VMonarch: Efficient Video Diffusion Transformers with Structured Attention

Supplementary Material

A. Python code of MonarchAttention

In this section, we show the Python-like code of MonarchAttention [60] in Fig. 6.

```
def R_update(
    aR, # (m, b, d)
    cR, # (m, b)
    Kb, # (m, b, d)
    ): # Computes R, aL, cL from aR, cR
    R = softmax(bmm(aR, Kb.transpose(1, 2)) \
        / cR[:, :, None], dim=2)
    cL = sum(R * log(R), dim=2).transpose(0, 1)
    aL = bmm(R, Kb).transpose(0, 1)
    return aL, cL, R
```

```
def L_update(
    aL, # (b, m, d)
    cL, # (b, m)
    Qb, # (b, m, d)
    ): # Computes L, aR, cR from aL, cL
    L = softmax(bmm(Qb, aL.transpose(1, 2)) \
        - cL[:, None, :], dim=2)
    cR = sum(L, dim=1).transpose(0, 1)
    aR = bmm(L.transpose(1, 2), Qb).transpose(0, 1)
    return aR, cR, L
```

```
def monarch_attention(Q, K, V, m, b, T):
    # Q, K, V: (N, d), m * b = N
    # T > 0: number of steps
    Qb = Q.reshape(m, b, d).transpose(0, 1)
    Kb = K.reshape(m, b, d)
    Vb = V.reshape(m, b, d)
    aR = Q.reshape(m, b, d)
    cR = ones(m, b)

    for t in range(T):
        aL, cL, R = R_update(aR, cR, Kb)
        aR, cR, L = L_update(aL, cL, Qb)

    y = bmm(R, Vb).transpose(0, 1)
    O = bmm(L, y).transpose(0, 1)
    O = O.reshape(N, d)
    return O
```

Figure 6. Python-like code of MonarchAttention.

B. Optimized Entropy Flash Attention

The **online softmax algorithm** computes the softmax function over a vector $\mathbf{x} = [x_1, \ldots, x_N]$ in a single pass without storing the entire vector. This is achieved by maintaining running statistics. For numerical stability, the algorithm uses the shift-invariant property of softmax with a running maximum $m = \max_k\{x_k\}$. After processing i elements, it maintains a state tuple (m_i, S_i), where $m_i = \max_{k=1}^{i}\{x_k\}$ is the running maximum and $S_i = \sum_{k=1}^{i} e^{x_k - m_i}$ is the denominator sum normalized by m_i. When a new element x_{i+1} arrives, the state is updated. The new maximum is $m_{i+1} = \max(m_i, x_{i+1})$. If the maximum changes ($m_{i+1} > m_i$), the previous sum S_i must be rescaled by a factor $\alpha = e^{m_i - m_{i+1}}$. The general update for S_i is:

$$S_{i+1} = S_i \cdot e^{m_i - m_{i+1}} + e^{x_{i+1} - m_{i+1}} \tag{10}$$

We extend the online softmax algorithm to **online entropy**, which computes the Shannon entropy $H(p) = -\sum p_j \log p_j$ in a single pass. The entropy can be expressed in terms of the final statistics of the online softmax:

$$H(p) = -\sum_{j=1}^{N} p_j \log p_j \tag{11}$$

$$= -\sum_{j=1}^{N} p_j \left((x_j - m_N) - \log S_N\right) \tag{12}$$

$$= \log S_N - \frac{1}{S_N} \sum_{j=1}^{N} e^{x_j - m_N}(x_j - m_N) \tag{13}$$

To compute this online, we introduce a third running statistic, L_i, which is the sum of logits weighted by their unnormalized probabilities: $L_i = \sum_{k=1}^{i} e^{x_k - m_i}(x_k - m_i)$.

We now present the complete update for the state (m_i, S_i, L_i) upon receiving a new element x_{i+1}. First, find the new maximum $m_{i+1} = \max(m_i, x_{i+1})$.

