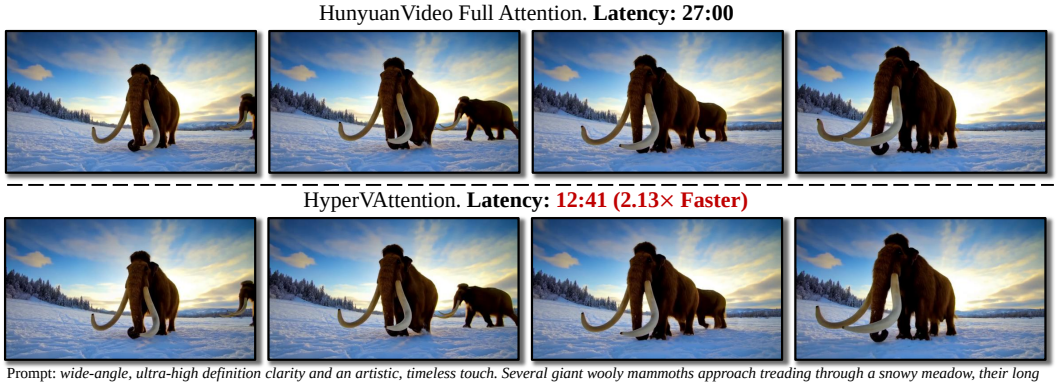
HyperVAttention: Efficient Sparse Attention with Spatio-Temporal Clustering for Video Diffusion

Dongyeun Lee[1] **Amir Zandieh**[2] **Vahab Mirrokni**[2] **Junmo Kim**[1†] **Insu Han**[1†]

[1]KAIST [2]Google Research



Prompt: *wide-angle, ultra-high definition clarity and an artistic, timeless touch. Several giant wooly mammoths approach treading through a snowy meadow, their long wooly fur lightly blows in the wind as they walk, snow covered trees and dramatic snow capped mountains in the distance, mid afternoon light with wispy clouds and a sun high in the distance creates a warm glow, the low camera view is stunning capturing the large furry mammal with beautiful photography, depth of field.*

Figure 1: HyperVAttention is a training-free sparse attention framework for video diffusion that achieves $2.13\times$ speedup on HunyuanVideo while preserving the visual fidelity of full attention.

Abstract

Video Diffusion Transformers (VDiTs) have demonstrated significant capabilities in high-fidelity video generation. However, their ability to produce long-duration videos is fundamentally constrained by the quadratic complexity of the self-attention mechanism. Recent clustering-based sparse attention methods improve the quality-speed trade-off by grouping semantically similar tokens, but their practical efficiency remains limited by two bottlenecks: substantial clustering overhead and low CTA utilization caused by irregular cluster-induced blocks. We propose HyperVAttention (HVA), a training-free sparse attention framework that addresses both bottlenecks jointly. To reduce clustering overhead, we introduce 3D local-window clustering, which exploits the spatio-temporal locality of video tokens to restrict centroid search to fixed local neighborhoods, and implement it with a custom Triton kernel for efficient execution. We further propose a hybrid clustering strategy that performs full clustering only at anchor steps and updates only subset tokens at intermediate steps, leveraging the temporal stability of cluster assignments across denoising steps. To improve CTA utilization, we present hardware-aware cluster merging that minimizes CTA-aligned execution cost through parallel agglomerative merging, improving block density and approximation fidelity by utilizing idle tile capacity. Together, these components reduce clustering overhead, avoid redundant updates, and better align sparse attention with the fixed tile structure of modern GPU kernels. Experiments on video generation show that HVA advances the Pareto frontier of training free sparse attention, achieving higher fidelity than SVG2 at comparable latency and a $2.13\times$ speedup over full attention on HunyuanVideo, or $2.23\times$ with its faster variant HVA-Flash.

†Corresponding Authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

1 Introduction

Diffusion Transformers (DiTs) [33] have recently emerged as a powerful backbone for high-fidelity video generation [12, 32, 26, 51, 18, 41, 70, 3]. However, practical deployment remains severely constrained by spatio-temporal self-attention, which jointly attends over all tokens across frames and spatial locations, resulting in extremely long token sequences and quadratic complexity. To address this bottleneck, various prior works [4, 59, 43, 46, 63, 36, 45] construct block-sparse masks over the original token layout, using blocks aligned with Cooperative Thread Arrays (CTAs), the tile-level execution units of FlashAttention kernels [10]. these approaches treat all tokens in a block as a coarse unit and fail to capture fine-grained token-level relevance, often compromising visual quality. To overcome this limitation, recent studies [13, 50, 71] have introduced clustering-based sparse attention, which cluster tokens by fine-grained relevance and permute them so that similar tokens become contiguous, enabling sparse attention to be computed over clustered token groups, which are then processed as blocks by the attention kernel. These clustering-based methods generally offer a better quality-speed trade-off than sparse attention over the original token layout.

Despite this progress, existing clustering-based approaches still face two key limitations that constrain their practical speedups. First, existing methods incur substantial clustering overhead by failing to fully exploit spatio-temporal locality and temporal stability. Prior methods often rely on generic global procedures such as *k*-means that compare every token against every centroid without exploiting the spatio-temporal locality of video tokens, making clustering itself a major bottleneck even with highly optimized kernels. They also overlook the temporal stability of token assignments across adjacent denoising steps, leading to redundant recomputation even though most assignments remain unchanged over time. Second, cluster-induced blocks often lead to low thread utilization and substantial efficiency loss. As noted above, clustering-based sparse attention uses token clusters as logical attention blocks. In practice, these cluster-induced blocks are executed after padding to fixed CTA tile sizes, as illustrated in Fig. 2(a). Because cluster sizes are often misaligned with the CTA tile shape of attention kernels [10, 11, 52], threads assigned to the padded region outside the true cluster extent remain underutilized, preventing theoretical sparsity from fully translating into actual speedup. Together, these limitations motivate a method that exploits spatio-temporal redundancy and temporal stability while improving the execution efficiency of sparse attention on modern GPUs.

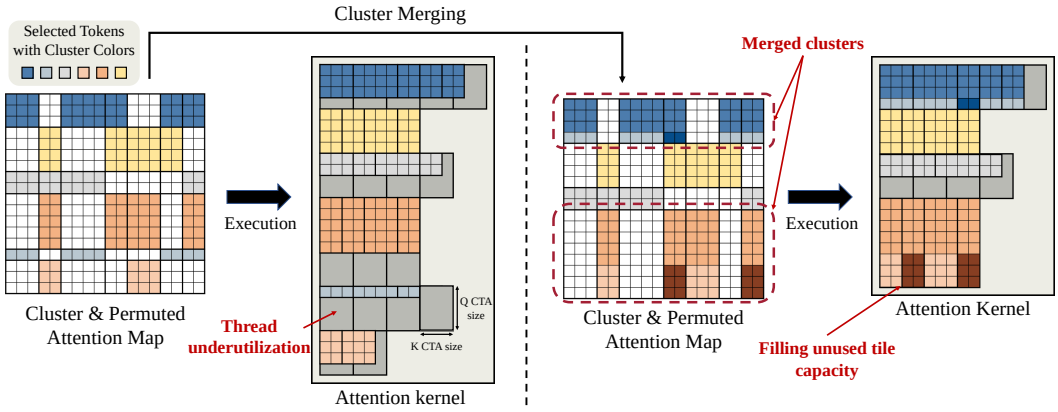
To this end, we present HyperVAttention(HVA), an efficient sparse attention framework for video diffusion that addresses both clustering overhead and low thread utilization. To reduce clustering overhead, we first introduce *3D local-window clustering*, which restricts centroid search to local neighborhoods by exploiting the spatio-temporal locality of video tokens. Compared with generic global clustering (e.g. *k*-means), it converges faster, has lower complexity, and is implemented with a custom Triton kernel for efficient execution. We further introduce a *hybrid clustering* strategy that leverages the temporal stability of token assignments across denoising steps by performing full clustering only at anchor steps and clustering only a subset of tokens at intermediate steps. To address low thread utilization from irregular cluster blocks, we further propose *hardware-aware cluster merging*, which formulates the tiled execution cost of sparse attention as a hardware cost minimization problem and approximately solves it with efficient parallel agglomerative merging. By merging clusters to reduce hardware cost, this step improves CTA utilization and latency, while using unused tile capacity to accommodate a more accurate attention approximation at no extra hardware cost. Together, these components enable HVA to advance the Pareto frontier of training free sparse attention for video generation, achieving higher fidelity than SVG2 at comparable latency and a $2.13\times$ speedup over full attention on HunyuanVideo [18], or $2.23\times$ with its faster variant HVA-Flash.

