AB-ATTENTION: ACCURATE BLOCK ATTENTION FOR ACCELERATING VIDEO DIFFUSION TRANSFORMERS

Anonymous authors
Paper under double-blind review

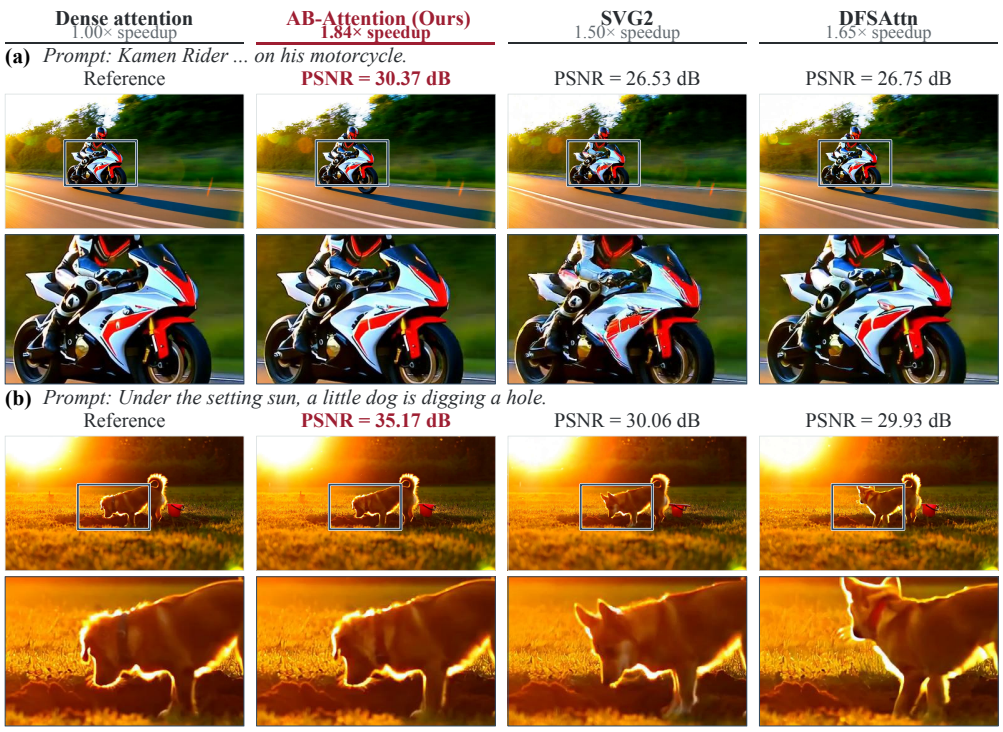


Figure 1: Examples on Wan2.1-T2V-14B, with one frame and an enlarged region per method. AB-Attention is closest to dense attention in full-video PSNR while achieving the highest speedup.

ABSTRACT

Video diffusion transformers generate high-quality videos, but the quadratic cost of 3D full attention dominates their inference. Dynamic block-sparse attention reduces this cost by selecting key/value blocks from the current activations and computing exact attention only on the selected ones. However, its fidelity degrades under aggressive sparsity: blocks are scored from coarse summaries or formed by costly global clustering, pruned blocks vanish from the softmax, and whole tiles are kept or dropped even when only a few of their keys matter. We present Accurate Block Attention (**AB-Attention**), a training-free method that addresses each limitation. Its Intra-Block Clustering Selector (IBCS) estimates block importance faithfully yet cheaply by clustering queries only within each query block, without permuting tokens. Sampling with Inclusion-Probability Weighting (SIP) then samples blocks in proportion to these estimates and reweights them by inverse inclusion probability, so that one pass estimates dense attention rather than truncating it. Finally, its kernel gathers key/value blocks much smaller than a tile into full tensor-core tiles, refining selection at no extra tensor-core cost. On Wan2.1 and HunyuanVideo, AB-Attention skips 89–92% of attention, whereas the baselines compute 1.8–3.6× as much, yet it still improves PSNR by up to 4.4 dB over the strongest baseline and achieves up to 2.23× end-to-end speedup.

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.

1 INTRODUCTION

Video diffusion transformers (Wan et al., 2025; Kong et al., 2024) primarily rely on 3D full attention to capture spatial and temporal dependencies, enabling high-quality generation. However, the computational cost of full attention grows quadratically with the sequence length. As the video resolution and number of frames increase, attention becomes increasingly expensive. Since this computation is repeated across transformer layers and denoising steps, attention has long been a major inference bottleneck (Xi et al., 2025; Zhang et al., 2025c). Reducing the cost of attention while preserving generation quality is therefore essential for generating longer and higher-resolution videos.

Sparse attention reduces this cost by computing only a subset of query–key pairs. Early static methods apply predefined spatial or temporal patterns, such as local 3D windows (Zhang et al., 2025c), energy-decay masks (Li et al., 2025), or per-head spatial and temporal masks (Xi et al., 2025). Their regular layouts map well onto GPU kernels, but attention patterns vary across prompts, heads, and denoising steps, so a fixed mask must retain many unimportant pairs to avoid missing important ones. Dynamic methods instead select key/value blocks from the current activations. Since block-sparse kernels execute attention in fixed-size tiles, they partition queries and keys into tile-aligned blocks, estimate block importance cheaply from mean-pooled tokens (Zhang et al., 2025b) or antidiagonal attention samples (Xu et al., 2025), and compute exact attention only on the most important blocks. SVG2 (Yang et al., 2025) instead forms the blocks by clustering and permuting tokens.

Despite this progress, dynamic methods remain limited by three design choices: **1) Importance estimates are coarse or costly.** Pooling a block into one representative mixes tokens with different attention patterns and can misrank blocks, whereas clustering yields faithful representatives only through sequence-wide k-means and token permutation (Yang et al., 2025). **2) Top-k selection truncates attention.** Discarding a block removes its contribution from both the attention numerator and the softmax normalizer, which better scores cannot recover. Correction-based methods (Liu et al., 2025b; Li et al., 2026; Zhou et al., 2026) approximate the discarded remainder, but add a separate approximation branch to the exact sparse computation. **3) The selection unit is the execution tile.** A tile-sized key block is kept or discarded as a whole even when only a few of its keys matter, and reordering tokens makes blocks more coherent without changing this unit. Finer units need not slow execution, as a kernel can gather them into full tiles, but they multiply the candidates to score.

