A. Appendix

A.1. Compactness with Different Request

To verify the effectiveness of the method proposed in Section 3.2.1, we selected the first three prompts from Penguin-VideoBenchmark as inputs and tested the compactness of each layer of the HunyuanVideo model under different prompts during the inference of 720p videos. The results are presented in Figure 9. We can observe that for different inputs, the compactness of the model layers exhibits similar trends under the same configuration, which validates the effectiveness of the method proposed in Section 3.2.1.

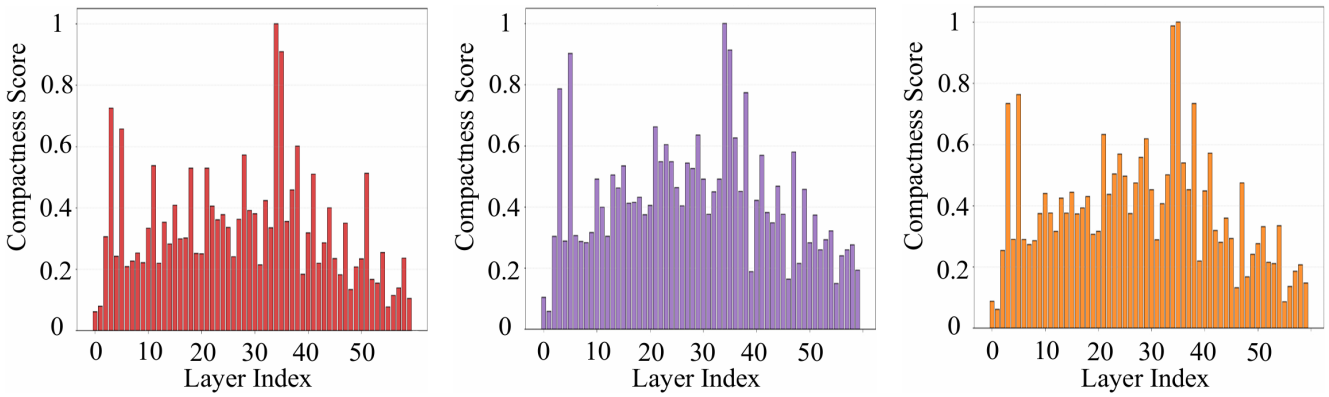

Figure 9. Compactness of each model layer under different inputs. (a) Subfigure 1 (red): Prompt is "In the large cage, two puppies were wagging their tails at each other." (b) Subfigure 2 (purple): Prompt is "A flock of bats flies over the village, captured in medium long shot." (c) Subfigure 3 (orange): Prompt is "Above the sea, a school of silver flying fish leaped out of the water."

A.2. Theoretical and Empirical Justification of Query Normalization

Theoretical justification. Let $q \in \mathbb{R}^d$ be a query vector, $k \in \mathbb{R}^d$ a key vector, and $s = q^\top k$ the attention score. After normalizing q to $\hat{q} = q/\|q\|_2$, the score becomes $\hat{s} = \hat{q}^\top k = s/\|q\|_2$. Since $\|q\|_2 > 0$, for any pair of keys k_a, k_b we have $q^\top k_a > q^\top k_b \iff \hat{q}^\top k_a > \hat{q}^\top k_b$. Thus the relative ordering of scores—and hence the Top-K selection—is invariant under query normalization, which justifies the statement in Sec. 3.1 that "the relative magnitude of query–key scores is independent of the query vector length."

Empirical evidence (Figure 10). We compare clustering unnormalized queries ("Original") versus clustering normalized queries ("Normalized," our method). Both approaches are evaluated in the same normalized space. The left panel of Figure 10 plots the average intra-cluster distance as a function of the number of clusters K, showing that normalized queries yield consistently tighter clusters for all K. The right panel reports the Davies–Bouldin (DB) index, whose definition for K clusters is

$$DB = \frac{1}{K}\sum_{i=1}^{K}\max_{j\neq i}\frac{S_i + S_j}{M_{ij}},$$

where S_i is the average intra-cluster distance of cluster i and M_{ij} denotes the distance between centroids i and j. A smaller DB index indicates more compact clusters with better inter-cluster separation; again, normalization clearly improves the result across all K.