2 Preliminaries

Let $Q, K, V \in \mathbb{R}^{n\times d}$ denote the query, key, and value matrices of a multi-head self-attention with h heads, where n is the number of tokens and d is the per-head dimension. Yang et al. [50] proposed SVG2, a clustering-based sparse attention approximation for VDiT, which applies *k*-means clustering to query and key embeddings respectively, and consequently each one is partitioned into N_q and N_k clusters. Using the corresponding permutation matrices Π^q and Π^k, the reordered tensors are

$$Q' = \Pi^q Q, \qquad K' = \Pi^k K, \qquad V' = \Pi^k V, \tag{1}$$

which place semantically similar tokens contiguously in memory. This produces cluster-induced blocks for block-sparse attention. Let s_i^q and s_j^k denote the query and key cluster sizes, and let μ_i^q



(a) Clustering-based Sparse Attention (b) w/ Hardware-Aware Cluster Merging

Figure 2: Illustration of how existing clustering-based sparse attention methods lead to thread underutilization. (a) These methods form cluster-induced blocks, whose irregular sizes often misalign with fixed CTA tiles and leave padded regions idle. (b) Merging similar clusters better matches cluster-induced blocks to CTA tiles, reducing tiled execution cost. Darker regions show additional tokens placed in unused tile capacity, improving approximation fidelity without increasing CTA tiles.

and μ_j^k denote their corresponding centroids. A common approximation is to estimate cluster-pair importance using centroid-level scores

$$S_{ij} = \frac{\langle \mu_i^q, \mu_j^k \rangle}{\sqrt{d}}, \qquad P'_{ij} = \frac{s_j^k \exp(S_{ij})}{\sum_{o=1}^{N_k} s_o^k \exp(S_{io})}. \tag{2}$$

where μ_i^q and μ_j^k are the centroids of the i-th query cluster and j-th key cluster, respectively. The cluster-level probabilities are converted into a sparse mask by a row-wise Top-p rule with threshold $p \in (0, 1]$. For each query cluster i, let $\mathrm{TopP}(P'_{i,:}; p)$ denote the smallest set of key-cluster indices, taken in descending order of $P'_{i,:}$, whose cumulative probability is at least p. We define

$$\mathcal{A}_i = \mathrm{TopP}(P'_{i,:}; p), \qquad m_{ij} = \mathbb{1}[j \in \mathcal{A}_i]\,. \tag{3}$$

This yields a cluster-level sparse mask $M = [m_{ij}] \in \{0, 1\}^{N_q \times N_k}$. The selected key clusters define the block-sparse attention computation:

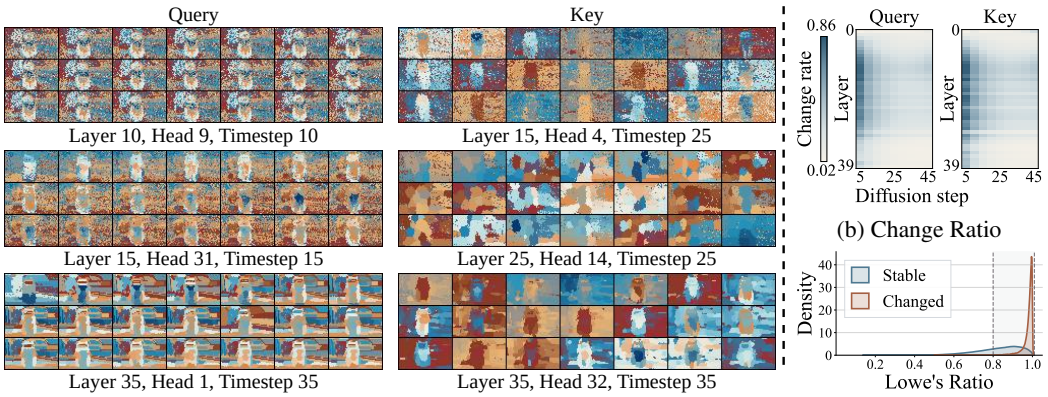
$$O'_i = \mathrm{Attn}\big(Q'_i, \{K'_j, V'_j\}_{j \in \mathcal{A}_i}\big)\,, \qquad O = \Pi^{q-1} O'. \tag{4}$$

Here, Attn denotes standard scaled dot-product attention restricted to the selected key/value clusters. However, the resulting cluster-induced blocks often have sizes misaligned with fixed CTA tile shapes, causing thread underutilization. Our method follows this clustering-based sparse attention pipeline, while reducing clustering overhead and improving hardware efficiency through cluster merging.

3 Motivation

Existing clustering-based sparse attention improves the quality-speed tradeoff by grouping semantically similar tokens, but its realized speedup remains limited by clustering overhead and low utilization of fixed GPU tiles. We analyze three observations behind these limits. Video tokens are locally coherent in space and time, cluster assignments are stable across denoising steps, and token sparsity does not always translate into efficient CTA execution.

Spatio-Temporal Locality of Video Tokens. Videos exhibit substantial redundancy in space and time: nearby regions are often similar, and consecutive frames change gradually. To verify whether this property also holds in diffusion latent space, we cluster tokens with *k*-means and visualize the assignments by projecting 128-dimensional head features into RGB space using Principal Component Analysis (PCA). As shown in Fig. 3a, spatially and temporally adjacent latent tokens tend to fall into the same or similar clusters. This observation suggests that meaningful assignments can often be identified from local neighborhoods alone, which motivates replacing global clustering with a local alternative that exploits the intrinsic 3D structure of video tokens and reduces clustering overhead.



(a) Clustering Results of Video Latent Tokens. (b) Change Ratio (c) Lowe's Ratio

Figure 3: (a) Clustering results of video latent tokens. Frames are ordered from the top left to the bottom right. (b) Ratio of cluster assignment changes over a five-step interval (t vs. $t-5$). (c) Distributions of Lowe's ratio (see Eq. (6)) for reassigned and unchanged tokens.

Temporal Stability across Denoising Steps. Recent works [27, 42, 67, 30] show that diffusion latent features are stable across adjacent denoising steps. We find that this stability extends to per-token cluster assignments. As shown in Fig. 3b, cluster assignments remain stable across most layers and timesteps. Beyond denoising step 10, only 25.8% of query assignments and 27.6% of key assignments change on average. Moreover, to examine which tokens remain stable, we use Lowe's ratio [29], defined as the ratio between the nearest- and second-nearest-centroid distances. A larger ratio indicates that the token lies near a cluster boundary and has an unstable assignment. As shown in Fig. 3c, tokens that later change clusters tend to have higher Lowe's ratios, making it a useful signal for subset selection. This motivates a hybrid update rule with infrequent full clustering and partial updates of only a subset of tokens at intermediate steps.

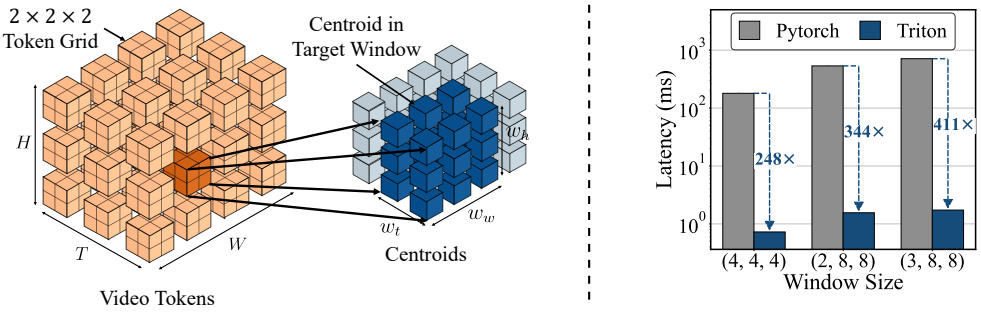
From Theoretical Density to Hardware Efficiency. In clustering-based sparse attention, low theoretical token density is not sufficient to yield end-to-end speedup. Attention is executed on GPUs at the granularity of cooperative thread arrays (CTAs), whereas these methods construct sparse blocks from token clusters. These cluster-induced blocks are generally misaligned with the fixed CTA tile sizes of FlashAttention kernels (e.g., 128 queries and 96 keys on Hopper), resulting in thread underutilization. As detailed in Fig. 5c, a pattern with 39.7% theoretical density still incurs 50.4% CTA density. This motivates a hardware-aware post-processing step that better aligns cluster-induced blocks with CTA tile sizes while preserving attention approximation quality.

4 HyperVAttention

4.1 3D Local-Window Clustering

As discussed in Section 3, video tokens exhibit strong spatio-temporal redundancy, making an expensive global centroid search often unnecessary. Building on this observation, we propose an efficient 3D local-window clustering, which restricts clustering to local neighborhoods instead of performing a global centroid search. Inspired by over-segmentation methods that partition images into a regular grid [72, 1], we extend this principle to a 3D spatio-temporal token lattice, a design visually supported by the spatially coherent cluster assignments in Fig. 3a. As illustrated in Fig. 4a, we partition the spatio-temporal token lattice into regular 3D grids of size (k_t, k_h, k_w), so that each token belongs to exactly one grid. The number of centroids is identical to the number of grids. Each centroid is initialized from the token at the center of a grid, and the resulting centroids preserve the same 3D topology as the underlying grid layout. Let $x_\ell \in \mathbb{R}^d$ denote the embedding of token ℓ, and let $\mu_u \in \mathbb{R}^d$ denote the centroid associated with grid u. For a token ℓ at coordinate (t_ℓ, h_ℓ, w_ℓ), let $g(\ell)$ denote the grid containing token ℓ, and define its local assignment as

$$g(\ell) = \left(\left\lfloor \frac{t_\ell}{k_t} \right\rfloor, \left\lfloor \frac{h_\ell}{k_h} \right\rfloor, \left\lfloor \frac{w_\ell}{k_w} \right\rfloor \right), \qquad a_\ell = \operatorname*{argmin}_{u \in \mathcal{N}(g(\ell))} \|x_\ell - \mu_u\|_2^2, \tag{5}$$



(a) 3D Local-Window Clustering

(b) Kernel Speed Comparison

Figure 4: (a) Illustration of 3D Local-Window Clustering. (b) Latency comparison between PyTorch and Triton implementations of 3D Local-Window Clustering varying window sizes.

where $\mathcal{N}(g(\ell))$ denotes the set of centroids in a local 3D window of size (w_t, w_h, w_w) around the grid $g(\ell)$. This restricts the search to a local spatio-temporal neighborhood and avoids redundant comparisons against distant centroids. The centroids are then updated by averaging the embeddings of their assigned tokens, while the underlying grid partition remains fixed throughout the iterations.