We therefore propose Accurate Block Attention (**AB-Attention**), a training-free method that directly addresses these limitations. The *Intra-Block Clustering Selector* (IBCS) scores key/value blocks with lightweight query clusters formed only within each query block, avoiding costly global clustering and token permutation. *Sampling with Inclusion-Probability Weighting* (SIP) samples blocks in proportion to these scores and reweights them by inverse inclusion probability. Every block can thus contribute, and a single sparse attention pass yields unbiased estimates of both the attention numerator and the softmax normalizer. AB-Attention further performs *fine-grained selection with tiled execution*: it selects key/value blocks much smaller than an execution tile and gathers them into full tensor-core tiles in its kernel, without reordering any tokens. The sampling plans are reused across a few nearby denoising steps, while attention is recomputed at every step (Xia et al., 2025).

Extensive experiments on representative video diffusion transformers, including Wan2.1 (Wan et al., 2025) and HunyuanVideo (Kong et al., 2024), demonstrate that AB-Attention significantly advances the quality–efficiency frontier of training-free sparse attention. It skips 89–92% of attention computation, whereas the competing baselines compute 1.8–3.6× as much, and delivers 1.84–2.23× end-to-end speedups on a single H20 GPU, while achieving the highest fidelity to dense attention and improving PSNR by 1.6–4.4 dB over the strongest baselines, including SVG2 (Yang et al., 2025) and DFSAttn (Hu et al., 2026). The examples in Figure 1 visually illustrate these fidelity gains.

2 RELATED WORK

Sparse Attention. Sparse attention accelerates video diffusion transformers by restricting query–key interactions to a subset that can be computed efficiently. Static methods exploit video geometry with fixed sparsity patterns: Sliding Tile Attention (Zhang et al., 2025c) uses tiled local 3D windows, Radial Attention (Li et al., 2025) constructs a static multiscale mask from spatiotemporal energy decay, and Sparse VideoGen (Xi et al., 2025) profiles attention heads online to assign each head a predefined spatial or temporal pattern. Dynamic methods instead infer sparsity from the current

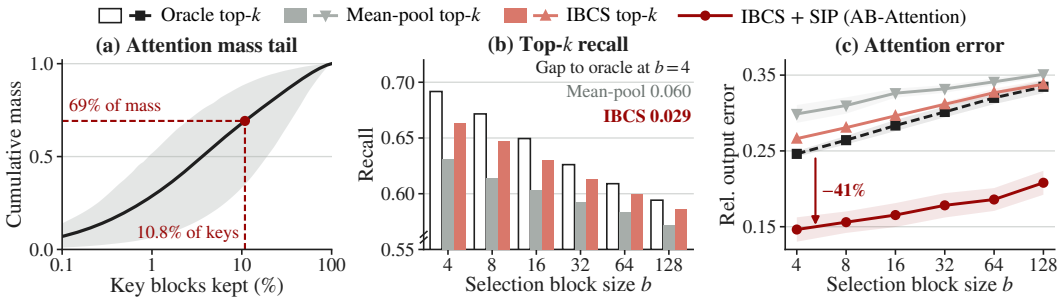


Figure 2: **Attention mass, recall, and error on Wan2.1-T2V-14B** at 89.16% sparsity. (a) Sorted cumulative attention mass of 4-token key blocks; the band spans the 10th–90th percentiles of query blocks. (b) Top-k recall: the fraction of each query block's dense attention mass covered by the selected key blocks. Finer blocks keep more mass, and IBCS halves the gap of mean-pool to the oracle at b=4. (c) Relative ℓ_2 error of the attention output against dense attention. Sampling with SIP on IBCS scores lowers the error 41% below oracle top-k at b=4.

activations. XAttention (Xu et al., 2025) estimates block importance by summing attention scores along strided antidiagonals, and SpargeAttention (Zhang et al., 2025b) combines coarse prediction from mean-compressed self-similar blocks with softmax-aware filtering. To improve block representations, SVG2 (Yang et al., 2025) clusters tokens by semantic similarity with k-means and permutes each cluster into contiguous blocks, and DFSAttn (Hu et al., 2026) combines 3D Hilbert-curve reordering with hierarchical sub-block scoring. SPADE (Liu et al., 2026) adapts block summaries and sparsity policies to each input. AdaSpa (Xia et al., 2025) and RAPID (Lin et al., 2026) amortize sparsity search by reusing and adapting masks across denoising steps. Across these designs, selection granularity, grouping overhead, and hardware-friendly execution remain coupled.

Correction-Based Sparse Attention. Pruning removes the skipped interactions from the attention numerator and the softmax normalizer. Training-free correction methods approximate these missing contributions: Rectified SpaAttn (Liu et al., 2025b) rectifies the amplified weights of retained blocks and recovers those of pruned blocks from pooled attention, PISA (Li et al., 2026) approximates the remaining blocks with a block-wise Taylor expansion, and SVG-EAR (Zhou et al., 2026) compensates pruned blocks with cluster centroids and routes exact computation to the blocks with the largest error. All three compute selected blocks exactly and approximate the rest, whereas AB-Attention estimates both the numerator and the normalizer from one weighted sample of all blocks.

Other Acceleration Methods. Diffusion inference can also be accelerated by reducing denoising steps, reusing intermediate computation, or improving arithmetic and memory efficiency. Fast solvers such as DPM-Solver (Lu et al., 2022) reduce the number of model evaluations, while progressive distillation (Salimans & Ho, 2022) repeatedly trains a student to halve the number of sampling steps. TeaCache (Liu et al., 2025a) reuses cached model outputs across denoising steps while the accumulated difference of timestep-modulated inputs stays below a threshold. FlashAttention (Dao et al., 2022) reduces memory traffic by computing exact attention tile by tile in on-chip memory, and SageAttention (Zhang et al., 2025a) accelerates attention by quantizing queries and keys to INT8. These approaches are complementary to sparse attention and can be combined with it.

3 Preliminaries

Dense attention. Video diffusion transformers apply self-attention to tokens across space and time. Given n tokens, let $Q, K, V \in \mathbb{R}^{n\times d}$ denote the query, key, and value matrices for one attention head. Dense attention computes

$$O = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right)V. \tag{1}$$

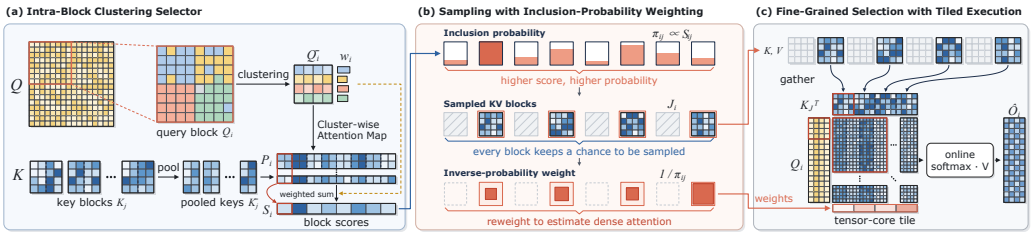


Figure 3: **Overview of AB-Attention.** (a) IBCS clusters each query block Q_i into centroids $\bar{Q}_i$ with weights w_i, and scores every key/value block by the weighted attention of the centroids to its pooled keys. (b) SIP samples each block with probability π_{ij} proportional to its score S_{ij}, so no block is excluded, and weights it by $1/\pi_{ij}$ to estimate full attention. (c) The kernel gathers the sampled blocks into full tensor-core tiles and attends to them with logits shifted by $-\log \pi_{ij}$.

