IMPROVING VIDEO SPARSE ATTENTION WITH FINE-GRAINED ROUTER AND SPARSE REBASING

Anonymous authors
Paper under double-blind review

ABSTRACT

We present VSA2, a frontier trainable sparse attention for video DiTs. VSA2 includes a variety of new architectural features and training procedures that we apply across all stages of DiT development cycle – including pretraining, RL, and inference – to produce a DiT with comparable or better quality than a full attention counterpart. Architecturally, VSA2 introduces a fine-grained router that improves the precision of identifying critical tokens and supports dynamic computation by allowing each query to attend to a variable number of key–value pairs. On training, we identify a Hard-to-Easy Curriculum, where models trained under high sparsity and later evaluated with lower sparsity during inference not only generalize effectively, but also outperform models trained with full attention in motion quality. VSA2 is also flexible: it can replace full attention during the middle of progressive low-to-high resolution pretraining, rebasing early-stage full-attention checkpoints. Experiments show that VSA2 reduces attention computation by half over VSA with lower loss. On 720P videos, it accelerates attention by 8.9x and end-to-end generation by 4.62x compared to the FlashAttention3 baseline, while achieving comparable or better video quality.

1 INTRODUCTION

Video diffusion transformers (DiTs) are rapidly approaching practical utility. Recent systems, such as Genie3 (Ball et al., 2025), generate interactive, minute-long, high-resolution video streams while maintaining persistent memory of earlier events. Scaling video DiTs to this regime is nontrivial: even a 5-second HD clip exceeds 100K tokens; this makes self-attention, whose cost is quadratic with sequence length, the dominating cost at both training and inference (Yang et al., 2024; Kong et al., 2024; Wang et al., 2025). Practically, most entries in $\mathrm{Softmax}(\mathbf{QK}^\top/\sqrt{\mathbf{D}})$ contribute negligibly to attention output, with a small set of *critical tokens* that carry the signal (Zhang et al., 2023; Jiang et al., 2024; Ding et al., 2025). This observation has driven a wave of *inference-only* sparse attention methods (Zhang et al., 2025c; Xi et al., 2025; Xu et al., 2025; Zhang et al., 2025f) that prune low-weight interactions to accelerate DiT. While effective for inference, these methods leave the costly pretraining phase untouched and thus cannot unlock long-context training.

To reduce *training* FLOPs, the field has turned to trainable sparse attention Lu et al. (2025a); Zhang et al. (2025g); Zhan et al. (2025); Zhang et al. (2025a). A common design is a two-stage, coarse-to-fine framework: a coarse branch acts as a router to pool neighboring tokens into tiles and computes full attention at tile granularity to form a coarse attention map; based on the map, the coarse branch applies TopK to identify tiles that likely contain critical tokens. Then, a fine branch computes token-level block-sparse attention only within the selected tiles. These methods have managed to sparsify full attention by 80% at post-training, but run into a sparsity ceiling for two mechanism-level reasons.

First, for hardware efficiency, the fine branch must use block-sparse kernels with block size B (e.g. 64, 128) aligned with hardware characteristics. In most prior routers, the pooling stride of the coarse branch implicitly matches B, so each coarse token represents a $(B_t \times B_h \times B_w)$ cube. This coarse view blurs the sparse structures in $\mathbf{QK}^\top$. At long horizon and high resolution, this aliasing either hurts the recall of truly critical tokens, results in quality drop, or forces more tiles to be kept, leading to sparsity drop. Second, standard per-token topK forces every query to attend to the same number of KVs, regardless of difficulty. Easy queries waste compute on redundant context; hard queries are

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.

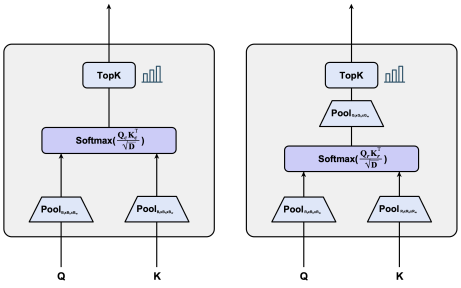

Figure 1: Illustration of the sparse attention router used in Zhang et al. (2025g;c); Zhan et al. (2025); Zhang et al. (2025a) versus the fine-grained router in our VSA2.

Symbol	Description
T, H, W	Video dimensions after VAE compression.
L	Total number of video tokens. $L = T \times H \times W$.
$B = B_t B_h B_w$	Block size and corresponding 3D cube size.
$N = N_t N_h N_w$	Number of cubes along each dimension. $N_t = T/B_t$, $N_h = H/B_h$, $N_w = W/B_w$.
$R = R_t R_h R_w$	Router input-pooling size. $R_t < B_t$, $R_h < B_h$, $R_w < B_w$.
$G = G_t G_h G_w$	Router score-pooling size. $G_t = B_t/R_t$, $G_h = B_h/R_h$, $G_w = B_w/R_w$.

Table 1: Notation summary for Video Sparse Attention 2.

under-provisioned. This caps performance at high sparsity because raising sparsity uniformly starves the very queries that need more context.

We present Video Sparse Attention 2 (VSA2), a sparse attention mechanism that allows for greater sparsity and more precise critical token identification. This is enabled by a finer-grained router decoupled from hardware block size, and a sequence-level budget that is fixed overall but adaptively allocated across queries. As shown in Figure 1 and Table 1, VSA2 decouples the mean-pooling size of the router from hardware-efficient block size. We use a hardware-efficient block size B for sparse attention and a smaller pooling size R for the router's queries and keys. After softmax, we aggregate the router scores using $G \times G$ pooling, where $G = B/R$. In parallel, we replace per-token topK with per-sequence topK, keeping the sequence level computation the same but allowing to assign variable compute budget for different query tokens. With a fine-grained router and per-sequence topK, VSA2 achieves lower loss than full attention while being 90% to 95% sparse.

Our systematic studies on pretraining with VSA2 further identify an effective training recipe named SPARSE REBASING. We find that VSA2 functions as a drop-in replacement for full attention within a progressive low-to-high resolution training pipeline: reuse full-attention checkpoints from the early image and low-resolution video training stages, and only introduce VSA2 in the most compute-demanding phases involving high-resolution, long-duration videos, avoiding training from scratch. A crucial component of SPARSE REBASING is HARD-TO-EASY CURRICULUM, where we find that training models with aggressive sparsity and later relaxing sparsity at inference leads to better motion quality compared to the full-attention baseline. We train a video DiT on 480-720P videos and RL preference pairs to verify the recipe end-to-end. The final model matches its full attention counterpart in human evaluations and sometimes outperforms it by producing better motion. At 720P, VSA2 accelerates the attention operation by 8.9× and the end-to-end generation by 4.62×. Further, despite being trained on 5–12s clips, the model directly generates 30s videos, indicating headroom toward minute-long generation.

In summary, this paper makes the following contribution: (1) We propose VSA2, an improved video sparse attention with a fine-grained router. (2) We identify practical training recipes, SPARSE REBASING, for pretraining with VSA2. (3) Powered by VSA2, we train a video DiT that matches or outperforms the full attention counterpart while being up to 95% sparse. To our knowledge, VSA2 is the first trainable sparse attention that is end-to-end verified in various stages of video DiT development.

