TAQ: Static-Deployable Temporal-Aware Quantization for Real-World Video Super-Resolution

Jinwoo Chung, Sangho An, Sungyeop Jung, and Jangho Kim*

Kookmin University, Seoul, Republic of Korea
{imaboybut, do753951, jshjsy01, jangho.kim}@kookmin.ac.kr

Abstract. Real-world video super-resolution (VSR) faces in-the-wild degradations whose artifacts accumulate over time, yet edge deployment requires static execution, and quantization parameters must be fixed after compilation for low-overhead inference. We propose Temporal-Aware Quantization (TAQ), a novel static post-training quantization framework that uses video structure only in offline calibration. TAQ calibrates sequence-specific activation bounds, refines them with a temporal consistency objective we propose that aligns inter-frame changes between floating-point and quantized outputs without weight retraining, and ensembles the refined bounds into one deployable set of static parameters. Across REDS, SPMCS, UDM10, and VideoLQ, TAQ improves perceptual and temporal quality, reducing LPIPS by up to 0.0334 and TLPIPS by up to 4.7344 over Static PTQ baselines under identical calibration. On NVIDIA Jetson Orin Nano, TAQ enables TensorRT INT8 with up to 3.56× speedup (1.15 fps), while a dynamic INT8 baseline is slower than FP32 (0.79×, 0.266 fps vs. 0.325 fps). Code is available at https://github.com/imaboybut/TAQ.

Keywords: Static Quantization · Video Super-Resolution · Edge device deployment

1 Introduction

Video super-resolution (VSR) [3, 12] is central to applications such as mobile quality enhancement, streaming cost reduction, preservation and restoration, and real-time production. While recent models achieve strong restoration quality, their multi-frame architectures inevitably incur substantial compute and memory costs, which makes low-latency deployment on resource-constrained edge hardware particularly difficult [14, 15, 35]. Post-training quantization (PTQ) is thus an attractive deployment path, because it converts a pretrained floating-point model into a low-precision model without full retraining, enabling practical efficiency improvements with minimal training burden.

To make PTQ practically useful on edge devices, we must in practice respect **static execution**: once the inference engine is built or compiled, quantization

* Corresponding author

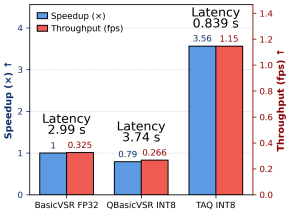

Fig. 1: TensorRT speed on NVIDIA Jetson Orin Nano. with the *BasicVSR backbone (higher is better).*

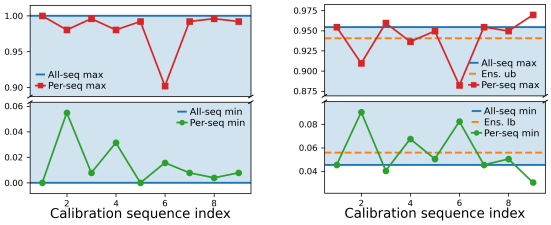

Fig. 2: Sequence-wise multimodality. (a) Per-seq clipped min/max varies widely (range heterogeneity). (b) Ensembling consolidates heterogeneous per-seq optima into a robust deployable bound.

parameters must remain fixed at runtime. Input-dependent range/scale updates ("dynamic activation quantization") incur runtime overhead and can weaken static compilation pipelines (e.g., TensorRT), diminishing practical speedups. In fact, Fig. 1 shows that a dynamic INT8 baseline (QBasicVSR [36]) can be even slower than the original FP32 model on NVIDIA Jetson Orin Nano (e.g., with the *BasicVSR backbone* [15], dynamic INT8 achieves only 0.79× speedup at 0.266 fps vs. FP32 at 0.325 fps), because input-dependent runtime range/scale handling offsets the gains from INT8 kernels. Motivated by this practical deployment behavior, we focus on static PTQ, where all quantization parameters are determined only during offline calibration and remain unchanged at inference. **This static design enables substantial acceleration in practice:** our proposed Temporal-Aware Quantization (TAQ) INT8 achieves 3.56× speedup and 1.15 fps in Fig. 1.

This static constraint, however, exposes VSR-specific challenges. Without input-dependent range adaptation at runtime, static PTQ must cope with (i) sequence-wise multimodality and (ii) temporal error drift. Videos exhibit sequence-wise multimodality: activation ranges vary substantially across sequences (Fig. 2a), so treating heterogeneous sequences as a single calibration distribution yields biased bounds—overly wide for typical clips (rounding noise) or overly tight for others (clipping). Fig. 2b corroborates this mismatch by contrasting per-sequence (and our refined) bounds against the mixed global bound. Moreover, quantization perturbations exhibit temporal error drift that accumulates over time as flicker and jitter, which image-centric PTQ cannot suppress because it ignores the temporal axis (Fig. 3).

To satisfy both the **static edge deployment constraint** in practice and the **video-specific PTQ challenge** (multimodality and temporal drift), we propose Temporal Aware Quantization (TAQ), a **static-deployable** PTQ framework for real-world VSR. TAQ leverages video structure only during **offline calibration** but produces a **static** quantized model for inference, with no runtime range/scale recomputation or input-dependent adaptation. Specifically, TAQ (i) *specializes* quantizer bounds per calibration sequence to respect multimodal clip statistics, (ii) *refines* them using a temporal consistency objective that aligns

inter-frame changes between floating-point and quantized outputs to suppress drift, and (iii) *aggregates* the per-sequence solutions into one deployable set of static quantization parameters via bound ensembling. As visualized in Fig. 3 using y–t profiles, TAQ mitigates banding/flicker patterns and yields smoother trajectories closer to the ground truth, while preserving static, hardware-friendly inference.

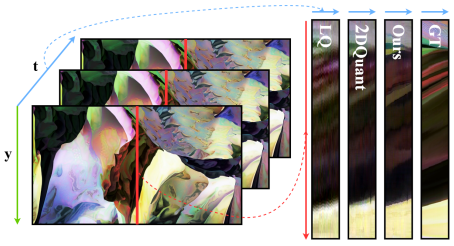

Fig. 3: Temporal stability via y–t profiles. *2DQuant* 8-bit shows banding/flicker from frame-wise error drift; *Ours* 8-bit yields smoother trajectories closer to GT.

Our specialize-then-aggregate design resolves a fundamental tension in static VSR quantization: per-sequence calibration is needed to respect heterogeneous statistics, yet edge deployment permits only a *single static* parameter set. Optimizing bounds per sequence reduces the bias induced by multimodality, and aggregating these per-sequence optima produces a stable static solution that is less sensitive to any single calibration clip. Importantly, each sequence bound is estimated from percentile-clipped activation statistics, limiting the influence of extreme tails before aggregation. In particular, simple averaging provides a natural and effective aggregation under a mixture-of-sequences view of deployment; we further analyze and validate this choice in Sec. 3.5. Compared to QBasicVSR, TAQ improves perceptual and temporal quality under a comparable mixed-precision budget (AvgBit 6/7.14 vs. fixed 6/7), while targeting edge-deployable *static quantization* for efficient accelerator execution in Sec. 4.3.