In our deployment we fix the normalized approach to $K = 65$ clusters (the setting used in Sec. 3.1) and ask how many clusters the unnormalized approach needs to match the same intra-cluster distance in the normalized space. On average the unnormalized clustering requires over 235 clusters, corresponding to an effective compression ratio of about $3.6\times$. These observations confirm that normalizing queries before clustering not only preserves Top-K correctness, but also makes the queries substantially easier to cluster—yielding higher compression ratios and tighter clusters as claimed in Sec. 3.1.

To further assess the practical effect of query normalization on reconstruction quality, we conduct an ablation study on the Hunyuan dataset. The following table compares key metrics with and without query normalization (all other components remain identical):

Table 5. Impact of query normalization on reconstruction quality (Hunyuan)

Metric	Without Normalization	With Normalization (Ours)
PSNR ↑	29.56	**30.58**
SSIM ↑	0.763	**0.835**
LPIPS ↓	0.317	**0.203**

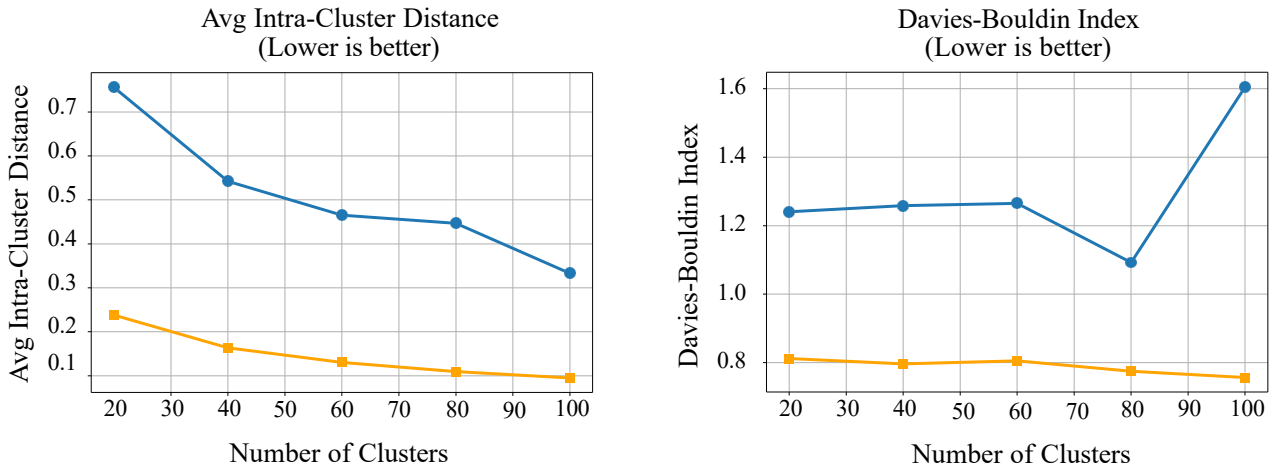

Figure 10. Left: average intra-cluster distance (lower is better). Right: Davies–Bouldin index (lower is better). Both curves compare clustering unnormalized queries ("Original") versus clustering normalized queries ("Normalized") under identical numbers of clusters.

A.3. Number of Clusters for k Under Different Videos Configuration

On the Hunyuan model, we measured the number of K clusters for each layer under different configurations, as shown in Figure 11. It can be observed that the trends of the number of K clusters across model layers are roughly the same under different configurations, though there are still slight differences. The dynamic clustering approach can better balance the accuracy and speed of video generation.