2 METHOD

2.1 BACKGROUND

Most existing video DiTs employ 3D full attention to capture dependencies across the entire spatiotemporal volume. A video latent of shape (T, H, W) is first flattened into a 1D sequence of length $L = THW$, and then the attention output is computed as: $\mathbf{S} = \mathbf{Q}\mathbf{K}^\top/\sqrt{D}, \mathbf{P} = \text{Softmax}(\mathbf{S} + \mathbf{M}), \mathbf{O} = \mathbf{P}\mathbf{V}$. In *full attention*, all entries in M are zero, allowing dense interactions

throughout the sequence. *Sparse attention* instead introduces $-\infty$ entries in $\mathbf{M}$, which can reduce FLOPs by skipping the corresponding computation in both $\mathbf{QK}^\top$ and $\mathbf{PV}$.

Since attention scores are highly non-uniform, full attention can be approximated by sparse attention if we construct a mask $\mathbf{M}$ that preserves entries with large values in $\mathbf{P}$ while discarding those with negligible contribution. This is analogous to the router in mixture-of-experts Shazeer et al. (2017); Lu et al. (2025a), where each token must be routed to a subset of important experts; here, each query block must be routed to a subset of high-contribution key–value pairs. Moreover, since modern accelerators are optimized for dense computation, unstructured sparsity rarely translates into real speedups. Block-sparse attention (Dao et al., 2022) is necessary for hardware efficiency, where every (B, B) block of $\mathbf{M}$ shares the same value. Thus, the sparse attention problem reduces to designing an accurate router that produces the block-level mask $\mathbf{M}$, consisting of $(\frac{L}{B}, \frac{L}{B})$ boolean entries, since all tokens within a (B, B) tile share the same value.

VSA Zhang et al. (2025g) addresses this by applying mean pooling, producing coarse-branch representations $\mathbf{Q}_c, \mathbf{K}_c, \mathbf{V}_c \in \mathbb{R}^{(L/B)\times D}$, where each token corresponds to a 3D cube (B_t, B_h, B_w). The coarse branch then perform dense attention $\mathbf{S}_c = \mathbf{Q}_c\mathbf{K}_c^\top/\sqrt{D}, \mathbf{P}_c = \text{Softmax}(\mathbf{S}_c), \mathbf{O}_c = \mathbf{P}_c\mathbf{V}_c$. The router reuses $\mathbf{P}_c$ and applies a topK selection to identify the index of the highest-scoring key-value blocks for each query block: $\mathbf{M} = \text{TopK}(\mathbf{Q_c}{\mathbf{K_c}}^\top, \text{K}, \text{dim} = -1)$. A fine branch then uses $\mathbf{M}$ to perform token-level block-sparse attention.

Normally, the router topK, like those in mixture-of-expert language models Shazeer et al. (2017), is learned with gradients propagated from the expert outputs. However, in sparse attention, while the coarse branch outputs $\mathbf{O}_c$ carry gradients during backpropagation, its router topK outputs $\mathbf{M}$ do not. Consequently, there is no gradient path from the fine branch back through the routing decision to update the coarse branch's routing capability. This naturally raises two questions: how does routing quality emerge and how can we further improve it? We consider two hypotheses. First, *locality heuristics*: neighboring tokens tend to be similar, so mean pooling already yields informative cube features and the router need not receive trainable parameters nor gradients, in line with observations from MoBA Lu et al. (2025a) and Quest Tang et al. (2024). Second, *auxiliary supervision*: the router shares parameters with the coarse branch, and the gradients from $\mathbf{O}_c$ benefit the router, despite the fact that the gradients do not come from $\mathbf{M}$, as in the case of NSA Yuan et al. (2025). In §3.3, we perform apple-to-apple ablations that isolate these factors and find that explicit gradient flow to the router is unnecessary. Based on this result, VSA2 therefore focuses on improving the accuracy *of gradient-free routing* by decoupling the pooling granularity of the router from the granularity of the block-sparse attention, so that each query block reliably identifies the most contributing key–value blocks.

2.2 VSA2

Table 1 summarizes our notation. As sketched in Figure 2, VSA2 has three components—a *coarse branch*, a *fine branch*, and a *router*—that work together to turn dense 3D attention into hardware-aligned block sparse attention while preserving key dependencies.

Permutation. We first permute the tokens into cube-major order: $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{THW\times D} = \mathbb{R}^{N_t N_h N_w \times B_t B_h B_w \times D} = \mathbb{R}^{N\times B\times D}$ Zhang et al. (2025e;g). This permutation can be applied once at the beginning of the transformer with proper permutation of RoPE embeddings Su et al. (2024), as attention is the only operation that depends on token order.

Coarse Branch. The coarse branch captures high-level structure via cube-level full attention, which is the same as VSA:

$$[\mathbf{Q}_c, \mathbf{K}_c, \mathbf{V}_c] = \text{MeanPool}_B([\mathbf{Q}, \mathbf{K}, \mathbf{V}]) \in \mathbb{R}^{3N\times D}, \tag{1}$$

$$\mathbf{P}_c = \text{Softmax}\Big(\mathbf{Q}_c\mathbf{K}_c^\top/\sqrt{D}\Big) \in \mathbb{R}^{N\times N}, \tag{2}$$

$$\hat{\mathbf{O}}_c = \mathbf{P}_c\mathbf{V}_c \in \mathbb{R}^{N\times D}. \tag{3}$$

where MeanPool_B denotes mean pooling over the (B_t, B_h, B_w) cube. In our early experiments, we also explored alternative pooling methods, including those similar to Yuan et al. (2025) and

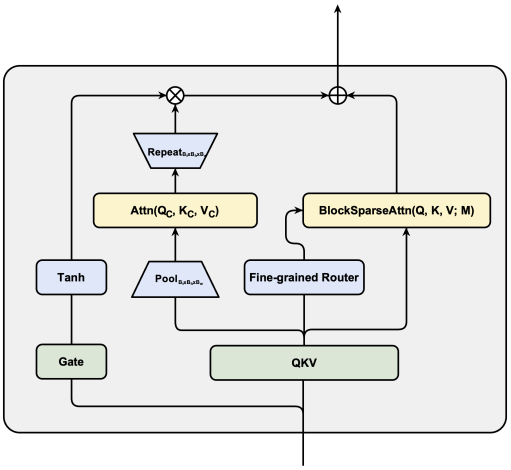

Figure 2: Architecture of Video Sparse Attention 2, comprising a coarse branch, a fine branch, and a separate router with a pool-softmax-pool structure.



Figure 3: Qualitative comparison after pretraining + RL. The top and bottom examples use image-to-video generation; the middle uses text-to-video.

attention-based pooling, but found them to be no better than simple mean pooling in video DiT. The coarse output is then broadcasted back to the original video resolution:

$$\mathbf{O}_c = \text{Repeat}_B(\hat{\mathbf{O}}_c) \in \mathbb{R}^{N\times B\times D}. \tag{4}$$

Router. In VSA, the coarse branch attention score $\mathbf{P}_c$ defines the cube-to-cube affinity score used to perform a Top-K selection and generate the block-sparse attention mask. In practice, we find the original pooling $B_t \times B_h \times B_w$ to be excessively large such that much information is lost in pooling. Instead, we decouple the router from the coarse branch and adopt a much smaller pooling size $R_t \times R_h \times R_w$:

$$[\mathbf{Q}_r, \mathbf{K}_r] = \text{MeanPool}_R([\mathbf{Q}, \mathbf{K}]) \in \mathbb{R}^{2N\times G\times D}, \tag{5}$$

$$\hat{\mathbf{P}}_r = \text{Softmax}\left(\frac{\mathbf{Q}_r\mathbf{K}_r^\top}{\sqrt{D}}\right) \in \mathbb{R}^{N\times N\times G\times G}. \tag{6}$$

After softmax, we aggregate the router score to block scores using another pooling operator:

$$\mathbf{P}_r = \text{MeanPool}_{G\times G}(\hat{\mathbf{P}}_r) \in \mathbb{R}^{N\times N}. \tag{7}$$

Based on $\mathbf{P}_r$, we select the topK key blocks at the sequence level rather than per token topK:

$$\mathbf{P}_r \in \mathbb{R}^{N\times N} \quad\longrightarrow\quad \mathbf{P}_r \in \mathbb{R}^{NN}, \tag{8}$$

$$\mathbf{M} = \text{TopK}_{NK}(\mathbf{P}_r), \tag{9}$$

which yields NK selected blocks per sequence. This design maintains a consistent compute budget while allowing each query to attend to different numbers of key-value blocks.