Our main contributions are summarized as follows:

- **Static-deployable PTQ for real-world VSR.** We formulate VSR quantization under an edge constraint of *static execution* and propose a PTQ pipeline that preserves static inference (no runtime activation-quantization adaptation).
- **Sequence-wise calibration with bound ensembling.** Edge deployment permits only one static parameter set, yet video activations are highly sequence-dependent; we estimate (ℓ, u) per sequence and ensemble them into a single deployable static set.
- **Temporal refinement via frame-difference alignment.** We refine bounds with a temporal loss that aligns consecutive-frame differences between FP and INT outputs, improving temporal stability while consistently benefiting image quality.
- **Empirical validation and edge readiness.** Across varied bit widths (2/3/4/8) and both degraded and in-the-wild settings, TAQ consistently surpasses image-centric PTQ baselines, and TensorRT INT8 inference on an edge platform (Jetson Orin Nano [22]) demonstrates practical deployability with real speedup.

2 Related work

Video Super-Resolution. Video super-resolution (VSR) aims not only to restore high-fidelity details in individual frames but also to maintain temporal consistency. Early approaches adopt recurrent or feedback architectures that propagate hidden states along the time axis to accumulate and propagate information [1, 2, 23]. Subsequent methods align multiple neighboring frames and fuse their features so that complementary cues from supporting frames benefit the reconstruction of the reference frame [8, 27]. More recently, transformer based models leverage attention to capture long range temporal dependencies and global context, achieving strong performance in VSR [2, 14, 15]. For real-world deployment, numerous studies have constructed realistic synthetic degradation pipelines, incorporating randomly shuffled or high order operators and codec artifacts, to approximate in-the-wild conditions for training and evaluation [30, 32, 35]. While these setups reduce domain shift and improve deployability, the diversity of scenes, motion, and codec conditions increases activation distribution variability; moreover, as sequence length grows, both computation and memory costs tend to rise.

Post-training Quantization. Quantization research is conventionally partitioned into quantization-aware training and post-training quantization. Quantization aware training learns quantizer parameters end-to-end [4, 6, 9, 10]. These methods reach strong accuracy at low precision but usually require training time and data access on the scale of the original model, which is costly for large VSR networks. Post-training quantization keeps the pre-trained model fixed and estimates clipping bounds from a small calibration set using observers such as Min–Max [7], percentile [11], and histogram-based criteria [18]. Many pipelines add bias correction and light refinements after calibration, which enables deployment without retraining [13, 19, 20].

Super-Resolution Quantization. Within super-resolution, recent work reduces accuracy loss by tailoring clipping to asymmetric and long-tailed activation distributions and by adding small refinements; representative examples include PTQ4SR [26] and 2DQuant [16]. However, most prior art targets image SR and does not directly address video-specific issues such as sequence-wise statistical mismatch, accumulation of frame-wise quantization error, and loss of temporal consistency. In practice, scene content, motion, and codec choices vary widely across sequences, so a single global calibration over mixed videos can produce either overly aggressive clipping or overly loose ranges. Recently, QBasicVSR [36] extends PTQ to VSR by introducing flow-guided bit adaptation and temporal shared-layer bit allocation. However, its bit-width selection relies on dynamic inference-time adaptation, which differs from the fully static setting we target in this work.

3 Method

The proposed Temporal-Aware Quantization (TAQ) builds on a standard affine PTQ quantizer (Sec. 3.1) and consists of three stages: (1) sequence-wise histogram-

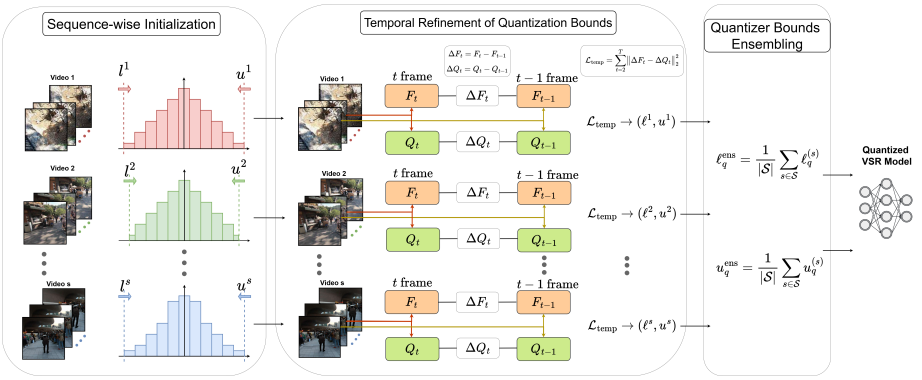

Fig. 4: Overview of our Temporal-Aware Quantization (TAQ) pipeline. *Sequence-wise Initialization* calibrates each sequence to estimate activation and weight bounds. *Temporal Refinement of Quantization Bounds* then refines the quantizers with a temporal loss on consecutive-frame differences, after which the per-sequence bounds are ensembled into a single deployable range for inference.

based initialization, (2) temporal refinement of quantization bounds, and (3) sequence-wise quantizer-bounds ensembling.

3.1 PTQ Quantizer

Uniform Asymmetric (Affine) Quantizer. Given bit-width b with $n = 2^b - 1$ and clipping bounds $\ell < u$, we use an affine uniform (fake) quantizer for a real-valued tensor v. Here, ℓ and u denote the tensor's clipping lower and upper bounds and determine the step size $\Delta = (u - \ell)/n$ (equivalently, the zero-point $z = -\ell/\Delta$). The forward quantization–dequantization is

$$v_c = \mathrm{clip}(v, \ell, u), \quad v_r = \mathrm{round}\left(\frac{v_c - \ell}{\Delta}\right), \quad v_q = \Delta\, v_r + \ell \tag{1}$$

where v_c is the clipped input, $v_r \in \{0, \ldots, n\}$ is the integer code, and v_q is the dequantized value used in the forward pass. We adopt the identity straight-through estimator (STE) in backpropagation. We denote this affine quantize–dequantize operator by $Q_b(v; \ell, u)$ in the following sections.

3.2 Sequence-wise Histogram-based Initialization (Video-aware)

We partition the calibration set *by sequence*. For each sequence, we replace every Conv/Linear layer with its quantized counterpart and run a single forward pass to collect per-layer activation statistics, while all model instances share the same pre-trained weights. For weight quantizers, bounds (ℓ, u) are initialized by minimizing the MSE between each pre-trained weight tensor and its quantized counterpart directly. For activation quantizers, captured activations are percentile-clipped (default 99.9 %) to obtain $\hat{x}$, which is then accumulated

into a histogram of K bins. Let c_k and w_k denote the center and sample count of bin k. We seek per-sequence bounds (ℓ, u) that minimize the weighted reconstruction error between each bin center and its affine quantize–dequantize output $Q_b(c_k; \ell, u)$ via Eq. (1):

$$J(\ell, u) = \sum_{k=1}^{K} w_k \left(c_k - Q_b(c_k; \ell, u)\right)^2, \qquad \ell < u. \tag{2}$$

Note that prior MSE-based calibration methods [16,26] pool all calibration samples into a single set of statistics, producing one global bound. In contrast, we solve Eq. (2) independently per sequence to preserve clip-specific statistics. We install the minimizer of $J(\ell, u)$ into the activation quantizers for that sequence. Repeating this over all sequences yields video-aware initial bounds, which serve as starting points for temporal refinement and subsequent ensembling.

