DRAFTATTENTION2: FAST VIDEO DIFFUSION WITH LOW-RESOLUTION-GUIDED MIXED-PRECISION ATTENTION

Anonymous authors
Paper under double-blind review

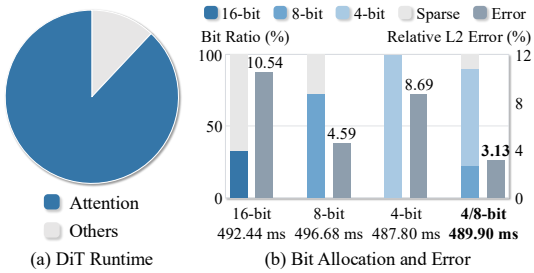
ABSTRACT

Video generation has broad applications in content creation and entertainment. Diffusion transformers have advanced the quality of generated videos, but attention over spatiotemporal tokens becomes increasingly expensive as video resolution and duration increase. We present DraftAttention2, a training-free framework that uses the low-resolution draft attention map to jointly select attention blocks and assign their numerical precision. Specifically, spatial 2D average- and max-pooled queries and keys capture complementary regional statistics to estimate block importance, and a shared ranking assigns higher precision to important blocks, lower precision to less important retained blocks, and skips the rest under configurable budgets. Our analysis separates sparsification error from attention-weighted quantization error, establishing when recovering skipped interactions with low-bit computation tightens the output-error bound. This analysis motivates retaining more interactions at low precision while reserving higher precision for blocks with larger attention mass. To translate these fine-grained assignments into practical speedups, we further develop fused operand preparation and a single attention kernel with precision-specific phases, sharing data movement, softmax statistics, and output accumulation across precisions. Experiments demonstrate that our method achieves a superior quality–efficiency trade-off over existing efficient video generation methods. Notably, its advantage is particularly pronounced for few-step video diffusion, where jointly combining sparsity with 4- and 8-bit mixed-precision computation substantially improves generation quality while retaining significant acceleration.

1 INTRODUCTION

Recent advances in video generation have enabled the synthesis of videos with increasingly realistic appearance and coherent motion (Kong et al., 2024; Wan et al., 2025). These models build on Diffusion Transformers (DiTs) (Peebles & Xie, 2023), using attention to model spatial and temporal dependencies. However, extending these capabilities to longer videos and higher resolutions remains computationally expensive. A major bottleneck is attention over spatiotemporal tokens: its computational cost scales quadratically with sequence length, and the computation is repeated at every denoising step. Figure 1(a) illustrates this bottleneck, with attention dominating the DiT runtime.

The sparsity offers a direct way to reduce this cost by retaining a subset of attention interactions as discussed in the previous works (Xi et al., 2025; Xia et al., 2025; Shen et al., 2025). Meanwhile, the low-precision computation provides a complementary opportunity: higher matrix-multiplication throughput at lower precisions translates into faster attention. Low-bit attention exploits this hardware capability through quantization (Zhang et al., 2025a;b;c). Combining sparsity with quantization enables sparse low-bit attention (Zhang et al., 2025d), while se-



(a) DiT Runtime (b) Bit Allocation and Error

Figure 1: Attention cost and precision allocation. (a) DiT runtime breakdown. (b) Block allocation and relative L2 output error at similar latency.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

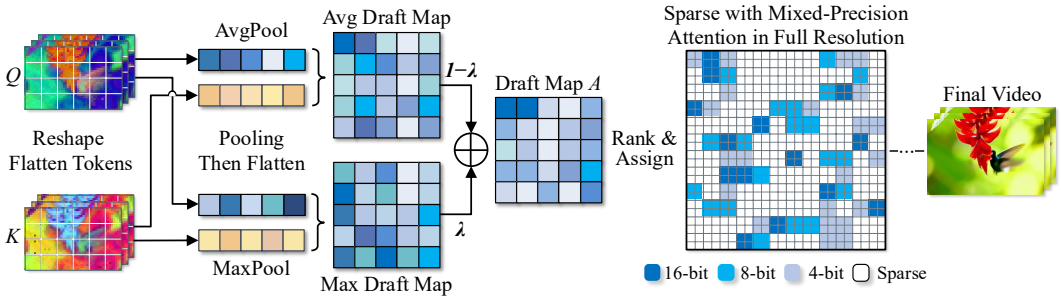


Figure 2: Overview of DraftAttention2. Average- and max-pooled queries and keys produce draft attention maps, whose weighted mixture guides block selection and precision allocation.

lective mixed-precision methods further improve accuracy by assigning higher precision to important interaction blocks (Sharratt, 2026).

Building on block-sparse attention, we generalize the binary retain-or-skip decision into a joint sparsity–precision allocation, where each block is skipped or computed at a selected numerical precision. Low-precision computation replaces part of the sparsification, while reserving a small high-precision budget for the most important blocks. Figure 1(b) illustrates this trade-off: at similar measured latency, mixed 4/8-bit attention achieves lower output error than the single-precision sparse and dense allocations. This design introduces two key challenges: efficiently estimating block importance to guide both block selection and precision assignment, and executing heterogeneous precisions without incurring substantial switching or output-merging overhead.

Framework overview. We present **DraftAttention2**, a training-free framework for sparse and mixed-precision attention in video diffusion (Figure 2). The key idea is to construct a low-resolution draft attention map from complementary average- and max-pooled queries and keys, and use it to jointly guide block selection and precision allocation. The resulting regional importance scores are ranked across all region pairs within each attention head: higher-scoring blocks are assigned higher precision, lower-ranked retained blocks use lower precision, and the rest are skipped under configurable budgets. This shared ranking allows both the number of retained blocks and their precision assignments to adapt across query regions under a common head-level budget. Finally, spatial token reordering maps these regional routing decisions to contiguous attention blocks for efficient full-resolution computation.

Theoretical analysis. Our analysis connects joint sparsity–precision allocation to attention-output error. Average pooling preserves mean regional logits, while the max pooling supplies variation information beyond the mean, motivating the combined draft. We then decompose the output error into sparsification and attention-weighted quantization terms. This decomposition shows that computing moderately important interactions at low precision can be a particularly favorable replacement for sparsification: recovering these interactions with low-bit computation can substantially tighten the output-error bound relative to the original DraftAttention formulation. Finally, under fixed precision budgets, we show that assigning higher precision to interactions with larger attention mass minimizes the resulting error bound, providing theoretical justification for the draft-guided precision allocation.

Low-overhead mixed-precision execution. To efficiently execute the resulting fine-grained precision assignments, we fuse spatial reordering, pooling, and quantization into the preparation of draft descriptors and full-resolution attention operands. Retained blocks are grouped by precision and processed in consecutive phases within a single attention kernel, so the number of precision transitions depends only on the number of active precision formats rather than the number of blocks. All phases share online softmax statistics and a common output accumulator, with lightweight rescaling at phase boundaries. This design avoids separate per-precision outputs and an additional merging kernel, substantially reducing the overhead of mixed-precision execution.

Experimental results. We evaluate DraftAttention2 on HunyuanVideo and MiniMax-H3, measuring video generation quality, similarity to dense-attention outputs, and DiT inference speed. The results show that draft-guided sparse and mixed-precision attention accelerates video generation with competitive quality. Allocating a small fraction of blocks to higher precision improves output similarity while preserving most of the speed benefit of low-precision computation, making precision allocation an effective way to control the quality–speed trade-off. Notably, on few-step video

diffusion, where approximation errors are less likely to be corrected over repeated denoising steps, DraftAttention2 exhibits a substantially stronger advantage over existing efficient attention methods, demonstrating its effectiveness in this more practical generation setting.

Our contributions are summarized as follows:

- We introduce DraftAttention2, which uses low-resolution guidance to jointly allocate sparsity and numerical precision across attention blocks under configurable budgets. Our analysis shows that computing moderately important interactions at low precision offers a particularly favorable error–efficiency trade-off: replacing their sparsification with low-bit computation can substantially tighten the output-error bound, leading to an overall tighter error bound than the previous DraftAttention.
- We develop an execution scheme that supports multiple precisions within the same attention computation with low overhead. Fused preparation shares data movement between draft construction and operand quantization, while precision-specific phases share normalization and output accumulation.
- Experiments on HunyuanVideo and MiniMax-H3 demonstrate faster DiT inference with competitive generation quality and show that a small high-precision budget improves fidelity to dense-attention outputs. DraftAttention2 further delivers particularly strong gains on few-step video diffusion, substantially improving generation quality over existing efficient attention methods while maintaining significant speedup.

2 Related Work

Sparse attention for video generation. Video diffusion models exhibit local spatiotemporal structure and data-dependent attention sparsity. Sparse VideoGen (Xi et al., 2025) uses online profiling to select spatial or temporal attention patterns for each head. DraftAttention (Shen et al., 2025) uses spatially average-pooled queries and keys to select blocks for full-resolution sparse attention. AdaSpa (Xia et al., 2025) searches for block-sparse masks online and reuses them between designated search steps. Sparse VideoGen2 (Yang et al., 2025) clusters and reorders query and key tokens, using cluster-centroid attention estimates for top-p selection.

Low-bit and mixed-precision attention. SageAttention2 (Zhang et al., 2025a) combines per-thread INT4 quantization of queries and keys with FP8 quantization of attention weights and values. SageAttention3 (Zhang et al., 2025b) applies NVFP4 quantization to both attention matrix multiplications, with two-level scaling of the attention weights. SpargeAttention (Zhang et al., 2025d) combines quantization with block selection that skips both matrix multiplications and an online softmax-aware filter that further skips value accumulation. Block-level mixed-precision methods allocate higher precision to selected interactions. Diagonal-Tiled Mixed-Precision Attention (DMA) (Ding et al., 2026) retains higher-precision query–key computation near the diagonal. BMAttn (Wang et al., 2026) and ThriftAttention (Sharratt, 2026) both adopt block-level mixed precision, with BMAttn partitioning attention into high-precision, low-precision, and skipped regions using offline-calibrated per-head boundaries, while ThriftAttention uses pooled queries and keys to assign FP16 to top-k key blocks and FP4 to the remaining unmasked blocks. Our method jointly selects attention blocks and assigns them multiple precisions using a shared low-resolution importance ranking over all blocks in each attention head. This allows both the number of retained blocks and their precision allocation to vary across query regions under a common per-head budget.

3 Methodology

As shown in Figure 2, DraftAttention2 extends block selection to a joint allocation of sparsity and precision. A low-resolution draft attention map ranks regional interactions, assigning higher precision to important blocks, lower precision to other retained blocks, and skipping the remainder. We first describe this allocation, then relate it to attention-weighted quantization error, and finally show how to execute the resulting mixed-precision attention with shared operand preparation and normalization.