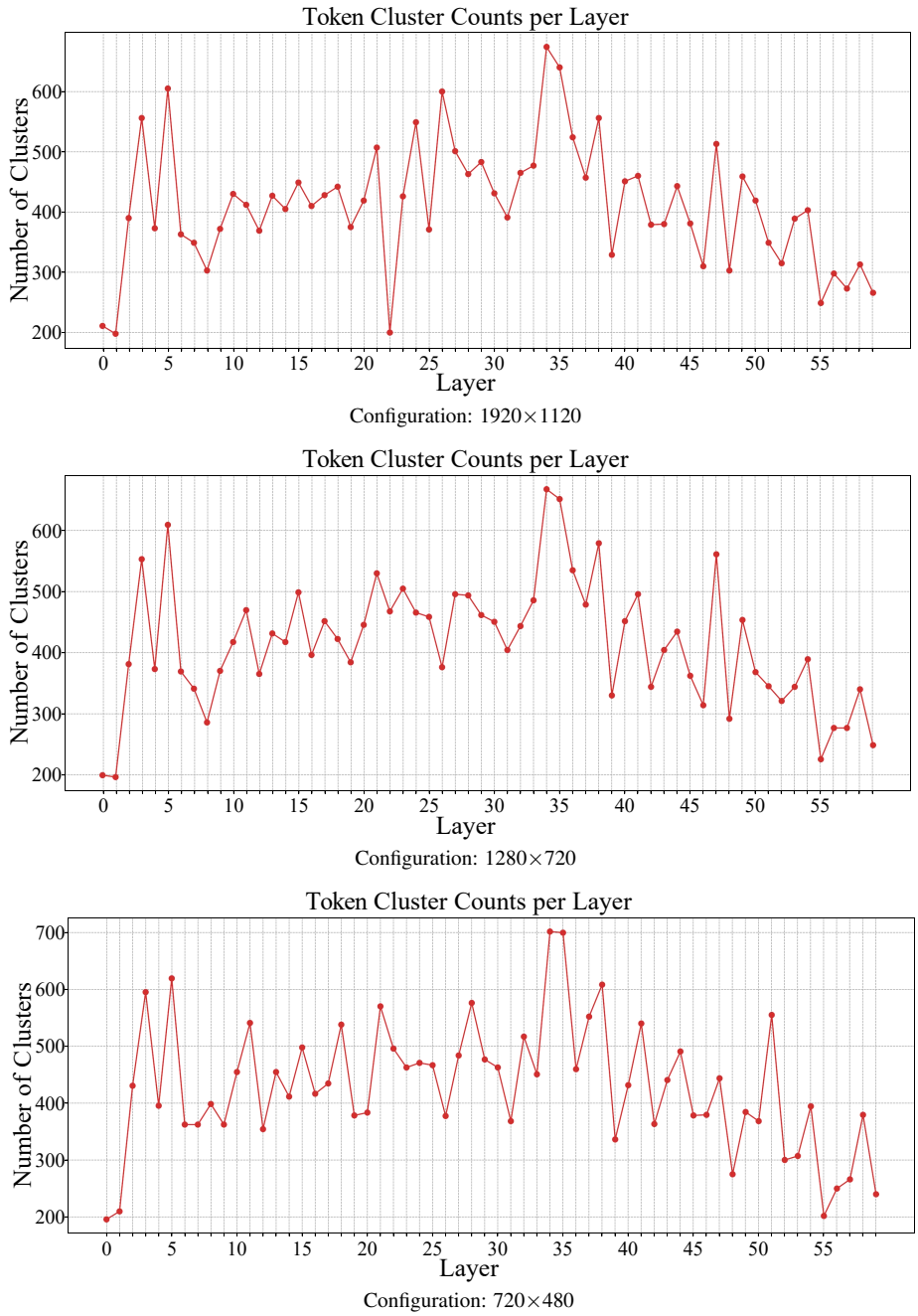

Figure 11. Number of clusters for k under different videos configuration

A.4. Component Ablation

To better understand the contribution of each component in AdaCluster, we summarize the ablation results of three key components in Table 6: (1) adaptive key clustering, (2) TensorQuest-based cluster selection, and (3) query normalization.

Starting from a baseline with mean-based clustering and without TensorQuest, introducing TensorQuest improves token selection quality and reconstruction accuracy. Replacing mean-based clustering with our adaptive clustering strategy further boosts performance, demonstrating the benefit of adaptive key grouping. Finally, applying query normalization stabilizes similarity estimation and provides the largest improvement across all metrics.