This avoids the usual quality-cost tradeoff in generic clustering, where more centroids improve quality but increase assignment cost. In methods such as k-means clustering, each of the n tokens must be compared against all N_c centroids, resulting in $\mathcal{O}(n\,N_c)$ complexity. In video diffusion, where n is already large, this quickly becomes a bottleneck. By contrast, 3D local-window clustering restricts the search volume of each token to a fixed local window with volume $w_t w_h w_w$, yielding $\mathcal{O}(n\,w_t w_h w_w)$ assignment cost independent of the total number of centroids. For HunyuanVideo at 720p, with $n = 118.8$K, $N_c = 1215$, and $(w_t, w_h, w_w) = (3, 8, 8)$, this yields a $6.3\times$ reduction in the per-token search space. This decouples clustering granularity from assignment overhead, allowing a larger centroid set to improve clustering quality without increasing assignment cost.

Efficient GPU Kernel Implementation. We implement 3D local-window clustering with a custom Triton kernel that performs local assignment and centroid accumulation directly on the native tensor layout. This avoids explicitly constructing overlapping 3D local grids from tokens stored in flattened $T \times H \times W$ layout, which would incur high memory overhead. Our design is inspired by implicit GEMM [40]. Following the same principle, the kernel forms 3D neighborhood tiles on the fly through index arithmetic rather than explicitly materializing overlapping local grids, and uses them for distance evaluation and centroid-statistics accumulation. Compared with a naive PyTorch implementation that explicitly rearranges tokens into overlapping local grids, our Triton kernel achieves up to a $411\times$ speedup (Fig. 4b). Implementation details are provided in Section A.2.

4.2 Hybrid Clustering across Denoising Steps

As discussed in Section 3, most token assignments remain stable across adjacent denoising steps, and Lowe's ratio [29] provides a useful signal for identifying the subset most likely to change. Based on this observation, we adopt a hybrid clustering strategy that performs full-token 3D local-window clustering only at anchor steps and updates only a subset of tokens at intermediate steps, as illustrated in Fig. 3c. Specifically, we use Lowe's ratio,

$$\rho_\ell = \frac{d_{\ell,1}}{d_{\ell,2}}, \qquad d_{\ell,1} = \min_u \|x_\ell - \mu_u\|_2, \qquad d_{\ell,2} = \min_{u \neq u_\ell^\star} \|x_\ell - \mu_u\|_2, \tag{6}$$

where $u_\ell^\star = \arg\min_u \|x_\ell - \mu_u\|_2$ denotes the nearest centroid of token ℓ.

At each first denoising step, we perform full-token 3D local-window clustering, compute the value ρ_ℓ for all tokens, and cache the indices of a subset of tokens with the largest ρ_ℓ values. At subsequent intermediate steps, we update only this subset by performing a standard k-means reassignment against the current centroid set and then updating the corresponding centroids. This substantially reduces computation, since most tokens are not revisited between anchor steps. It also complements 3D local-window clustering with a global refinement step: whereas local-window clustering restricts each

token to nearby centroids, subset refinement allows unstable tokens to be reassigned using the full centroid set. In this way, the hybrid strategy combines low overhead with periodic global correction.

4.3 Hardware-Aware Cluster Merging

As discussed in Section 3, cluster sizes are often misaligned with the fixed CTA tile shapes of attention kernels, leading to thread underutilization. To address this issue, we formulate the hardware-efficiency objective of sparse attention as a cost minimization problem, and then present a parallel agglomerative heuristic for approximately solving it.

Hardware Cost Minimization. Since the softmax operation is applied independently to each row of the attention matrix, each query cluster constitutes a single block row and is executed independently. For each row, efficient modern sparse attention kernel [52] packs the selected key cluster blocks into contiguous memory, enabling efficient computation in units of CTA tiles while avoiding thread underutilization. Accordingly, the hardware cost of a single query cluster i can be written as

$$C(i) = \left(\tau_q \left\lceil \frac{s_i^q}{\tau_q} \right\rceil\right) \left(\tau_k \left\lceil \frac{\sum_{j=1}^{N_k} m_{ij} s_j^k}{\tau_k} \right\rceil\right), \tag{7}$$

where N_q and N_k denote the numbers of query and key clusters, τ_q and τ_k are the CTA tile sizes, $M = [m_{ij}] \in \{0,1\}^{N_q \times N_k}$ is the cluster-level sparse mask, and s_i^q and s_j^k are the sizes of query cluster i and key cluster j, respectively. In Eq. (7), the two terms are the CTA-aligned sizes of the query block and packed key blocks. Thus, $C(i)$ measures tiled work rather than token sparsity.

Parallel Agglomerative Merging. A straightforward way to reduce this cost is to fill underutilized CTA tiles by merging query clusters with compatible key selections. We extend the row-wise cost in Eq. (7) from a single query cluster to a merged group. When multiple query clusters are merged, their query sizes add up, while the selected key clusters are given by the union of their row masks. For a group $S \subseteq \mathcal{I}_q$ with $\mathcal{I}_q = \{1, \ldots, N_q\}$, we define $s^q(S) = \sum_{i \in S} s_i^q$ and $\chi_j(S) = \max_{i \in S} m_{ij}$. Then, the corresponding execution cost becomes

$$C(S) = \left(\tau_q \left\lceil \frac{s^q(S)}{\tau_q} \right\rceil\right) \left(\tau_k \left\lceil \frac{\sum_{j=1}^{N_k} \chi_j(S) s_j^k}{\tau_k} \right\rceil\right). \tag{8}$$

This induces the following partition problem over query clusters:

$$\min_{\mathcal{P}\text{: Partition of } \mathcal{I}_q} \sum_{S \in \mathcal{P}} C(S). \tag{9}$$

Since Eq. (9) is a combinatorial set partitioning problem with a non-additive block cost, approximation strategies are possible including exact combinatorial solvers and sequential greedy agglomerative merging. In our setting, however, cluster merging is useful only if its overhead remains negligible relative to the hardware cost it saves. This makes both exact optimization and sequential greedy updates unattractive in practice. Thus, our goal is to avoid sequential single-merge updates and instead use a parallel merge scheme. To this end, we propose a parallel agglomerative algorithm: in each round, we randomly bipartition the active groups, evaluate only cross-partition candidates, and merge multiple disjoint cost-reducing pairs in parallel. Disjointness ensures that no group participates in more than one accepted merge and can therefore be applied in parallel without conflict. More formally, for two disjoint groups A and B, define the merge gain

$$\Delta C(A, B) := C(A \cup B) - C(A) - C(B). \tag{10}$$

If $\Delta C(A, B) < 0$, merging A and B reduces hardware cost. For each group in one subset, we select the group in the other subset with the smallest merge gain. If multiple groups select the same target, we keep only the best match, so the accepted pairs remain disjoint and can be executed in parallel. Repeating this procedure progressively builds larger groups over multiple rounds. Implementation details are provided in Algorithm 1.

Although this procedure does not guarantee the global optimum of Eq. (9), it is highly parallelizable and directly targets the execution overhead caused by tile misalignment. Since merging takes the union of the selected clusters, it can introduce additional attention computations within idle tile capacity, improving approximation fidelity while reducing hardware cost.

Algorithm 1 Parallel Agglomerative Merging for Hardware Cost Minimization

Inputs: query cluster indices $\mathcal{I}_q = \{1, \dots, N_q\}$, sizes $\{s_i^q\}$, $\{s_j^k\}$, mask $M = [m_{ij}]$, tile sizes τ_q, τ_k, rounds R
Output: a disjoint partition $\mathcal{P}$ of $\mathcal{I}_q$
$\mathcal{P} \leftarrow \big\{\{i\} \mid i \in \mathcal{I}_q\big\}$ ▷ Initialize every query cluster as a singleton group
Initialize $\chi_j(\{i\}) \leftarrow m_{ij}$ for all $i \in \mathcal{I}_q$ and $j = 1, \dots, N_k$
for $r = 1$ to R **do**
  Randomly bipartition the active groups in $\mathcal{P}$ into $\mathcal{A}$ and $\mathcal{B}$
  for each $A \in \mathcal{A}$ **do**
    for each $B \in \mathcal{B}$ **do**
      $s^q(A \cup B) \leftarrow s^q(A) + s^q(B)$ ▷ Sum query sizes
      $\chi_j(A \cup B) \leftarrow \max(\chi_j(A), \chi_j(B))$ for $j = 1, \dots, N_k$ ▷ Logical OR
      $C(A \cup B) \leftarrow \left(\tau_q \left\lceil \frac{s^q(A \cup B)}{\tau_q} \right\rceil\right) \left(\tau_k \left\lceil \frac{\sum_{j=1}^{N_k} \chi_j(A \cup B) s_j^k}{\tau_k} \right\rceil\right)$
      $\Delta C(A, B) \leftarrow C(A \cup B) - \big(C(A) + C(B)\big)$
    $\pi(A) \leftarrow \arg\min_{B \in \mathcal{B}} \Delta C(A, B)$
  $\mathcal{M} \leftarrow \{(A, \pi(A)) \mid \Delta C(A, \pi(A)) < 0 \text{ and } A = \arg\min_{A' : \pi(A') = \pi(A)} \Delta C(A', \pi(A))\}$
  $\mathcal{R} \leftarrow \bigcup_{(A,B) \in \mathcal{M}} \{A, B\}$
  $\mathcal{P} \leftarrow (\mathcal{P} \setminus \mathcal{R}) \cup \{A \cup B \mid (A, B) \in \mathcal{M}\}$ ▷ Execute merges in parallel
return $\mathcal{P}$

5 Experiments

Experimental Settings. We evaluate HyperVAttention (HVA) using Wan2.2-I2V/T2V-A-14B [41] and HunyuanVideo-T2V-13B [18] for video generation on a single NVIDIA H200 GPU. We generate 81 and 129 frames at 720p resolution for Wan2.2 and HunyuanVideo, corresponding to token lengths of $n = 75{,}600$ and $118{,}800$, respectively. For Text-to-Video, we utilize the Penguin Benchmark[3], which provides optimized prompts derived from VBench [16] dataset. For image-to-video, we employ center-cropped images from VBench at 720p. We randomly sample 50 instances for each task as in [50]. We compare against state-of-the-art training-free sparse attention baselines for video diffusion, including SpargeAttn [59] and SVG [45], which operate on the original token layout, and SVG2 [50], a clustering-based baseline. We tune HVA for higher fidelity at latency comparable to SVG2 and reduce the minimum key cluster retention ratio (`min_kc`) in its faster variant HVA-Flash to maximize speed while retaining higher fidelity than SVG2. Detailed hyperparameters are provided in Table 2.