The intuition behind the pool-softmax-pool operation in Eq.(5)–(7) is simple. In the compute-unbounded case, an oracle router would first compute token-level attention and only then aggregate to block-level affinities:

$$\mathbf{P} = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{D}}\right) \in \mathbb{R}^{N\times N\times B\times B}, \tag{10}$$

$$\mathbf{P}_{oracle} = \text{MeanPool}_{B\times B}(\hat{\mathbf{P}}) \in \mathbb{R}^{N\times N}. \tag{11}$$

Equations (1)–(2) approximate $\mathbf{P}_{oracle}$ by moving the pooling step: instead of the full attention matrix $\hat{\mathbf{P}}$, it pools $\mathbf{Q}$ and $\mathbf{K}$ *before* applying the dot product and softmax. This reordering would be

Exp	Stage	Configuration				T2V			I2V	
		Train TopK	Infer TopK	Attn Sparsity	E2E Speedup	Motion	Following	Aesthetics	Motion	Following
1	480P	64	64	0.90	2.09×	-1.34%	1.34%	-0.67%	8.72%	1.34%
2	480P	64	128	0.82	1.92×	22.1%	-3.36%	-4.7%	6.71%	-8.72%
3	480P	32	128	0.82	1.92×	7.38%	-15.4%	-4.7%	11.4%	-9.4%
4	480P	32	64	0.90	2.09×	12.1%	-10.1%	-7.38%	4.03%	-4.03%
5	480P RL	64	64	0.90	2.09×	-4.03%	1.34%	-2.01%	8.05%	-4.7%
6	480P RL	64 + 128	128	0.82	1.92×	7.38%	-2.01%	0.00%	8.05%	-2.69%
7	720P	64	64	0.95	4.62×	6.71%	-2.01%	1.34%	0.67%	-1.34%

Table 2: Human evaluation results of VSA2 against full attention across various stages of video DiT training. A positive score means the model trained with VSA2 has better quality than its full attention counterpart at that stage.

exact if attention were linear, but because softmax is nonlinear, averaging $\mathbf{Q}$ and $\mathbf{K}$ is not equivalent to averaging $\hat{\mathbf{P}}$. As a result, the method provides an approximation rather than an identity. Its accuracy depends on a locality assumption: tokens within a cube $B_t \times B_h \times B_w$ are similar enough that pre-averaging them introduces minimal distortion.

Our fine-grained router, shown in Figure 1 and defined by equations (5)–(7), occupies a middle ground between the oracle scores $\mathbf{P}_{\text{oracle}}$ and the coarse-branch approximation $\mathbf{P}_c$. The key idea is to decouple the router's pooling size R from the block size B used in block sparse attention. We first apply a much *smaller* pooling $R = R_t \times R_h \times R_w$ to $\mathbf{Q}$ and $\mathbf{K}$, which reduces sequence length and router FLOPs while preserving substantially more intra-cube detail than B-pooling. We then compute the router's attention $\hat{\mathbf{P}}_r$ and aggregate them with a $G{\times}G$ operator to obtain block-level scores $\mathbf{P}_r \in \mathbb{R}^{N\times N}$. This design preserves the fine-grained heterogeneity that the coarse branch overlooks, yet keeps routing cost negligible.

Fine Branch. The fine branch only includes a block-sparse attention computed as:

$$\mathbf{O}_f = \text{BlockSparseAttn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}; \mathbf{M}) \in \mathbb{R}^{N\times B\times D}, \tag{12}$$

where BlockSparseAttn denotes block-sparse attention restricted to the selected block pairs in $\mathbf{M}$. This generalizes to cross attention for text conditioning as in MMDiT Esser et al. (2024), as Eq.(12) extends to include full video-to-text and text-to-video attention components.

Gated Merge. The coarse and fine branch outputs are combined through a lightweight gating mechanism:

$$\mathbf{G} = \tanh\big(\mathbf{X}\mathbf{W}_g\big), \tag{13}$$

$$\mathbf{O} = \mathbf{G} \odot \mathbf{O}_c + \mathbf{O}_f. \tag{14}$$

Here, $\mathbf{X} \in \mathbb{R}^{L\times D}$ are the hidden states of the transformer layer, and $\mathbf{W}_g \in \mathbb{R}^{D\times 1}$ is a single-channel projection. This produces a gate per-token per-head $\mathbf{G} \in \mathbb{R}^{L\times 1}$ (Qiu et al., 2025), which is broadcasted to $\mathbb{R}^{L\times D}$ to modulate $\mathbf{O}_c$; $\odot$ denotes element-wise multiplication.

Kernel Implementation. For non-divisible (T, H, W), we pad each dimension to the nearest multiple of (B_t, B_h, B_w) and ensure the attention to padding tokens are masked out in our kernels. Block-sparse attention is implemented in ThunderKittens Spector et al. (2024), following (Zhang et al., 2025g). To avoid materializing $\hat{\mathbf{P}}_r$, we fuse the GEMM, softmax, and $(G{\times}G)$-pooling into a single kernel implemented with CuTe DSL. The fused kernel computes $\mathbf{Q}_r\mathbf{K}_r^\top$ twice (Figure 5): a first pass to accumulate log-sum-exp (LSE) statistics, and a second to apply softmax, perform score pooling, and write back. Although this introduces one additional compute pass, it substantially cuts I/O and yields a clear speedup over the unfused baseline.

2.3 Improved Training Recipe

Instead of training from scratch, we develop a more efficient strategy, which we term Sparse Rebasing for our main experiments in §3.2. This approach initializes the model from a pre-trained 256p video checkpoint that uses full attention and then introduces VSA2 for subsequent training on 480P and 720P resolutions, and reinforcement learning. The transition is seamless: the gating

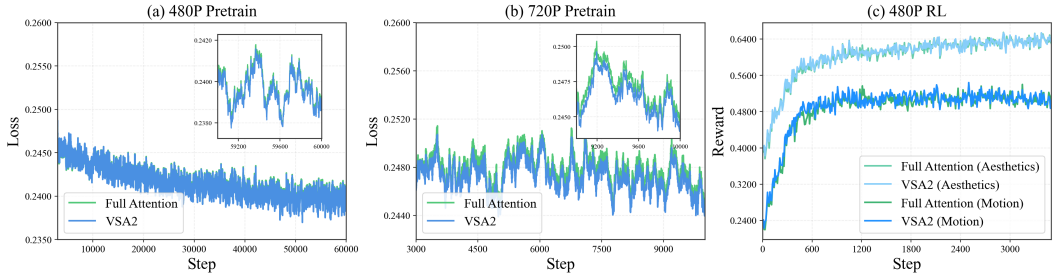

Figure 4: Training loss and reward scores of VSA2 vs. full attention across various stages of video DiT training. With a sparsity ratio of 90% to 95%, VSA2 shows lower loss and similar reward scores compared to full attention.

weights $\mathbf{W_g}$ in Eq.(13) which are the only newly added parameters, are initialized to zero, ensuring the model's output contains only $\mathbf{O_f}$ initially. This strategy offers two significant advantages. First, it allows us to leverage existing text-to-image models that are trained with full attention, which is a common practice for state-of-the-art video model training pipelines Wang et al. (2025); Zheng et al. (2024). Second, it is flexible, as the benefits of sparse attention are minimal at the early, low-resolution stage of video training, where sequence lengths are still short. As demonstrated in §3.2, a model trained through SPARSE REBASING even achieves better quality than its full-attention counterpart. Furthermore, we find that reducing the sparsity level during inference—a schedule we call HARD-TO-EASY CURRICULUM (train-hard/test-easy)—improves motion quality, even surpassing the performance of the original full-attention model.