Its $\mathcal{O}(n^2 d)$ cost becomes expensive for high-resolution videos. FlashAttention (Dao et al., 2022) makes exact attention more memory-efficient by reducing data movement and intermediate storage. It still computes all query–key pairs and therefore retains quadratic computation.

Sparse attention. Sparse attention reduces this arithmetic by computing only selected query–key pairs. Conceptually, it adds a mask $\mathcal{M} \in \{-\infty, 0\}^{n \times n}$ to the attention logits:

$$O_{\text{sparse}} = \text{softmax}\left(\frac{QK^\top}{\sqrt{d}} + \mathcal{M}\right) V. \tag{2}$$

An entry $\mathcal{M}_{uv} = 0$ keeps the interaction between query u and key v, while $\mathcal{M}_{uv} = -\infty$ skips it. Block-sparse attention makes this mask hardware-friendly: it partitions the queries into blocks of B_q tokens and the keys and values into tiles of B_k tokens, typically of equal size (e.g., $B_q = B_k = 128$) to match the tiles of FlashAttention kernels (Dao et al., 2022), and keeps or skips each block–tile pair as a whole, so every kept pair runs as dense tensor-core matrix multiplications. Its central trade-off is granularity: large blocks keep the tensor cores fully utilized but select coarsely, whereas small blocks select more precisely but yield small matrix multiplications that underutilize the GPU.

4 Methodology

As illustrated in Figure 3, AB-Attention selects key/value blocks much smaller than an execution tile and estimates the dense attention of each query block from a weighted sample of them. Section 4.1 introduces the Intra-Block Clustering Selector (IBCS), which scores these blocks with query clusters formed within each query block. Section 4.2 introduces Sampling with Inclusion-Probability Weighting (SIP), which samples blocks in proportion to these scores and reweights them by inverse inclusion probability, so that one block-sparse pass estimates dense attention rather than truncating it. Section 4.3 introduces fine-grained selection with tiled execution: its kernels run these small blocks on full tensor-core tiles without reordering tokens, preserving the efficiency of block-sparse attention. Algorithm 1 summarizes the method for one attention head at one denoising step.

4.1 Intra-Block Clustering Selector (IBCS)

We divide the queries into blocks $Q_i \in \mathbb{R}^{B_q \times d}$ of B_q tokens, and the keys and values into much smaller blocks $K_j, V_j \in \mathbb{R}^{b \times d}$ of b tokens. The two sides are sized differently on purpose. Small key/value blocks select the attention more precisely: as Figure 2(b) shows, shrinking them from 128 to 4 tokens raises the attention mass kept by the oracle top-k from 59% to 69%. They also cost no extra tensor-core work, because the attention kernel of Section 4.3 packs the sampled blocks into full tiles. Small query blocks, however, bring no such benefit: queries that share a tile must either compute the union of their selections or run in small tiles that leave the tensor cores idle. AB-Attention therefore keeps each query block as tall as a tile and shrinks only the key/value blocks.

To score the key/value blocks for Q_i, IBCS needs a compact summary of its B_q queries. Simply averaging the queries of the block (Zhang et al., 2025b) or of its positional sub-blocks (Hu et al., 2026) is cheap, but mixes queries that attend to very different keys. Clustering tokens over the whole sequence (Yang et al., 2025) keeps such queries apart, but runs k-means over all tokens and reorders them, which is costly for long, high-resolution videos. IBCS clusters queries only within each query block and uses the clusters only for scoring, so attention itself still runs on the original token order.

Intra-block query clustering. IBCS partitions the queries of Q_i into R clusters $\mathcal{C}_{i1}, \ldots, \mathcal{C}_{iR}$ with a single k-means iteration: starting from R well-separated queries as initial centers, it assigns each query to its nearest center. Each cluster $\mathcal{C}_{ir}$ is summarized by its centroid $\bar{q}_{ir}$, the mean of its member queries, and its weight w_{ir}, the fraction of the queries of Q_i that it contains. Stacking them over all R clusters gives the full centroid matrix $\bar{Q}_i \in \mathbb{R}^{R\times d}$ and the corresponding weight vector $w_i \in \mathbb{R}^R$.

Block scores. For each query block, IBCS scores all $M = n/b$ key/value blocks of the sequence. Each block holds only b tokens, so the keys need far less summarization than the queries: IBCS only mean-pools small groups of adjacent keys, as detailed in Appendix A.1. $\bar{K}_j$ denotes the pooled keys of block K_j, and $\bar{K} = [\bar{K}_1; \ldots; \bar{K}_M]$ stacks those of all blocks. The score of block j is the attention mass on its pooled keys, averaged over the centroids with the cluster weights:

$$S_{ij} = \sum_{r=1}^{R} w_{ir} \frac{\sum_{\bar{k}\in\bar{K}_j} \exp\left(\bar{q}_{ir}\bar{k}^\top/\sqrt{d}\right)}{\sum_{\bar{k}\in\bar{K}} \exp\left(\bar{q}_{ir}\bar{k}^\top/\sqrt{d}\right)}, \tag{3}$$

where the fraction is the softmax attention that centroid $\bar{q}_{ir}$ assigns to the pooled keys of block j. A block that only one cluster attends to thus still keeps a score in proportion to that cluster's weight, whereas a single mean query of the whole block would average it away with the other clusters. As Figure 2(b) shows, IBCS thereby halves the recall gap of mean pooling to the oracle top-k.

4.2 Sampling with Inclusion-Probability Weighting (SIP)

Top-k, threshold, and static-mask methods compute a fixed set of blocks and simply drop all the rest from both the attention numerator and the softmax normalizer, a loss that better scores can shrink but never fully remove. Attention mass, however, has a long tail: as Figure 2(a) shows, even the oracle top-k keeps only about 69% of it with 10.8% of the keys. Recent correction methods (Liu et al., 2025b; Li et al., 2026; Zhou et al., 2026) approximate the dropped blocks in a separate, second computation path. SIP instead samples from all blocks and reweights each sampled block to stand in for the unsampled ones, so one block-sparse pass directly estimates the full dense attention.