Metrics. To measure fidelity in generated videos against the full-attention baseline, we employ LPIPS [64], Structural Similarity Index Measure (SSIM) and PSNR. We also use scores in VBench [16] to evaluate overall video generation quality. We additionally report an LLM-as-a-Judge score, inspired by recent LLM-based evaluation protocols [69, 5], to capture complementary aspects such as subject consistency, aesthetics, and temporal flickering. We choose `gemini-3-flash preview` [39] and more details including evaluation prompt are provided in Section C.

5.1 Results on Text/Image-to-Video Generation

Table 1 reports the benchmarking results evaluating visual quality against inference efficiency across Wan 2.2 and HunyuanVideo. Our main configuration, HVA, achieves the best fidelity to full attention among sparse methods across the evaluated settings. On HunyuanVideo text-to-video generation, it reaches 33.97 PSNR and 0.046 LPIPS, compared with 32.66 PSNR and 0.056 LPIPS for SVG2. On Wan 2.2 image-to-video and text-to-video generation, it also gives the lowest perceptual degradation among sparse methods. SpargeAttn obtains strong VBench scores in some settings, but its fidelity metrics are consistently worse than ours, indicating larger deviation from the full-attention output. In contrast, under the LLM-as-a-Judge evaluation based on a recent vision-language model, HVA achieves higher overall quality scores than the other sparse methods. HVA-Flash achieves

[3] https://github.com/Tencent-Hunyuan/HunyuanVideo/blob/main/assets/PenguinVideoBenchmark.csv

Table 1: Quantitative comparison of quality and efficiency across various methods. **Bold** and underline denote the best and second-best results, respectively.

Methods	Quality					Efficiency	
	PSNR ↑	SSIM ↑	LPIPS ↓	VBench ↑	LLM-as-a-Judge ↑	Density ↓	Speedup ↑
Wan 2.2 (*14B, Image-to-Video, 81 Frames, 720p, n =75,600*)							
Full Attention	-	-	-	0.816	8.24	100%	1×
SpargeAttn	24.95	0.869	0.106	0.815	8.11	30.15%	1.77×
SVG	23.06	0.817	0.143	0.814	7.47	34.29%	1.76×
SVG2	25.55	0.894	0.095	0.814	7.97	29.54%	1.80×
HVA	**26.84**	**0.916**	**0.077**	**0.816**	**8.15**	36.95%	1.80×
HVA-Flash	25.92	0.903	0.090	0.814	7.95	33.47%	**1.86**×
Wan 2.2 (*14B, Text-to-Video, 81 Frames, 720p, n =75,600*)							
Full Attention	-	-	-	0.845	9.20	100%	1×
SpargeAttn	21.69	0.784	0.167	0.845	9.08	30.15%	1.77×
SVG	20.16	0.717	0.227	**0.846**	9.05	34.29%	1.78×
SVG2	24.33	0.866	0.117	0.843	9.15	33.57%	1.71×
HVA	**25.50**	**0.894**	**0.094**	0.845	**9.35**	42.05%	1.72×
HVA-Flash	24.42	0.867	0.116	0.843	9.18	36.72%	**1.81**×
HunyuanVideo (*13B, Text-to-Video, 129 Frames, 720p, n =118,800*)							
Full Attention	-	-	-	0.821	9.05	100%	1×
SpargeAttn	28.12	0.894	0.103	**0.822**	8.96	30.19%	1.94×
SVG	27.20	0.872	0.124	0.819	8.86	29.70%	1.91×
SVG2	32.66	0.947	0.056	0.817	8.89	26.27%	2.12×
HVA	**33.97**	**0.958**	**0.046**	0.819	**9.18**	31.34%	2.13×
HVA-Flash	32.92	0.949	0.055	0.818	8.92	28.35%	**2.23**×

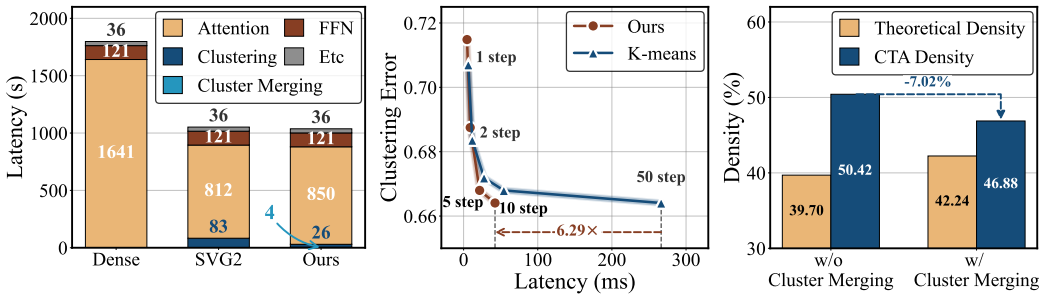
speedups of $1.86\times$ on Wan2.2 image conditioned generation, $1.81\times$ on Wan2.2 text conditioned generation, and $2.23\times$ on HunyuanVideo. Notably, our theoretical density is slightly higher than SVG2, both of ours achieve superior wall-clock speedups. This empirical finding validates our cluster merging strategy, showing that GPU-friendly block utilization can effectively overcome the operational bottlenecks of clustering-based sparse attention.

5.2 Ablation Study

End-to-End Speedup. As illustrated in Fig. 5a, the latency breakdown reveals the source of our efficiency. While SVG2 suffers from a heavy 83 s clustering overhead, our hybrid strategy with 3D local-window clustering requires only 26 s, accelerating the clustering stage by over $3.1\times$. Furthermore, cluster merging improves hardware efficiency by aligning cluster block sizes with CTA tile sizes, with only 4 s of additional overhead. Consequently, this combination of reduced clustering latency and efficient cluster merging enables HVA-Flash to achieve up to a $2.23\times$ speedup over full attention, delivering the fastest inference among the evaluated methods, as shown in Table 1.

Clustering Efficiency. To demonstrate the effectiveness of our 3D local-window clustering, we compare clustering error, which is closely correlated with the final output quality [50], against Flash k-means [50] across varying iterations in Fig. 5b. Leveraging the spatio-temporal locality of video tokens, our 3D local-window clustering strategy proves highly effective and converges rapidly. Our method requires only 10 iterations ($\sim$40 ms) to achieve a clustering error comparable to what k-means reaches after 50 iterations ($>$250 ms). This achieves a remarkable $6.29\times$ latency reduction, effectively resolving the computational bottleneck of iterative token clustering.

3D Local-Window Clustering Kernel Speed Evaluation. As shown in Fig. 4b, when evaluated across varying local window sizes, the custom Triton implementation of 3D local-window clustering consistently outperforms its PyTorch counterpart, achieving up to a $411\times$ speedup. This gap is primarily a systems effect rather than an algorithmic one. The PyTorch baseline realizes the local search through a sequence of fine-grained tensor ops, incurring repeated kernel launches, global-



(a) End-to-End Latency Breakdown (b) Clustering Efficiency (c) Density Comparison

Figure 5: Efficiency analysis on diverse component of HyperVAttention.

memory round trips for intermediate tensors, and poor reuse across neighbor comparisons. In contrast, our Triton kernel keeps the computation on the native $T \times H \times W$ layout and fuses neighbor traversal, distance evaluation, Lowe's ratio computation, and centroid accumulation into a single pass, so each token feature is loaded once and consumed immediately. This substantially improves locality and arithmetic intensity while eliminating most launch overhead and redundant memory traffic.

Hardware-Aware Cluster Merging. Fig. 5c compares the theoretical token density and the actual CTA density of the attention kernel with and without cluster merging. Due to the misalignment between cluster size and thread-block tiling, underutilization is inevitable and leads to computational inefficiency. By integrating cluster merging, our approach accommodates a more accurate approximation of attention (theoretical density: $39.70\% \rightarrow 42.24\%$) within idle tile capacity, while reducing the actual CTA density from 50.42% to 46.88%, corresponding to a 7.02% relative reduction.

6 Related Works

Existing methods are broadly divided into training-based [62, 56, 61, 60, 2, 22, 37, 25] and training-free approaches [4, 59, 43, 46, 63, 36, 45, 50, 71]. Among training-free methods, several works [4, 59, 43, 46, 63, 36, 45] construct block-sparse masks over the original token layout, which is hardware-friendly but often miss fine-grained token relevance. Clustering-based methods [50, 71] instead cluster and permute tokens to form cluster-induced blocks, better preserving semantic structure but remaining bottlenecked by iterative clustering and inefficient CTA utilization. Our method targets these two bottlenecks jointly.

Sparse Attention in LLMs has been extensively explored for long-context inference. Several studies identify critical tokens or local attention sinks and retain them within constrained KV caches [47, 65], estimate block- or page-level importance using proxy metrics to adaptively allocate attention budgets [38, 17, 19, 49], or reorder and hashe tokens to cluster relevant features efficiently [13, 24]. However, these methods rely on 1D causal structure and target prefill or KV-cache efficiency, whereas video diffusion attention remains compute-bound over 3D spatiotemporal dependencies.

