, L.V., Timofte, R.: Learning for video compression with hierarchical quality and recurrent enhancement. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6628–6637 (2020)
54. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018)
55. Zheng, Y., Yang, H., Zhang, T., Bao, J., Chen, D., Huang, Y., Yuan, L., Chen, D., Zeng, M., Wen, F.: General facial representation learning in a visual-linguistic manner. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18697–18709 (2022)