Score-proportional sampling. For each query block Q_i, SIP samples s of the M key/value blocks, with s set by the compute budget. Block j is included with probability

$$\pi_{ij} = \min(1, \lambda_i S_{ij}), \qquad \textstyle\sum_{j=1}^{M} \pi_{ij} = s, \tag{4}$$

where λ_i scales the scores to the budget. High-scoring blocks reach $\pi_{ij} = 1$ and are always kept, as in top-k, while every other block keeps a nonzero chance, which the estimator below requires. SIP thus approaches top-k when the scores are concentrated and uniform sampling when they are flat. It draws the sample J_i by systematic sampling (Madow, 1949), which includes each block with probability π_{ij} and returns exactly s distinct blocks, so the sample fills whole tiles. All queries of Q_i share J_i, so the sample executes as block-sparse attention. Appendix A.2 details both steps.

Inverse-inclusion-probability weighting. Let $A_{ij} = \exp(Q_i K_j^\top/\sqrt{d})$. Block j contributes $N_{ij} = A_{ij}V_j$ to the attention numerator and the row sums D_{ij} of A_{ij} to the softmax normalizer. Dense attention sums both over all blocks, $N_i = \sum_{j=1}^{M} N_{ij}$ and $D_i = \sum_{j=1}^{M} D_{ij}$, and computes $O_i = N_i/D_i$ with row-wise division. Attending only to the sampled blocks would drop every unsampled block from both sums, so the sample would again prune attention rather than estimate it. SIP therefore weights each sampled block by $1/\pi_{ij}$, so that it also stands in for the blocks left out:

$$\widehat{O}_i = \frac{\widehat{N}_i}{\widehat{D}_i} = \frac{\sum_{j\in J_i} N_{ij}/\pi_{ij}}{\sum_{j\in J_i} D_{ij}/\pi_{ij}} = \mathrm{softmax}\Big(\frac{Q_i K_{J_i}^\top}{\sqrt{d}} - \log \pi_{iJ_i}\Big) V_{J_i}, \tag{5}$$

Algorithm 1 AB-Attention for one attention head at denoising step t

Require: $Q, K, V \in \mathbb{R}^{n \times d}$; block sizes B_q and b; clusters R; budget s; plan cycle C

if t is a plan-rebuild step **then** ▷ every C steps
  $\bar{K}_j \leftarrow \textsc{MeanPool}(K_j)$ for $j = 1, \dots, M$
  for each query block Q_i **do**
    $(\bar{Q}_i, w_i) \leftarrow \textsc{Cluster}(Q_i, R)$ ▷ IBCS
    $S_{ij} \leftarrow$ score of block K_j (Eq. 3), $j = 1, \dots, M$
    $\pi_{ij} \leftarrow$ inclusion probability of K_j (Eq. 4), $j = 1, \dots, M$ ▷ SIP
    $J_i \leftarrow$ sample s distinct blocks with $\Pr(j \in J_i) = \pi_{ij}$ ▷ systematic sampling
  end for
end if
for each query block Q_i **do** ▷ recomputed at every step
  gather K_{J_i}, V_{J_i} and pack them into full tiles
  $\widehat{O}_i \leftarrow \mathrm{softmax}\big(Q_i K_{J_i}^\top / \sqrt{d} - \log \pi_{iJ_i}\big) V_{J_i}$
end for
return $\widehat{O}$

where K_{J_i} and V_{J_i} stack the sampled blocks. A block with $\pi_{ij} = 1$ keeps weight one, while a block sampled with probability $1/4$ counts four times. Since $\Pr(j \in J_i) = \pi_{ij}$, $\widehat{N}_i$ and $\widehat{D}_i$ are unbiased Horvitz–Thompson estimates (Horvitz & Thompson, 1952) of N_i and D_i for fixed activations, and their variance shrinks as the scores become proportional to the attention mass of the blocks. The ratio $\widehat{O}_i$ itself is biased in general, but as the last form shows, it is ordinary softmax attention over the sampled blocks with the logits of block j shifted by $-\log \pi_{ij}$, so it runs in a single standard attention pass. Appendix A.3 gives the derivation. As Figure 2(c) shows, explicitly accounting for the tail brings the attention error of SIP 41% below that of the oracle top-k at the same block size.

4.3 Fine-Grained Selection with Tiled Execution

Tensor cores need full tiles, but the keys in a tile need not be contiguous. AB-Attention therefore decouples the selection unit from the execution unit: it selects small key/value blocks, and packs the blocks sampled for a query block into full tiles. A single fused attention kernel, set out in Algorithm 1, gathers them directly on chip, applies each block's logit shift, and runs the usual tiled online softmax. A query block thus runs exactly as many tensor-core tiles as tile-level selection at the same budget, without reordering any tokens or ever materializing the gathered keys and values.

The selector kernel decouples the same way: scoring one query block against one key/value block involves only a few centroids against a few pooled keys, far too small for a tile, so the kernel tiles the score matrix across several blocks on each side and groups the partial results afterwards. The block size then drops out of the cost of both kernels and can be chosen for selection quality alone.

Plan construction is amortized by rebuilding the sampling plan every C steps from the first sparse step and reusing it in between, while attention is recomputed at every step from current activations.

5 Experiments

5.1 Experimental Setup

Models and datasets. We evaluate three text-to-video diffusion transformers: Wan2.1-T2V-1.3B and Wan2.1-T2V-14B (Wan et al., 2025), and HunyuanVideo-T2V-13B (Kong et al., 2024). Every generated video is sampled at 720p resolution and contains 81 frames on Wan2.1 or 129 frames on HunyuanVideo. Text prompts are the 600 prompts of the Penguin Benchmark (Kong et al., 2024).

Metrics. Since a sparse method only approximates full attention, we compare each sparse video with the corresponding full-attention video from the same prompt and seed and report PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018). To evaluate the videos themselves, independent of the reference, we further report five VBench (Huang et al., 2024) dimensions: aesthetic quality (AQ), background consistency (BC), imaging quality (IQ), motion smoothness (MS),

Table 1: Quality and efficiency comparison of AB-Attention and baselines on Wan2.1-T2V-1.3B, Wan2.1-T2V-14B, and HunyuanVideo-T2V-13B. Shaded rows report the full-attention reference.