3 EXPERIMENTS

3.1 SETUP

Model Training. We largely follow the architecture of MMDiTs Esser et al. (2024). Training uses flow matching Lipman et al.; Liu et al. (2022) with a velocity prediction objective and joint text-to-video and image-to-video supervision on a video dataset. Timestep sampling follows a logit-normal distribution with resolution-aware shifting Esser et al. (2024). We employ FSDP Zhao et al. (2023), sequence parallelism Jacobs et al. (2023), activation recomputation Chen et al. (2016), and `torch.compile` Ansel et al. (2024) for scalable multi-node training. Training proceeds in three stages: 480p and 720p pretraining, followed by an RL stage Xu et al. (2023) initialized from 480p checkpoints. Additional details on the RL stage are provided in the supplementary. Across all stages, training clips span 5–12 seconds with diverse aspect ratios.

Baselines. In main results, we compare VSA2 against full attention baselines trained under exactly the same hyperparameter, data, and iterations. Architecturally, VSA2 can be understood as VSA Zhang et al. (2025d) with fine-grained router and per-sequence topK, and we ablate those changes in §3.3. We also compare VSA2 with other choices of efficient attention designs, including key architectures of NSA Yuan et al. (2025). To ensure a fair comparison, all models are configured to have a similar parameter.

Evaluation Metric. We primarily use flow matching loss as the evaluation metric, which correlates well with human preference Polyak et al. (2025). We additionally conduct human evaluation on 149 curated prompts comparing VSA2 and full attention, with each pair rated as *good*, *same*, or *bad*. The final score is computed as $(\#\text{good} - \#\text{bad})/149$ and reported in Table 2. The prompts are intentionally challenging to better distinguish model performance. For evaluation, we use EMA checkpoints to generate 10-second videos at resolutions of 480x864 (99k tokens) and 720x1280 (220k tokens).

3.2 MAIN RESULTS

As demonstrated in Table 2 Exp 1, 5, 7 and Figure 4, VSA2 performs comparably to full attention in all training stages, including 480p, 720p, and RL. This strong performance is consistent across

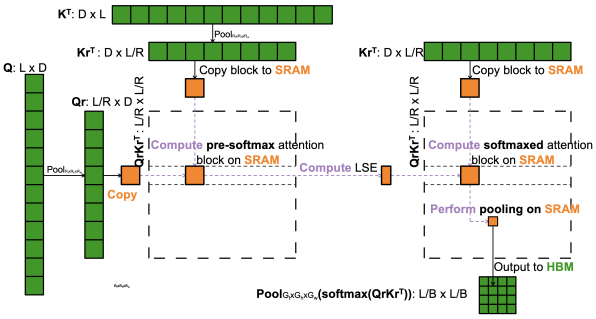

Figure 5: Kernel implementation of VSA2's router, which consists of two GEMM passes. The first pass computes the LSE for softmax and the second pass calculates the softmax score with on-chip pooling.

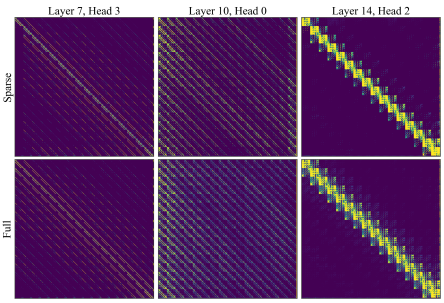

Figure 6: Attention visualization of VSA2 vs. full attention. Their attention patterns are highly similar even after pretraining.

human evaluations, flow-matching loss, and RL rewards. A key finding is that as the resolution increases from 480p to 720p, VSA2 does not require higher topK value to achieve a lower loss than the full-attention baseline. At 720p resolution, this efficiency results in a remarkable 95% sparsity and a 4.62× increase in end-to-end inference speed.

Comparing Exp 1–4 in Table 2, we observe a consistent trend: employing a higher topK value during inference than during training—a strategy we term HARD-TO-EASY CURRICULUM—significantly improves motion quality compared to the full-attention baseline. In particular, in Exp 2, human raters judged 22.1% of VSA2 text-to-video samples to exhibit better motion than those from the full attention model. A model trained with top32 and tested with top64 (Exp 4) achieves better motion scores than one both trained and tested with top64 (Exp 1). This improvement, however, is accompanied by a trade-off in prompt following and aesthetic quality.

We hypothesize that the improved motion quality stems from a regularization effect similar to structured dropout Ghiasi et al. (2018), while reduced prompt adherence arises from a training–inference mismatch. In MMDiT, video and text tokens share the same self-attention space; increasing the number of attended video tokens at inference effectively reduces attention paid to text tokens. A lightweight fine-tuning stage with a higher topK can mitigate this issue. Indeed, in Exp. 6 of Table 2, RL training with top128 initialized from a top64 checkpoint restores prompt following and aesthetic quality while preserving superior motion quality over the full-attention baseline. Figure 3 shows human-evaluation samples from Exp. 6, with additional results in the supplementary material. RL at a higher TopK allows the model to re-adapt attention allocation under denser contexts, effectively bridging the gap between sparse pretraining and less sparse inference. We further show in the supplementary that VSA2, trained on 5–12s videos, can directly generate 30s outputs without noticeable degradation, providing early evidence that VSA2 scales toward the minute-long regime.

To further analyze model behavior, we visualize and compare attention maps from VSA2 and a full-attention model in Figure 6. Unlike prior work that profiles full-attention models at inference time Xi et al. (2025); Zhang et al. (2025c;f), we compare two distinct checkpoints from Exp. 1 in Table 2, one trained with VSA2 and the other with full attention. As shown in Figure 6, the attention maps remain remarkably similar even after 60K training steps, providing strong evidence that VSA2 preserves training dynamics comparable to full attention during pretraining.

3.3 ABLATION STUDIES

In this section, we present a series of ablation studies to justify the key design choices of VSA2. We begin by analyzing the token grouping and router mechanisms, which represent the primary architectural departures from NSA Yuan et al. (2025). Subsequently, we ablate the specific enhancements that differentiate VSA2 from VSA Zhang et al. (2025g), providing a comprehensive validation of our attention design.