3.1 Draft-Guided Precision Allocation

For one attention head, let $Q, K, V \in \mathbb{R}^{N\times d}$, with dense attention $P = \mathrm{softmax}_{\mathrm{row}}(QK^\top/\sqrt{d})$ and output $O^{\mathrm{dense}} = PV$. Following DraftAttention (Shen et al., 2025), we estimate regional importance before evaluating token-level attention.

Spatial grouping and draft attention. Spatially neighboring video tokens provide a natural unit for pooling and block-wise computation. We partition each frame into connected regions $\{\mathcal{R}_a\}$, with balanced token counts bounded by the attention block capacity. Section 3.3 describes the spatial partition and its alignment with attention blocks.

For each region, we form an average descriptor and a channel-wise maximum descriptor:

$$\bar{Q}_a = \frac{1}{|\mathcal{R}_a|} \sum_{u \in \mathcal{R}_a} Q_u, \qquad Q^{\max}_{a,c} = \max_{u \in \mathcal{R}_a} Q_{u,c}, \tag{1}$$

with $\bar{K}$ and $K^{\max}$ defined analogously. We compute a draft map from each pair of pooled query and key tensors, normalize each map over key regions, and combine them to obtain the block importance scores.

Definition 3.1 (Draft-Guided Precision Allocation)**.** Let $\bar{Q}, \bar{K}$ and $Q^{\max}, K^{\max}$ be the regional average and channel-wise maximum descriptors. For a pooling weight $\lambda \in [0, 1]$, define

$$A = (1-\lambda)\,\mathrm{softmax}_{\mathrm{row}}\left(\frac{\bar{Q}\bar{K}^\top}{\sqrt{d}}\right) + \lambda\,\mathrm{softmax}_{\mathrm{row}}\left(\frac{Q^{\max}(K^{\max})^\top}{\sqrt{d}}\right). \tag{2}$$

Given head-level precision quotas T_{16}, T_8, T_4, rank video-region pairs by A_{ab}. Assign the first T_{16} pairs to the 16-bit path, the next T_8 to INT8, and the next T_4 to NVFP4; skip the rest.

Remark 3.2 (Complementary Regional Statistics). By bilinearity, $\bar{Q}_a^\top \bar{K}_b/\sqrt{d}$ equals the mean token-pair logit between regions $\mathcal{R}_a$ and $\mathcal{R}_b$. Regions with the same average descriptor can nevertheless have different channel-wise maxima. The Max branch exposes these variations to the draft, while λ controls their contribution; $\lambda = 0$ recovers average-only guidance.

Algorithm 1: Draft-guided sparsity and precision allocation

Input: Q, K; regions $\{\mathcal{R}_a\}_{a=1}^{R}$; λ; quotas T_{16}, T_8, T_4
Output: Precision assignments $p \in \{0, 4, 8, 16\}^{R \times R}$

$\bar{Q}, \bar{K} \leftarrow \mathrm{AvgPool}(Q, \{\mathcal{R}_a\}), \mathrm{AvgPool}(K, \{\mathcal{R}_a\})$
$Q^{\max}, K^{\max} \leftarrow \mathrm{MaxPool}(Q, \{\mathcal{R}_a\}), \mathrm{MaxPool}(K, \{\mathcal{R}_a\})$
$A \leftarrow (1-\lambda)\,\mathrm{softmax}_{\mathrm{row}}(\bar{Q}\bar{K}^\top/\sqrt{d}) + \lambda\,\mathrm{softmax}_{\mathrm{row}}(Q^{\max}(K^{\max})^\top/\sqrt{d})$
$I \leftarrow \mathrm{argsort}(\mathrm{vec}(A),\ \mathrm{descending} = \mathrm{True},\ \mathrm{stable} = \mathrm{True})$
$T \leftarrow T_{16} + T_8 + T_4$; $p \leftarrow \mathbf{0}_{R^2}$
$p[I[:T_{16}]] \leftarrow 16$
$p[I[T_{16}:T_{16}+T_8]] \leftarrow 8$
$p[I[T_{16}+T_8:T]] \leftarrow 4$
return $\mathrm{reshape}(p, R, R)$

Joint allocation of sparsity and precision. For R video regions, the draft contains R^2 scores, replacing token-level scoring for the allocation decision. The total retained budget is $T = T_{16} + T_8 + T_4$, giving a retained fraction of T/R^2. The individual quotas specify how many blocks use each precision. Since ranking is performed across the whole map within each head, query regions can receive different numbers of retained blocks and different precision allocations under the same head-level budget.

Each selected region pair is then evaluated at full resolution using its assigned precision. All retained blocks for a query contribute to a common softmax normalization and output, including blocks assigned to different precisions. The pooled descriptors guide this computation; the output is formed from the full-resolution values. Algorithm 1 gives the draft-guided allocation, where $p_{ab} \in \{0, 4, 8, 16\}$ specifies the precision of region pair (a, b) and 0 denotes a skipped pair. Section 3.3 describes attention computation with these assignments. Routing details and numerical formats are given in Appendix C.

3.2 Quantization and Attention Importance

DraftAttention selects which interactions to compute at the reference precision. DraftAttention2 also chooses how accurately to compute them: a skipped block can instead contribute through a low-bit path. We analyze when the recovered attention mass outweighs the additional quantization error, and how precision should be distributed among retained blocks.

Illustrative example. Figure 1(b) compares four allocations on the same captured video-attention input, using average-only draft guidance ($\lambda = 0$). At similar measured latency, sparse 16-bit attention loses information through omitted interactions, whereas dense 4-bit attention retains every interaction but incurs larger quantization error. The mixed 4/8-bit allocation balances these effects, reducing relative output error to 3.132%. The 16-bit label denotes the reference-precision path; the analysis isolates the additional error from low-bit quantization.

From sparsification to quantization error. Quantization perturbs both the attention logits and the weighted value aggregation. We take the original BF16 inputs as the reference and leave the 16-bit path unquantized in the reconstruction model; ordinary floating-point roundoff is treated separately in Appendix B.4. For a query u with nonempty retained support Ω_u, let P^{Ω}_{uv} be the exact attention probability renormalized over Ω_u. Let $\Omega^{(p)}_u$ contain the retained keys assigned precision p, including any retained prefix interactions. Define

$$m^{(p)}_u = \sum_{v \in \Omega^{(p)}_u} P^{\Omega}_{uv}, \qquad \delta_u = \sum_{v \notin \Omega_u} P_{uv}, \qquad \sum_{p \in \{4,8,16\}} m^{(p)}_u = 1. \tag{3}$$

Here δ_u measures omitted dense attention mass, while $m^{(p)}_u$ measures the share of retained attention assigned to each precision.

Lemma 3.3 (Attention-Weighted Quantization Error)**.** *Suppose $\|V_v\|_2 \le V_{\max}$ and the reconstructed operands satisfy the error bounds in Appendix B.1. Let ε_p be the uniform error envelope for low-bit precision $p \in \{4, 8\}$ in Equation* (12)*. The reconstructed mixed-precision output satisfies*

$$\|O^{\text{dense}}_u - \widehat{O}_u\|_2 \le \underbrace{2V_{\max}\delta_u}_{\textit{sparsification}} + \underbrace{m^{(8)}_u \varepsilon_8 + m^{(4)}_u \varepsilon_4}_{\textit{quantization}}. \tag{4}$$

The quantization term weights each precision's error by the attention mass computed at that precision. The envelopes account jointly for logit, value, and probability-weight reconstruction, consistent with the shared normalization used by the kernel. The two terms expose the choice between omitting an interaction and retaining a quantized approximation to it.

Proposition 3.4 (From Sparse to Mixed-Precision Attention)**.** *Compare a DraftAttention (v1) sparse allocation with a DraftAttention2 allocation that preserves the same 16-bit blocks and computes additional blocks at 8 or 4 bits. For one query, let δ_{v2} be the dense attention mass skipped by v2 and τ_p the dense mass assigned to its p-bit path, $p \in \{4, 8\}$. Assume $0 < \tau = \tau_8 + \tau_4 < 1 - \delta_{\text{v2}}$, so the shared 16-bit support is nonempty. Lemma 3.3 gives the respective error bounds*

$$B_{\text{v1}} = 2V_{\max}(\delta_{\text{v2}} + \tau), \qquad B_{\text{v2}} = 2V_{\max}\delta_{\text{v2}} + \frac{\tau_8\varepsilon_8 + \tau_4\varepsilon_4}{1 - \delta_{\text{v2}}}. \tag{5}$$

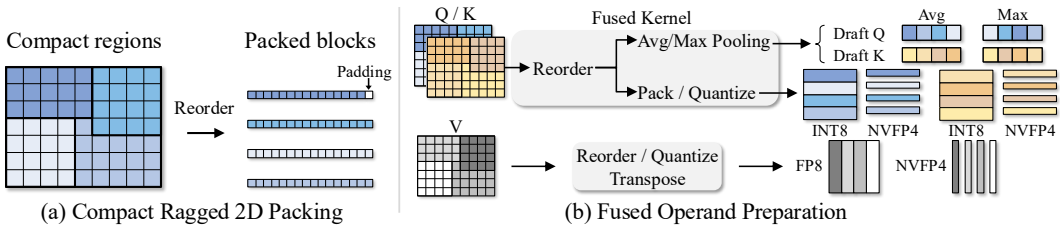
In particular, $B_{\text{v2}} < B_{\text{v1}}$ whenever

$$\tau_8\varepsilon_8 + \tau_4\varepsilon_4 < 2V_{\max}\tau(1 - \delta_{\text{v2}}). \tag{6}$$

Remark 3.5 (Interpreting the Comparison). The two allocations preserve the same reference-precision blocks but have different omitted attention masses: $\delta_{\text{v1}} = \delta_{\text{v2}} + \tau_4 + \tau_8$. Low-bit recovery reduces the sparsification bound by $2V_{\max}\tau$ and introduces the quantization term in Equation (5); the bound improves when the latter is smaller than the former. For these matched allocations, the skipped fractions of video-region pairs satisfy $\rho_{\text{v1}} = \rho_{\text{v2}} + \rho_4 + \rho_8$, where ρ_p is the fraction assigned to precision p by v2. These block fractions differ from the attention masses above. The comparison motivates computing moderately important interactions at low precision while leaving the least important interactions sparse. Figure 1(b) illustrates the resulting trade-off at similar latency.

Allocating precision within the retained support. Once the support is fixed, the same decomposition determines where higher precision is most useful. For a retained region pair (a, b), define its query-averaged mass $W_{ab} = N^{-1}_{\text{vid}} \sum_{u \in \mathcal{R}_a} \sum_{v \in \mathcal{R}_b} P^{\Omega}_{uv}$, where N_{vid} is the number of video queries.