3.3 Temporal Refinement of Quantization Bounds

A single, globally shared clipping range can be brittle when scene statistics and motion patterns vary across frames in the same sequence. Starting from the *sequence-wise histogram initialization*, we therefore *refine* the clipping bounds (ℓ, u) for *both weights and activations* on a *per-sequence (clip)* basis. Let $\mathcal{Q}$ denote the collection of quantizers in the network. Given the same input, we run the floating-point reference model F and the quantized model Q_θ in parallel and treat the clipping bounds of weight/activation quantizers as the sole learnable variables; biases and bit-widths remain fixed.

Temporal Differences. Let F_t and Q_t denote the final RGB outputs of F and Q_θ at time t, respectively. We define raw frame differences

$$\Delta F_t = F_t - F_{t-1}, \qquad \Delta Q_t = Q_t - Q_{t-1}. \tag{3}$$

For the boundary frame (t=1), the temporal term is omitted and the sum starts at t=2.

Temporal Consistency Loss. A natural alternative would be to penalize per-frame output discrepancy $\|F_t - Q_t\|$; however, even small per-frame errors can fluctuate across time and manifest as flicker. By instead matching inter-frame changes, static bias cancels in the difference and temporal artifacts are directly suppressed. We therefore minimize a temporal consistency loss that aligns inter-frame changes of Q_θ to those of F:

$$\mathcal{L}_{\text{temp}} = \sum_{t=2}^{T} \left\| \Delta F_t - \Delta Q_t \right\|_2^2. \tag{4}$$

Gradients are propagated to the quantizer bounds $\theta = \{(\ell_q, u_q)\}_{q \in \mathcal{Q}}$ of *both* weights and activations; the affine quantization form is unchanged. After refinement, we cache the per-sequence bounds $\hat{\theta}^{(s)} = \{(\ell_q^{(s)}, u_q^{(s)})\}_{q \in \mathcal{Q}}$ for all weight and activation quantizers, which are subsequently aggregated via *sequence-wise* bound averaging in Sec. 3.4.

3.4 Sequence-Wise Quantizer Bounds Ensembling

Let $\mathcal{S}$ be the set of calibrated sequences (clips), and let $\mathcal{Q}$ denote the collection of quantizers in the network. After the per-sequence temporal refinement in Sec. 3.3, each quantizer $q \in \mathcal{Q}$—whether weight or activation—is associated with sequence-specific bounds $\left(\ell_q^{(s)}, u_q^{(s)}\right)$ for every $s \in \mathcal{S}$. We then perform *uniform ensembling* over sequences, preserving one-to-one matching by quantizer key (module path or layer name):

$$\ell_q^{\text{ens}} = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \ell_q^{(s)}, \quad u_q^{\text{ens}} = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} u_q^{(s)}. \tag{5}$$

With uniform weights, this ensembling reduces to arithmetic averaging of the per-sequence bounds. The final *ensembled* model installs $\left(\ell_q^{\text{ens}}, u_q^{\text{ens}}\right)$ back into each quantizer q without any scale or zero point conversion. Ensembling the per-sequence bounds (ℓ, u) directly and re-applying them to each quantizer keeps the implementation lightweight and preserves the statistics observed during calibration and temporal refinement. Conceptually, this moves between a single global calibration (high bias) and per-clip calibrations (high variance), while retaining the temporal signals injected by the refinement stage. All quantizers remain affine. The ensembling consolidates heterogeneous per-sequence optima into a single robust range for deployment, reducing variance without reintroducing global-calibration bias. The full procedure is summarized in Algorithm 1.

3.5 Analysis of Sequence-Wise Bound Ensembling

Why a consolidation objective? After sequence-wise refinement, each sequence s yields an optimum $\hat{\theta}_q^{(s)}$ for each quantizer $q \in \mathcal{Q}$, where $\theta_q = (\ell_q, u_q)$. However, edge deployment requires a *single static* set of quantization parameters. We therefore consolidate $\{\hat{\theta}_q^{(s)}\}_{s=1}^{S}$ by choosing a static θ_q that stays close to the sequence-specific optima in an average sense:

$$\theta_q^\star = \arg\min_{\theta_q} \sum_{s=1}^{S} \alpha_s \|\theta_q - \hat{\theta}_q^{(s)}\|_2^2, \qquad \alpha_s \geq 0, \ \sum_{s=1}^{S} \alpha_s = 1. \tag{6}$$

Proposition 1 (Weighted mean as the exact minimizer). *The minimizer of Eq. (6) is* $\theta_q^\star = \sum_{s=1}^{S} \alpha_s \hat{\theta}_q^{(s)}$. *For* $\alpha_s = 1/S$, $\theta_q^\star$ *reduces to the arithmetic mean.*

Outlier control. To mitigate sensitivity to atypical clips, we apply percentile clipping (default: 99.9%) [11] to activations *before* histogram construction, which suppresses extreme tails and stabilizes per-sequence bound estimates. Moreover, since the deployable bound is an arithmetic average over S calibration sequences, the influence of any single outlier on $(\ell_q^{\text{ens}}, u_q^{\text{ens}})$ is bounded and decreases with $1/S$.

Empirical robustness. We empirically confirm this robustness by comparing mean ensembling against robust alternatives, including the median and

Algorithm 1 Temporal-Aware Quantization

Input: Full-precision VSR model F with quantizers $\mathcal{Q}$, calibration sequences $D = \{S_i\}_{i=1}^{N}$, bit-width b ($n = 2^b - 1$), histogram bins K, percentile thresholds (p_ℓ, p_u) for activation clipping (default $p_\ell = 0.1\%$, $p_u = 99.9\%$), refinement epochs E.
Output: Final quantized model Q_θ.