Orthogonal acceleration techniques include linear attention [48, 7], high-compression auto-encoders [8], quantization [34, 23, 15, 21, 20, 44, 66, 14, 35, 58, 55, 57], caching [27, 42, 28, 67, 30, 9, 31], and distillation [54, 53, 6, 68]. Our work is complementary to these directions and focuses specifically on the quality-efficiency trade-off of sparse attention in video diffusion transformers.

7 Conclusion

We presented HyperVAttention (HVA), a training-free sparse attention framework that accelerates video diffusion by reducing clustering overhead and improving CTA utilization. It combines 3D local-window clustering, hybrid clustering strategy, and hardware-aware cluster merging to improve both algorithmic and kernel-level execution efficiency. Experiments on Wan2.2 and HunyuanVideo show that HVA advances the Pareto frontier of training free sparse attention, achieving higher fidelity than SVG2 at comparable latency and a $2.13\times$ speedup over full attention on HunyuanVideo, or $2.23\times$ with its faster variant HVA-Flash. As a limitation, benefits are most pronounced for long videos where attention dominates runtime, and gains may be smaller for images or very short videos.

8 Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2025-23523958, RS-2026-25470670) and the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2026-25617720, AI Star Fellowship Support Program).

References

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012. 4

[2] Krish Agarwal, Zhuoming Chen, Cheng Luo, Yongqi Chen, Haizhong Zheng, Xun Huang, Atri Rudra, and Beidi Chen. Monarchrt: Efficient attention for real-time video generation. *arXiv preprint arXiv:2602.12271*, 2026. 9

[3] Vladimir Arkhipkin, Vladimir Korviakov, Nikolai Gerasimenko, Denis Parkhomenko, Viacheslav Vasilev, Alexey Letunovskiy, Nikolai Vaulin, Maria Kovaleva, Ivan Kirillov, Lev Novitskiy, et al. Kandinsky 5.0: A family of foundation models for image and video generation. *arXiv preprint arXiv:2511.14993*, 2025. 2

[4] 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. 2, 9

[5] Dongping Chen, Ruoxi Chen, Shilin Zhang, Yaochen Wang, Yinuo Liu, Huichi Zhou, Qihui Zhang, Yao Wan, Pan Zhou, and Lichao Sun. Mllm-as-a-judge: Assessing multimodal llm-as-a-judge with vision-language benchmark. In *Forty-first International Conference on Machine Learning*, 2024. 7, 19

[6] Junsong Chen, Shuchen Xue, Yuyang Zhao, Jincheng Yu, Sayak Paul, Junyu Chen, Han Cai, Song Han, and Enze Xie. Sana-sprint: One-step diffusion with continuous-time consistency distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16185–16195, 2025. 9

[7] Junsong Chen, Yuyang Zhao, Jincheng Yu, Ruihang Chu, Junyu Chen, Shuai Yang, Xianbang Wang, Yicheng Pan, Daquan Zhou, Huan Ling, et al. Sana-video: Efficient video generation with block linear diffusion transformer. *arXiv preprint arXiv:2509.24695*, 2025. 9

[8] Junyu Chen, Han Cai, Junsong Chen, Enze Xie, Shang Yang, Haotian Tang, Muyang Li, Yao Lu, and Song Han. Deep compression autoencoder for efficient high-resolution diffusion models. *arXiv preprint arXiv:2410.10733*, 2024. 9

[9] Zhiyuan Chen, Keyi Li, Yifan Jia, Le Ye, and Yufei Ma. Accelerating diffusion transformer via increment-calibrated caching with channel-aware singular value decomposition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18011–18020, 2025. 9

[10] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35:16344–16359, 2022. 2

[11] Juechu Dong, Boyuan Feng, Driss Guessous, Yanbo Liang, and Horace He. Flex attention: A programming model for generating optimized attention kernels. *arXiv preprint arXiv:2412.05496*, 2(3):4, 2024. 2

[12] Google DeepMind. Veo 3.1: Text-to-video generation model. https://gemini.google/overview/video-generation/, 2026. 2

[13] Insu Han, Rajesh Jayaram, Amin Karbasi, Vahab Mirrokni, David Woodruff, and Amir Zandieh. Hyperattention: Long-context attention in near-linear time. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Eh0Od2BJIM. 2, 9

[14] Yefei He, Jing Liu, Weijia Wu, Hong Zhou, and Bohan Zhuang. Efficientdm: Efficient quantization-aware fine-tuning of low-bit diffusion models. In *International Conference on Learning Representations*, 2024. 9

[15] Yushi Huang, Ruihao Gong, Jing Liu, Tianlong Chen, and Xianglong Liu. Tfmq-dm: Temporal feature maintenance quantization for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7362–7371, 2024. 9

[16] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 7

[17] Huiqiang Jiang, Yucheng Li, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H. Abdi, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. MInference 1.0: Accelerating pre-filling for long-context LLMs via dynamic sparse attention. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=fPBACAbqSN. 9

[18] 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. 2, 7

[19] Xunhao Lai, Jianqiao Lu, Yao Luo, Yiyuan Ma, and Xun Zhou. Flexprefill: A context-aware sparse attention mechanism for efficient long-sequence inference. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=OfjIlbelrT. 9

[20] Dongyeun Lee, Jiwan Hur, Hyounguk Shon, Jae Young Lee, and Junmo Kim. Dmq: Dissecting outliers of diffusion models for post-training quantization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18510–18520, 2025. 9

[21] 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. 9

[22] 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 for long video generation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=hYovE4nHTt. 9

[23] Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-diffusion: Quantizing diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17535–17545, 2023. 9

[24] Yucheng Li, Huiqiang Jiang, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Amir H Abdi, Dongsheng Li, Jianfeng Gao, Yuqing Yang, and Lili Qiu. MMIference: Accelerating pre-filling for long-context vlms via modality-aware permutation sparse attention. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=me6PfbATWM. 9

[25] Cheng Liang, Haoxian Chen, Liang Hou, Qi Fan, Gangshan Wu, Xin Tao, and Limin Wang. Vmonarch: Efficient video diffusion transformers with structured attention. *arXiv preprint arXiv:2601.22275*, 2026. 9

[26] Lightricks. Ltx-2: Real-time video generation model. https://www.lightricks.com/ltxv, 2025. 2

[27] 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)*, pages 7353–7363, June 2025. 4, 9

[28] Haozhe Liu, Wentian Zhang, Jinheng Xie, Francesco Faccio, Mengmeng Xu, Tao Xiang, Mike Zheng Shou, Juan-Manuel Perez-Rua, and Jürgen Schmidhuber. Faster diffusion via temporal attention decomposition. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=xXs2GKXPnH. 9

[29] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 4, 5

[30] 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. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=W49UjcpGxx. 4, 9

[31] Xuran Ma, Yexin Liu, Yaofu Liu, Xianfeng Wu, Mingzhe Zheng, Zihao Wang, Ser-Nam Lim, and Harry Yang. Model reveals what to cache: Profiling-based feature reuse for video diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17150–17159, 2025. 9

[32] OpenAI. Sora: Creating video from text. https://openai.com/sora, 2024. 2

[33] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023. 2

[34] Yuzhang Shang, Zhihang Yuan, Bin Xie, Bingzhe Wu, and Yan Yan. Post-training quantization on diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1972–1981, 2023. 9

[35] Yang Sui, Yanyu Li, Anil Kag, Yerlan Idelbayev, Junli Cao, Ju Hu, Dhritiman Sagar, Bo Yuan, Sergey Tulyakov, and Jian Ren. Bitsfusion: 1.99 bits weight quantization of diffusion model. *arXiv preprint arXiv:2406.04333*, 2024. 9

[36] Wenhao Sun, Rong-Cheng Tu, Yifu Ding, Jingyi Liao, Zhao Jin, Shunyu Liu, and Dacheng Tao. VORTA: Efficient video diffusion via routing sparse attention. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=gY9yOGYB48. 2, 9

[37] 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. *arXiv preprint arXiv:2502.07590*, 2025. 9

[38] 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. 9

[39] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. 7

[40] Vijay Thakkar, Pradeep Ramani, Cris Cecka, Aniket Shivam, Honghao Lu, Ethan Yan, Jack Kosaian, Mark Hoemmen, Haicheng Wu, Andrew Kerr, Matt Nicely, Duane Merrill, Dustyn Blasig, Aditya Atluri, Fengqi Qiao, Piotr Majcher, Paul Springer, Markus Hohnerbach, Jin Wang, and Manish Gupta. CUTLASS, January 2023. URL https://github.com/NVIDIA/cutlass. 5

[41] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 2, 7

[42] Felix Wimbauer, Bichen Wu, Edgar Schoenfeld, Xiaoliang Dai, Ji Hou, Zijian He, Artsiom Sanakoyeu, Peizhao Zhang, Sam Tsai, Jonas Kohler, et al. Cache me if you can: Accelerating diffusion models through block caching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6211–6220, 2024. 4, 9

[43] Jianzong Wu, Liang Hou, Haotian Yang, Ye Tian, Pengfei Wan, Di ZHANG, and Yunhai Tong. VMoBA: Mixture-of-block attention for video diffusion models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=oQaRElUdmh. 2, 9

[44] Junyi Wu, Haoxuan Wang, Yuzhang Shang, Mubarak Shah, and Yan Yan. Ptq4dit: Post-training quantization for diffusion transformers. In *NeurIPS*, 2024. 9

[45] Haocheng Xi, Shuo Yang, Yilong Zhao, Chenfeng Xu, Muyang Li, Xiuyu Li, Yujun Lin, Han Cai, Jintao Zhang, Dacheng Li, et al. Sparse video-gen: Accelerating video diffusion transformers with spatial-temporal sparsity. In *International Conference on Machine Learning*, pages 68208–68224. PMLR, 2025. 2, 7, 9

[46] 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*, pages 15982–15993, 2025. 2, 9