Theorem 3.6 (Importance-Guided Precision Allocation)**.** *Fix the retained support and precision quotas. If $0 \le \varepsilon_8 \le \varepsilon_4$, then the assignment-dependent error bound $\sum_{(a,b):\, p_{ab} \in \{4,8\}} W_{ab}\varepsilon_{p_{ab}}$ is minimized by assigning the 16-bit path to the T_{16} largest masses, INT8 to the next T_8, and NVFP4 to the remainder.*



(a) Compact Ragged 2D Packing (b) Fused Operand Preparation

Figure 3: Compact spatial packing and fused operand preparation. (a) Compact spatial regions are packed into contiguous blocks. (b) Q/K reordering, pooling, and quantization share a kernel; V reordering and transposition are fused with operand packing and loading. Colors denote regions; gray cells denote padding.

The theorem motivates the precision ordering in Definition 3.1: larger attention mass makes an interaction more sensitive to quantization. Computing the exact masses requires full-resolution attention, so the router uses draft scores A_{ab} as low-cost proxies. Together, the results motivate retaining more interactions at low precision while reserving higher precision for the most important blocks. Proofs and the reconstruction-error model are in Appendix B.

3.3 Hardware-Efficient Execution

To efficiently execute the joint sparsity–precision allocation, spatial regions must align with contiguous attention blocks, while blocks assigned different precisions must share normalization and output accumulation. We address these requirements through fused operand preparation and phased mixed-precision execution.

Fused quantization and reordering. Tokens belonging to one spatial region can be scattered in raster order, whereas attention kernels operate on fixed-size contiguous blocks. We construct a ragged 2D partition with balanced token counts across regions, prioritizing the smallest total perimeter among candidate partitions (Figure 3). These criteria maximize block utilization while preserving 2D spatial locality. The partition adapts region shapes to the frame dimensions and uses the minimum number of blocks, with less than one block of total padding per frame. We pack each region into a contiguous block, using the same spatial indices for pooling and operand preparation. The partition is reused across layers and denoising steps.

Computing pooled descriptors and preparing each numerical format separately would repeatedly move the same token features through global memory. Our Q/K preparation kernel fuses spatial gathering, average- and max-pooling, and quantization, producing the draft descriptors together with packed operands and quantization scales. The value path similarly fuses spatial reordering with operand preparation. To reconcile the $QK^\top$ accumulator layout with the operand layout of the probability–value multiplication (Shah et al., 2024), we align the intra-block K/V layouts so that the attention weights can be converted and packed within each thread after softmax. Value transposition is incorporated into operand packing and loading. Thus, the data movement needed to prepare attention operands also supplies the draft descriptors, without standalone spatial reordering, pooling, or value-transposition kernels.

Phased execution. To limit precision switching, we group retained key blocks for each query block into NVFP4, INT8, and 16-bit segments. A single attention kernel processes the nonempty segments in this order, using the corresponding operand-loading and matrix-multiplication path. If n_{phase} phases are nonempty, the number of precision transitions is

$$n_{\text{switch}} = \max(n_{\text{phase}} - 1, 0) \le 2. \tag{7}$$

The number of precision transitions therefore depends only on the active formats, independently of the number of retained blocks. Consecutive phases also reuse a shared-memory workspace, with operand fragments and quantization temporaries local to each phase.

Cross-precision accumulation. To accumulate contributions from different precisions into a common attention output, all phases share FP32 row maxima, exponential sums, and output accumulators using online softmax (Dao, 2024). Let $\widetilde{P}$ denote the unnormalized exponential weights, and let U_c and

Table 1: Video generation quality and DiT speedup. VBench Avg is reported as a percentage. Bit allocations give percentages of 0-, 4-, 8-, and 16-bit attention blocks, where 0-bit denotes sparse and 16-bit denotes dense.

HunyuanVideo							**MiniMax-H3**						
Method	Bit (%) 0 / 4 / 8 / 16	VBench Avg↑	PSNR ↑	SSIM ↑	LPIPS ↓	Speedup ↑	Method	Bit (%) 0 / 4 / 8 / 16	VBench Avg↑	PSNR ↑	SSIM ↑	LPIPS ↓	Speedup ↑
Dense	0/0/0/100	77.06	–	–	–	1.00×	Dense	0/0/0/100	78.37	–	–	–	1.00×
XAttn	85.6/0/0/14.4	73.90	22.90	0.679	0.251	1.74×	XAttn	79.9/0/0/20.1	78.03	19.64	0.622	0.267	1.76×
SVG2	83.6/0/0/16.4	65.70	24.34	0.711	0.382	2.06×	SVG2	79.9/0/0/20.1	77.02	19.87	0.626	0.293	1.99×
Sol-Attn	85.8/0/0/14.2	76.81	25.21	0.805	0.178	2.16×	Sol-Attn	80/0/0/20	78.25	20.95	0.691	0.198	2.04×
SVG-EAR	85.2/0/0/14.8	66.49	25.03	0.739	0.348	2.06×	SVG-EAR	79.9/0/0/20.1	76.97	20.40	0.644	0.275	1.97×
SVOO	85.3/0/0/14.7	71.37	26.72	0.816	0.174	2.15×	SVOO	79.9/0/0/20.1	78.07	20.56	0.662	0.238	2.04×
Ours	85/0/0/15	74.55	26.19	0.822	0.151	2.28×	**Ours**	80/0/0/20	77.90	21.01	0.692	0.199	2.23×
Ours	85/0/7.5/7.5	74.70	26.23	0.824	0.149	2.38×	**Ours**	80/0/10/10	77.84	20.95	0.690	0.200	2.33×
SpargeAttn	85/0/15/0	74.36	26.02	0.824	0.159	2.50×	SpargeAttn	80/0/20/0	77.47	20.23	0.662	0.232	2.59×
Ours	85/0/15/0	74.83	25.20	0.797	0.155	2.50×	**Ours**	80/0/20/0	78.15	20.29	0.667	0.216	2.47×
Ours	80.7/14.4/4.8/0	75.96	26.37	0.814	0.157	2.51×	**Ours**	74.3/19.3/6.4/0	77.74	21.02	0.690	0.205	2.60×
SageAttn3	0/100/0/0	75.42	32.30	0.919	0.068	1.88×	SageAttn3	0/100/0/0	77.40	23.75	0.779	0.149	1.94×
Ours	13.4/64.9/21.6/0	75.75	34.77	0.945	0.043	1.90×	**Ours**	13.6/64.8/21.6/0	77.43	25.07	0.817	0.122	1.95×

ℓ be the output numerator in channel c and the softmax denominator. We rescale the accumulators at phase boundaries to account for the probability and value quantization scales. For example, the INT8 path uses E4M3 operands for $\widetilde{P}V$. With probability dequantization scale η and nonzero per-channel value dequantization scale δ_c^V, its entry and exit transformations are

$$\text{Enter:} \quad U_c^{\text{enc}} = \frac{U_c}{\eta \delta_c^V}, \quad \ell^{\text{enc}} = \frac{\ell}{\eta}, \qquad \text{Exit:} \quad U_c \;= \eta \delta_c^V U_c^{\text{enc}}, \ell \;= \eta \ell^{\text{enc}}. \tag{8}$$

The normalized output satisfies $U_c/\ell = \delta_c^V U_c^{\text{enc}}/\ell^{\text{enc}}$, so each phase can update the shared output accumulator after rescaling. The denominator accumulates exponential weights before their conversion to matrix operands, as in the reconstruction model of Section 3.2. We normalize once after all retained blocks, avoiding per-precision output writes and a separate merging kernel. Appendix C details the spatial layout, quantization, and phase-specific scaling.

4 Experiments

4.1 Experimental Setup

Model family. We evaluate DraftAttention2 on HunyuanVideo-13B (Kong et al., 2024) and MiniMax-H3 (MiniMaxAI, 2026), including 4-step and 8-step MiniMax-H3 variants using FL2VA Turbo LoRAs (ModelTC, 2026). We generate 129-frame videos at 1280×720 resolution with HunyuanVideo and 362-frame videos at 1344×768 resolution with MiniMax-H3 and its few-step variants. We retain dense attention for the first 10 denoising steps of the standard models and the first quarter of steps in the few-step variants. We compare with dense attention, XAttention (Xu et al., 2025), SVG2 (Yang et al., 2025), SVG-EAR (Zhou et al., 2026), SVOO (Luo et al., 2026), Sol-Attn (Li et al., 2026), SpargeAttention (Zhang et al., 2025d), and SageAttention3 (Zhang et al., 2025b). For clustering-based baselines, we select the numbers of query and key clusters to obtain comparable inference speedups.

Metrics and prompts. We evaluate the quality of generated videos with VBench (Huang et al., 2024) and their similarity to dense-attention outputs with PSNR, SSIM, and LPIPS (Zhang et al., 2018). We report eight VBench dimensions and their arithmetic mean. For similarity evaluation, we use dense references generated with matching prompts and random seeds. All videos are generated using prompts from VBench, and latency is measured on an NVIDIA RTX 5090 GPU.

4.2 Main Results

Higher performance on recent video diffusion models. Table 1 compares generation quality and DiT speedup on HunyuanVideo and MiniMax-H3. Compared with sparse-attention methods

Table 2: Video generation quality and DiT speedup for MiniMax-H3 with FL2VA Turbo LoRAs (1344×768, 362 frames). Bit allocations follow the 0 / 4 / 8 / 16 order; 0-bit denotes sparse and 16-bit denotes dense.