$L[q] \leftarrow \emptyset,\ U[q] \leftarrow \emptyset \quad \forall q \in \mathcal{Q}$
for all weight quantizer $q_w \in \mathcal{Q}$ **do**
  $(\ell_{q_w}, u_{q_w}) \leftarrow \arg\min_{\ell<u} \|w_{q_w} - Q_b(w_{q_w}; \ell, u)\|_2^2$
end for
for $S_i \in D$ **do**
  for all activation quantizer $q_a \in \mathcal{Q}$ **do**
    $x_{q_a} \leftarrow$ activations of quantizer q_a from forward pass on S_i
    $\hat{x}_{q_a} \leftarrow \mathrm{clip}\big(x_{q_a},\ \mathrm{percentile}(x_{q_a}, p_\ell),\ \mathrm{percentile}(x_{q_a}, p_u)\big)$
    $\{(c_k, w_k)\}_{k=1}^{K} \leftarrow \mathrm{Histogram}(\hat{x}_{q_a},\ K)$
    $J_{q_a}(\ell, u) \leftarrow \sum_{k=1}^{K} w_k \big(c_k - Q_b(c_k; \ell, u)\big)^2$
    $(\ell_{q_a}^{(i)}, u_{q_a}^{(i)}) \leftarrow \arg\min_{\ell<u} J_{q_a}(\ell, u)$
  end for
  for $e = 1$ **to** E **do**
    Update $\{(\ell_q, u_q)\}_{q\in\mathcal{Q}}$ by minimizing Eq. (4)
  end for
  for all $q \in \mathcal{Q}$ **do**
    $L[q].\mathrm{append}(\ell_q^{(i)}), \quad U[q].\mathrm{append}(u_q^{(i)})$
  end for
end for
for all $q \in \mathcal{Q}$ **do**
  $\ell_q^{\mathrm{ens}} \leftarrow \frac{1}{N}\sum_{i=1}^{N} \ell_q^{(i)}, \quad u_q^{\mathrm{ens}} \leftarrow \frac{1}{N}\sum_{i=1}^{N} u_q^{(i)}$
end for
$Q_\theta \leftarrow \mathrm{Quantize}\big(F; \{(\ell_q^{\mathrm{ens}}, u_q^{\mathrm{ens}})\}_{q\in\mathcal{Q}}\big)$
return Q_θ

removing the most extreme sequence in Sec. 4.4. In our setting, the mean performs best, supporting the choice of simple averaging as a stable deployable aggregator. For completeness, we provide a local quadratic surrogate motivation for mean ensembling in the supplementary material A.

4 Experiments

4.1 Experimental Setup

Datasets and splits. We evaluate on REDS [21], SPMCS [24], and UDM10 [31], applying the same real-world degradation pipeline to all datasets. For calibration, we use 9 sequences from REDS validation set #21—#29 with 20 frames per sequence. For testing, we use REDS validation set #0—#20, all 30 sequences of SPMCS validation set, and all 10 sequences of UDM10 validation set. In addition, we evaluate on VideoLQ [3] to assess robustness on in-the-wild videos

without paired ground truth and report no-reference metrics on its validation set.

Evaluation metrics. PSNR is a fidelity metric that expresses the MSE between the restored image and the ground truth in dB. SSIM [28] measures structural similarity (luminance, contrast, and structure) in the range $[0, 1]$. LPIPS [34] is a perceptual distance computed in a pre-trained CNN feature space.

Temporal metrics. TLPIPS [5] uses LPIPS$(\widehat{Y}_t, \widehat{Y}_{t-1})$ and LPIPS(Y_t, Y_{t-1}) for consecutive frame pairs, takes their absolute difference, and averages over time. TOF [5] estimates optical flow with RAFT [25] for Flow$(\widehat{Y}_t, \widehat{Y}_{t-1})$ and Flow(Y_t, Y_{t-1}), then averages the per-pixel absolute difference. Both TLPIPS and TOF directly evaluate sequence-level temporal consistency and are therefore essential for verifying temporal stability in VSR.

No-reference metrics. ILNIQE [33] models generalized natural scene statistics (NSS) and predicts perceptual quality without using distorted–reference pairs or subjective scores, directly from local statistical features. NRQM [17] is a learning-based metric trained on human subject studies for single-image super-resolution, using spatial/frequency features to regress perceptual quality.

Implementation details. We use the official pre-trained RealViformer [35] model. Architectural details are provided in the appendix. All experiments use $\times 4$ upscaling. We denote the bit-width configuration as b_w/b_a (weight/activation). Unless otherwise noted, we use uniform precision $b_w{=}b_a{=}b$. We apply post-training quantization at $b \in \{2, 3, 4, 8\}$, keeping the original hyperparameters unless otherwise noted. For activation range initialization, we build histograms with $K{=}1024$ bins after percentile clipping. In the second stage, we optimize quantizer bounds with Adam (learning rate 5e-4). The total number of optimization steps is 9 sequences $\times 20$ frame differences $\times 5$ epochs.

4.2 Results

Baselines. We use standard implementations for MinMax [7] and Percentile [11]. We port the official PTQ4SR [26] and 2DQuant-SR [16] code to the RealViformer backbone and apply the same quantizer family as ours, where 2DQuant-SR is a state-of-the-art method for *image* super-resolution. All methods use the same sequence list and the same number of calibration frames, and we keep each method's public hyperparameters unchanged.

Evaluation. Under identical calibration and evaluation settings, we report PSNR, SSIM, LPIPS, TLPIPS, and TOF at 2/3/4/8 bits for all methods. For the no-reference VideoLQ dataset, we additionally report ILNIQE and NRQM.

Quantitative results. Tab. 1 presents extensive comparisons on RealViformer at 2/3/4/8 bits under identical calibration/evaluation settings. Across REDS, SPMCS, and UDM10, our method consistently surpasses image-centric PTQ baselines on both image quality (LPIPS, PSNR/SSIM) and temporal consistency (TLPIPS, TOF), with the gains most pronounced at low bit-widths. For instance, on SPMCS at 4-bit, TLPIPS drops from 24.1784 (2DQuant) to 19.6660, and on UDM10 at 3-bit, TOF falls from 126.6288 (2DQuant) to 83.1930.

Table 1: Comprehensive comparison grouped by bit. For REDS/SPMCS/UDM10 we report PSNR↑ / SSIM↑ / LPIPS↓ / TLPIPS↓ / TOF↓. For VideoLQ we report ILNIQE↓ / NRQM↑. Best and second-best are in **bold** and underlined.