Group Head vs. Group Neighbour. Sparse attention in language models, such as NSA Yuan et al. (2025), is based on grouped-query attention (GQA). NSA adopts a "group head" approach, where

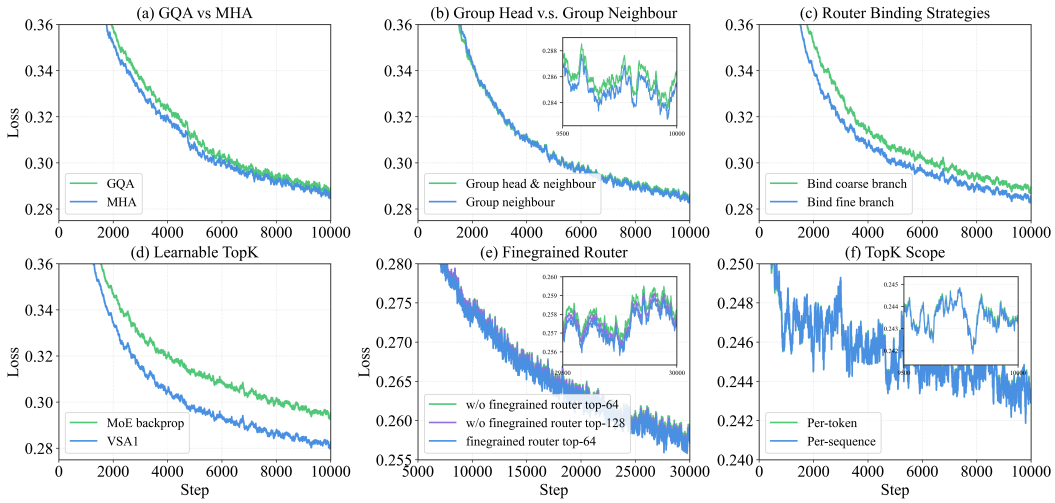

Figure 7: Ablation studies of various design choices in video sparse attention. All models are trained from scratch except those in plot (f), which are initialized from a 256P video checkpoint.

query heads within the same group share the same sparsity pattern. Conversely, VSA2 utilizes Multi-Head Attention (MHA) and adopts a "group neighbour" strategy, where spatially and temporally neighboring query tokens attend to a common set of KV tokens. To validate our architectural choice of MHA+group neighbour over GQA+group head, we conducted two ablation studies. First, we compare GQA vs MHA under a full attention setting. As shown in Figure 7(a), MHA achieves a significantly lower training loss than GQA (with 4 groups), suggesting that it is inherently better suited for video generation. Second, we investigate the grouping strategy itself. Figure 7(b) demonstrates that even within a GQA framework, the "group neighbour" approach outperforms a hybrid "group head & neighbour" strategy. For this experiment, the "group neighbour" method used larger spatiotemporal cubes, while the hybrid method combined smaller cubes with head grouping. Both configurations maintained the same effective group size. The better performance of the pure "group neighbour" configuration confirms that it is the more effective strategy for video models.

Learnable Router vs. Gradient-free Router. As discussed in §2.1, VSA/NSA Yuan et al. (2025) use a learnable router that shares parameter with the coarse branch and receives gradients from $\mathbf{O_c}$. We conducted ablation studies, presented in Figures 7(c) and (d), to determine if such a learnable router is really necessary for video models. First, we decouple the QKV projection layers for the coarse and fine attention branches. We compared two configurations: (1) share the router QKV with the coarse branch's QKV, where the router parameters receive gradients from $\mathbf{O_c}$, and (2) share the router QKV to the fine branch's QKV, which is analogous to a gradient-free router design in MoBA Lu et al. (2025a). The results in Figure 7(c) demonstrate that binding the router to the fine branch achieves a lower training loss, indicating that the router does not need to receive gradients from $\mathbf{O_c}$. Next, we investigate an alternative learnable router design inspired by mixture-of-experts in FFN. In this experiment, the coarse branch, the router and the fine branch were assigned an independent set of QKV parameters. The output of the fine branch was re-weighted by the router's output logits, allowing the router logits to receive direct gradients based on the importance it assigns to each block. As shown in Figure 7(d), this MoE-style backpropagation approach did not outperform VSA. We thus conclude that a complex, learnable router is not necessary for video diffusion models. A simpler, effectively gradient-free router is sufficient and yields better performance.

Gradient-Free Router Design. To optimize the gradient-free router design, we introduce *fine-grained router*. The efficacy of this design is demonstrated in Figure 7(e), which shows that models incorporating the fine-grained router achieve a significantly lower training loss compared to the non-fine-grained router baseline. Notably, the fine-grained router with a top64 outperforms the top128 baseline router. Furthermore, we investigated the optimal scope for the topK selection strategy. As illustrated in Figure 7(f), applying topK selection on a *per-sequence* basis yields a lower loss than the conventional *per-token* approach. We also tried some top-p-based methods during preliminary studies, but did not yield positive results. These ablation studies, which favor the fine-grained router and per-sequence topK, collectively inform the final architectural design of VSA2.

3.4 Attention Speed

Table 2 presents the model-level speedup of VSA2, and Figure 8 compares its attention-level performance against FlashAttention3 Shah et al. (2024). For this comparison, we report the total time of all operations in Figure 2, excluding the QKV projections. We also provide a detailed runtime breakdown for the coarse branch, fine branch, and router components. VSA2 achieves an $8.9\times$ acceleration at a sequence length of 220K (corresponding to a 720P–10s video). The fine-grained router accounts for 22% of the total attention runtime at 220K and 30% at 436K. Considering that the fine-grained router reduces the fine branch computation by half (Figure 7 (e)), its additional cost is well justified.

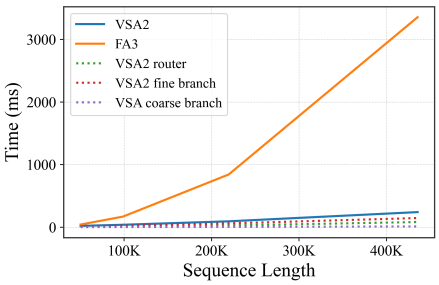

Figure 8: Speed comparison on a single H800 GPU with batch size 1, number of heads 20, head dimension 128 and top64.

4 Related Work

Video DiTs. Modern video DiTs follow a multi-stage training recipe. First, large-scale pretraining adopts a progressive curriculum: training starts on images, then advances to short low-resolution clips, and finally scales to long high-resolution videos Lin et al. (2024); Zheng et al. (2024); Kong et al. (2024); Wang et al. (2025); Gao et al. (2025). Second, *supervised fine-tuning* (SFT) on curated text–video pairs improves motion, instruction-following, and style control. *Third*, *reinforcement learning* tailors generation to human feedback with diffusion-adapted RL objectives, which report consistent gains in semantic alignment and temporal coherence Wu et al. (2025); Xue et al. (2025); Shen et al. (2025); Liu et al. (2025). Finally, *step distillation* reduces the diffusion steps Lu et al. (2025b); Yin et al. (2024a;b). In contrast to LLMs-where most pretraining FLOPs occur on short contexts, video DiTs expend the bulk of compute on high resolution training, making quadratic attention a bottleneck at both pretraining and inference,

DiT Inference Acceleration. The iterative denoising process in DiTs is computationally intensive. Step distillation methods, such as progressive distillation Salimans & Ho (2022) and consistency distillation Song et al. (2023); Song & Dhariwal (2023); Wang et al. (2024), train a student model to replicate a teacher's output in fewer steps Salimans & Ho (2022). More recent data-free methods align student and teacher models by matching intermediate distributions Yin et al. (2024b;a). In addition to step reduction, caching methods exploit temporal redundancy by reusing intermediate activations from previous denoising steps, thus avoiding redundant computations Ma et al. (2024b;a); Lv et al. (2024); Kahatapitiya et al. (2025). Lastly, post-training quantization has enabled W8A8 or even W4A4 precision for DiTs with minimal quality loss Wang et al. (2025); Li et al. (2024); Mehta et al. (2025), and attention-specific 8-bit/4-bit quantization offers plug-and-play acceleration Zhang et al. (2024b;a; 2025b). Sparse attention, as the focus of this paper, is largely orthogonal to those methods Team (2025).