Configuration	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**ImageQual ↑**
AvgCluster + w/o Quest + w/o Norm	28.94	0.687	0.410	64.06%
AvgCluster + TensorQuest + w/o Norm	29.01	0.724	0.378	64.79%
AdaCluster + TensorQuest + w/o Norm	29.56	0.763	0.317	65.03%
AdaCluster + TensorQuest + Norm (Ours)	**30.58**	**0.835**	**0.203**	**65.11%**

Table 6. Component analysis of AdaCluster.

A.5. Hyperparameter Sensitivity Analysis

To provide a thorough understanding of AdaCluster's robustness, we conduct a grid search over five key hyperparameters on the Hunyuan reconstruction task. The table below reports the reconstruction quality (PSNR, SSIM, LPIPS) and inference speedup under different configurations. The setting used in the main paper is highlighted in **bold**.

Table 7. Hyperparameter sensitivity analysis on Hunyuan. The configuration used in the main experiments is highlighted in **bold**.

Sparsity	KV Thresh	Q Clus	Initial K	TopK	PSNR↑	SSIM↑	LPIPS↓	Speedup
74.8%	5.5	30	50	64	30.21	0.820	0.220	1.72×
77.5%	3.5	65	100	64	31.88	0.850	0.180	1.69×
76.4%	**5.5**	**65**	**100**	**64**	**30.58**	**0.835**	**0.203**	**1.68×**
60.1%	8.0	65	100	64	30.58	0.825	0.233	1.55×
83.2%	5.5	65	200	32	28.82	0.635	0.395	1.90×

As shown, the selected configuration (76.4% sparsity, KV threshold 5.5, 65 query clusters, initial Top-100, codebook size 64) achieves a strong balance between reconstruction fidelity and inference speedup. Higher sparsity or larger initial TopK tends to yield greater acceleration at the cost of quality degradation, while more aggressive KV thresholding or fewer query clusters can improve quality in some regimes but reduces overall speedup.

A.6. Additional Results on H100 GPUs

A.6.1. Setup

To further validate the scalability of **AdaCluster** on larger models and longer sequence lengths, we conduct additional experiments on NVIDIA H100 GPUs (80GB HBM). We evaluate two representative high-capacity Diffusion Transformer based Text-to-Video models: Wan2.1-T2V-14B and HunyuanVideo.

Models and Configurations. For **Wan2.1-T2V-14B**, we generate 5-second videos at 832×480 resolution using 50 inference steps. For **HunyuanVideo**, we generate videos at 1280×720 resolution using 30 inference steps (consistent with the main experiments).

All other hyperparameters remain identical to those described in Section 4.1: $k_{\max}$ is set such that the top 15% of layers use full attention, $\tau = 1.5\times$ the average token-to-cluster-center distance from the first inference step, the number of query clusters is fixed at 65, and key clusters are dynamically adjusted. Cluster centers from the previous timestep are reused for subsequent steps. We integrate the same Triton-based clustering kernel and FlashAttention fallback for skipped layers.

Datasets and Metrics. We use the same prompt set from PenguinVideoBenchmark as in the main experiments. Efficiency is measured by end-to-end generation speedup (relative to the FlashAttention baseline) and total attention computation in PFLOPs.

A.6.2. Results and Analysis

Table 8 summarizes the efficiency and quality results on H100.

Table 8. Efficiency and quality results of AdaCluster and baselines on H100.

Model	Method	Speedup↑	Attention (PFLOPs)↓
Wan2.1-14B	SVG2	1.61×	427.43
Wan2.1-14B	AdaCluster	**1.81**×	**404.94**
HunyuanVideo	SVG2	1.58×	335.36
HunyuanVideo	AdaCluster	**1.67**×	**322.21**

The results in Table 8 demonstrate that AdaCluster consistently outperforms SVG2 across both models. On Wan2.1-14B, AdaCluster achieves a speedup of 1.81× (vs. 1.61× for SVG2) while reducing attention PFLOPs from 427.43 to 404.94. Similarly, on HunyuanVideo, AdaCluster attains a speedup of 1.67× (vs. 1.58×) and lowers attention PFLOPs to 322.21 from 335.36. These improvements confirm the effectiveness of adaptively allocating cluster numbers per layer in enhancing both efficiency and generation quality.