		2-bit					3-bit				
Dataset	Metric	MinMax	Percentile	PTQ4SR	2DQuant	Ours	MinMax	Percentile	PTQ4SR	2DQuant	Ours
REDS	PSNR↑	7.5872	7.9975	11.9190	24.0987	**24.5386**	9.5231	16.3431	21.7145	22.0758	**22.2444**
	SSIM↑	0.2067	0.0542	0.4200	0.5209	**0.5944**	0.0375	0.2574	0.3735	0.4243	**0.4252**
	LPIPS↓	1.0172	1.2244	0.9186	0.7298	**0.7142**	1.2899	0.7746	0.6965	0.5848	**0.5757**
	TLPIPS↓	23.4412	24.0689	16.7200	10.0091	**5.7869**	20.4510	26.6831	22.8983	20.0603	**15.3259**
	TOF↓	92.5622	880.5197	210.8610	43.8434	**31.7238**	1288.6210	1029.1262	88.9967	52.9386	**45.2512**
SPMCS	PSNR↑	7.5412	7.9866	14.5251	24.6055	**24.7626**	8.2103	16.2307	7.6570	**22.4048**	22.3462
	SSIM↑	0.1744	0.0481	0.4738	0.6054	**0.6126**	0.0243	0.2425	0.0275	0.4067	**0.4462**
	LPIPS↓	1.0354	1.2773	0.8898	0.6931	**0.6730**	1.3195	0.8075	1.3956	**0.5675**	0.5710
	TLPIPS↓	27.9528	42.7062	26.4721	17.6929	**14.0521**	33.8040	44.2389	37.9850	28.3180	**26.1131**
	TOF↓	34.9713	952.5176	33.1535	14.0130	**5.8894**	1433.5659	880.0272	632.9210	55.8117	**30.7197**
UDM10	PSNR↑	9.1668	9.3714	13.2019	28.2226	**28.3204**	9.1056	17.3223	7.3805	24.7232	**24.7274**
	SSIM↑	0.3372	0.0752	0.6104	0.7683	**0.7744**	0.0309	0.3516	0.0312	**0.5553**	0.5462
	LPIPS↓	0.9401	1.3360	0.8888	0.6009	**0.5971**	1.3705	0.8323	1.4398	**0.5867**	0.6063
	TLPIPS↓	21.8818	38.4365	23.7475	14.6868	**11.3868**	29.1477	38.3209	31.2210	23.8387	**21.6668**
	TOF↓	99.1825	878.1625	99.3990	85.8837	**62.4279**	1377.2106	937.4745	656.0068	126.6288	**83.1930**
VideoLQ	ILNIQE↓	81.8523	79.5250	75.0065	40.7119	**39.1392**	55.7329	56.6197	33.3424	29.6925	**29.3606**
	NRQM↑	3.7145	2.5073	2.9741	2.6489	**4.7491**	3.7225	3.6481	4.2188	4.0320	**6.6185**

		4-bit					8-bit				
Dataset	Metric	MinMax	Percentile	PTQ4SR	2DQuant	Ours	MinMax	Percentile	PTQ4SR	2DQuant	Ours
REDS	PSNR↑	10.7379	23.1268	23.3999	23.1155	**23.8949**	24.9587	25.4449	25.7334	25.7180	**25.7768**
	SSIM↑	0.1440	0.4689	0.5329	0.5306	**0.5480**	0.6675	0.6493	0.6694	0.6767	**0.6772**
	LPIPS↓	0.8517	0.5555	0.6793	0.4828	**0.4724**	0.2566	0.3122	0.2856	0.2660	**0.2517**
	TLPIPS↓	36.1503	15.3454	18.9417	12.9041	**10.0249**	1.8291	1.9872	2.7159	1.4566	**1.2161**
	TOF↓	1385.4387	55.4030	29.4118	29.0666	**23.6122**	6.4111	6.3244	12.0487	6.9301	**5.9729**
SPMCS	PSNR↑	11.8819	23.1022	22.5544	24.0131	**24.0377**	25.3908	25.4711	25.9204	25.9200	**25.9252**
	SSIM↑	0.0899	0.4960	0.4460	0.5512	**0.5709**	0.6840	0.6635	0.6869	0.6932	**0.6941**
	LPIPS↓	1.2338	0.5478	0.7142	0.4647	**0.4582**	0.2892	0.3370	0.3515	0.2979	**0.2754**
	TLPIPS↓	33.4934	24.2999	29.1236	24.1784	**19.6660**	5.6968	6.1193	7.0000	4.2171	**3.8087**
	TOF↓	1325.1817	15.6654	72.7156	34.5345	**12.5978**	1.7482	1.8737	7.4582	1.9064	**1.5042**
UDM10	PSNR↑	13.0904	25.5344	24.7417	27.1506	**27.2359**	29.5723	29.8206	29.4053	30.3029	**30.3708**
	SSIM↑	0.1045	0.6203	0.5559	0.6964	**0.7244**	0.8570	0.8413	0.8585	0.8601	**0.8664**
	LPIPS↓	1.3088	0.5875	0.7519	0.4763	**0.4604**	0.2592	0.2744	0.2478	0.2495	**0.2161**
	TLPIPS↓	30.4998	19.8927	25.9396	19.7242	**15.5704**	3.0982	3.9375	3.5400	3.3868	**2.0095**
	TOF↓	1173.7991	153.6336	99.8883	89.3016	**71.7848**	33.2438	34.4643	27.2276	28.7950	**25.0208**
VideoLQ	ILNIQE↓	41.2283	34.7752	32.5161	27.8252	**27.4141**	27.3130	27.1688	27.0469	26.3192	**26.2863**
	NRQM↑	4.6891	5.8520	5.9645	4.7833	**6.0281**	4.3962	4.6805	4.1943	4.5939	**4.9738**

On REDS at 4-bit, PSNR rises from 23.1155 dB (2DQuant) to 23.8949 dB and SSIM from 0.5306 to 0.5480, while LPIPS decreases from 0.4828 to 0.4724. At 8-bit, our model matches or surpasses the strongest baselines while preserving temporal stability. Taken together, sequence-wise initialization, temporal refinement, and bound ensembling jointly reduce temporal error drift and sustain image quality at low bit-widths. We report acceleration results for RealViformer in the supplementary material B.1.

Qualitative results. Fig. 5 compares ×4 reconstructions at 8-bit on REDS, SPMCS, and UDM10. Percentile and PTQ4SR often cause oversmoothing or ringing, while MinMax and 2DQuant reduce artifacts but still exhibit texture collapse and repeated patterns in fine details. In contrast, our method preserves sharper edges and more faithful micro-textures, reproducing contours and periodic structures closest to GT without amplifying noise. Specifically, on REDS the brick mortar lines remain distinct, on SPMCS the building façades retain window lattices, and on UDM10 the arch scene maintains fine grooves without blotchy artifacts. These visual trends align with the quantitative LPIPS/SSIM improvements. Additional comparisons are provided in the supplementary material.

Fig. 5: Visual comparisons on UDM10, REDS, and SPMCS. Left: full frame with ROI (red box). Right: crop comparisons (LR, MinMax, Percentile, PTQ4SR, 2DQuant, Ours, FP, GT).

4.3 Comparison with VSR-Specific Dynamic Quantization

We compare TAQ with QBasicVSR [36], a VSR-specific PTQ method that determines bit-widths via flow-gradient video bit adaptation and temporal shared-layer bit adaptation at inference time. Since QBasicVSR does not release its training code, we evaluate using its public inference code and the released MP6 weights. Both methods are evaluated on the BasicVSR [1] backbone under identical conditions. To ensure a fair comparison, we follow the QBasicVSR calibration protocol and use 100 sequences from REDS and Vimeo-90K [29], rather than the 9 REDS sequences used in Sec. 4.2. TAQ runs the same sequence-wise calibration and temporal refinement on each sequence, producing per-sequence bounds, which are then ensembled into a single deployable model.