4-Step Diffusion							8-Step Diffusion						
Method	Bit (%) 0 / 4 / 8 / 16	VBench Avg↑	PSNR ↑	SSIM ↑	LPIPS ↓	Speedup ↑	Method	Bit (%) 0 / 4 / 8 / 16	VBench Avg↑	PSNR ↑	SSIM ↑	LPIPS ↓	Speedup ↑
Dense	0/0/0/100	74.14	–	–	–	1.00×	Dense	0/0/0/100	79.87	–	–	–	1.00×
Sol-Attn	65/0/0/35	74.19	19.54	0.664	0.191	1.46×	Sol-Attn	79.9/0/0/20.1	79.91	18.19	0.575	0.262	1.80×
SVG2	64/0/0/36	73.80	19.02	0.603	0.231	1.33×	SVG2	79.6/0/0/20.4	79.34	17.80	0.531	0.333	1.66×
SVG2	85.8/0/0/14.2	73.13	17.70	0.511	0.306	1.59×	SVG2	86.7/0/0/13.3	79.04	17.36	0.509	0.371	1.81×
SVG-EAR	66.4/0/0/33.6	73.81	19.39	0.619	0.224	1.32×	SVG-EAR	79.6/0/0/20.4	79.51	18.39	0.557	0.309	1.68×
SVG-EAR	86.1/0/0/13.9	73.73	18.28	0.543	0.283	1.59×	SVG-EAR	88.4/0/0/11.6	78.81	17.87	0.530	0.350	1.82×
SVOO	65/0/0/35	73.80	19.59	0.637	0.217	1.44×	SVOO	80.1/0/0/19.9	79.24	17.99	0.532	0.338	1.80×
SVOO	76.3/0/0/23.7	73.74	18.41	0.561	0.267	1.60×	**Ours**	80/0/0/20	79.88	18.75	0.596	0.247	1.84×
Ours	65/0/0/35	74.11	20.19	0.691	0.175	1.63×	**Ours**	74.3/0/12.8/12.8	79.85	18.85	0.601	0.243	1.87×
Ours	63.4/0/18.3/18.3	73.89	19.84	0.671	0.188	1.66×	**Ours**	57.9/0/42.1/0	79.69	19.15	0.615	0.245	1.87×
Ours	53.2/0/37.5/9.4	73.85	20.60	0.702	0.173	1.64×	**Ours**	40.5/59.5/0/0	79.73	19.75	0.633	0.240	1.87×
Ours	29.6/35.2/35.2/0	74.05	22.38	0.779	0.128	1.65×	**Ours**	50.8/24.6/24.6/0	79.82	20.76	0.690	0.188	1.87×
SpargeAttn	65/0/35/0	73.74	19.26	0.641	0.206	1.86×	SpargeAttn	80/0/20/0	79.70	18.15	0.566	0.277	2.12×
Ours	65/0/35/0	73.65	19.21	0.616	0.239	1.85×	**Ours**	80/0/20/0	79.81	18.46	0.582	0.262	2.12×
Ours	54.5/34.1/11.4/0	74.00	20.28	0.687	0.179	1.84×	**Ours**	74/13/13/0	79.89	18.83	0.599	0.247	2.12×
SageAttn3	0/100/0/0	74.00	21.06	0.721	0.162	1.65×	SageAttn3	0/100/0/0	79.72	20.99	0.693	0.195	1.73×
Ours	13.1/65.2/21.7/0	73.83	22.63	0.786	0.126	1.64×	**Ours**	13/65.3/21.8/0	79.80	22.57	0.758	0.152	1.73×

including XAttention, SVG2, Sol-Attn, SVG-EAR, and SVOO, DraftAttention2 achieves higher inference speed while maintaining competitive generation quality and fidelity to dense attention. For example, our sparse configurations reach up to $2.38\times$ and $2.33\times$ speedup on HunyuanVideo and MiniMax-H3, respectively, exceeding the speedups of these sparse-only baselines. We further introduce mixed-precision allocation to improve quality at higher acceleration. At speedups matched to SpargeAttention, DraftAttention2 consistently achieves better generation quality and dense-output fidelity on both HunyuanVideo and MiniMax-H3. We additionally compare against SageAttention3 under matched speedups, where DraftAttention2 achieves better generation quality with comparable acceleration. These results demonstrate that jointly allocating sparsity and numerical precision provides a better quality–efficiency trade-off than sparse-only or uniform low-precision attention.

Strong performance on few-step generation. Table 2 evaluates DraftAttention2 on 4- and 8-step MiniMax-H3 generation, where aggressive attention approximation is more challenging because fewer denoising steps are available to correct accumulated errors. Across both settings, DraftAttention2 achieves a stronger quality–efficiency trade-off than sparse-only baselines by replacing part of the sparsification with low-bit computation. For 4-step generation, our mixed-precision configurations closely preserve dense quality while substantially improving fidelity over sparse baselines at similar acceleration. For 8-step generation, DraftAttention2 exhibits the same trend, consistently improving generation quality and dense-output fidelity over sparse-only baselines while maintaining comparable or higher acceleration. At speedups comparable to SpargeAttention, mixed 4/8-bit allocation further improves dense-output fidelity. For SageAttention3, DraftAttention2 achieves better generation quality at comparable speedups, further demonstrating the advantage of selective sparsity–precision allocation over uniform low-bit attention. These results highlight the particular benefit of joint sparsity and mixed-precision allocation for practical few-step video generation, where preserving moderately important interactions becomes increasingly critical.

Visualization. Figure 4 compares DraftAttention2 with competing methods on few-step video generation. As highlighted by the red boxes, DraftAttention2 preserves the rail junction and the cat's appearance, producing results close to the dense baseline.

4.3 Ablation Study

We study pooling and precision allocation on a dense 4-step trajectory capture with $C = 128$. We measure relative L2 error against the dense output and complete attention-operator latency, including operand preparation and routing.

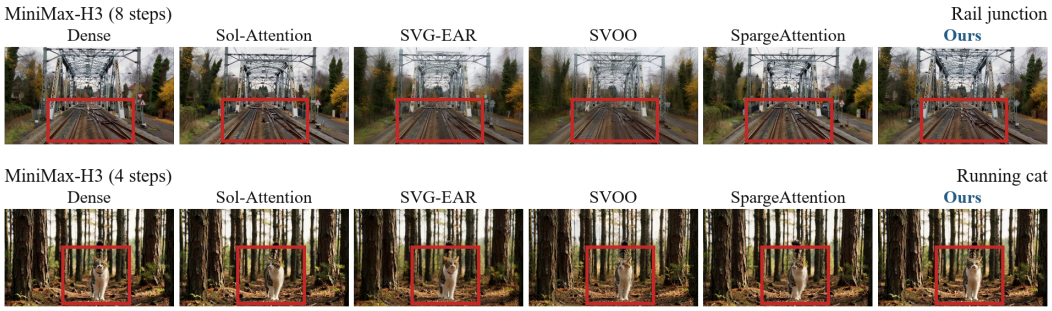


Figure 4: Visual comparison on 8-step (top) and 4-step (bottom) diffusion with MiniMax-H3 FL2VA Turbo LoRAs. Red boxes highlight structural details.

Table 3: MaxPool ablation on MiniMax-H3 at 80% sparsity. The 8/16-bit configuration assigns equal budgets to the two precisions.

Bit	Max Pool	VBench Avg. (%)↑	PSNR ↑	SSIM ↑	LPIPS ↓
16	w/o	78.20	20.89	0.687	0.202
	w/	77.90	21.01	0.692	0.199
8/16	w/o	77.68	20.82	0.684	0.204
	w/	77.84	20.95	0.690	0.200
8	w/o	78.11	20.15	0.660	0.221
	w/	78.15	20.29	0.667	0.216

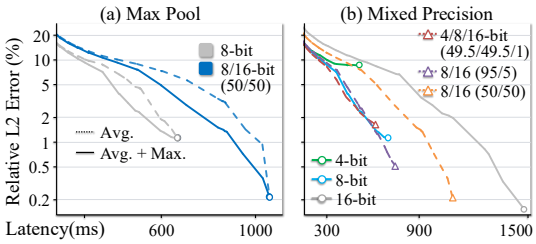


Figure 5: Attention output error versus latency. (a) Average pooling with and without MaxPool. (b) Single- and mixed-precision attention with MaxPool. Curves vary sparsity at fixed precision ratios; open circles mark zero sparsity. Ratios are percentages of retained connections.

Effect of MaxPool. Figure 5(a) compares average pooling with the average–max mixture ($\lambda = 0.2$). Adding MaxPool improves the error–latency trade-off for both 8-bit and mixed 8/16-bit attention. The improvement at a fixed block budget demonstrates the value of the complementary max-pooled draft for guiding block selection. Table 3 further shows consistent improvements in video generation performance at fixed sparsity and precision budgets. The improvements extend to both 16-bit and mixed 8/16-bit attention, showing that the additional guidance benefits different precision allocations.

Effect of mixed precision. Figure 5(b) shows the benefit of a small high-precision budget. Across a range of latency budgets, introducing a small fraction of higher-precision computation consistently lowers attention output error compared with single-precision alternatives. Selective high-precision computation thus reduces error beyond the zero-sparsity endpoint of pure 8-bit attention. Reducing sparsity alone cannot remove the quantization error that remains when every block is computed at low precision. Draft-guided precision allocation addresses this error by reserving higher precision for important interactions, improving accuracy within the measured attention latency budget.

5 Conclusion

We presented DraftAttention2, a training-free attention acceleration framework that uses a low-resolution draft to jointly allocate sparsity and numerical precision in video diffusion. The draft map provides a unified importance ranking for both block selection and precision assignment, enabling adaptive allocation under configurable computation budgets. Our theoretical analysis decomposes attention-output error into sparsification and attention-weighted quantization terms, showing that recovering selected sparse interactions with low-bit computation can substantially tighten the output-error bound and yield a tighter bound than the original DraftAttention under matched allocations. Fused operand preparation and shared accumulation across precision phases reduce the overhead of executing these fine-grained assignments, translating the proposed allocation into practical acceleration. Experimental results on HunyuanVideo and MiniMax-H3 demonstrate a strong quality–efficiency trade-off over existing efficient attention methods. Notably, DraftAttention2 shows particularly strong advantages on few-step video diffusion, where mixed-precision recovery substantially improves generation quality and dense-output fidelity while retaining significant acceleration.

AI USE STATEMENT

We used generative AI tools to assist with refining the presentation of theoretical arguments and mathematical claims, discussing research methodology and experimental design, and drafting and editing portions of the manuscript. We also used these tools for brainstorming, literature search, and improving the clarity and organization of the paper.

All AI-assisted work was critically reviewed and verified by the authors. Mathematical claims and theoretical arguments were independently checked, and all implementations, experimental settings, measurements, and reported results were validated by the authors. Generative AI was not used to generate or fabricate experimental data. We take full responsibility for the final content of this work, including all text, claims, code, results, and other artifacts produced with the aid of generative AI.

REFERENCES

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*, 2024.

Yifu Ding, Xinhao Zhang, Yinglei Wang, Yuanpei Chen, and Jinyang Guo. Diagonal-tiled mixed-precision attention for efficient low-bit MXFP inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 3420–3428, 2026. URL https://openaccess.thecvf.com/content/CVPR2026W/EDGE/papers/Ding_Diagonal-Tiled_Mixed-Precision_Attention_for_Efficient_Low-Bit_MXFP_Inference_CVPRW_2026_paper.pdf.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, Kathrina Wu, Qin Lin, Junkun Yuan, Yanxin Long, Aladdin Wang, Andong Wang, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Hongmei Wang, Jacob Song, Jiawang Bai, Jianbing Wu, Jinbao Xue, Joey Wang, Kai Wang, Mengyang Liu, Pengyu Li, Shuai Li, Weiyan Wang, Wenqing Yu, Xinchi Deng, Yang Li, Yi Chen, Yutao Cui, Yuanbo Peng, Zhentao Yu, Zhiyu He, Zhiyong Xu, Zixiang Zhou, Zunnan Xu, Yangyu Tao, Qinglin Lu, Songtao Liu, Dax Zhou, Hongfa Wang, Yong Yang, Di Wang, Yuhong Liu, Jie Jiang, and Caesar Zhong. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024. URL https://arxiv.org/abs/2412.03603.

Haopeng Li, Yitong Li, Junsong Chen, Tian Ye, Haozhe Liu, Jincheng Yu, Duomin Wang, Ruihua Zhang, Zeke Xie, Enze Xie, and Song Han. Sol-Attn: Accelerating video generation inference via on-the-fly attention sparsification. *arXiv preprint arXiv:2607.24027*, 2026. URL https://arxiv.org/abs/2607.24027.