Method	Perceptual Metrics			VBench					Efficiency		
	PSNR ↑	SSIM ↑	LPIPS ↓	AQ ↑	BC ↑	IQ ↑	MS ↑	SC ↑	Sparsity ↑	Latency (s) ↓	Speedup ↑
Wan2.1 1.3B	∞	1.0000	0.0000	59.11	95.96	65.48	98.99	95.96	0%	962.39	1.00×
Radial	27.856	0.8672	0.1704	59.65	95.97	66.56	99.02	95.63	73.63%	549.91	1.75×
DFSAttn	27.734	0.8508	0.2193	57.83	95.78	67.30	98.93	95.45	81.03%	548.75	1.75×
SVG2	30.021	0.8858	0.1576	54.59	94.82	58.25	98.90	93.84	80.12%	536.47	1.79×
AB-Attention	**34.436**	**0.9328**	**0.0734**	58.02	95.75	63.92	98.96	95.84	**89.16%**	479.11	2.01×
Wan2.1 14B	∞	1.0000	0.0000	59.39	96.32	66.84	99.16	96.18	0%	4743.52	1.00×
Radial	26.696	0.8595	0.1708	59.43	96.19	66.56	99.14	95.90	73.63%	2907.31	1.63×
DFSAttn	27.708	0.8784	0.1464	58.50	96.12	67.04	99.15	95.82	81.03%	2866.46	1.65×
SVG2	29.236	0.8950	0.1274	58.08	95.90	65.27	99.09	95.66	67.89%	3169.10	1.50×
AB-Attention	**30.788**	**0.9169**	**0.0964**	58.92	96.23	66.09	99.14	96.11	**89.16%**	2571.52	1.84×
HunyuanVideo 13B	∞	1.0000	0.0000	57.45	96.12	66.13	99.16	95.39	0%	4399.03	1.00×
Radial	27.686	0.8802	0.1655	57.21	96.04	65.89	99.16	95.07	73.03%	2412.27	1.82×
DFSAttn	30.251	0.9141	0.1151	57.22	96.04	65.90	99.14	95.27	81.29%	2129.13	2.07×
SVG2	30.230	0.8967	0.1529	54.97	95.66	62.18	99.07	94.84	72.37%	2464.84	1.78×
AB-Attention	**34.347**	**0.9480**	**0.0700**	57.03	96.07	65.29	99.15	95.36	**92.35%**	1974.61	2.23×

and subject consistency (SC). For efficiency, sparsity is the share of video-attention computation skipped in the sparse denoising steps, and speedup is the ratio of dense to sparse end-to-end latency.

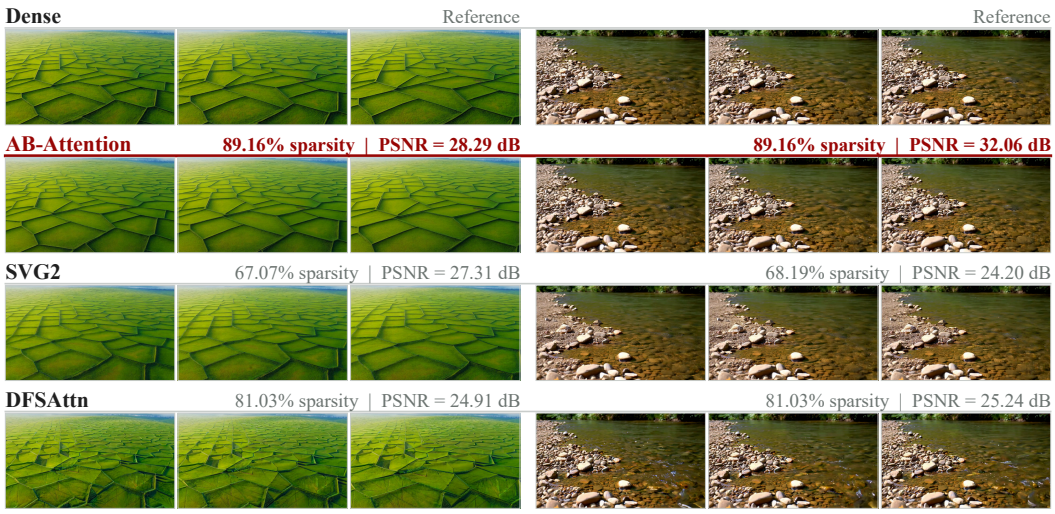
Baselines. We compare against three recent state-of-the-art training-free sparse attention methods. RadialAttention (Li et al., 2025) applies a static energy-decay mask, whereas SVG2 (Yang et al., 2025) and DFSAttn (Hu et al., 2026) build their masks dynamically from the current queries and keys. We use the official released code and the default hyperparameters of each baseline, as detailed in Appendix B.2. The officially released DFSAttn executes its masks with a FlashAttention-2-based block-sparse kernel (Guo et al., 2024); we instead run them with the same FlashAttention-3 block-sparse kernel from FlashInfer (Ye et al., 2025), which RadialAttention and SVG2 also use.

Implementation details. Each video is generated on NVIDIA H20 GPUs in BF16 precision with PyTorch 2.6 and CUDA 12.4. Dense attention, including the dense steps of every method, uses FlashAttention-3 (Shah et al., 2024). Following prior work (Li et al., 2025; Hu et al., 2026), all methods keep attention dense in the first transformer block and during the first 10 of the 50 denoising steps. AB-Attention uses a key/value block size of 4, a query block size of 128, and eight query clusters per query block, and rebuilds its sampling plan every four denoising steps. Its sampling budget is kept fixed for each model, resulting in an 89.16% sparsity on both Wan2.1 models and 92.35% on HunyuanVideo. End-to-end latency is measured following the protocol in Appendix C.

5.2 Main Results

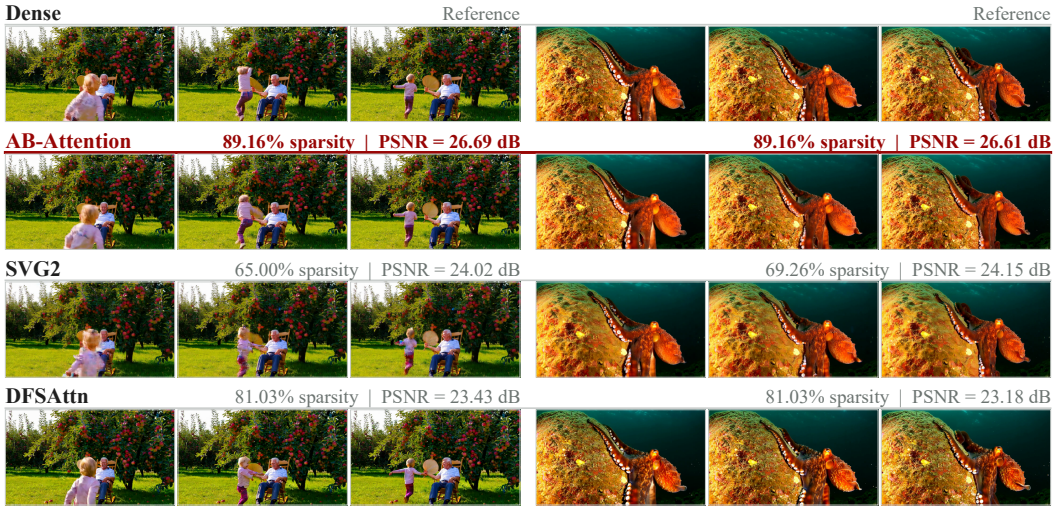
Table 1 reports the quality and efficiency of every method on all three models. AB-Attention attains the best PSNR, SSIM, and LPIPS of any method on every model and consistently improves PSNR over the strongest baseline by 1.6–4.4 dB. It does so while skipping 89–92% of attention computation, more than any baseline: every competing baseline computes 1.8–3.6 times as much attention. Figures 1 and 4 show the same effect visually: AB-Attention stays close to the dense reference where the baselines lose fine detail. On VBench, AB-Attention remains close to dense attention on all five dimensions, showing that the videos it generates retain their overall visual quality.