Accuracy. In this comparison we denote each configuration as b_w/b_a, where b_w and b_a are the weight and activation bit-widths respectively. Tab. 2 compares TAQ with QBasicVSR under its mixed-precision MP6 configuration (6/7.14 on average). TAQ 6/7 with fixed uniform precision uses fewer average bits, yet

Table 2: Comparison with QBasicVSR. We quantize **BasicVSR** following QBasicVSR's protocol. Official QBasicVSR provides only MP6 (mixed-precision) weights.

Dataset	Metric	QBasicVSR	TAQ (ours)	TAQ (ours)
AvgBit (W/A)		6 / 7.14 (MP-A)	6 / 6 (fixed)	6 / 7 (fixed)
REDS4	PSNR↑/SSIM↑	31.26/0.8877	31.15/0.8832	**31.32**/**0.8882**
	LPIPS↓/TLPIPS↓	0.1704/10.27	0.1735/10.40	**0.1691**/**9.81**
Vid4	PSNR↑/SSIM↑	27.13/0.8225	26.98/0.8189	**27.24**/**0.8247**
	LPIPS↓/TLPIPS↓	0.2140/14.17	0.2256/14.13	**0.2112**/**12.60**
VideoLQ	ILNIQE↓/NRQM↑	29.66/3.772	29.58/3.774	**28.87**/**3.955**

Table 3: TensorRT deployment speed on NVIDIA Jetson Orin Nano. All methods use the BasicVSR backbone with 32 frames from Vid4.

Method	Precision	Dataset	Frame	Throughput (fps)↑	Latency (s)↓	Speedup↑
BasicVSR	FP32	Vid4	32	0.3245	2.987	–
QbasicVSR	INT8 (Dynamic)			0.2661	3.738	0.79×
TAQ (ours)	INT8 (Static, ours)			1.1459	0.839	3.56×

surpasses QBasicVSR in PSNR, LPIPS, and TLPIPS across REDS4, Vid4, and VideoLQ, with the temporal consistency gains most pronounced.

Deployment speed. Beyond accuracy, practical edge deployment demands real inference speedup. However, dynamic quantization with input-dependent range/scale updates inherently conflicts with static compilation pipelines, diminishing practical speedups. QBasicVSR combines dynamic activation quantization with mixed-precision bit allocation, both of which prevent static compilation from applying uniform kernel optimization and require runtime recomputation of activation ranges. Tab. 3 confirms this empirically on an NVIDIA Jetson Orin Nano: when built with TensorRT, the dynamic INT8 baseline achieves only 0.79× speedup over FP32, actually running slower, whereas TAQ static INT8 achieves 3.56× speedup at 1.15 fps, consistent with Fig. 1. These results highlight that even when accuracy is comparable, static quantization is essential for practical edge deployment. We provide INT8 results on the BasicVSR backbone in the supplementary material B.2.

4.4 Analysis

Backbone Generalization. Our PTQ is backbone-agnostic. Beyond the transformer based RealViformer, it also surpasses MinMax and 2DQuant at 4/8-bit on the CNN-based RealBasicVSR [3], as shown in Tab. 4. This indicates that sequence-wise initialization, temporal refinement, and quantizer bounds ensembling generalize across transformer and preprocessing-plus-propagation CNN backbones and support practical portability across diverse VSR architectures.

Ablation. Tab. 5 isolates the two core components of TAQ: sequence-wise calibration and temporal refinement. Sequence-wise calibration accounts for most gains over the global single-range baseline, with PSNR rising by 0.68 dB, TLPIPS

Table 4: Backbone generalization on RealBasicVSR (×4). 4/8-bit with MinMax, 2DQuant, and Ours. Best and second-best are in **bold** and underlined.

		4-bit			8-bit		
Dataset	Metric	MinMax	2DQuant	Ours	MinMax	2DQuant	Ours
REDS	PSNR↑	23.3486	24.1553	**24.3244**	24.9063	24.9148	**24.9984**
	SSIM↑	0.4549	0.6057	**0.6083**	0.6688	0.6687	**0.6697**
	LPIPS↓	0.7558	0.4873	**0.4804**	0.2554	0.2585	**0.2552**
	TLPIPS↓	8.7117	5.4051	**4.2680**	1.0129	0.9524	**0.9220**
	TOF↓	62.9974	11.1770	**9.8612**	5.5583	5.3500	**5.3287**
SPMCS	PSNR↑	23.4604	**24.8612**	24.8359	25.3284	25.3497	**25.4446**
	SSIM↑	0.4708	0.6314	**0.6354**	0.6854	0.6940	**0.6951**
	LPIPS↓	0.7347	0.4572	**0.4466**	0.2705	0.2736	**0.2689**
	TLPIPS↓	17.2236	10.2953	**9.3207**	3.0021	3.0108	**2.8794**
	TOF↓	23.4923	2.4408	**2.4283**	1.5765	1.4988	**1.4799**
UDM10	PSNR↑	26.3084	27.9595	**28.0597**	28.0033	28.3154	**28.3427**
	SSIM↑	0.5678	0.7903	**0.7948**	0.8372	0.8366	**0.8375**
	LPIPS↓	0.7574	0.4229	**0.4093**	0.2565	0.2556	**0.2545**
	TLPIPS↓	11.7563	6.9206	**6.6938**	1.9791	1.9844	**1.9123**
	TOF↓	42.5702	15.2992	**13.1039**	27.3640	26.9500	**25.5977**
VideoLQ	ILNIQE↓	30.2577	28.0459	**27.8858**	27.2038	27.0081	**26.9304**
	NRQM↑	4.9976	5.1893	**5.2307**	5.5270	5.6576	**5.7643**

Table 5: Ablation of the two components: sequence-wise quantizer bounds ensembling (*Seq-wise*) and temporal refinement with a temporal consistency loss (*TempRef*).

Method	PSNR↑	SSIM↑	LPIPS↓	TLPIPS↓	TOF↓
Global (single range)	23.1268	0.4689	0.5555	15.3454	55.4030
Seq-wise	23.8104	0.5479	0.4915	11.3696	27.5246
Seq-wise + TempRef	23.8949	0.5480	0.4724	10.0249	23.6122

dropping by 3.98, and TOF falling by 27.9, mitigating the distributional mismatch caused by heterogeneous sequences. Adding temporal refinement further improves temporal stability, with TLPIPS dropping by an additional 1.34 and TOF by 3.9, while modestly improving image quality. Overall, combining both components yields the best results across all metrics, confirming that the two are complementary.

Calibration time. Tab. 6 reports the total calibration time on an A6000 GPU for 9 clips. TAQ takes 32 minutes, slightly faster than 2DQuant (36 minutes), despite including sequence-wise initialization and temporal refinement.

Table 6: Total calibration time. A6000, 9 clips.