A.7. Video Results with Different Methods

We selected HunyuanVideo as the baseline and compared the performance of different methods across various prompts, where all videos were produced on a single NVIDIA A40 GPU with configuration under 1280×720, 81 frames. The results demonstrate that AdaCluster achieves satisfactory accuracy, with high similarity to full attention. SVG2 performs reasonably well in most scenarios, but static clustering often encounters issues, particularly evident in the handling of video details. SpargeAttn, however, delivers poor performance due to neglecting embedding similarity.

Figure 12. Prompt: Two dolphins are swimming in the blue sea.

Figure 13. Prompt: A flying wild goose captured from a low-angle shot.

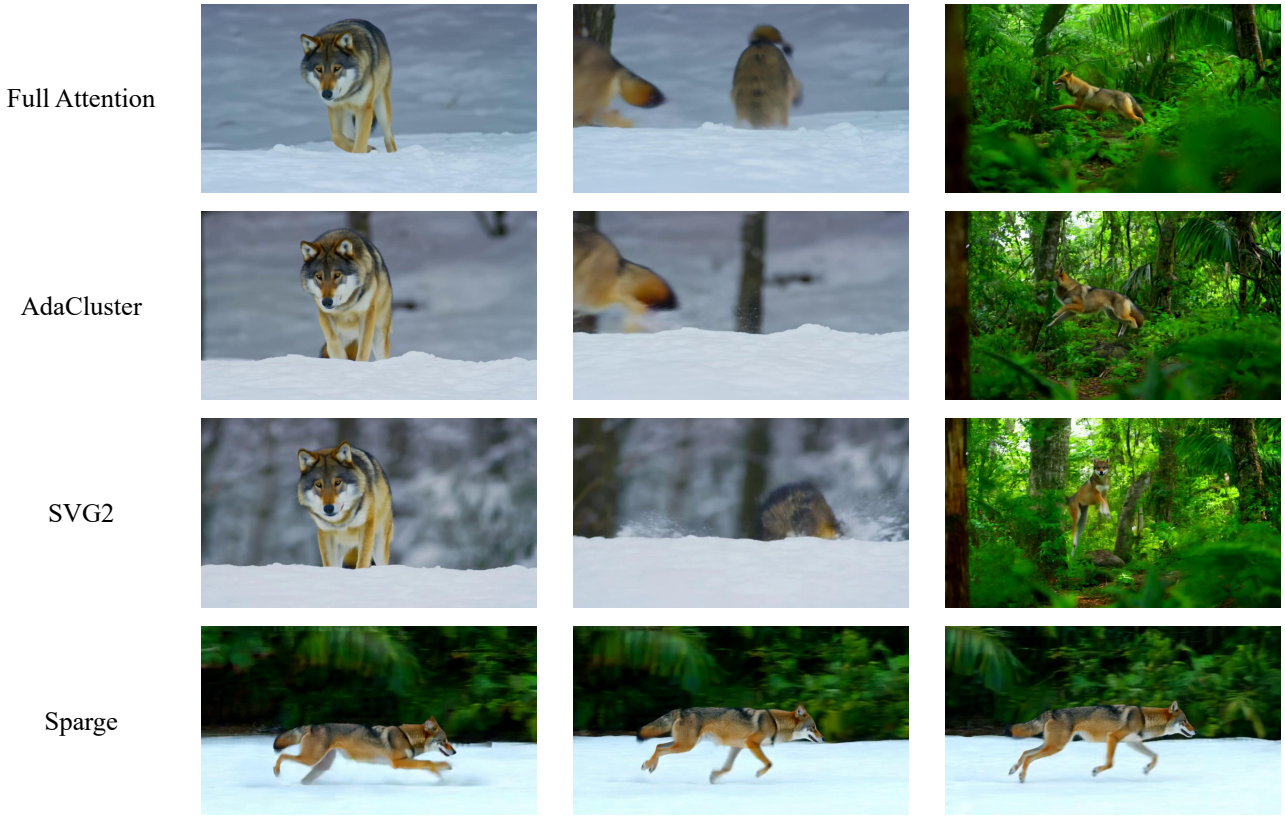

Figure 14. Prompt: Tracking shot of a wolf slowly moving through a tranquil snowy landscape, leaving footprints with each step. Suddenly, the wolf accelerates into a run and leaps. At the moment of the jump, the camera cuts to reveal the wolf landing in the midst of a lush, verdant tropical rainforest.