Jiayi Luo, Jiayu Chen, Jiankun Wang, Cong Wang, Hanxin Zhu, Qingyun Sun, Chen Gao, Zhibo Chen, and Jianxin Li. Attention sparsity is input-stable: Training-free sparse attention for video generation via offline sparsity profiling and online QK co-clustering. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2603.18636.

MiniMaxAI. MiniMax-H3. https://huggingface.co/MiniMaxAI/MiniMax-H3, 2026. Hugging Face model repository.

ModelTC. MiniMax-H3-Turbo: Few-step turbo LoRAs for MiniMax-H3, 2026. URL https://huggingface.co/lightx2v/Minimax-h3-Turbo.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4195–4205, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Peebles_Scalable_Diffusion_Models_with_Transformers_ICCV_2023_paper.html.

Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. FlashAttention-3: Fast and accurate attention with asynchrony and low-precision. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/7ede97c3e082c6df10a8d6103a2eebd2-Abstract-Conference.html.

Joe Sharratt. ThriftAttention: Selective mixed precision for long-context FP4 attention. *arXiv preprint arXiv:2605.23081*, 2026. URL https://arxiv.org/abs/2605.23081.

Xuan Shen, Chenxia Han, Yufa Zhou, Yanyue Xie, Yifan Gong, Quanyi Wang, Yiwei Wang, Yanzhi Wang, Pu Zhao, and Jiuxiang Gu. DraftAttention: Fast video diffusion via low-resolution attention guidance. *arXiv preprint arXiv:2505.14708*, 2025. URL https://arxiv.org/abs/2505.14708.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Zining Wang, Haojie Duanmu, Fanliu Kong, Zhihang Yuan, Jinyang Guo, Size Zheng, Yang Zhang, Shouda Liu, and Xianglong Liu. BMAttn: Block-aligned mixed-precision attention quantization for LLM inference. ICLR 2026 submission, 2026. URL https://openreview.net/pdf?id=isBH8kP5AX.

Haocheng Xi, Shuo Yang, Yilong Zhao, Chenfeng Xu, Muyang Li, Xiuyu Li, Yujun Lin, Han Cai, Jintao Zhang, Dacheng Li, Jianfei Chen, Ion Stoica, Kurt Keutzer, and Song Han. Sparse video-gen: Accelerating video diffusion transformers with spatial-temporal sparsity. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 68208–68224, 2025. URL https://proceedings.mlr.press/v267/xi25c.html.

Yifei Xia, Suhan Ling, Fangcheng Fu, Yujie Wang, Huixia Li, Xuefeng Xiao, and Bin Cui. Training-free and adaptive sparse attention for efficient long video generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15982–15993, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Xia_Training-free_and_Adaptive_Sparse_Attention_for_Efficient_Long_Video_Generation_ICCV_2025_paper.html.

Ruyi Xu, Guangxuan Xiao, Haofeng Huang, Junxian Guo, and Song Han. Xattention: Block sparse attention with antidiagonal scoring. *arXiv preprint arXiv:2503.16428*, 2025.

Shuo Yang, Haocheng Xi, Yilong Zhao, Muyang Li, Jintao Zhang, Han Cai, Yujun Lin, Xiuyu Li, Chenfeng Xu, Kelly Peng, Jianfei Chen, Song Han, Kurt Keutzer, and Ion Stoica. Sparse videogen2: Accelerate video generation with sparse attention via semantic-aware permutation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 96965–96991, 2025. doi: 10.52202/085713-3243. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/8bd5148caced2d73cea7b6961a874a49-Abstract-Conference.html.

Jintao Zhang, Haofeng Huang, Pengle Zhang, Jia Wei, Jun Zhu, and Jianfei Chen. SageAttention2: Efficient attention with thorough outlier smoothing and per-thread INT4 quantization. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 75097–75119, 2025a. URL https://proceedings.mlr.press/v267/zhang25ae.html.

Jintao Zhang, Jia Wei, Haoxu Wang, Pengle Zhang, Xiaoming Xu, Haofeng Huang, Kai Jiang, Jianfei Chen, and Jun Zhu. SageAttention3: Microscaling FP4 attention for

inference and an exploration of 8-bit training. In *Advances in Neural Information Processing Systems*, volume 38, pp. 53901–53931, 2025b. doi: 10.52202/085713-1799. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/4db397e0f760cc573c681e81a01a3dba-Abstract-Conference.html.

Jintao Zhang, Jia Wei, Pengle Zhang, Jun Zhu, and Jianfei Chen. SageAttention: Accurate 8-bit attention for plug-and-play inference acceleration. In *The Thirteenth International Conference on Learning Representations*, 2025c. URL https://openreview.net/forum?id=OL44KtasKc.

Jintao Zhang, Chendong Xiang, Haofeng Huang, Jia Wei, Haocheng Xi, Jun Zhu, and Jianfei Chen. SpargeAttention: Accurate and training-free sparse attention accelerating any model inference. In *International Conference on Machine Learning (ICML)*, 2025d. URL https://proceedings.mlr.press/v267/zhang25ch.html.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

Xuanyi Zhou, Qiuyang Mang, Shuo Yang, Haocheng Xi, Jintao Zhang, Huanzhi Mao, Joseph E. Gonzalez, Kurt Keutzer, Ion Stoica, and Alvin Cheung. SVG-EAR: Parameter-free linear compensation for sparse video generation via error-aware routing. *arXiv preprint arXiv:2603.08982*, 2026. URL https://arxiv.org/abs/2603.08982.

APPENDIX

A ADDITIONAL EXPERIMENTAL DETAILS

Generation and attention settings. We use the released HunyuanVideo-13B and MiniMax-H3 checkpoints. For HunyuanVideo, we use 50 denoising steps, a guidance scale of 6, and a timestep shift of 7. MiniMax-H3 follows its released sampling schedule with a video timestep shift of 12. The few-step models apply FL2VA Turbo 4-step and 8-step v1.0 768p LoRAs to MiniMax-H3, with a video timestep shift of 6 for both variants. The standard models retain dense attention for the first 10 denoising steps. For the few-step models, we retain dense attention for the first step in the 4-step variant and the first two steps in the 8-step variant, as well as the first two layers of every step. We measure latency over the complete DiT denoising trajectory, including attention preparation and retained dense attention, and report speedup relative to the corresponding dense baseline.

Quality evaluation. We evaluate generation quality using the VBench implementation and the prompt sets associated with the reported dimensions. The eight dimensions are subject consistency (SC), background consistency (BC), temporal flickering (TF), motion smoothness (MS), aesthetic quality (AQ), imaging quality (IQ), dynamic degree (DD), and overall consistency (OC). We report each score as a percentage and use their arithmetic mean as Avg. To measure fidelity to dense generation, each output is paired with its dense-attention reference using the same prompt and random seed. PSNR and SSIM are computed on RGB frames normalized to $[0, 1]$, and LPIPS uses the AlexNet backbone. We evaluate similarity over all frames, average the frame scores within each video, and then average across videos.

Attention error analysis. For attention error analysis, we use a dense-trajectory capture from the 4-step model, with 109,376 tokens, 56 heads, and head dimension 128. Relative L2 error is $\|\widehat{O} - O^{\text{dense}}\|_F / \|O^{\text{dense}}\|_F$, reported as a percentage. Latency is the median of three measurements of the complete attention operator. Each curve connects non-dominated measurements at a fixed precision ratio as sparsity varies. For comparisons under a latency budget, we select the lowest measured error among configurations whose measured latency does not exceed that budget; the single-precision reference includes the 4-, 8-, and 16-bit paths.

Table 4: Detailed VBench scores (%) for HunyuanVideo and MiniMax-H3. Configurations correspond to Table 1.

Method	Bit (%) 0 / 4 / 8 / 16	VBench↑								
		SC	BC	TF	MS	AQ	IQ	DD	OC	AVG
HunyuanVideo										
Dense	0/0/0/100	93.87	96.60	98.97	99.11	61.54	67.87	72.22	26.28	77.06
XAttn	85.6/0/0/14.4	93.76	95.57	99.05	98.76	60.10	66.38	51.39	26.16	73.90
SVG2	83.6/0/0/16.4	93.38	96.88	99.57	99.37	50.30	51.24	9.72	25.15	65.70
Sol-Attn	85.8/0/0/14.2	93.62	96.49	98.95	99.05	61.52	67.55	70.83	26.50	76.81
SVG-EAR	85.2/0/0/14.8	93.83	96.99	99.65	99.43	51.41	52.91	12.50	25.20	66.49
SVOO	85.3/0/0/14.7	94.30	96.68	99.61	99.38	58.82	63.27	33.33	25.58	71.37
Ours	85/0/0/15	94.76	97.23	99.69	99.48	60.96	68.10	50.00	26.16	74.55
Ours	85/0/7.5/7.5	94.83	97.33	99.70	99.49	60.96	67.81	51.39	26.10	74.70
SpargeAttn	85/0/15/0	94.86	97.33	99.71	99.50	60.93	67.89	48.61	26.05	74.36
Ours	85/0/15/0	94.93	97.15	99.70	99.50	60.79	66.54	54.17	25.86	74.83
Ours	80.7/14.4/4.8/0	93.53	96.31	99.21	99.30	60.83	65.90	69.44	23.14	75.96
SageAttn3	0/100/0/0	93.72	96.63	99.24	99.29	60.76	65.89	65.28	22.53	75.42
Ours	13.4/64.9/21.6/0	93.71	96.55	99.21	99.28	61.14	66.73	66.67	22.70	75.75
MiniMax-H3										
Dense	0/0/0/100	90.38	88.41	97.92	98.95	62.95	70.67	91.67	26.01	78.37
XAttn	79.9/0/0/20.1	89.36	89.23	98.19	98.66	62.40	68.47	91.67	26.30	78.03
SVG2	79.9/0/0/20.1	89.22	89.24	98.11	98.64	60.48	65.31	88.89	26.29	77.02
Sol-Attn	80/0/0/20	90.33	88.50	97.89	98.90	63.08	70.95	90.28	26.07	78.25
SVG-EAR	79.9/0/0/20.1	89.45	89.16	98.19	98.78	60.85	65.52	87.50	26.29	76.97

Table 4 (continued)

Method	Bit (%) 0 / 4 / 8 / 16	VBench↑								
		SC	BC	TF	MS	AQ	IQ	DD	OC	AVG
SVOO	79.9/0/0/20.1	89.86	89.63	98.06	98.78	61.98	68.30	91.67	26.25	78.07
Ours	80/0/0/20	90.45	88.84	97.94	98.95	62.80	70.53	87.50	26.15	77.90
Ours	80/0/10/10	90.39	88.46	97.94	98.94	62.88	70.47	87.50	26.13	77.84
SpargeAttn	80/0/20/0	90.36	89.50	98.00	98.85	62.64	69.55	84.72	26.15	77.47
Ours	80/0/20/0	90.23	88.88	97.82	98.82	62.86	70.16	90.28	26.13	78.15
Ours	74.3/19.3/6.4/0	90.35	88.19	98.01	98.91	62.69	70.10	87.50	26.15	77.74
SageAttn3	0/100/0/0	90.13	88.09	98.03	98.93	62.43	69.46	86.11	25.99	77.40
Ours	13.6/64.8/21.6/0	90.29	87.42	98.04	98.94	62.71	69.86	86.11	26.04	77.43

Table 5: Detailed VBench scores (%) for few-step generation. Configurations correspond to Table 2.