Sparse Attention. Sparse attention mitigates the quadratic complexity of self-attention by restricting the computation to a smaller, more relevant subset of critical tokens. In LLMs, sparse attention has been extensively studied, with methods ranging from fixed sparsity patterns that target specific token arrangements Yang et al.; Xiao et al. (2023); Jiang et al. (2024) to input-adaptive approaches that dynamically identify salient regions for attention computation (Tang et al., 2024; Gao et al., 2024; Zhang et al., 2023; Lu et al., 2025a; Yuan et al., 2025). Adapting these techniques to video introduces unique spatio-temporal challenges. Many recent training-free methods for video models introduce sparsity by exploiting locality or pre-defined temporal patterns Yuan et al. (2024); Zhang et al. (2025e); Ding et al. (2025); Xu et al. (2025); Li et al. (2025); Xi et al. (2025). Others generate sparse masks dynamically during inference Xi et al. (2025); Zhao et al. (2025); Cai et al. (2025). Although pioneering works like DSV Tan et al. (2025) and VSA Zhang et al. (2025g) have explored integrating sparse attention directly into the pre-training phase, their studies have been confined to smaller-scale experiments, with evaluation relying primarily on the loss metric. To our knowledge, VSA2 is the first sparse attention mechanism validated at various stages of the DiT development cycle, including pre-training, subsequent RL with human preference, and inference.

AI USE STATEMENT

We used generative AI tools for language polishing and improving the readability of the manuscript. AI-based tools were also used during training data preparation for data captioning. The authors take full responsibility for the final content of this work.

REFERENCES

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, et al. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, pp. 929–947, 2024.

Philip J. Ball, Jakob Bauer, Frank Belletti, Bethanie Brownfield, Ariel Ephrat, Shlomi Fruchter, Agrim Gupta, Kristian Holsheimer, Aleksander Holynski, Jiri Hron, Christos Kaplanis, Marjorie Limont, Matt McGill, Yanko Oliveira, Jack Parker-Holder, Frank Perbet, Guy Scully, Jeremy Shar, Stephen Spencer, Omer Tov, Ruben Villegas, Emma Wang, Jessica Yung, Cip Baetu, Jordi Berbel, David Bridson, Jake Bruce, Gavin Buttimore, Sarah Chakera, Bilva Chandra, Paul Collins, Alex Cullum, Bogdan Damoc, Vibha Dasagi, Maxime Gazeau, Charles Gbadamosi, Woohyun Han, Ed Hirst, Ashyana Kachra, Lucie Kerley, Kristian Kjems, Eva Knoepfel, Vika Koriakin, Jessica Lo, Cong Lu, Zeb Mehring, Alex Moufarek, Henna Nandwani, Valeria Oliveira, Fabio Pardo, Jane Park, Andrew Pierson, Ben Poole, Helen Ran, Tim Salimans, Manuel Sanchez, Igor Saprykin, Amy Shen, Sailesh Sidhwani, Duncan Smith, Joe Stanton, Hamish Tomlinson, Dimple Vijaykumar, Luyu Wang, Piers Wingfield, Nat Wong, Keyang Xu, Christopher Yew, Nick Young, Vadim Zubov, Douglas Eck, Dumitru Erhan, Koray Kavukcuoglu, Demis Hassabis, Zoubin Gharamani, Raia Hadsell, Aäron van den Oord, Inbar Mosseri, Adrian Bolton, Satinder Singh, and Tim Rocktäschel. Genie 3: A new frontier for world models. 2025.

Shengqu Cai, Ceyuan Yang, Lvmin Zhang, Yuwei Guo, Junfei Xiao, Ziyan Yang, Yinghao Xu, Zhenheng Yang, Alan Yuille, Leonidas Guibas, et al. Mixture of contexts for long video generation. *arXiv preprint arXiv:2508.21058*, 2025.

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness, 2022. URL https://arxiv.org/abs/2205.14135.

Hangliang Ding, Dacheng Li, Runlong Su, Peiyuan Zhang, Zhijie Deng, Ion Stoica, and Hao Zhang. Efficient-vdit: Efficient video diffusion transformers with attention tile. *arXiv preprint arXiv:2502.06155*, 2025.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Yizhao Gao, Zhichen Zeng, Dayou Du, Shijie Cao, Peiyuan Zhou, Jiaxing Qi, Junjie Lai, Hayden Kwok-Hay So, Ting Cao, Fan Yang, et al. Seerattention: Learning intrinsic sparse attention in your llms. *arXiv preprint arXiv:2410.13276*, 2024.

Yu Gao, Haoyuan Guo, Tuyen Hoang, Weilin Huang, Lu Jiang, Fangyuan Kong, Huixia Li, Jiashi Li, Liang Li, Xiaojie Li, et al. Seedance 1.0: Exploring the boundaries of video generation models. *arXiv preprint arXiv:2506.09113*, 2025.

Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. Dropblock: A regularization method for convolutional networks. *Advances in neural information processing systems*, 31, 2018.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *arXiv preprint arXiv:2506.08009*, 2025.

Sam Ade Jacobs, Masahiro Tanaka, Chengming Zhang, Minjia Zhang, Shuaiwen Leon Song, Samyam Rajbhandari, and Yuxiong He. Deepspeed ulysses: System optimizations for enabling training of extreme long sequence transformer models. *arXiv preprint arXiv:2309.14509*, 2023.

Huiqiang Jiang, Yucheng Li, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir Abdi, Dongsheng Li, Chin-Yew Lin, et al. Minference 1.0: Accelerating pre-filling for long-context llms via dynamic sparse attention. *Advances in Neural Information Processing Systems*, 37:52481–52515, 2024.

Kumara Kahatapitiya, Haozhe Liu, Sen He, Ding Liu, Menglin Jia, Chenyang Zhang, Michael S Ryoo, and Tian Xie. Adaptive caching for faster video generation with diffusion transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15240–15252, 2025.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

Muyang Li, Yujun Lin, Zhekai Zhang, Tianle Cai, Xiuyu Li, Junxian Guo, Enze Xie, Chenlin Meng, Jun-Yan Zhu, and Song Han. Svdquant: Absorbing outliers by low-rank components for 4-bit diffusion models. *arXiv preprint arXiv:2411.05007*, 2024.

Xingyang Li, Muyang Li, Tianle Cai, Haocheng Xi, Shuo Yang, Yujun Lin, Lvmin Zhang, Songlin Yang, Jinbo Hu, Kelly Peng, et al. Radial attention: O (nlog n) sparse attention with energy decay for long video generation. *arXiv preprint arXiv:2506.19852*, 2025.

Bin Lin, Yunyang Ge, Xinhua Cheng, Zongjian Li, Bin Zhu, Shaodong Wang, Xianyi He, Yang Ye, Shenghai Yuan, Liuhan Chen, et al. Open-sora plan: Open-source large video generation model. *arXiv preprint arXiv:2412.00131*, 2024.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*.

Hao Liu, Matei Zaharia, and Pieter Abbeel. Ring attention with blockwise transformers for near-infinite context. *arXiv preprint arXiv:2310.01889*, 2023.