A.8. Video Results with Different Models and Prompts

We provide video screenshots generated by AdaCluster and full attention for comparison, where all videos were produced on a single NVIDIA A40 GPU. The comparison covers three different models: HunyuanVideo, CogvideoX and Wan-2.1.

Figure 15 shows the results on CogVideoX, Figure 16 presents the outputs of HunyuanVideo, and Figure 17 illustrates the results obtained with Wan-2.1. As shown, AdaCluster maintains a high degree of similarity to full attention across all models, demonstrating strong generalization capability. Notably, the original output quality of CogVideoX is inferior to that of HunyuanVideo and Wan-2.1. Nevertheless, AdaCluster consistently achieves stable and highly aligned results even on the lower-quality baseline.

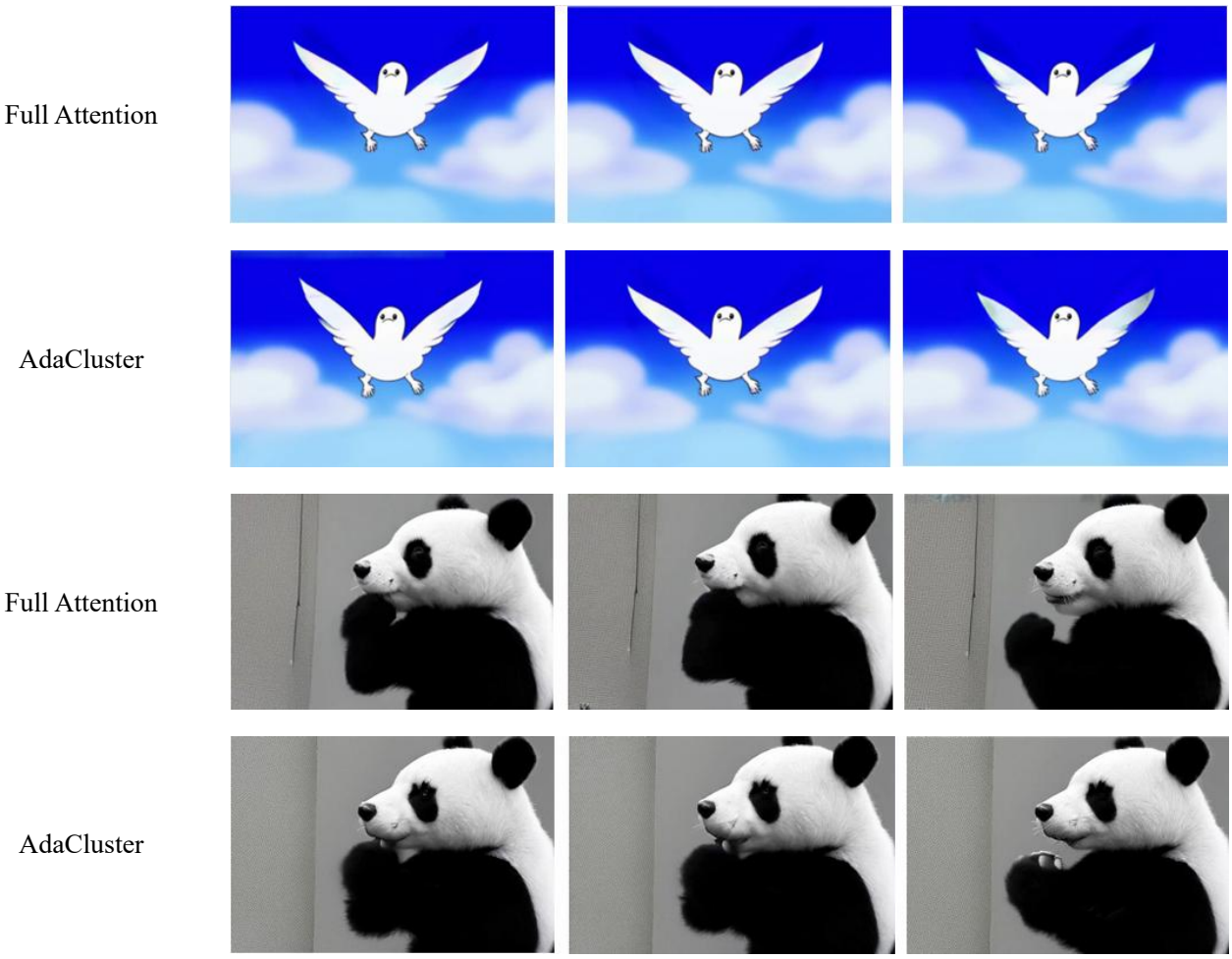

Figure 15. CogvideoX model screenshot

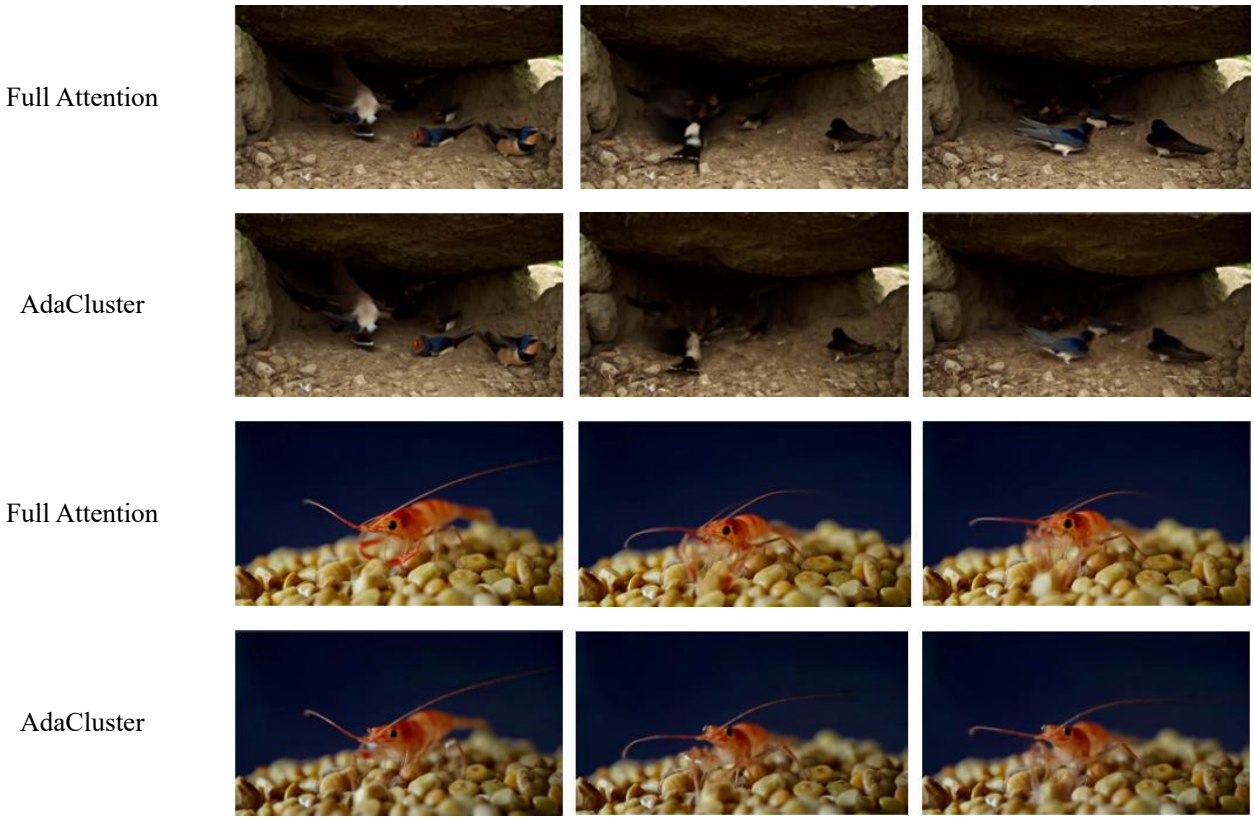

Figure 16. HunyuanVideo model screenshot

Figure 17. Wan-2.1 model screenshot