Method	MinMax	2DQuant	TAQ
Time (min)↓	3	36	32

Effect of percentile clipping. Tab. 7 shows the effect of 99.9% percentile clipping before histogram construction. Without clipping, extreme activation tails dominate the range estimation, leading to overly wide quantization bounds and larger rounding errors for normal activations. Percentile clipping suppresses these rare outliers, stabilizes sequence-wise bound estimation, and makes the final ensembling less sensitive to atypical clips. As a result, clipping improves all

Table 7: Effect of percentile clipping (99.9%). 4-bit RealViformer on REDS. Clipping suppresses extreme tails before histogram construction.

Method	PSNR↑	SSIM↑	LPIPS↓	TLPIPS↓	TOF↓
2DQuant (no clip)	22.3764	0.4747	0.5150	17.8592	34.9974
2DQuant (99.9%)	23.1155	0.5306	0.4828	12.9041	29.0666
TAQ (no clip)	22.6066	0.4823	0.5047	11.2559	30.2009
TAQ (99.9%)	23.8949	0.5480	0.4724	10.0249	23.6122

Table 8: Ensembling strategy. 4-bit RealViformer on REDS. Comparison of bound aggregation methods after per-sequence temporal refinement.

Method	PSNR↑	SSIM↑	LPIPS↓	TLPIPS↓	TOF↓
Median	23.0800	0.5355	0.4821	10.1316	23.6815
Trimmed mean	23.8851	0.5451	0.4786	10.0418	24.2053
Mean	23.8949	0.5480	0.4724	10.0249	23.6122

metrics for both 2DQuant and TAQ, confirming its importance for robust static VSR quantization.

Ensembling strategy. Tab. 8 compares three aggregation strategies: median, trimmed mean (removing the most extreme sequence), and arithmetic mean. TAQ does not average model weights; it only consolidates the clip-specific quantizer bounds (ℓ, u) into one static deployable quantizer set. This is sufficient for static deployment because the sequence-wise bounds already encode clip-specific activation statistics and the temporal refinement signal, while the final aggregation reduces the variance of heterogeneous sequence optima without requiring any runtime adaptation. Median discards distributional spread, yielding the lowest PSNR and highest LPIPS. Trimmed mean matches the arithmetic mean on temporal metrics but slightly underperforms on PSNR and SSIM; since each per-sequence histogram is already percentile-clipped at 99.9%, additional outlier removal offers marginal benefit. We also considered learned aggregation weights, but they slightly improved PSNR at the cost of worse temporal consistency (23.8996/10.2315 vs. 23.8949/10.0249 in PSNR/TLPIPS), suggesting overfitting to the small calibration set and its fidelity proxy. Therefore, the arithmetic mean achieves the best overall balance across all five metrics, supporting its use as the default static aggregator for simplicity, stability, and deployability.

Calibration set size. Tab. 9 ablates the number and diversity of calibration sequences under the 4/4-bit setting, using $N = 5, 9, 20$ REDS calibration sequences and Mix20, which combines 10 REDS and 10 Vimeo-90K sequences. TAQ consistently outperforms MinMax and 2DQuant across all budgets on both REDS and distribution-shifted UDM10. Even with only 5 sequences, TAQ reduces TLPIPS from 14.09 to 10.20 on REDS and from 23.35 to 16.27 on UDM10. Performance improves smoothly with larger budgets, while Mix20 remains competitive on UDM10, showing that diverse calibration clips can stabilize the ensembled bounds. These results confirm that 9 sequences are a sufficient and practical operating point.

Table 9: Calibration budget ablation. REDS N: N seqs from REDS val or train set; Mix20: REDS 10 + Vimeo-90K 10.

Calib count	Dataset	Metric	MinMax	2DQuant	TAQ (ours)
Bit (W/A)			4/4	4/4	4/4
REDS 5	REDS	PSNR/SSIM↑	12.75/0.083	22.91/0.503	**23.70**/**0.520**
		LPIPS/TLPIPS↓	1.204/22.18	0.496/14.09	**0.476**/**10.20**
	UDM10	PSNR/SSIM↑	13.40/0.097	26.08/0.652	**26.82**/**0.688**
		LPIPS/TLPIPS↓	1.299/31.55	0.518/23.35	**0.493**/**16.27**
REDS 9	REDS	PSNR/SSIM↑	10.73/0.144	23.11/0.531	**23.89**/**0.548**
		LPIPS/TLPIPS↓	0.852/36.15	0.483/12.90	**0.472**/**10.02**
	UDM10	PSNR/SSIM↑	13.09/0.105	27.15/0.696	**27.23**/**0.724**
		LPIPS/TLPIPS↓	1.309/30.49	0.476/19.72	**0.460**/**15.57**
REDS 20	REDS	PSNR/SSIM↑	12.88/0.096	23.45/0.539	**24.10**/**0.570**
		LPIPS/TLPIPS↓	1.188/27.08	0.488/11.67	**0.463**/**9.28**
	UDM10	PSNR/SSIM↑	14.06/0.135	27.33/0.729	**27.63**/**0.754**
		LPIPS/TLPIPS↓	1.279/39.86	0.453/17.51	**0.439**/**13.24**
Mix 20	REDS	PSNR/SSIM↑	12.77/0.063	23.40/0.544	**24.07**/**0.564**
		LPIPS/TLPIPS↓	1.242/26.69	0.486/10.97	**0.469**/**9.19**
	UDM10	PSNR/SSIM↑	13.82/0.077	27.43/0.741	**27.58**/**0.749**
		LPIPS/TLPIPS↓	1.341/38.61	0.450/15.94	**0.447**/**13.20**

5 Conclusion

To accelerate video super-resolution on real edge hardware, PTQ must rely on static quantization that avoids input-dependent recomputation of activation ranges at inference time. In this work, we propose a novel method, Temporal-Aware Quantization (TAQ), that satisfies this deployment constraint by calibrating sequence-specific bounds, refining them with a temporal-consistency objective based on inter-frame difference alignment, and consolidating the refined optima into a single deployable set of static quantization parameters via bound ensembling. Across REDS, SPMCS, UDM10, and the in-the-wild VideoLQ benchmark, TAQ consistently improves both perceptual quality and temporal stability over static PTQ baselines, reducing LPIPS by up to 0.0334 and TLPIPS by up to 4.7344 under identical calibration conditions. Crucially, we show that static quantization is a deployment requirement rather than a mere modeling preference: on NVIDIA Jetson Orin Nano with TensorRT, TAQ achieves up to 3.56× speedup (1.15 fps), whereas an input-dependent dynamic INT8 baseline attains only 0.79× speedup (0.266 fps vs. 0.325 fps for FP32), i.e., it can be slower than the original model in practice. TAQ also generalizes across architectures, outperforming strong baselines on both transformer and CNN backbones.

Acknowledgements

This work was supported by the LG CTO, and supported by the Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (o.RS-2025-02219317, AI Star Fellowship(Kookmin University)), and the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (RS-2025-23524783).