If the maximum does not change ($m_{i+1} = m_i$), the updates are additive:

$$S_{i+1} = S_i + e^{x_{i+1} - m_{i+1}} \tag{14}$$

$$L_{i+1} = L_i + e^{x_{i+1} - m_{i+1}}(x_{i+1} - m_{i+1}) \tag{15}$$

If the maximum increases ($m_{i+1} > m_i$), the previous sums must be rescaled before adding the new term. Let $\alpha = e^{m_i - m_{i+1}}$. The updates are:

$$S_{i+1} = S_i \cdot \alpha + e^{x_{i+1} - m_{i+1}} \tag{16}$$

$$L_{i+1} = L_i \cdot \alpha + S_i \cdot \alpha \cdot (m_i - m_{i+1}) + e^{x_{i+1} - m_{i+1}}(x_{i+1} - m_{i+1}) \tag{17}$$

Noting that $\log(\alpha) = m_i - m_{i+1}$ and letting $p_{i+1} = e^{x_{i+1} - m_{i+1}}$, the update for L_{i+1} can be expressed more

Algorithm 2 Flash-Entropy-Attention Backward

Require: $\mathbf{Q}, \mathbf{K}, \mathbf{V}, \mathbf{O}, \mathbf{dO} \in \mathbb{R}^{N \times d}$, L, $H, dH \in \mathbb{R}^{N}$.
Ensure: $\mathbf{dQ}, \mathbf{dK}, \mathbf{dV} \in \mathbb{R}^{N \times d}$.

$T_r \leftarrow \lceil N/B_r \rceil, T_c \leftarrow \lceil N/B_c \rceil$.
Compute $D \in \mathbb{R}^{N}$ as $\mathrm{rowsum}(\mathbf{dO} \odot \mathbf{O})$.
Initialize $\mathbf{dQ} \leftarrow \mathbf{0}_{N \times d}$ in HBM.
for $j = 1 \rightarrow T_c$ **do**
  Load $\mathbf{K}_j, \mathbf{V}_j$ to SRAM.
  Initialize $\mathbf{dK}_j, \mathbf{dV}_j \leftarrow \mathbf{0}_{B_c \times d}$.
  for $i = 1 \rightarrow T_r$ **do**
    Load $\mathbf{Q}_i, \mathbf{O}_i, \mathbf{dO}_i, L_i, D_i, H_i, dH_i$ to SRAM.
    $\mathbf{S}_{ij} \leftarrow \mathbf{Q}_i \mathbf{K}_j^T$.
    $\mathbf{P}_{ij} \leftarrow \exp(\mathbf{S}_{ij} - L_i)$.
    $\mathbf{dV}_j \leftarrow \mathbf{dV}_j + \mathbf{P}_{ij}^T \mathbf{dO}_i$.
    $\mathbf{dP}_{ij} \leftarrow \mathbf{dO}_i \mathbf{V}_j^T$.
    $\mathbf{dS}_{ij} \leftarrow \mathbf{P}_{ij} \odot (\mathbf{dP}_{ij} - D_i)$. ▷ Standard Gradient
    $\mathbf{dS}_{ij} \leftarrow \mathbf{dS}_{ij} - dH_i \odot \mathbf{P}_{ij} \odot (\mathbf{S}_{ij} - L_i + H_i)$
    Load $\mathbf{dQ}_i$ from HBM.
    $\mathbf{dQ}_i \leftarrow \mathbf{dQ}_i + \mathbf{dS}_{ij} \mathbf{K}_j$.
    $\mathbf{dK}_j \leftarrow \mathbf{dK}_j + \mathbf{dS}_{ij}^T \mathbf{Q}_i$.
    Write $\mathbf{dQ}_i$ to HBM.
  end for
  Write $\mathbf{dK}_j, \mathbf{dV}_j$ to HBM.
end for

compactly in a form analogous to the entropy formula itself:

$$L_{i+1} = L_i \cdot \alpha + S_i \cdot \alpha \log(\alpha) + p_{i+1} \log(p_{i+1}) \tag{18}$$

After processing all N elements, the final state is (m_N, S_N, L_N). The Shannon entropy is then calculated as:

$$H(p) = \log S_N - \frac{L_N}{S_N} \tag{19}$$

C. Backward pass of Flash-Entropy-Attention

We provide the backward pass algorithm of FlashAttention with Online Entropy in Algorithm 2. However, the entropy term is not numerically stable; meanwhile, the gradient from entropy tends to dominate the gradient rather than the original attention output, which caused training instability. Therefore, we do not backpropagate through the entropy calculation in practice. We still provide the full backward algorithm here for completeness.