Runtao Liu, Haoyu Wu, Ziqiang Zheng, Chen Wei, Yingqing He, Renjie Pi, and Qifeng Chen. Videodpo: Omni-preference alignment for video diffusion generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 8009–8019, 2025.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Enzhe Lu, Zhejun Jiang, Jingyuan Liu, Yulun Du, Tao Jiang, Chao Hong, Shaowei Liu, Weiran He, Enming Yuan, Yuzhi Wang, et al. Moba: Mixture of block attention for long-context llms. *arXiv preprint arXiv:2502.13189*, 2025a.

Yanzuo Lu, Xin Xia, Manlin Zhang, Huafeng Kuang, Jianbin Zheng, Yuxi Ren, and Xuefeng Xiao. Hyper-bagel: A unified acceleration framework for multimodal understanding and generation. *arXiv preprint arXiv:2509.18824*, 2025b.

Zhengyao Lv, Chenyang Si, Junhao Song, Zhenyu Yang, Yu Qiao, Ziwei Liu, and Kwan-Yee K Wong. Fastercache: Training-free video diffusion model acceleration with high quality. *arXiv preprint arXiv:2410.19355*, 2024.

Xinyin Ma, Gongfan Fang, Michael Bi Mi, and Xinchao Wang. Learning-to-cache: Accelerating diffusion transformer via layer caching. *Advances in Neural Information Processing Systems*, 37: 133282–133304, 2024a.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15762–15772, 2024b.

Gaurav Mehta, Jack Xin, Rifat Islam, Yashwant Zhang, Aadil Baig, Ankit Goel, and Stefano Cavallari. Nvidia tensorrt unlocks fp4 image generation for nvidia blackwell geforce rtx 50 series gpus. https://developer.nvidia.com/blog/nvidia-tensorrt-unlocks-fp4-image-generation-for-nvidia-blackwell-geforce-rtx-50-series-gpus/, May 2025. NVIDIA Developer Blog.

Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, David Yan, et al. Movie gen: A cast of media foundation models, 2025. URL https://arxiv.org/abs/2410.13720.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, et al. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. *arXiv preprint arXiv:2505.06708*, 2025.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. Flashattention-3: Fast and accurate attention with asynchrony and low-precision. *Advances in Neural Information Processing Systems*, 37:68658–68685, 2024.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Xiangwei Shen, Zhimin Li, Zhantao Yang, Shiyi Zhang, Yingfang Zhang, Donghao Li, Chunyu Wang, Qinglin Lu, and Yansong Tang. Directly aligning the full diffusion trajectory with fine-grained human preference. *arXiv preprint arXiv:2509.06942*, 2025.

Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. *arXiv preprint arXiv:2310.14189*, 2023.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. 2023.

Benjamin F Spector, Simran Arora, Aaryan Singhal, Daniel Y Fu, and Christopher Ré. Thunderkittens: Simple, fast, and adorable ai kernels. *arXiv preprint arXiv:2410.20399*, 2024.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Xin Tan, Yuetao Chen, Yimin Jiang, Xing Chen, Kun Yan, Nan Duan, Yibo Zhu, Daxin Jiang, and Hong Xu. Dsv: Exploiting dynamic sparsity to accelerate large-scale video dit training, 2025. URL https://arxiv.org/abs/2502.07590.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. Quest: Query-aware sparsity for efficient long-context llm inference. *arXiv preprint arXiv:2406.10774*, 2024.

FastVideo Team. Fastwan: Generating a 5-second video in 5 seconds via sparse distillation. https://hao-ai-lab.github.io/blogs/fastvideo_post_training/, August 2025. Hao AI Lab @ UCSD Blog.

Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Fu-Yun Wang, Zhaoyang Huang, Alexander Bergman, Dazhong Shen, Peng Gao, Michael Lingelbach, Keqiang Sun, Weikang Bian, Guanglu Song, Yu Liu, et al. Phased consistency models. *Advances in neural information processing systems*, 37:83951–84009, 2024.

Jie Wu, Yu Gao, Zilyu Ye, Ming Li, Liang Li, Hanzhong Guo, Jie Liu, Zeyue Xue, Xiaoxia Hou, Wei Liu, et al. Rewarddance: Reward scaling in visual generation. *arXiv preprint arXiv:2509.08826*, 2025.

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, 2025. URL https://arxiv.org/abs/2502.01776.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:15903–15935, 2023.

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

Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, et al. Dancegrpo: Unleashing grpo on visual generation. *arXiv preprint arXiv:2505.07818*, 2025.

S Yang, Y Sheng, JE Gonzalez, I Stoica, and L Zheng. Post-training sparse attention with double sparsity, 2024b. *URL https://arxiv. org/abs/2408.07092*.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *CoRR*, 2024.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T Freeman. Improved distribution matching distillation for fast image synthesis. In *NeurIPS*, 2024a.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *CVPR*, 2024b.

Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, YX Wei, Lean Wang, Zhiping Xiao, et al. Native sparse attention: Hardware-aligned and natively trainable sparse attention. *arXiv preprint arXiv:2502.11089*, 2025.

Zhihang Yuan, Hanling Zhang, Lu Pu, Xuefei Ning, Linfeng Zhang, Tianchen Zhao, Shengen Yan, Guohao Dai, and Yu Wang. Ditfastattn: Attention compression for diffusion transformer models. *Advances in Neural Information Processing Systems*, 37:1196–1219, 2024.

Chenlu Zhan, Wen Li, Chuyu Shen, Jun Zhang, Suhui Wu, and Hao Zhang. Bidirectional sparse attention for faster video diffusion training. *arXiv preprint arXiv:2509.01085*, 2025.

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. *arXiv preprint arXiv:2411.10958*, 2024a.

Jintao Zhang, Jia Wei, Haofeng Huang, Pengle Zhang, Jun Zhu, and Jianfei Chen. Sageattention: Accurate 8-bit attention for plug-and-play inference acceleration. *arXiv preprint arXiv:2410.02367*, 2024b.

Jintao Zhang, Haoxu Wang, Kai Jiang, Shuo Yang, Kaiwen Zheng, Haocheng Xi, Ziteng Wang, Hongzhou Zhu, Min Zhao, Ion Stoica, Joseph E. Gonzalez, Jun Zhu, and Jianfei Chen. Sla: Beyond sparsity in diffusion transformers via fine-tunable sparse-linear attention. *arXiv preprint arXiv:2509.24006*, 2025a.

Jintao Zhang, Jia Wei, Pengle Zhang, Xiaoming Xu, Haofeng Huang, Haoxu Wang, Kai Jiang, Jun Zhu, and Jianfei Chen. Sageattention3: Microscaling fp4 attention for inference and an exploration of 8-bit training. *arXiv preprint arXiv:2505.11594*, 2025b.

Jintao Zhang, Chendong Xiang, Haofeng Huang, Jia Wei, Haocheng Xi, Jun Zhu, and Jianfei Chen. Spargeattn: Accurate sparse attention accelerating any model inference. In *International Conference on Machine Learning (ICML)*, 2025c.

Peiyuan Zhang, Yongqi Chen, Haofeng Huang, Will Lin, Zhengzhong Liu, Ion Stoica, Eric Xing, and Hao Zhang. Vsa: Faster video diffusion with trainable sparse attention. *arXiv preprint arXiv:2505.13389*, 2025d.

Peiyuan Zhang, Yongqi Chen, Runlong Su, Hangliang Ding, Ion Stoica, Zhenghong Liu, and Hao Zhang. Fast video generation with sliding tile attention, 2025e. URL https://arxiv.org/abs/2502.04507.

Peiyuan Zhang, Yongqi Chen, Runlong Su, Hangliang Ding, Ion Stoica, Zhengzhong Liu, and Hao Zhang. Fast video generation with sliding tile attention. *arXiv preprint arXiv:2502.04507*, 2025f.