Efficiency follows the same ordering. AB-Attention reaches end-to-end speedups of 1.84–2.23×, ahead of every baseline on every model, and cuts the generation time of a HunyuanVideo video from 4399 s to 1975 s. Its closest competitor, DFSAttn on HunyuanVideo, reaches 2.07× only while computing 2.4 times as much attention, 18.71% versus 7.65% of dense attention. AB-Attention is thus at once the sparsest, the most faithful, and the fastest method on every model in Table 1.



(a) *A vast grassland where every blade of grass is made up of geometric shapes ...*

(b) *The stream flows gently, with pebbles scattered along its bed.*



(c) *In the yard, there is an apple tree ... A child is running and playing around the apple tree.*

(d) *In the deep sea, there is a large rock ... To the right of the rock is a large octopus ...*

Figure 4: Qualitative comparison on Wan2.1-T2V-14B. Panels (a)–(d) are four Penguin Benchmark prompts, abbreviated with ellipses. Within a panel, rows are methods and columns are frames 12, 40, and 68 of the same video. Each header reports that video's own sparsity and its full-video PSNR against the dense reference in the first row.

5.3 Ablation Study

Table 2 ablates the three core designs of AB-Attention one at a time, keeping the other two fixed: the query clusters of IBCS, the sampling and weighting of SIP, and the fine-grained key/value selection unit. All variants use Wan2.1-T2V-14B with 100 prompts randomly sampled from the 600 Penguin Benchmark prompts and run at the same 89.16% sparsity of the main results, so the differences reflect how each variant spends the same budget rather than how much attention it computes.

Query clustering. IBCS summarizes each query block with its eight query clusters. To test whether grouping queries by feature similarity matters, we instead average eight consecutive 16-query intervals, as in positional sub-block pooling (Hu et al., 2026), keeping the number of repre-

Table 2: Ablation of the AB-Attention components on Wan2.1-T2V-14B. Each variant replaces one component at the same 89.16% sparsity.

	Variant	PSNR ↑		SSIM ↑	LPIPS ↓
	AB-Attention	**29.705**		**0.9154**	**0.0956**
IBCS	Query pooling	28.793	−0.91	0.9024	0.1112
SIP	w/o weights	25.706	−4.00	0.8427	0.1911
	Uniform w/o weights	22.452	−7.25	0.7437	0.4749
	Top-k selection	25.711	−3.99	0.8420	0.1996
Tiling	Tile-level selection	28.686	−1.02	0.9006	0.1113

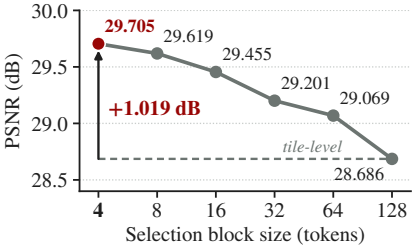


Figure 5: PSNR versus selection block size on Wan2.1-T2V-14B.

sentatives and hence the overall scoring cost unchanged. This *query pooling* variant lowers PSNR by 0.91 dB, with SSIM and LPIPS degrading accordingly. A positional interval mixes queries that attend to different keys, so its single mean can underrate a key block that only some of them attend to, whereas a cluster gathers queries with similar features, which tend to attend to similar keys.

Sampling and weighting. We compare SIP with three simpler alternatives at the same budget: score-proportional sampling without the inverse-inclusion weights, uniform sampling without weights, and deterministic top-k selection. Removing the weights lowers PSNR by 4.00 dB, to the level of top-k selection: without them, the blocks outside the sample are simply dropped, and sampling becomes another form of pruning. Discarding the scores as well then costs a further 3.25 dB and more than doubles LPIPS, because uniform sampling spends most of its fixed budget on the many blocks that receive little attention, as Figure 2(a) shows. The scores thus decide where to spend the limited budget, and the weights account for the blocks it leaves out; SIP needs both.

Selection granularity. *Tile-level selection*, which scores and samples whole execution tiles as conventional block-sparse attention does, lowers PSNR by 1.02 dB. As Figure 5 shows, PSNR rises steadily as the selection block shrinks from a full tile toward the 4-token blocks of AB-Attention, and Appendix D shows the same trend for SSIM and LPIPS. Smaller key/value blocks let the sample keep the few important keys in a region without spending any budget on their neighbors, and since the kernel of Section 4.3 packs them into full tiles, this gain costs no extra tensor-core work.

Overall, replacing SIP costs by far the most, 3.99–7.25 dB of PSNR, followed by tile-level selection at 1.02 dB and query pooling at 0.91 dB. Every variant is worse than AB-Attention on all three metrics, so at the same budget each component of AB-Attention contributes to its fidelity.

6 Conclusion

We presented AB-Attention, a training-free sparse attention method for video diffusion transformers that directly estimates dense attention rather than truncating it. IBCS scores key/value blocks with lightweight query clusters formed only within each query block, and SIP samples blocks in proportion to these scores and reweights them by their inverse inclusion probabilities, so that a single block-sparse pass yields unbiased estimates of the attention numerator and normalizer. AB-Attention further decouples selection from execution: it selects key/value blocks much smaller than a tile and packs them into full tensor-core tiles in its kernel. On both Wan2.1 and HunyuanVideo, AB-Attention skips 89–92% of attention, whereas the baselines compute 1.8–3.6× as much, yet it stays closest to dense attention and reaches the highest end-to-end speedups of 1.84–2.23×.

Limitations and future work. Despite these encouraging results, AB-Attention remains limited in both its current implementation and application scope. Our kernel is built for the H20 GPUs used in our experiments and currently supports only forward inference, while our evaluation covers three text-to-video diffusion transformers. Future work will address these limitations through kernel support for further hardware, evaluation on a broader range of models, and a trainable implementation with backward support that enables fine-tuning under AB-Attention.