D. Experiments details

We select Wan2.1-1.3B [47] as our primary base model for training-free and post-training experiments, while also extending our verification to the larger Wan2.1-14B model [47] and the architecturally different Wan2.2-5B model [47]. For the Wan2.1-1.3B post-training, we tune the model with a $61 \times 448 \times 832$ input size for 1500 steps on the Fastvideo Wan14B-Syn-600k dataset [69], using a learning rate of 1×10^{-6} and a batch size of 8. For ablation studies on this model, we fine-tune for 500 steps with the same learning rate and batch size, calculating validation loss on 32 samples split from the training dataset. Regarding the Wan2.1-14B and Wan2.2-5B models, we conduct fine-tuning on the Wan2.2-Syn-32k dataset [69] with a resolution of $93 \times 704 \times 1280$. To handle the increased computational load, we employ sequence parallelism with a degree of 8. Regarding the sparse attention baselines, we follow their default setting. For VSA [69], we employ a sparsity annealing schedule that gradually increases the sparsity ratio to 90%, and use its Triton version as the CUDA version is incompatible with our hardware. For VMoBA, we adopt its default cyclic partitioning strategy, alternating between 1D, 2D, and 3D attention blocks. We set the chunk sizes to 4, (7, 13), and (4, 7, 13), respectively, and use a top-p (0.25) strategy for block selection for both training and inference. We test all models on VBench-Long [21] prompts with the same setting: 50 inference steps, guidance scale 5.0, and flow shift 3.0, the negative prompt is "Bright tones, overexposed, static, blurred details, subtitles, style, works, paintings, images, static, overall gray, worst quality, low quality, JPEG compression residue, ugly, incomplete, extra fingers, poorly drawn hands, poorly drawn faces, deformed, disfigured, misshapen limbs, fused fingers, still picture, messy background, three legs, many people in the background, walking backwards".

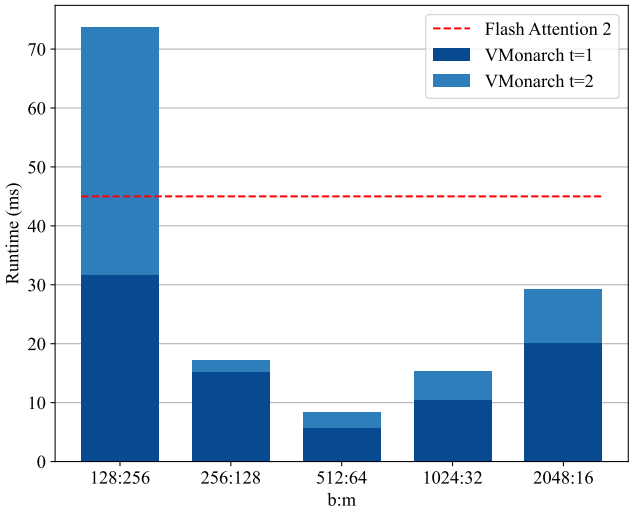


Figure A. Various computational budgets compared with FA2 at a sequence length of 2^{15}. In practice, b is usually larger than 256.

E. Various Monarch Computation Budgets

Our kernel supports flexible attention decomposition, and its computational budget can be adjusted by varying the monarch factor sizes m, b. Theoretically, the computational complexity is minimized when $m = b = \sqrt{N}$, and the I/O

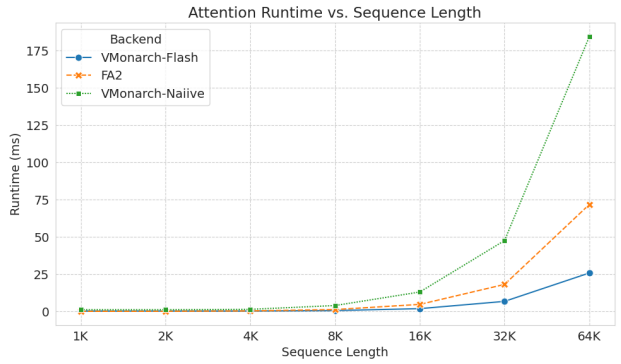


Figure 8. Speed comparison of FlashAttention2(FA2) [8], VMonarch-Naive, and our optimized VMonarch-Fast (Triton) implementation. After applying kernel optimization, VMonarch-Fast achieves an approximately $8\times$ speedup over the naive implementation and surpasses the performance of FA2.

complexity decreases as b increases. As the I/O complexity decreases as m decreases, the best m lies in $[1, \sqrt{N}]$. As shown in Fig. A, we profile the kernel inference time with various computation budgets at a sequence length of 32k. In the video modality, setting $b = 1\text{F}$ (where $\text{F} = HW$) corresponds to a specific decomposition ($b : m \approx 1024 : 32$), which can be further accelerated by reducing b via spatial downsampling.