References

1. Chan, K.C., Wang, X., Yu, K., Dong, C., Loy, C.C.: Basicvsr: The search for essential components in video super-resolution and beyond. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4947–4956 (2021) 4, 11
2. Chan, K.C., Zhou, S., Xu, X., Loy, C.C.: Basicvsr++: Improving video super-resolution with enhanced propagation and alignment. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5972–5981 (2022) 4
3. Chan, K.C., Zhou, S., Xu, X., Loy, C.C.: Investigating tradeoffs in real-world video super-resolution. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5962–5971 (2022) 1, 8, 12
4. Choi, J., Wang, Z., Venkataramani, S., Chuang, P.I.J., Srinivasan, V., Gopalakrishnan, K.: Pact: Parameterized clipping activation for quantized neural networks. arXiv preprint arXiv:1805.06085 (2018) 4
5. Chu, M., Xie, Y., Mayer, J., Leal-Taixé, L., Thuerey, N.: Learning temporal coherence via self-supervision for gan-based video generation. ACM Transactions on Graphics (TOG) **39**(4), 75–1 (2020) 9
6. Esser, S.K., McKinstry, J.L., Bablani, D., Appuswamy, R., Modha, D.S.: Learned step size quantization. arXiv preprint arXiv:1902.08153 (2019) 4
7. Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., Kalenichenko, D.: Quantization and training of neural networks for efficient integer-arithmetic-only inference. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2704–2713 (2018) 4, 9
8. Jo, Y., Oh, S.W., Kang, J., Kim, S.J.: Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3224–3232 (2018) 4
9. Jung, S., Son, C., Lee, S., Son, J., Han, J.J., Kwak, Y., Hwang, S.J., Choi, C.: Learning to quantize deep networks by optimizing quantization intervals with task loss. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4350–4359 (2019) 4
10. Lee, J., Kim, D., Ham, B.: Network quantization with element-wise gradient scaling. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 6448–6457 (2021) 4
11. Li, R., Wang, Y., Liang, F., Qin, H., Yan, J., Fan, R.: Fully quantized network for object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2810–2819 (2019) 4, 7, 9
12. Li, X., Liu, Y., Cao, S., Chen, Z., Zhuang, S., Chen, X., He, Y., Wang, Y., Qiao, Y.: Diffvsr: Revealing an effective recipe for taming robust video super-resolution against complex degradations. arXiv preprint arXiv:2501.10110 (2025) 1
13. Li, Y., Gong, R., Tan, X., Yang, Y., Hu, P., Zhang, Q., Yu, F., Wang, W., Gu, S.: Brecq: Pushing the limit of post-training quantization by block reconstruction. arXiv preprint arXiv:2102.05426 (2021) 4
14. Liang, J., Cao, J., Fan, Y., Zhang, K., Ranjan, R., Li, Y., Timofte, R., Van Gool, L.: Vrt: A video restoration transformer. IEEE Transactions on Image Processing **33**, 2171–2182 (2024) 1, 4
15. Liang, J., Fan, Y., Xiang, X., Ranjan, R., Ilg, E., Green, S., Cao, J., Zhang, K., Timofte, R., Gool, L.V.: Recurrent video restoration transformer with guided de-

formable attention. Advances in Neural Information Processing Systems **35**, 378–393 (2022) 1, 2, 4
16. Liu, K., Qin, H., Guo, Y., Yuan, X., Kong, L., Chen, G., Zhang, Y.: 2dquant: Low-bit post-training quantization for image super-resolution. Advances in Neural Information Processing Systems **37**, 71068–71084 (2024) 4, 6, 9
17. Ma, C., Yang, C.Y., Yang, X., Yang, M.H.: Learning a no-reference quality metric for single-image super-resolution. Computer Vision and Image Understanding **158**, 1–16 (2017) 9
18. Migacz, S.: 8-bit inference with tensorrt. In: GPU technology conference. vol. 2 (2017) 4
19. Nagel, M., Amjad, R.A., Van Baalen, M., Louizos, C., Blankevoort, T.: Up or down? adaptive rounding for post-training quantization. In: International conference on machine learning. pp. 7197–7206. PMLR (2020) 4
20. Nagel, M., Baalen, M.v., Blankevoort, T., Welling, M.: Data-free quantization through weight equalization and bias correction. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 1325–1334 (2019) 4
21. Nah, S., Baik, S., Hong, S., Moon, G., Son, S., Timofte, R., Mu Lee, K.: Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. pp. 0–0 (2019) 8
22. NVIDIA Corporation: NVIDIA Jetson Orin Nano super developer kit. https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/nano-super-developer-kit/ (2025), accessed: 2026-03-03 3
23. Sajjadi, M.S., Vemulapalli, R., Brown, M.: Frame-recurrent video super-resolution. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6626–6634 (2018) 4
24. Tao, X., Gao, H., Liao, R., Wang, J., Jia, J.: Detail-revealing deep video super-resolution. In: Proceedings of the IEEE international conference on computer vision. pp. 4472–4480 (2017) 8
25. Teed, Z., Deng, J.: Raft: Recurrent all-pairs field transforms for optical flow. In: European conference on computer vision. pp. 402–419. Springer (2020) 9
26. Tu, Z., Hu, J., Chen, H., Wang, Y.: Toward accurate post-training quantization for image super resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5856–5865 (2023) 4, 6, 9
27. Wang, X., Chan, K.C., Yu, K., Dong, C., Change Loy, C.: Edvr: Video restoration with enhanced deformable convolutional networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. pp. 0–0 (2019) 4
28. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing **13**(4), 600–612 (2004) 9
29. Xue, T., Chen, B., Wu, J., Wei, D., Freeman, W.T.: Video enhancement with task-oriented flow. International Journal of Computer Vision **127**(8), 1106–1125 (2019) 11
30. Yang, X., Xiang, W., Zeng, H., Zhang, L.: Real-world video super-resolution: A benchmark dataset and a decomposition based learning scheme. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4781–4790 (2021) 4
31. Yi, P., Wang, Z., Jiang, K., Jiang, J., Ma, J.: Progressive fusion video super-resolution network via exploiting non-local spatio-temporal correlations. In: Pro-

ceedings of the IEEE/CVF international conference on computer vision. pp. 3106–3115 (2019) 8
32. Zhang, K., Liang, J., Van Gool, L., Timofte, R.: Designing a practical degradation model for deep blind image super-resolution. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4791–4800 (2021) 4
33. Zhang, L., Zhang, L., Bovik, A.C.: A feature-enriched completely blind image quality evaluator. IEEE Transactions on Image Processing **24**(8), 2579–2591 (2015) 9
34. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 586–595 (2018) 9
35. Zhang, Y., Yao, A.: Realviformer: Investigating attention for real-world video super-resolution. In: European Conference on Computer Vision. pp. 412–428. Springer (2024) 1, 4, 9
36. Zhang, Z., Shang, F., Liu, H., Wan, L., Feng, W., Hui, Y.: Qbasicvsr: Temporal awareness adaptation quantization for video super-resolution. In: The Thirty-ninth Annual Conference on Neural Information Processing Systems 2, 4, 11