Method	Bit (%) 0 / 4 / 8 / 16	VBench↑								
		SC	BC	TF	MS	AQ	IQ	DD	OC	AVG
4-Step Diffusion										
Dense	0/0/0/100	91.65	93.35	98.77	99.06	62.56	73.80	50.00	23.92	74.14
Sol-Attn	65/0/0/35	91.68	93.35	98.75	99.03	62.76	73.97	50.00	23.99	74.19
SVG2	64/0/0/36	91.48	93.37	98.71	98.89	62.50	72.95	48.61	23.86	73.80
SVG2	85.8/0/0/14.2	91.33	93.16	98.75	98.90	62.29	72.42	44.44	23.75	73.13
SVG-EAR	66.4/0/0/33.6	91.28	93.56	98.76	98.97	62.65	72.83	48.61	23.84	73.81
SVG-EAR	86.1/0/0/13.9	91.50	93.43	98.80	98.94	62.47	72.24	48.61	23.81	73.73
SVOO	65/0/0/35	91.54	93.62	98.73	98.95	62.33	72.75	48.61	23.84	73.80
SVOO	76.3/0/0/23.7	91.52	93.43	98.77	98.96	62.61	72.10	48.61	23.93	73.74
Ours	65/0/0/35	91.63	93.16	98.75	99.03	62.60	73.78	50.00	23.90	74.11
Ours	63.4/0/18.3/18.3	91.54	93.09	98.72	98.96	62.51	73.79	48.61	23.89	73.89
Ours	53.2/0/37.5/9.4	91.47	93.39	98.71	98.81	62.38	73.49	48.61	23.93	73.85
Ours	29.6/35.2/35.2/0	91.62	93.21	98.73	98.96	62.25	73.68	50.00	23.97	74.05
SpargeAttn	65/0/35/0	91.62	93.18	98.75	99.01	62.53	73.78	47.22	23.81	73.74
Ours	65/0/35/0	91.43	93.45	98.65	98.45	61.77	73.07	48.61	23.77	73.65
Ours	54.5/34.1/11.4/0	91.61	93.16	98.71	98.93	62.21	73.57	50.00	23.82	74.00
SageAttn3	0/100/0/0	91.47	93.51	98.71	98.89	62.16	73.41	50.00	23.84	74.00
Ours	13.1/65.2/21.7/0	91.66	93.14	98.72	98.93	62.19	73.51	48.61	23.92	73.83
8-Step Diffusion										
Dense	0/0/0/100	91.94	89.76	96.72	98.64	64.28	72.92	98.61	26.06	79.87
Sol-Attn	79.9/0/0/20.1	91.87	90.07	96.65	98.57	64.40	73.01	98.61	26.11	79.91
SVG2	79.6/0/0/20.4	91.61	90.02	97.22	98.67	63.80	69.99	97.22	26.21	79.34
SVG2	86.7/0/0/13.3	91.46	89.77	97.31	98.73	62.91	68.80	97.22	26.15	79.04
SVG-EAR	79.6/0/0/20.4	91.80	90.66	97.25	98.74	63.95	70.21	97.22	26.24	79.51
SVG-EAR	88.4/0/0/11.6	91.57	90.04	97.35	98.76	63.16	69.02	94.44	26.17	78.81
SVOO	80.1/0/0/19.9	91.67	90.39	97.33	98.68	63.26	69.27	97.22	26.11	79.24
Ours	80/0/0/20	91.93	89.79	96.74	98.64	64.49	72.71	98.61	26.11	79.88
Ours	74.3/0/12.8/12.8	91.92	89.64	96.83	98.60	64.42	72.73	98.61	26.08	79.85
Ours	57.9/0/42.1/0	91.71	90.52	97.08	98.28	63.66	71.63	98.61	26.06	79.69
Ours	40.5/59.5/0/0	91.86	90.36	97.11	98.66	64.12	71.07	98.61	26.03	79.73
Ours	50.8/24.6/24.6/0	91.91	89.94	96.88	98.59	64.31	72.34	98.61	25.95	79.82
SpargeAttn	80/0/20/0	92.00	90.31	96.90	98.65	64.53	71.99	97.22	25.96	79.70
Ours	80/0/20/0	91.78	90.16	96.88	98.46	64.28	72.11	98.61	26.17	79.81
Ours	74/13/13/0	91.99	90.03	96.85	98.60	64.52	72.45	98.61	26.08	79.89
SageAttn3	0/100/0/0	91.81	90.08	97.02	98.59	64.08	71.61	98.61	25.98	79.72
Ours	13/65.3/21.8/0	91.84	90.21	97.00	98.58	64.12	72.03	98.61	26.00	79.80

B Quantization Analysis and Proofs

We analyze one attention head and one video query at a time. All sums exclude padding and include the retained prefix interactions. The support is fixed by the router before the mixed-precision computation. As in DraftAttention (Shen et al., 2025), we separate the effect of removing interactions from the approximation on the retained support. Tracking both the output numerator and the softmax denominator also underlies the approximation analysis in Sol-Attn (Li et al., 2026). Here the retained interactions are computed with quantized operands, and the analysis explicitly accounts for the different weights used by the denominator and the probability–value multiplication.

B.1 Reconstruction-Error Model

We take the stored BF16 tensors Q, K, V as reference inputs. The reconstruction model isolates additional low-bit quantization: on the 16-bit path, $\widehat{s}_{uv} = s_{uv}$, $\widehat{V}_v = V_v$, and $\widehat{w}_{uv} = w_{uv}$. Thus $\alpha_{ub} = \nu_{ub} = \omega_{ub} = 0$ on this path. Ordinary floating-point effects, including any reference-path format conversion, are accounted for separately in $\varepsilon_u^{\mathrm{arith}}$ below. For completeness, the exact dense operator on these reference inputs is

$$P = \mathrm{softmax}_{\mathrm{row}}(QK^\top/\sqrt{d}), \qquad O^{\mathrm{dense}} = PV. \tag{9}$$

For a query u, write $s_{uv} = Q_u^\top K_v/\sqrt{d}$ and $P_{uv}^\Omega = e^{s_{uv}}/\sum_{r\in\Omega_u} e^{s_{ur}}$ on its retained support. For a retained key block $\mathcal{B}_b$, write $m_{ub} = \sum_{v\in\mathcal{B}_b} P_{uv}^\Omega$. The precision-level mass in Equation (3) is $m_u^{(p)} = \sum_{b:p_{ub}=p} m_{ub}$. Let $\widehat{s}_{uv}$ and $\widehat{V}_v$ be the reconstructed logit and value. At a common row reference h_u, set $w_{uv} = e^{\widehat{s}_{uv}-h_u}$. The denominator sums w_{uv} before weight quantization; the numerator uses reconstructed weights $\widehat{w}_{uv}$, including subsequent online rescalings. For each retained block, assume

$$\begin{aligned} |\widehat{s}_{uv} - s_{uv}| \le \alpha_{ub}, \qquad & \|\widehat{V}_v - V_v\|_2 \le \nu_{ub}, \\ \sum_{v\in\mathcal{B}_b} |\widehat{w}_{uv} - w_{uv}| \le \omega_{ub} \sum_{v\in\mathcal{B}_b} w_{uv}. & \end{aligned} \tag{10}$$

These are bounds on reconstructed operands in their original units. Define $\alpha_u = \max_b \alpha_{ub}$ and the block error cost

$$c_{ub} = 2V_{\max}(e^{\alpha_{ub}} - 1) + e^{\alpha_{ub}}\big[\nu_{ub} + (V_{\max} + \nu_{ub})\omega_{ub}\big]. \tag{11}$$

For the format-level statement in the main text, assume common bounds $\alpha_{ub} \le \alpha_p$, $\nu_{ub} \le \nu_p$, and $\omega_{ub} \le \omega_p$ whenever block b uses low-bit precision $p \in \{4, 8\}$. These bounds cover all admissible assignments under consideration. With $\bar{\alpha} = \max(\alpha_4, \alpha_8)$, define

$$\varepsilon_p = e^{\bar{\alpha}}\left\{2V_{\max}(e^{\alpha_p} - 1) + e^{\alpha_p}\big[\nu_p + (V_{\max} + \nu_p)\omega_p\big]\right\}. \tag{12}$$

For notation in the proofs, set $\varepsilon_{16} = 0$: reference-precision blocks have no additional low-bit quantization cost. For small errors, the leading terms are $2V_{\max}\alpha_p + \nu_p + V_{\max}\omega_p$. The ordering of these envelopes is an assumption of Theorem 3.6, not a consequence of bit width alone.

B.2 Reconstructed Operands and Logit Error

For a fixed query u, the retained key supports $\{\mathcal{B}_b\}$ are disjoint and partition Ω_u. All quantities below are expressed in the original numerical units after restoring their quantization scales. For a block assigned precision p, let

$$\widehat{Q}_u^{(p)} = Q_u + \Delta Q_u^{(p)}, \qquad \widehat{K}_v^{(p)} = K_v + \Delta K_v^{(p)}, \qquad \widehat{V}_v = V_v + \Delta V_v.$$

The reconstructed logit is $\widehat{s}_{uv} = (\widehat{Q}_u^{(p)})^\top \widehat{K}_v^{(p)}/\sqrt{d}$. Expanding the inner product gives

$$|\widehat{s}_{uv} - s_{uv}| \le \frac{\|Q_u\|_2\|\Delta K_v^{(p)}\|_2 + \|K_v\|_2\|\Delta Q_u^{(p)}\|_2 + \|\Delta Q_u^{(p)}\|_2\|\Delta K_v^{(p)}\|_2}{\sqrt{d}}. \tag{13}$$

Taking the maximum over $v \in \mathcal{B}_b$ provides a valid α_{ub} in Equation (10). The value error is bounded by $\nu_{ub} = \max_{v\in\mathcal{B}_b} \|\Delta V_v\|_2$. For the low-bit paths, these residuals include errors from operand conversion and the represented quantization scales.