[47] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv*, 2023. 9

[48] Enze Xie, Junsong Chen, Junyu Chen, Han Cai, Haotian Tang, Yujun Lin, Zhekai Zhang, Muyang Li, Ligeng Zhu, Yao Lu, et al. Sana: Efficient high-resolution image synthesis with linear diffusion transformers. *arXiv preprint arXiv:2410.10629*, 2024. 9

[49] Ruyi Xu, Guangxuan Xiao, Haofeng Huang, Junxian Guo, and Song Han. Xattention: Block sparse attention with antidiagonal scoring. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025. 9

[50] Shuo Yang, Haocheng Xi, Yilong Zhao, Muyang Li, Jintao Zhang, Han Cai, Yujun Lin, Xiuyu Li, Chenfeng Xu, Kelly Peng, et al. Sparse videogen2: Accelerate video generation with sparse attention via semantic-aware permutation. *arXiv preprint arXiv:2505.18875*, 2025. 2, 7, 8, 9

[51] 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. *arXiv preprint arXiv:2408.06072*, 2024. 2

[52] Zihao Ye, Lequn Chen, Ruihang Lai, Wuwei Lin, Yineng Zhang, Stephanie Wang, Tianqi Chen, Baris Kasikci, Vinod Grover, Arvind Krishnamurthy, et al. Flashinfer: Efficient and customizable attention engine for llm inference serving. *Proceedings of Machine Learning and Systems*, 7, 2025. 2, 6

[53] Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in neural information processing systems*, 37:47455–47487, 2024. 9

[54] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6613–6623, 2024. 9

[55] Jintao Zhang, Haofeng Huang, Pengle Zhang, Jia Wei, Jun Zhu, and Jianfei Chen. Sageattention2: Efficient attention with thorough outlier smoothing and per-thread int4 quantization. In *International Conference on Machine Learning (ICML)*, 2025. 9

[56] Jintao Zhang, Haoxu Wang, Kai Jiang, Shuo Yang, Kaiwen Zheng, Haocheng Xi, Ziteng Wang, Hongzhou Zhu, Min Zhao, Ion Stoica, et al. Sla: Beyond sparsity in diffusion transformers via fine-tunable sparse-linear attention. *arXiv preprint arXiv:2509.24006*, 2025. 9

[57] 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*, 2025. 9

[58] Jintao Zhang, Jia Wei, Pengle Zhang, Jun Zhu, and Jianfei Chen. Sageattention: Accurate 8-bit attention for plug-and-play inference acceleration. In *International Conference on Learning Representations (ICLR)*, 2025. 9

[59] Jintao Zhang, Chendong Xiang, Haofeng Huang, Jia Wei, Haocheng Xi, Jun Zhu, and Jianfei Chen. Spargeattention: Accurate and training-free sparse attention accelerating any model inference. In *International Conference on Machine Learning*, pages 76397–76413. PMLR, 2025. 2, 7, 9

[60] Jintao Zhang, Kai Jiang, Chendong Xiang, Weiqi Feng, Yuezhou Hu, Haocheng Xi, Jianfei Chen, and Jun Zhu. Spargeattention2: Trainable sparse attention via hybrid top-k+ top-p masking and distillation fine-tuning. *arXiv preprint arXiv:2602.13515*, 2026. 9

[61] Jintao Zhang, Haoxu Wang, Kai Jiang, Kaiwen Zheng, Youhe Jiang, Ion Stoica, Jianfei Chen, Jun Zhu, and Joseph E Gonzalez. Sla2: Sparse-linear attention with learnable routing and qat. *arXiv preprint arXiv:2602.12675*, 2026. 9

[62] Peiyuan Zhang, Yongqi Chen, Haofeng Huang, Will Lin, Zhengzhong Liu, Ion Stoica, Eric P. Xing, and Hao Zhang. Faster video diffusion with trainable sparse attention. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=VrYCLQ5inI. 9

[63] Peiyuan Zhang, Yongqi Chen, Runlong Su, Hangliang Ding, Ion Stoica, Zhengzhong Liu, and Hao Zhang. Fast video generation with sliding tile attention. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=U74MOXPEJd. 2, 9

[64] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 7

[65] 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. 9

[66] Tianchen Zhao, Tongcheng Fang, Enshu Liu, Wan Rui, Widyadewi Soedarmadji, Shiyao Li, Zinan Lin, Guohao Dai, Shengen Yan, Huazhong Yang, et al. Vidit-q: Efficient and accurate quantization of diffusion transformers for image and video generation. *arXiv preprint arXiv:2406.02540*, 2024. 9

[67] Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-time video generation with pyramid attention broadcast. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=hDBrQ4DApF. 4, 9

[68] Kaiwen Zheng, Yuji Wang, Qianli Ma, Huayu Chen, Jintao Zhang, Yogesh Balaji, Jianfei Chen, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Large scale diffusion distillation via score-regularized continuous-time consistency. *arXiv preprint arXiv:2510.08431*, 2025. 9

[69] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023. 7, 19

[70] 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. 2

[71] Xuanyi Zhou, Qiuyang Mang, Shuo Yang, Haocheng Xi, Jintao Zhang, Huanzhi Mao, Joseph E Gonzalez, Kurt Keutzer, Ion Stoica, and Alvin Cheung. Svg-ear: Parameter-free linear compensation for sparse video generation via error-aware routing. *arXiv preprint arXiv:2603.08982*, 2026. 2, 9

[72] C Lawrence Zitnick and Sing Bing Kang. Stereo for image-based rendering using image over-segmentation. *International Journal of Computer Vision*, 75(1):49–65, 2007. 4

A Implementation Details

A.1 Hyperparameters

For hybrid clustering, we perform full clustering every two denoising steps and cache the top 15% most ambiguous tokens, ranked by Lowe's ratio, for intermediate-step updates. For hardware-aware cluster merging, we use $R = 3$ rounds of parallel agglomerative merging. Table 2 summarizes the hyperparameters used for all methods and evaluation settings. To ensure a fair comparison, we apply the same warmup strategy to every sparse baseline. Below we clarify the meaning of each table entry.

Time warmup / Layer warmup. Time warmup uses full attention during the first few denoising timesteps, and layer warmup uses full attention in the first few transformer layers. Each entry is reported as warmup length / total number of timesteps or layers, with stage-specific values listed separately when applicable. To ensure a fair comparison, we apply identical warmup strategies across all evaluated methods.

Q/K grid size. The 3D temporal-height-width grid used to initialize the query and key centroids.

Q/K window size. The corresponding local search window used by 3D local-window clustering.

N_q, N_k. The numbers of query and key centroids, respectively.

Top-*p*. Following SVG2, the cumulative centroid-selection threshold used to build the cluster-level sparse map. For each query centroid, we compute centroid-to-centroid attention scores, apply a size-weighted softmax over key centroids, and keep the highest-probability key centroids until their cumulative probability reaches p.

Min-*kc*. The minimum retained key-cluster ratio for each query row. We always preserve at least this fraction of key clusters even if the Top-*p* threshold is reached earlier.

Density. A parameter for SpargeAttn and SVG that controls their target block density.

Table 2: Hyperparameters for all experiments including ours and baselines.

Methods	Time warmup	Layer warmup	Q grid size	K grid size	Q window size	K window size	N_q	N_k	Top-*p*	Min-*kc*	Density
Wan 2.2 (*14B, Image-to-Video, 81 Frames, 720p, n =75,600*)											
SpargeAttn	10/50	1/40	–	–	–	–	–	–	–	–	0.3
SVG	10/50	1/40	–	–	–	–	–	–	–	–	0.3
SVG2	10/50	1/40	–	–	–	–	300	1000	0.9	0.1	–
HVA	10/50	1/40	(7, 6, 6)	(6, 4, 4)	(3, 8, 8)	(3, 8, 8)	336	960	0.9	0.18	–
HVA-Flash	10/50	1/40	(7, 6, 6)	(6, 4, 4)	(3, 8, 8)	(3, 8, 8)	336	960	0.9	0.12	–
Wan 2.2 (*14B, Text-to-Video, 81 Frames, 720p, n =75,600*)											
SpargeAttn	10/50	1/40	–	–	–	–	–	–	–	–	0.3
SVG	10/50	1/40	–	–	–	–	–	–	–	–	0.3
SVG2	10/50	1/40	–	–	–	–	300	1000	0.9	0.1	–
HVA	10/50	1/40	(7, 6, 6)	(6, 4, 4)	(3, 8, 8)	(3, 8, 8)	336	960	0.9	0.2	–
HVA-Flash	10/50	1/40	(7, 6, 6)	(6, 4, 4)	(3, 8, 8)	(3, 8, 8)	336	960	0.9	0.1	–
HunyuanVideo (*13B, Text-to-Video, 129 Frames, 720p, n =118,800*)											
SpargeAttn	10/50	1/20, 1/40	–	–	–	–	–	–	–	–	0.3
SVG	10/50	1/20, 1/40	–	–	–	–	–	–	–	–	0.3
SVG2	10/50	1/20, 1/40	–	–	–	–	400	1000	0.9	0.1	–
HVA	10/50	1/20, 1/40	(11, 5, 5)	(11, 3, 3)	(3, 8, 8)	(3, 8, 8)	432	1215	0.9	0.15	–
HVA-Flash	10/50	1/20, 1/40	(11, 5, 5)	(11, 3, 3)	(3, 8, 8)	(3, 8, 8)	432	1215	0.9	0.1	–

A.2 Algorithms

3D Local-Window Clustering Kernel. Algorithm 2 summarizes the core computation of our 3D local-window clustering kernel. Rather than materializing overlapping local windows, the kernel derives local neighborhoods by index arithmetic on the native $T \times H \times W$ layout, performs local nearest-centroid assignment, and accumulates centroid statistics for the subsequent update step.