Peiyuan Zhang, Haofeng Huang, Yongqi Chen, Will Lin, Zhengzhong Liu, Ion Stoica, Eric P Xing, and Hao Zhang. Faster video diffusion with trainable sparse attention. *arXiv e-prints*, pp. arXiv–2505, 2025g.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36:34661–34710, 2023.

Tianchen Zhao, Ke Hong, Xinhao Yang, Xuefeng Xiao, Huixia Li, Feng Ling, Ruiqi Xie, Siqi Chen, Hongyu Zhu, Yichong Zhang, et al. Paroattention: Pattern-aware reordering for efficient sparse and quantized attention in visual generation models. *arXiv preprint arXiv:2506.16054*, 2025.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.

Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all. *arXiv preprint arXiv:2412.20404*, 2024.



Figure 9: More text-to-video samples of VSA2 vs. full attention.

A More Qualitative Results.

In Figure 9,10,11, we present more qualitative examples of VSA2 v.s. full attention generated from checkpoints in Exp 6, Table 2. We also include video files in our supplementary material zip file.

B Sparsity Computation

In this section, we describe how we compute the sparsity of VSA2 relative to full attention. Let L denote the sequence length, B the block size, and R the router input-pooling size. The sequence contains $N = L/B$ blocks, assuming divisibility for simplicity. We estimate the relative computational cost by summing the dominant FLOPs of the constituent branches.

- **Coarse Branch.** Pooling reduces the sequence length by a factor of B, so the coarse attention requires $1/B^2$ of the FLOPs of full attention. Under our configurations, this contribution and the overhead of gating, `tanh`, and pooling are negligible. We therefore omit them from the final sparsity estimate.



Prompt: The camera slowly pushes forward toward a glass kettle on the table. Inside, the water is violently boiling, with countless bubbles rising from the bottom, rolling upward, and merging into larger bubbles.



Prompt: In a painting-style dreamy scene, a top-down wide shot shows a little girl running on the water surface with a small cat beside her. The girl faces away from the camera, her long brown curls blowing in the wind as she runs in a white dress. Ahead of her, an orange-and-black striped cat with white wings glides gracefully in the air. The water shimmers gold and blue. The camera gently follows as the girl's footsteps create tiny splashes, and the winged cat glides forward with softly glowing wings. A low-angle view reveals golden sunlight spreading toward the right, blending into soft pink and beige tones.



Prompt: A full-body shot shows a wingsuit flyer soaring over a massive natural sinkhole. The flyer is back-facing the camera, head toward the left, wearing a blue flight suit, black goggles, a white helmet, and black shoes, with black straps along the sides. Two cliffs covered with vegetation frame the scene, with an arched opening between them. Sunlight brightens the clear sky above, and faint mist fills the lower left. The camera follows the flyer diving toward the huge natural arch of Tianmen Mountain. He subtly shifts his body, gently banking right to align with the center of the opening. The cliff walls rush past as the arch grows larger. He then flies through the opening, with mountains and clouds receding rapidly behind him.

Figure 10: More image-to-video samples of VSA2 vs. full attention.

- **Router.** The router operates on pooled queries and keys of length L/R. Its dominant cost comes from two GEMM operations: one to compute the softmax normalization statistics and another to recompute the attention scores for softmax and score pooling. Together, these require approximately $1/R^2$ of the FLOPs of full attention, whose dominant cost likewise consists of two GEMMs.
- **Fine Branch.** The per-sequence topK selection retains NK query–key block pairs in total, corresponding to K selected KV blocks per query block on average. Each selected block pair contains B^2 token pairs. The fine branch therefore computes attention over NKB^2 token pairs, compared to L^2 for full attention, yielding a relative computational cost of

$$\frac{NKB^2}{L^2} = \frac{KB}{L}. \tag{15}$$

  Individual query blocks may attend to different numbers of KV blocks, while the total computation remains fixed by the sequence-level budget.



Prompt: On a snowy mountain, a full-body shot follows a young white male skier. Positioned on the left side of the frame, he faces right, wearing a light blue ski suit, matching helmet, black gloves, and a black backpack. He glides downhill with snow spraying behind him. The sky is bright with white clouds, and distant snowy peaks and scattered pine trees fill the background. The camera tracks him as he swings his ski poles, turns, accelerates down the slope, and eventually glides toward the bottom of the course, his shadow stretching longer across the snow.



Prompt: A lively and joyful scene shows a full-body shot of a small dog playing in the water. The dog, with brown-and-white fur and drooping ears, runs forward with its mouth open in a happy expression, splashing water as it moves. A blue-and-orange ball floats on the water. The background is blurred with greenery. The dog vigorously chases the ball that is constantly pushed and tumbling by the current, shaking its tail as it splashes through the water.



Prompt: A man drives an old blue truck down a dusty rural road, sunlight casting a muted glow as clouds of dirt trail behind the vehicle.



Prompt: A man plays the piano in a cozy living room while friends gather around with drinks, enjoying the relaxed, intimate atmosphere.

Figure 11: Trained on 5-12s video, VSA2 can directly generate 30s videos.

We define sparsity as one minus the computational cost relative to full attention. Combining the router and fine-branch costs gives

$$\text{Sparsity} \approx 1 - \left(\frac{1}{R^2} + \frac{KB}{L} \right). \tag{16}$$

For fixed B, R, and K, the relative cost of the fine branch decreases as the sequence length grows, while the relative router cost remains constant.

C Reward Feedback Learning

We describe how we apply RL to pretrained checkpoints in Table 2. We use reward-feedback learning Xu et al. (2023) rather than DPO/PPO/GRPO variants. During training, the model predicts x_0 (the clean video), and multiple reward models evaluate the results. The objective maximizes a composite reward that combines a VLM-based RM with a CLIP-based RM. The gradients directly backpropagate through the reward model to video DiT. We tune the reward weights using the full-attention checkpoint and then reuse the same hyperparameters for VSA2 without any adjustment, ensuring a fair comparison across models.

D Social Impacts

VSA2 reduces the training and inference cost of video diffusion models, making high-quality video generation more widely accessible. This can broaden access to creative tools across education, animation, and independent media production. At the same time, scalable realistic video generation introduces risks, including potential misuse for deepfakes or misleading content. We highlight the need to build strong detection systems and ethical safeguards alongside technical progress to ensure responsible deployment.

E FUTURE WORK AND LIMITATION

We plan to apply VSA2 to even larger sequence length, such as 1080P videos. Besides, VSA2 is orthogonal to recent development of autoregressive video generation methods such as Self-Forcing Huang et al. (2025), which we also plan to explore in the future. While the computation is fully balanced under the sequence parallelism strategy Ulysses Jacobs et al. (2023), the dynamic sparsity pattern of VSA2 may introduce compatibility challenges for Ring-Attention Liu et al. (2023). Specifically, Ulysses performs sequence parallelism by gathering the full sequence on each GPU while sharding attention heads across devices. Since VSA2 maintains equal computation across attention heads, the workload remains naturally balanced. In contrast, Ring-Attention shards queries along the sequence dimension and overlaps attention computation with communication by sequentially gathering KV blocks. Because VSA2 does not enforce uniform KV selection across the sequence dimension, the selected KV blocks may become concentrated within a small subset of sequence partitions, leading to uneven computation workloads across GPUs. Another limitation is the router's fixed relative computational cost, approximately $1/R^2$ of full attention, which could become a bottleneck for extremely long sequences when sparsity is very high.