For a uniform nearest-integer quantizer with recovery step Δ and no clipping, each reconstructed scalar has error at most $\Delta/2$. A d-dimensional vector with a common step therefore has error at most $\sqrt{d}\,\Delta/2$. This observation applies to the INT8 operand path under its range assumptions. Floating-point formats such as E2M1 and E4M3 have nonuniform spacing; for these paths, Equation (13) uses the actual reconstruction residuals. The theoretical costs need not be computed by the router.

B.3 Weight Quantization under Shared Normalization

Choose a common final row reference h_u, for example the maximum reconstructed logit on Ω_u, and write

$$w_{uv} = e^{\widehat{s}_{uv} - h_u}, \qquad Z_u^{\mathrm{q}} = \sum_{v \in \Omega_u} w_{uv}.$$

The denominator uses these weights before their conversion to the operand format of the probability–value multiplication. Let $\widehat{w}_{uv}$ denote the reconstructed weight used in that multiplication, expressed at the same final reference. Then, in exact arithmetic on these reconstructed operands,

$$\widehat{O}_u = \frac{\sum_{v \in \Omega_u} \widehat{w}_{uv} \widehat{V}_v}{Z_u^{\mathrm{q}}}. \tag{14}$$

In general, $\sum_v \widehat{w}_{uv}$ differs from Z_u^{q}. Equation (14) retains this distinction.

To relate this representation to online execution, suppose block b is processed at running row maximum $h_u^{(b)}$. Its unquantized weights are $e^{\widehat{s}_{uv} - h_u^{(b)}}$. Every later increase of the row maximum rescales the previous numerator and denominator by the same factor. The product of these factors is $e^{h_u^{(b)} - h_u}$. Thus the block's final denominator contribution is $\sum_{v \in \mathcal{B}_b} w_{uv}$, and its reconstructed numerator weights are the locally quantized weights multiplied by $e^{h_u^{(b)} - h_u}$. This defines $\widehat{w}_{uv}$ for the actual traversal order. In particular, the block-relative error

$$\frac{\sum_{v \in \mathcal{B}_b} |\widehat{w}_{uv} - w_{uv}|}{\sum_{v \in \mathcal{B}_b} w_{uv}}$$

is unchanged by these common rescalings.

The phase-boundary transformations in Equation (8) are changes of numerical units. In exact arithmetic, the entry and exit scalings are inverse transformations, and the normalized output is preserved after restoring the value scale. Consequently, all precision phases contribute to the single numerator and denominator in Equation (14). Rounding in exponentiation, scale conversion, matrix accumulation, and final output conversion is accounted for separately below.

B.4 Proof of the Attention-Weighted Error Bound

Proof of Lemma 3.3. Fix a query u with nonempty retained support and define its exact retained-support output by

$$O_u^{\Omega} = \sum_{v \in \Omega_u} P_{uv}^{\Omega} V_v.$$

We first bound the support-removal error. Let $D_u = \sum_{v \notin \Omega_u} P_{uv} V_v$. Since the retained dense probability mass is $1 - \delta_u$,

$$O_u^{\mathrm{dense}} = (1 - \delta_u) O_u^{\Omega} + D_u.$$

The value bound implies $\|D_u\|_2 \le \delta_u V_{\max}$ and $\|O_u^{\Omega}\|_2 \le V_{\max}$, hence

$$\|O_u^{\mathrm{dense}} - O_u^{\Omega}\|_2 \le 2 V_{\max} \delta_u. \tag{15}$$

Next, let $z_{uv} = e^{s_{uv} - h_u}$ and $Z_u = \sum_{v \in \Omega_u} z_{uv}$. Thus $P_{uv}^{\Omega} = z_{uv}/Z_u$ and $\sum_{v \in \Omega_u} z_{uv}(V_v - O_u^{\Omega}) = 0$. Subtracting O_u^{Ω} from Equation (14) yields the exact identity

$$\begin{aligned} Z_u^{\mathrm{q}}(\widehat{O}_u - O_u^{\Omega}) = &\sum_{v \in \Omega_u} (w_{uv} - z_{uv})(V_v - O_u^{\Omega}) \\ &+ \sum_{v \in \Omega_u} w_{uv}(\widehat{V}_v - V_v) + \sum_{v \in \Omega_u} (\widehat{w}_{uv} - w_{uv}) \widehat{V}_v. \end{aligned} \tag{16}$$

The three terms respectively account for logit, value, and exponential-weight quantization, including the change in normalization caused by the perturbed logits.

For $v \in \mathcal{B}_b$, the logit bound gives

$$|w_{uv} - z_{uv}| \le z_{uv}(e^{\alpha_{ub}} - 1), \qquad w_{uv} \le e^{\alpha_{ub}} z_{uv}.$$

Also, $\|V_v - O_u^{\Omega}\|_2 \le 2V_{\max}$ and $\|\widehat{V}_v\|_2 \le V_{\max} + \nu_{ub}$. Applying these inequalities and Equation (10) to the contribution of each block in Equation (16) gives

$$Z_u^{\mathrm{q}} \|\widehat{O}_u - O_u^{\Omega}\|_2 \le \sum_b \left[2V_{\max}(e^{\alpha_{ub}} - 1) + e^{\alpha_{ub}}\{\nu_{ub} + (V_{\max} + \nu_{ub})\omega_{ub}\}\right] \sum_{v \in \mathcal{B}_b} z_{uv}.$$

Since $\alpha_u = \max_b \alpha_{ub}$,

$$Z_u^{\mathrm{q}} \ge e^{-\alpha_u} Z_u, \qquad \sum_{v \in \mathcal{B}_b} z_{uv} = Z_u m_{ub}.$$

Dividing by Z_u^{q} establishes

$$\|\widehat{O}_u - O_u^{\Omega}\|_2 \le e^{\alpha_u} \sum_b m_{ub} c_{ub}. \tag{17}$$

Combining Equations (15) and (17) gives the block-dependent bound. Since $e^{\alpha_u} c_{ub} \le \varepsilon_{p_{ub}}$ under the common format bounds, Equation (4) follows. □

Arithmetic roundoff and empty rows. Let O_u^{impl} denote the implemented output. If $\|O_u^{\mathrm{impl}} - \widehat{O}_u\|_2 \le \varepsilon_u^{\mathrm{arith}}$, then

$$\|O_u^{\mathrm{dense}} - O_u^{\mathrm{impl}}\|_2 \le 2V_{\max}\delta_u + e^{\alpha_u} \sum_b m_{ub} c_{ub} + \varepsilon_u^{\mathrm{arith}}.$$

This last term includes ordinary floating-point rounding and any reference-path format conversion, including effects on subsequent shared normalization. Omitting a 16-bit quantization term therefore does not assume exact hardware arithmetic. If a query has no valid retained key, the implementation returns zero, and $\|O_u^{\mathrm{dense}} - O_u^{\mathrm{impl}}\|_2 \le V_{\max}$ directly.

B.5 Proof of the v1-to-v2 Error-Bound Comparison

Proof of Proposition 3.4. Fix a query and let Ω_{16} be the support retained by v1 and preserved at 16 bits by v2. Let L_8, L_4 be disjoint subsets of its complement, with $\tau_p = \sum_{v \in L_p} P_{uv}$. The v2 support is $\Omega' = \Omega_{16} \cup L_8 \cup L_4$, with dense attention mass $1 - \delta_{\mathrm{v2}}$. Hence Ω_{16} has mass $1 - \delta_{\mathrm{v2}} - \tau > 0$. The v1 allocation skips mass $\delta_{\mathrm{v1}} = \delta_{\mathrm{v2}} + \tau$ and has no additional low-bit quantization cost, so Lemma 3.3 gives $B_{\mathrm{v1}} = 2V_{\max}(\delta_{\mathrm{v2}} + \tau)$. On Ω', the exact renormalized masses are

$$m_u^{(16)} = \frac{1 - \delta_{\mathrm{v2}} - \tau}{1 - \delta_{\mathrm{v2}}}, \qquad m_u^{(8)} = \frac{\tau_8}{1 - \delta_{\mathrm{v2}}}, \qquad m_u^{(4)} = \frac{\tau_4}{1 - \delta_{\mathrm{v2}}}.$$

Applying the lemma to v2 gives B_{v2} in Equation (5). Subtracting the two bounds yields

$$B_{\mathrm{v2}} - B_{\mathrm{v1}} = -2V_{\max}\tau + \frac{\tau_8\varepsilon_8 + \tau_4\varepsilon_4}{1 - \delta_{\mathrm{v2}}}. \tag{18}$$

Since $1 - \delta_{\mathrm{v2}} > 0$, Equation (6) is precisely the condition for this difference to be negative. □

The comparison concerns output-error bounds for reconstructed operators with matched 16-bit supports. It isolates v2's low-bit recovery from changes in draft scoring or support selection. If ρ_{v1} and ρ_{v2} denote the skipped fractions of all video-region pairs, and ρ_4, ρ_8 the fractions assigned to the low-bit paths by v2, this construction gives $\rho_{\mathrm{v1}} = \rho_{\mathrm{v2}} + \rho_4 + \rho_8$. The quantities $\delta_{\mathrm{v2}}, \tau_8, \tau_4$ are dense attention masses, not block fractions; equal sparsity ratios alone do not establish the comparison. Arithmetic roundoff can be added separately as in Appendix B.4; measured latency and relative output error are evaluated by the operator experiments.

B.6 Proof of Importance-Guided Precision Allocation

Fix the retained video-region pairs and the precision of all always-retained interactions. For the allocation objective, set $P^{\Omega}_{uv} = 0$ on rows with empty retained support; their zero outputs contribute a fixed support error. For a retained region pair (a, b), its contribution to query-averaged attention mass is

$$W_{ab} = \frac{1}{N_{\text{vid}}} \sum_{u \in \mathcal{R}_a} \sum_{v \in \mathcal{R}_b} P^{\Omega}_{uv}, \qquad N_{\text{vid}} = \sum_{a} |\mathcal{R}_a|.$$

The inner sum includes all physical blocks inherited from region b. With the support fixed, these weights are independent of precision. Averaging Lemma 3.3 over video queries gives the assignment-dependent objective

$$\sum_{(a,b) \text{ retained}} W_{ab} \varepsilon_{p_{ab}}. \tag{19}$$

The support error and always-retained interactions with fixed precision contribute constants.

Proof of Theorem 3.6. Consider two retained pairs with $W_1 \geq W_2$ and two format costs $\varepsilon_{\text{low}} \leq \varepsilon_{\text{high}}$. The excess cost of assigning the larger error cost to the larger weight is

$$(W_1 \varepsilon_{\text{high}} + W_2 \varepsilon_{\text{low}}) - (W_1 \varepsilon_{\text{low}} + W_2 \varepsilon_{\text{high}}) = (W_1 - W_2)(\varepsilon_{\text{high}} - \varepsilon_{\text{low}}) \geq 0.$$

Exchanging each inverted assignment preserves the quotas and cannot increase the objective. Repeated exchanges yield the stated ordering. □

The theorem concerns the error bound at fixed support and quotas. The implemented router approximates its mass ordering with A_{ab}; it does not evaluate W_{ab} or the error envelopes online. The pooling mixture controls this proxy and does not enter the quantization-error decomposition.