AI USE STATEMENT

We used AI tools to assist with grammar correction and language polishing. The authors reviewed and approved all AI-assisted edits and made all final decisions regarding the content and wording of the paper. The authors take full responsibility for the paper and its associated artifacts.

ETHICS STATEMENT

This work involves no human subjects and no new data, and uses publicly released models and prompts under their licenses. As inference-time acceleration, AB-Attention inherits these models' misuse risks, such as misleading synthetic media, without weakening their safeguards.

REPRODUCIBILITY STATEMENT

Section 4 and Algorithm 1 specify AB-Attention in full, and Section 5.1 with Appendices A and B give every experimental detail, including baseline configurations and the seeds and prompts shared by all methods. We will release our implementation upon acceptance of this paper.

REFERENCES

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 16344–16359. Curran Associates, Inc., 2022. doi: 10.52202/068431-1189. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/67d57c32e20fd0a7a302cb81d36e40d5-Paper-Conference.pdf.

Teofilo F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985. ISSN 0304-3975. doi: https://doi.org/10.1016/0304-3975(85)90224-5. URL https://www.sciencedirect.com/science/article/pii/0304397585902245.

Junxian Guo, Haotian Tang, Shang Yang, Zhekai Zhang, Zhijian Liu, and Song Han. Block Sparse Attention. https://github.com/mit-han-lab/Block-Sparse-Attention, 2024.

Daniel G. Horvitz and Donovan J. Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association*, 47(260):663–685, 1952. doi: 10.1080/01621459.1952.10483446.

Jie Hu, Zixiang Gao, Yutong He, and Kun Yuan. DFSAttn: Dynamic fine-grained sparse attention for efficient video generation. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=GO0G6bo9UO.

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*, pp. 21807–21818, 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. doi: 10.48550/arXiv.2412.03603. URL https://arxiv.org/abs/2412.03603.

Haopeng Li, Shitong Shao, Wenliang Zhong, Zikai Zhou, Lichen Bai, Hui Xiong, and Zeke Xie. Pisa: Piecewise sparse attention is wiser for efficient diffusion transformers, 2026. URL https://arxiv.org/abs/2602.01077.

Xingyang Li, Muyang Li, Tianle Cai, Haocheng Xi, Shuo Yang, Yujun Lin, Lvmin Zhang, Songlin Yang, Jinbo Hu, Kelly Peng, Maneesh Agrawala, Ion Stoica, Kurt Keutzer, and Song Han. Radial attention: $\mathcal{O}(n \log n)$ sparse attention with energy decay for long video generation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 16822–16852. Curran Associates, Inc., 2025. doi: 10.52202/085713-0568. URL https://papers.neurips.cc/paper_files/paper/2025/hash/18a367688479c1b08b001584218a4443-Abstract-Conference.html.

Shangran Lin, Lu Lu, Jian Chen, and Qiang Liu. Rapid: Reusing attention sparsity with inter-step adaptation for efficient video diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 36147–36156, June 2026.

Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep embedding tells: It's time to cache for video diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7353–7363, June 2025a.

Shanghao Liu, Renze Chen, Size Zheng, Yuanqiang Liu, Yun Liang, and Hailong Yang. SPADE: An input-adaptive sparse attention engine for fast video diffusion models inference. In *Proceedings of the 63rd ACM/IEEE Design Automation Conference*, 2026. doi: 10.1145/3770743.3804015. URL https://arxiv.org/abs/2608.03335.

Xuewen Liu, Zhikai Li, Jing Zhang, Mengjuan Chen, and Qingyi Gu. Rectified spaattn: Revisiting attention sparsity for efficient video generation, 2025b. URL https://arxiv.org/abs/2511.19835.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan LI, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 5775–5787. Curran Associates, Inc., 2022. doi: 10.52202/068431-0418. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/260a14acce2a89dad36adc8eefe7c59e-Paper-Conference.pdf.

William G. Madow. On the theory of systematic sampling, ii. *The Annals of Mathematical Statistics*, 20(3):333–354, 1949. doi: 10.1214/aoms/1177729988.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TIdIXIpzhoI.

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 A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 68658–68685. Curran Associates, Inc., 2024. doi: 10.52202/079017-2193. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/7ede97c3e082c6df10a8d6103a2eebd2-Paper-Conference.pdf.

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. doi: 10.48550/arXiv.2503.20314. URL https://arxiv.org/abs/2503.20314.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. doi: 10.1109/TIP.2003.819861.

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 videogen: Accelerating video diffusion transformers with spatial-temporal sparsity. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 68208–68224. PMLR, 13–19 Jul 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, October 2025.

Ruyi Xu, Guangxuan Xiao, Haofeng Huang, Junxian Guo, and Song Han. Xattention: Block sparse attention with antidiagonal scoring, 2025. URL https://arxiv.org/abs/2503.16428.

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 D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 96965–96991. Curran Associates, Inc., 2025. doi: 10.52202/085713-3243. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/8bd5148caced2d73cea7b6961a874a49-Paper-Conference.pdf.

Zihao Ye, Lequn Chen, Ruihang Lai, Wuwei Lin, Yineng Zhang, Stephanie Wang, Tianqi Chen, Baris Kasikci, Vinod Grover, Arvind Krishnamurthy, and Luis Ceze. Flashinfer: Efficient and customizable attention engine for LLM inference serving. In *Eighth Conference on Machine Learning and Systems*, 2025. URL https://openreview.net/forum?id=RXPofAsL8F.

Jintao Zhang, Jia wei, Pengle Zhang, Jun Zhu, and Jianfei Chen. Sageattention: Accurate 8-bit attention for plug-and-play inference acceleration. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 71566–71585, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/b286c344d38e10d2466c0514b78e2f36-Paper-Conference.pdf.

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 *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 76397–76413. PMLR, 2025b. URL https://proceedings.mlr.press/v267/zhang25ch.html.

Peiyuan Zhang, Yongqi Chen, Runlong Su, Hangliang Ding, Ion Stoica, Zhengzhong Liu, and Hao Zhang. Fast video generation with sliding tile attention. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 74714–74731. PMLR, 2025c. URL https://proceedings.mlr.press/v267/zhang25m.html.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. 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. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_The_Unreasonable_Effectiveness_CVPR_2018_paper.html.

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, 2026. URL https://arxiv.org/abs/2603.08982.