Algorithm 2 3D Local-Window Clustering in Triton

Inputs: token tensor $X \in \mathbb{R}^{T\times H\times W\times D}$, grid size (k_t, k_h, k_w), local-window size (w_t, w_h, w_w), iterations N_{iter}

Output: assignments $\{a_\ell\}_{\ell=1}^{N}$, centroids $\{\mu_u\}_{u\in\mathcal{G}}$

$K_t \leftarrow \lceil T/k_t \rceil,\ K_h \leftarrow \lceil H/k_h \rceil,\ K_w \leftarrow \lceil W/k_w \rceil$

$w_t \leftarrow \min(w_t, K_t),\ w_h \leftarrow \min(w_h, K_h),\ w_w \leftarrow \min(w_w, K_w)$

$\mathcal{G} \leftarrow [0, K_t-1] \times [0, K_h-1] \times [0, K_w-1]$

For token coordinate $\ell = (t_\ell, h_\ell, w_\ell)$, let

$$g(\ell) \leftarrow \left(\left\lfloor \frac{t_\ell}{k_t} \right\rfloor, \left\lfloor \frac{h_\ell}{k_h} \right\rfloor, \left\lfloor \frac{w_\ell}{k_w} \right\rfloor \right), \quad x_\ell \leftarrow X_{t_\ell, h_\ell, w_\ell, :}$$

for each grid $u = (u_t, u_h, u_w) \in \mathcal{G}$ **do**

  $c_t \leftarrow \min(u_t k_t + \lfloor k_t/2 \rfloor,\ T-1)$

  $c_h \leftarrow \min(u_h k_h + \lfloor k_h/2 \rfloor,\ H-1)$

  $c_w \leftarrow \min(u_w k_w + \lfloor k_w/2 \rfloor,\ W-1)$

  $\mu_u \leftarrow X_{c_t, c_h, c_w, :}$

for $\eta = 1$ to N_{iter} **do**

  $z_u \leftarrow 0 \in \mathbb{R}^D,\ n_u \leftarrow 0 \qquad \forall u \in \mathcal{G}$

  for each grid $u = (u_t, u_h, u_w) \in \mathcal{G}$ **in parallel do**

    $\bar{u}_t \leftarrow \text{clip}(u_t - \lfloor w_t/2 \rfloor,\ 0,\ K_t - w_t)$

    $\bar{u}_h \leftarrow \text{clip}(u_h - \lfloor w_h/2 \rfloor,\ 0,\ K_h - w_h)$

    $\bar{u}_w \leftarrow \text{clip}(u_w - \lfloor w_w/2 \rfloor,\ 0,\ K_w - w_w)$

    $\mathcal{N}(u) \leftarrow \{(\bar{u}_t + \delta_t, \bar{u}_h + \delta_h, \bar{u}_w + \delta_w) \mid \delta_t \in [0, w_t),\ \delta_h \in [0, w_h),\ \delta_w \in [0, w_w)\}$

    for each token coordinate $\ell = (t_\ell, h_\ell, w_\ell)$ such that $g(\ell) = u$ **do**

      $a_\ell \leftarrow \arg\min_{v\in\mathcal{N}(u)} \|x_\ell - \mu_v\|_2^2$

      $z_{a_\ell} \leftarrow z_{a_\ell} + x_\ell, \quad n_{a_\ell} \leftarrow n_{a_\ell} + 1$ ▷ atomic accumulation

  for each grid $u \in \mathcal{G}$ **do**

    $$\mu_u \leftarrow \begin{cases} z_u/n_u, & n_u > 0, \\ \mu_u, & n_u = 0 \end{cases}$$

return $\{a_\ell\}_{\ell=1}^{N}, \{\mu_u\}_{u\in\mathcal{G}}$

B Additional Quantitative Results

B.1 Tradeoff between Quality and Latency

We compare HVA and SVG2 across multiple sparsity settings to determine whether HVA improves generation fidelity at comparable total generation latency. We evaluate both methods using HunyuanVideo on an H200 GPU, generating 15 videos per configuration. We vary the minimum ratio of retained key clusters, denoted `min_kc`, while keeping the Top p threshold at 0.9.

Table 3 shows that HVA achieves higher fidelity across the evaluated latency range. At the same $2.19\times$ speedup, it improves PSNR from 30.76 to 33.83. At approximately $2.1\times$ speedup, it achieves 34.73 PSNR versus 33.11 for SVG2, with corresponding improvements in SSIM and LPIPS.

Table 3: Tradeoff between quality and latency for SVG2 and HVA using HunyuanVideo on an H200 GPU. Speedup is relative to full attention. **Bold** denotes the best result within each sweep.

		Quality			Efficiency	
Method	**min_kc**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**Density ↓**	**Speedup ↑**
Full Attention	–	–	–	–	100%	1.00×
SVG2	0.05	30.76	0.935	0.068	23.82%	**2.19**×
SVG2	0.0625	31.67	0.943	0.057	24.33%	2.18×
SVG2	0.075	32.44	0.950	0.050	24.94%	2.15×
SVG2	0.0875	32.79	0.953	0.047	25.54%	2.13×
SVG2	0.1	**33.11**	**0.955**	**0.044**	26.22%	2.10×
HVA-Flash	0.1	33.71	0.961	0.039	28.31%	**2.21**×
HVA	0.1125	33.83	0.961	0.038	29.00%	2.19×
HVA	0.125	34.22	0.964	0.037	29.81%	2.14×
HVA	0.1375	34.17	0.963	0.037	30.54%	2.14×
HVA	0.15	**34.73**	**0.967**	**0.034**	31.33%	2.11×

B.2 Component Ablation

We evaluate the incremental contributions of hybrid updates and cluster merging guided by CTA cost on top of clustering within 3D local windows. For each configuration, we generate 15 videos using HunyuanVideo on an H200 GPU. We start with local clustering alone, then enable hybrid updates, and finally add cluster merging to obtain the complete HVA configuration.

As shown in Table 4, hybrid updates retain generation fidelity while providing the improvement in inference speed. PSNR increases from 34.44 to 34.59, and speedup increases from $2.05\times$ to $2.06\times$. Adding cluster merging further improves all three fidelity metrics and raises speedup to $2.11\times$. Notably, merging increases theoretical token density from 28.83% to 31.33% while reducing latency. This behavior is consistent with the CTA cost objective because additional attention pairs can occupy otherwise idle CTA capacity, allowing higher fidelity without a proportional increase in tiled work.

B.3 Comparison of Merging Algorithms

We compare algorithms for cluster merging to examine how the merging strategy affects CTA cost, approximation error, and computational overhead. We consider deterministic greedy merging, which applies the same CTA cost objective sequentially, and weighted overlap merging, which estimates merge compatibility from the overlap between masks of selected key clusters. Our method evaluates CTA cost while accepting multiple disjoint merges in parallel. We measure merging and attention latency separately to distinguish optimization overhead from attention execution time.

Table 4: Cumulative component ablation using HunyuanVideo on an H200 GPU.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	Density ↓	Speedup ↑
Local Window Clustering	34.44	0.964	0.036	29.74%	2.05×
+ Hybrid Update	34.59	0.965	0.035	28.83%	2.06×
+ Cluster Merging (HVA)	**34.73**	**0.967**	**0.034**	31.33%	**2.11**×

As shown in Table 5, parallel agglomerative merging attains a CTA cost close to deterministic greedy and matches its L2 error of 0.627. Applying merges in parallel substantially reduces optimization overhead, with merging latency of 257.06 ms compared with 3294.86 ms for the sequential greedy procedure. Weighted overlap also reduces merging overhead relative to deterministic greedy, but increases CTA cost and attention latency relative to the unmerged pattern. This illustrates the limitation of using mask overlap as a proxy, since similar key selections do not necessarily yield efficient CTA alignment after merging. Our method combines direct CTA cost optimization with parallel execution, achieving the lowest attention and combined latency among the merging algorithms.

Table 5: Merging algorithm comparison using merging and attention timings. Bold latency values mark the best results among merging methods.

Method	Merging (ms)	Attention (ms)	Total (ms)	L2 Error	CTA Cost (10^9)	Density (%)
No Merging	–	69.28	69.36	0.674	12.79	19.88
Deterministic Greedy	3294.86	62.60	3357.47	**0.627**	**11.01**	22.27
Weighted Overlap	987.83	72.26	1060.09	0.654	13.74	22.56
Parallel Agglomerative (Ours)	**257.06**	**62.38**	**319.44**	**0.627**	11.23	22.46

B.4 Distilled Video Generation with Few Denoising Steps

We evaluate HVA on a distilled video diffusion model to test its effectiveness with a short denoising trajectory. We use distilled HunyuanVideo-1.5 to generate videos at 720p from input images with 12 denoising steps on H200.

Table 6 shows that HVA achieves the highest fidelity among the evaluated sparse methods, improving PSNR from SVG2's 25.73 to 27.58 and speedup from 1.89× to 2.00×. HVA-Flash further increases speedup to 2.10×, the fastest evaluated configuration, while maintaining better PSNR, SSIM, and LPIPS than SVG2. SVG exceeds the available GPU memory in this setting.

Table 6: Evaluation of distilled HunyuanVideo-1.5 with image conditioning on H200. Speedup is relative to full attention. OOM denotes out of memory.