C Implementation Details

Block-wise precision allocation requires multiple operand formats while retaining a common normalization across all selected blocks. We fuse operand preparation to reduce data movement and organize computation into precision-specific phases that share one online softmax state.

C.1 Routing and Precision Formats

For each spatial region $\mathcal{R}_a$, valid-token counts $n_a = |\mathcal{R}_a|$ exclude padding. The descriptors in Equation (2) are $\bar{Q}_a = n_a^{-1} \sum_{u \in \mathcal{R}_a} Q_u$ and $Q^{\max}_{a,c} = \max_{u \in \mathcal{R}_a} Q_{u,c}$, with keys defined analogously. Both branches normalize over key regions. The router ranks all R^2 video-region pairs within each head, resolving ties by region-pair index. Precision fractions determine integer quotas T_p satisfying $\sum_p T_p = T$. Each region pair's precision is assigned to all of its physical attention blocks, so the number of retained blocks and their precisions can vary across query regions. The labels 4, 8, and 16 denote the NVFP4, INT8, and 16-bit paths. The INT8 path uses INT8 for $QK^\top$ and FP8 for the probability–value multiplication. Prefix interactions lie outside the video-region budget: video queries retain all valid prefix keys, and prefix queries use non-sparse attention. Rows with no valid retained key return zero.

C.2 Fused Quantization and Reordering

Section 3.3 describes the fused preparation shown in Figure 3. Here we give the spatial packing and operand-layout details.

We partition each $H_v \times W_v$ frame into R_f connected regions and pack each region into a contiguous block of capacity C, where

$$R_f = \left\lceil \frac{H_v W_v}{C} \right\rceil, \qquad 0 \leq R_f C - H_v W_v < C. \tag{20}$$

This ragged 2D layout uses the minimum number of blocks without requiring a fixed patch shape to divide the frame dimensions. To preserve spatial locality, we prioritize the smallest total perimeter among candidate partitions. For a rectangular token grid,

$$\sum_{a=0}^{R_f-1} |\partial \mathcal{R}_a| = 2(H_v + W_v) + 2E_{\text{cut}}, \tag{21}$$

where $|\partial \mathcal{R}_a|$ counts unit boundary edges and E_{cut} counts horizontal or vertical neighbor pairs assigned to different regions. Reducing total perimeter therefore preserves more spatial neighbors within the same pooled region. Valid-token counts exclude padding from pooling and attention; the inverse index map restores the output order.

The $QK^\top$ accumulator layout can differ from the operand layout required by the subsequent probability–value multiplication (Shah et al., 2024). We align these layouts so that $\widetilde{P}$ can feed the subsequent matrix multiplication through thread-local conversion and packing. For an intra-block permutation matrix Π, the matching column and row permutations preserve the product:

$$(\widetilde{P}\Pi)(\Pi^\top V) = \widetilde{P}V. \tag{22}$$

We fuse the required intra-block K/V permutations with quantization, and incorporate V transposition into operand packing and loading.

C.3 Quantization Formats and Scales

Table 6 distinguishes matrix operand formats from accumulation formats. The row maxima, exponential sums, and output state shared across blocks and phases are stored in FP32. For the INT8 and FP16 paths, each physical key block's FP16 probability–value accumulator is converted to FP32 before being added to the persistent output state.

Table 6: Operand and accumulator formats in the mixed-precision kernel.

Path	Q, K	$QK^\top$ accumulator	$\widetilde{P}, V$	$\widetilde{P}V$ accumulator
NVFP4	E2M1	FP32	E2M1	FP32
INT8	INT8	INT32	E4M3	FP16
FP16	FP16	FP32	FP16	FP16

NVFP4 operands. We use E2M1 values with E4M3 local scales for groups of 16 elements along each matrix multiplication's reduction dimension. For Q and K, groups run along feature channels; for V, they run along key tokens within each output channel. Given a positive per-tensor scale g_X for $X \in \{Q, K, V\}$ and a quantization group G, the local scale and reconstructed values are

$$\sigma_{X,G} = \text{cast}_{\text{E4M3}}\left(\frac{\max_{x\in G}|x|}{6g_X}\right), \qquad \widehat{x} = g_X \sigma_{X,G}\, \text{cast}_{\text{E2M1}}\left(\frac{x}{g_X\sigma_{X,G}}\right). \tag{23}$$

A group whose encoded scale is zero is encoded as zero. The matrix instructions apply local scales, and $g_Q g_K/\sqrt{d}$ restores the scale of the attention logits.

The exponential weights $\widetilde{P}$ are quantized online. For a query u and a group G of 16 key positions, we use

$$\begin{aligned} \sigma_{P,u,G} &= \text{cast}_{\text{E4M3}}\left(448 \max_{v\in G} \widetilde{P}_{uv}\right), \\ \widehat{\widetilde{P}}_{uv} &= \frac{\sigma_{P,u,G}}{2688}\, \text{cast}_{\text{E2M1}}\left(\frac{2688\widetilde{P}_{uv}}{\sigma_{P,u,G}}\right). \end{aligned} \tag{24}$$

Here $2688 = 6 \cdot 448$ combines the maximum finite values of E2M1 and E4M3; zero-scale groups again produce zeros. The denominator is updated with the FP32 weights before this conversion.

INT8 and FP16 operands. The INT8 path quantizes Q and K with one FP32 scale per query block and physical key block, respectively. For either operand block X, the scale is $\delta_X = \max|X|/127 +$

10^{-7} and the encoding rounds X/δ_X to the nearest integer, with ties away from zero. The INT32 dot products are converted to FP32 and rescaled by $\delta_Q \delta_K / \sqrt{d}$. For the probability–value product, V is encoded in E4M3 with the per-channel scale $\delta_c^V = \max_v |V_{v,c}|/2.25$; all-zero channels are encoded as zero. Exponentiation uses base two with an additive offset of 8.807, amplifying the weights by approximately 448 before E4M3 conversion. The probability dequantization scale is $\eta = 0.0022326917$. The denominator accumulates the amplified FP32 weights, while the numerator uses E4M3 operands. The FP16 path uses FP16 Q, K, V and converts exponential weights to FP16 after updating the FP32 denominator.

C.4 Phased Mixed-Precision Execution

The NVFP4, INT8, and FP16 phases extend the same online-softmax state described in Section 3.3. Changes in the running row maximum rescale both the existing numerator and denominator before new block contributions are accumulated. Phase-boundary transformations account separately for the numerical scales of the probability and value operands.

The NVFP4 phase executes first. Its matrix instructions incorporate the local scales of $\widetilde{P}$ and V; multiplying the resulting numerator by $g_V/2688$ restores the common accumulation scale. Its denominator already accumulates unamplified exponential weights. The INT8 phase enters and leaves its probability and value scales using Equation (8); the running row maximum is unchanged by this conversion. The FP16 phase then continues the accumulation in the common scale. Final normalization divides the output numerator by the shared denominator, and the inverse spatial index map restores token order.

The common shared-memory workspace is sized for the largest phase or output-staging requirement. Each phase's operand fragments, quantization scales, and loading temporaries remain local to that phase, allowing their storage to be reused by subsequent phases.

D Additional Visual Comparisons

Figures 6–8 provide additional visual comparisons.

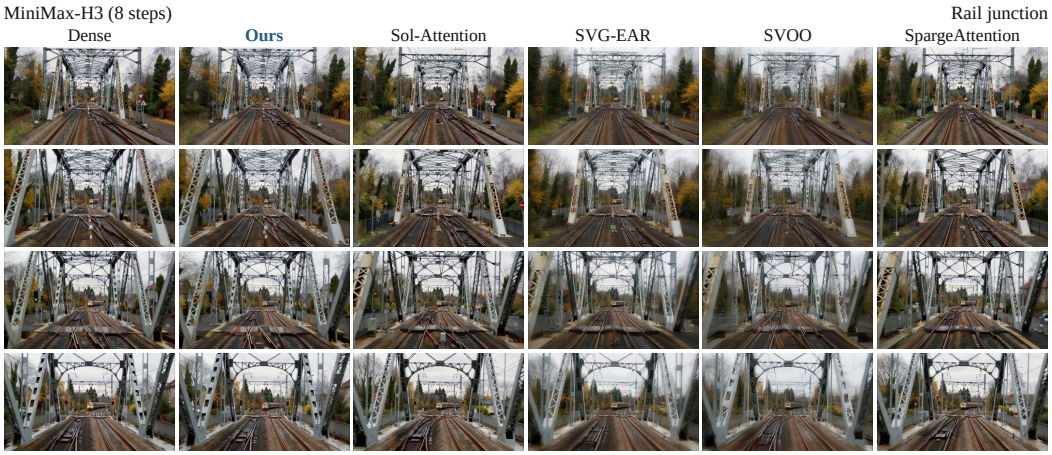


Figure 6: Visual comparison of rail junctions in 8-step MiniMax-H3 generation.



Figure 7: Visual comparison of a running cat in 4-step MiniMax-H3 generation.

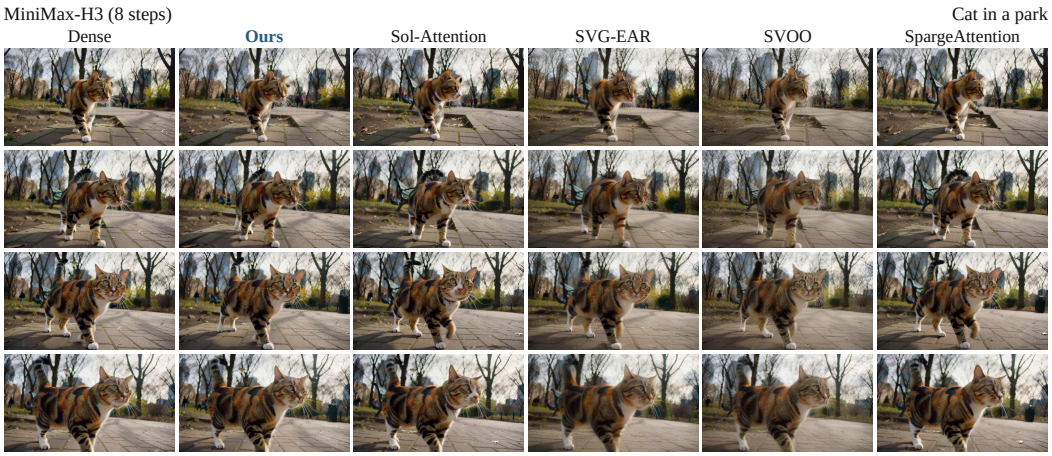


Figure 8: Visual comparison of a cat in a park in 8-step MiniMax-H3 generation.