APPENDIX

Appendix contents

A Sampling and Estimation Details **13**

A.1 Query Centroids and Pooled Keys 13

A.2 Systematic Sampling Without Replacement 13

A.3 Weighted Attention Estimation 13

B Implementation Details **14**

B.1 AB-Attention 14

B.2 Baselines 14

C Evaluation Protocol **14**

D Additional Ablation Results **15**

A SAMPLING AND ESTIMATION DETAILS

A.1 QUERY CENTROIDS AND POOLED KEYS

The single k-means iteration of IBCS starts from R centers chosen in each query block by farthest-first traversal (Gonzalez, 1985). For the four-token key blocks, the first two and the last two keys are mean-pooled into the two pooled keys of $\bar{K}_j$ in Equation 3. Pooling halves the scoring cost, and since every pooled key stands for two keys, key-side size weights cancel in the softmax.

A.2 SYSTEMATIC SAMPLING WITHOUT REPLACEMENT

Inclusion probabilities. The scale λ_i in Equation 4 is found iteratively: scale the remaining scores to the remaining budget, set $\pi_{ij} = 1$ for blocks whose scaled scores reach one, deduct them from the budget, and repeat. Scores are floored at 10^{-30}, so every π_{ij} stays positive, as Appendix A.3 requires. The budget s is a multiple of 32, so the sample fills whole 128-token tiles.

Systematic draw. Let $H_i = \{j : \pi_{ij} = 1\}$ and $s' = s - |H_i|$. Order the remaining blocks as $j_1, \ldots, j_L$ by index, let $F_0 = 0$ and $F_t = \sum_{h=1}^{t} \pi_{ij_h}$, so $F_L = s'$, draw $\xi \sim \mathrm{Uniform}[0, 1)$, and include j_t if

$$F_{t-1} \le \xi + r < F_t \quad \text{for some } r \in \{0, \ldots, s' - 1\}. \tag{6}$$

The intervals $[F_{t-1}, F_t)$ tile $[0, s')$, which contains the s' thresholds $\xi + r$ spaced one apart. An interval of length $\pi_{ij_t} < 1$ contains at most one threshold, and does so with probability π_{ij_t} (Madow, 1949). With H_i, the draw returns exactly s distinct blocks with inclusion probabilities π_{ij}. Independent per-block draws would make the sample size random, and sampling with replacement could repeat blocks, leaving tiles partly empty or wasting tensor-core work on duplicates.

A.3 WEIGHTED ATTENTION ESTIMATION

With the block contributions N_{ij} and D_{ij} defined before Equation 5, let I_{ij} indicate whether block j is sampled for query block i. The Horvitz–Thompson estimates

$$\widehat{N}_i = \sum_{j=1}^{M} \frac{I_{ij} N_{ij}}{\pi_{ij}}, \qquad \widehat{D}_i = \sum_{j=1}^{M} \frac{I_{ij} D_{ij}}{\pi_{ij}} \tag{7}$$

satisfy $\mathbb{E}[\widehat{N}_i] = N_i$ and $\mathbb{E}[\widehat{D}_i] = D_i$ for fixed activations, since $\mathbb{E}[I_{ij}] = \pi_{ij}$; this needs no independence between blocks, which systematic sampling lacks. The ratio $\widehat{O}_i = \widehat{N}_i / \widehat{D}_i$ is biased in general. Since $A_{ij}/\pi_{ij} = \exp(Q_i K_j^\top / \sqrt{d} - \log \pi_{ij})$, it is softmax attention over sampled blocks with the logits of block j shifted by $-\log \pi_{ij}$, the last form of Equation 5. Tiling the sampled blocks

changes neither their indices nor weights, so the online-softmax kernel of Algorithm 1 computes $\widehat{O}_i$ exactly.

B Implementation Details

B.1 AB-Attention

On Wan2.1, AB-Attention replaces the self-attention among video tokens, and cross-attention to the text remains dense. HunyuanVideo attends jointly over video and prompt tokens; AB-Attention samples only the video keys of video queries, while attention to prompt keys and all attention from prompt queries remain exact.

Both Wan2.1 models share one configuration. Queries are divided into blocks of $B_q = 128$ tokens and keys and values into blocks of $b = 4$ tokens. IBCS forms $R = 8$ query clusters per query block with one k-means iteration from farthest-first centers, and pools each key block into two keys, as described in Appendix A.1. SIP samples $s = 2048$ key/value blocks (8192 keys) per query block on Wan2.1 and $s = 2272$ (9088 keys) on HunyuanVideo, 10.84% and 7.65% of the 18,900 and 29,700 blocks of a video, which gives the sparsity of 89.16% and 92.35%. Attention stays dense in the first transformer block and in denoising steps 1–10. From step 11, the sampling plan is rebuilt every $C = 4$ steps, at steps 11, 15, . . . , 47, and reused for the steps in between. Sampling uses seed 42.

B.2 Baselines

RadialAttention (Li et al., 2025) uses a decay factor of 0.2 on Wan2.1 and 0.95 on HunyuanVideo, with the 720p video tokens zero-padded to a multiple of its 128-token mask block. SVG2 (Yang et al., 2025) groups queries and keys into 100 and 500 clusters and keeps key clusters up to a cumulative attention of 0.9, with a minimum key ratio of 0.1. DFSAttn (Hu et al., 2026) reorders tokens along a 3D Hilbert curve, scores blocks with 16-token sub-blocks, and refreshes its mask every 12 steps; the retained block fraction starts at 0.3 and drops by 0.1 per refresh down to 0.1.

C Evaluation Protocol

Quality metrics. PSNR, SSIM, and LPIPS with the VGG backbone are computed on RGB frames in $[0, 1]$, averaged over the frames of each video, and then averaged over prompts. The five VBench dimensions are evaluated per video and averaged over prompts.

Sparsity. Sparsity counts only the sparse denoising steps. It comes from the static mask for RadialAttention, from the allocated top-k budget for DFSAttn, from the token pairs kept by the recorded cluster masks for SVG2, and from the recorded block selections for AB-Attention.

Table 3: Fidelity to dense attention for each selection block size on Wan2.1-T2V-14B (100 prompts). Every row executes 8192 key tokens per query block in 128-token tiles (89.16% sparsity). Block size 4 is the full AB-Attention, and block size 128 is tile-level selection.

Block size	PSNR ↑	SSIM ↑	LPIPS ↓
4	**29.705**	**0.9154**	**0.0956**
8	29.619	0.9150	**0.0956**
16	29.455	0.9108	0.0985
32	29.201	0.9092	0.1015
64	29.069	0.9072	0.1032
128	28.686	0.9006	0.1113

Latency. Latency is measured end to end for one video on a single NVIDIA H20 GPU. The timed region covers text encoding, all 50 denoising steps, and VAE decoding, and excludes model loading, initial-noise loading, and video export. For AB-Attention, plan construction and sampling fall inside the denoising steps and are therefore timed.

D Additional Ablation Results

Table 3 reports PSNR, SSIM, and LPIPS for the block-size sweep of Figure 5; none of them degrades as the selection block shrinks from 128 to 4 tokens.