Method	Quality			Efficiency	
	PSNR ↑	SSIM ↑	LPIPS ↓	Density ↓	Speedup ↑
HunyuanVideo-1.5 (*Image-to-Video, 720p, Distilled, 12 Steps*)					
Full Attention	–	–	–	100%	1.00×
SpargeAttn	24.64	0.848	0.133	30.40%	1.94×
SVG (OOM)	–	–	–	–	–
SVG2	25.73	0.879	0.110	23.38%	1.89×
HVA	**27.58**	**0.914**	**0.083**	28.91%	2.00×
HVA-Flash	26.70	0.897	0.098	25.75%	**2.10**×

C LLMs-as-a-Judge Results

C.1 Evaluation Protocol

Inspired by recent LLM-as-a-Judge evaluation approaches [69, 5], we use an LLM judge to complement automatic metrics with perceptual assessments of generated videos. We use a subset of VBench prompts as the text conditions and ask the judge to evaluate each generated video along three axes: *subject consistency*, *aesthetic quality*, and *temporal flickering*. The judge assigns a score from 1 to 10 for each axis and returns a JSON object, from which we compute the overall LLM-as-a-Judge score as the average of the three scores. The detailed prompt is provided in Section C.3.

Table 7: Quantitative comparison across different dimensions for video generation tasks. **Bold** and underline denote the best and second-best results among sparse methods, respectively.

Methods	Subject Cons. ↑	Aesthetic Qual. ↑	Temp. Flickering ↑	Overall Score ↑
Wan 2.2 (*14B, Image-to-Video, 81 Frames, 720p, n =75,600*)				
Full Attention	7.49	8.68	8.55	8.24
SpargeAttn	**7.56**	8.13	**8.64**	<u>8.11</u>
SVG2	7.40	<u>8.27</u>	8.24	7.97
HVA	<u>7.42</u>	**8.43**	<u>8.60</u>	**8.15**
HVA-Flash	7.35	8.20	8.30	7.95
Wan 2.2 (*14B, Text-to-Video, 81 Frames, 720p, n =75,600*)				
Full Attention	9.29	8.89	9.42	9.20
SpargeAttn	<u>9.21</u>	8.62	9.41	9.08
SVG2	9.19	8.84	9.42	9.15
HVA	**9.48**	**8.95**	**9.62**	**9.35**
HVA-Flash	9.20	<u>8.89</u>	<u>9.45</u>	<u>9.18</u>
HunyuanVideo (*13B, Text-to-Video, 129 Frames, 720p, n =118,800*)				
Full Attention	9.09	8.65	9.41	9.05
SpargeAttn	8.88	<u>8.72</u>	<u>9.28</u>	<u>8.96</u>
SVG2	8.88	8.69	9.10	8.89
HVA	**9.15**	**8.84**	**9.55**	**9.18**
HVA-Flash	<u>8.90</u>	8.71	9.15	8.92

C.2 Detailed Evaluation Result

Table 7 reports the LLM-as-a-Judge results averaged over all evaluated samples. Overall, HyperVAttention achieves the best overall score among sparse methods across all settings, while Ours-Flash maintains competitive perceptual quality under a more aggressive speed-oriented configuration. These results suggest that our method preserves video quality beyond frame-level fidelity metrics.

C.3 Evaluation Prompt

```
1. System Role
- You are an expert Video Quality Assurance Engineer and Cinematographer. Your task is to
↪  evaluate an AI-generated video based on specific technical and artistic dimensions. You
↪  must provide objective, critical scores from 1 (Poor) to 10 (Excellent).

2. Evaluation Process
- Analyze the video frame-by-frame to detect inconsistencies.
- Assess the motion dynamics and temporal stability.
- Score each dimension based on the strict criteria provided.
- Calculate the final average.

3. Strict Evaluation Criteria

Dimension | Description & Scoring Guidelines

1. Subject Consistency | Does the main subject maintain its identity (features, clothing,
↪  shape) throughout the entire duration? Penalty for morphing.
2. Aesthestic Quality| General visual appeal, composition, lighting, and "cinematic" feel.
3. Temporal Flickering | Presence of high-frequency noise or "flashing" brightness/texture
↪  changes between frames.


4. Required Output Format

You MUST ONLY output a single, valid JSON object. Do NOT include any introductory text,
↪  analysis, or conversational response outside of the JSON.

The JSON object MUST contain three keys: overall_score, and metrics.
- overall_score: the simple average of all the metric scores
- metrics: a dictionary containing the required metrics and their scores (Type: Object)

Required JSON Schema:

{
    "overall_score": 0.0,
    "metrics": {
    "subject_consistency": 0,
    "aesthetic_quality": 0,
    "temporal_flickering": 0,
    },
}

Your response to the following video input MSUT be this JSON object and nothing else.
```

D Visual Results

Figs. 6 to 8 show additional qualitative comparisons on Wan2.2 Image-to-Video, Wan2.2 Text-to-Video, and HunyuanVideo Text-to-Video. Across all examples, Full Attention, Ours, and Ours-Flash produce nearly indistinguishable generations, preserving the same scene layout, subject appearance, and motion patterns. These results show that our sparse attention variants maintain the visual fidelity of full attention while enabling more efficient inference.



Figure 6: Visualization of generated videos from Wan2.2 on Image-to-Video generation.



Figure 7: Visualization of generated videos from Wan2.2 on Text-to-Video generation.

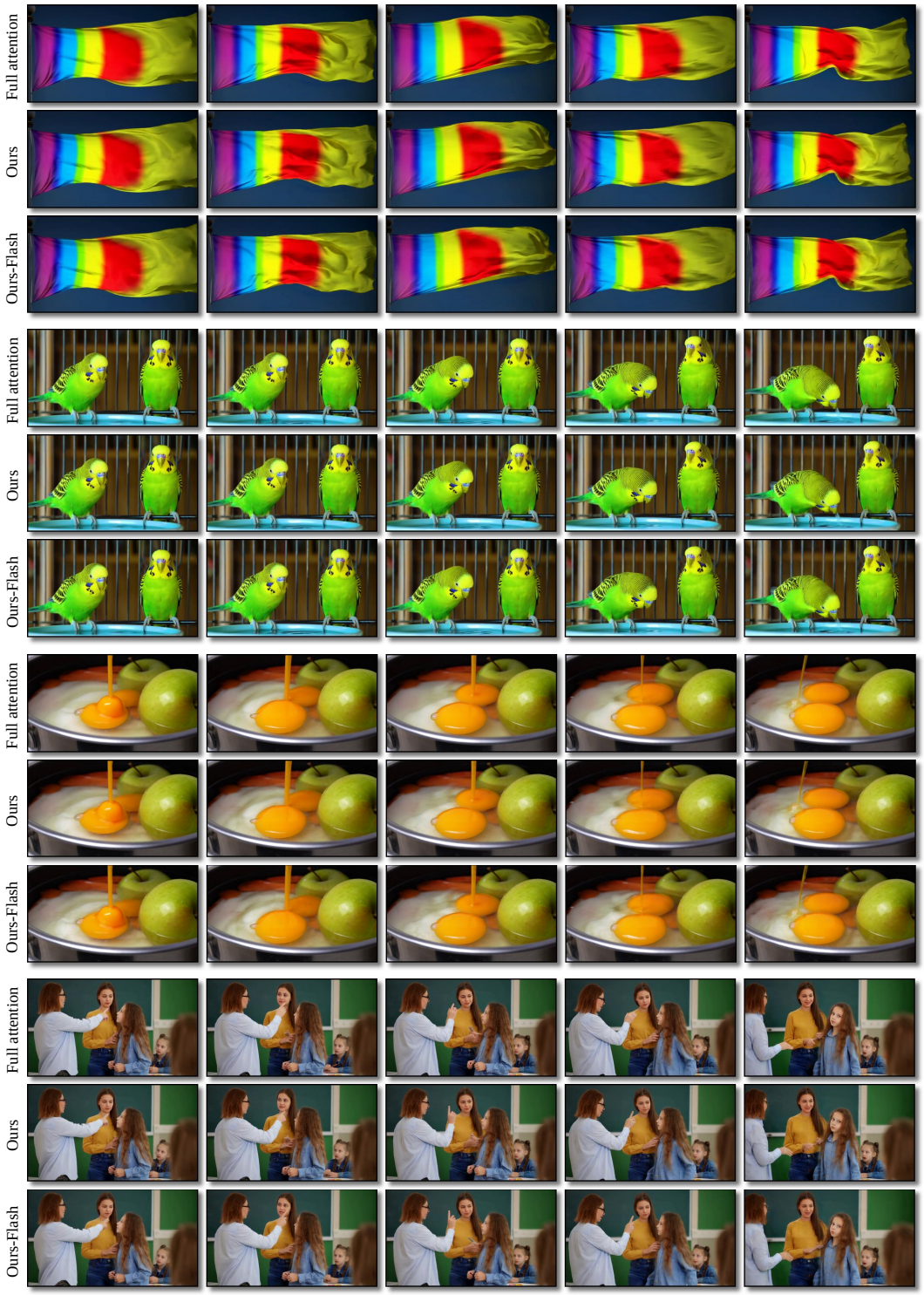


Figure 8: Visualization of generated videos from HunyuanVideo on Text-to-Video generation.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claim is clarified in Section 5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Our limitation is clarified in Section 7.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: This paper does not include theoretical results such as lemmas or theorems.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Implementation details are provided in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Our paper does not include data release. Our code will be released after publication.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Implementation details are provided in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We conducted tests over several random seeds and found the outcomes to be virtually identical. To keep the result clean and focused on the primary trends, we have excluded error stds.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All details are provided in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.

- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our paper conducted in the paper conform with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: As this study centers on improving architectural efficiency, its immediate social consequences are minimal. However, we acknowledge that the underlying technology of video generation has far-reaching effects on society, ranging from content creation to digital ethics.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

Justification: Our work centers on architectural efficiency and does not inherently increase the risk of model misuse. To ensure responsible deployment, we will include the official safety filters and usage guidelines from the base pre-trained models in our repository upon the release of our code

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All details are provided in Section 5.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: Our code will be released after publication.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: Our paper does not include crowdsourcing experiments and research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: Our paper does not include experiments conducted by participants.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: Usage of LLMs is not included in core approach of our method.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.