F. Kernel Optimization of VMonarch

To further enhance the efficiency of our Video Monarch Attention (VMonarch), we developed a fused kernel. We benchmark the speed of our optimized kernel against a naive PyTorch implementation. The benchmark is conducted with a batch size of 1, 12 attention heads, and a head dimension of 64. The number of iterations for VMonarch is set to 2. We test on sequence lengths ranging from 2^{10} to 2^{16}. The Monarch factor sizes m and b are set to $m = 32$ and $b =$ sequence length$/32$. As shown in Fig. 8, the optimized kernel provides a significant speedup, achieving approximately $8\times$ speedup over the naive implementation and surpassing the performance of FlashAttention2 (FA2) [8].

G. Detailed Speed and Memory Benchmarks

We benchmark the memory usage and speed of our Video Monarch Attention (VMonarch) against several baselines: Full Attention (FA2), Video Sparse Attention (VSA), and various VMoBA [50] configurations (VMoBA-1d, VMoBA-2d, VMoBA-3d, and VMoBA-3d-TopP). The comparisons are conducted on different video lengths, with results presented in Fig. 9 and Fig. 10. Our findings indicate that VMoBA encounters a significant memory overhead bottleneck at finer granularities. Furthermore, with random inputs (uniform distribution), the TopP strategy used in VMoBA-3d-TopP leads to a noticeable slowdown in speed and an increase in memory consumption. We compare with FA2 as our kernel extends its Triton implementation. Our Online-Entropy Attention is also compatible with FA3, as both are based on the Online-Softmax algorithm.

Table 4. Similarity comparison of our and other baseline methods. All methods are evaluated under the training-free setting with a resolution of $61 \times 448 \times 832$.

Method	Similarity	
	PSNR ↑	SSIM ↑
FlashAttention2 (FullAttn) [8]	-	-
Video Sparse Attention (VSA) [69]	9.71	0.32
Video MoBA (VMoBA) [50]	10.61	0.30
Video Monarch (VMonarch)	12.59	0.43

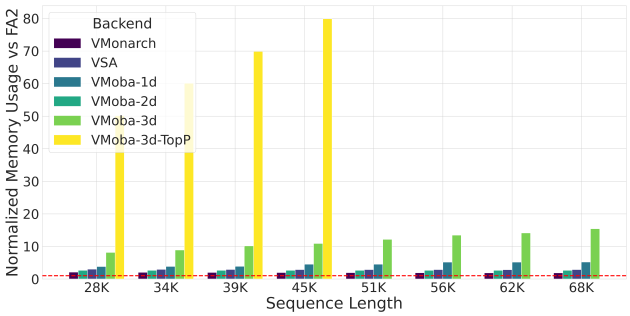


Figure 9. Memory benchmark of different attention mechanisms. We compare the relative memory usage against full attention.

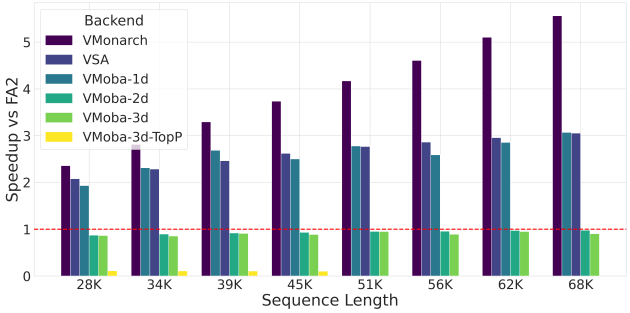


Figure 10. Speed benchmark of different attention mechanisms. We compare the relative speed against full attention.

H. Detailed Training-free comparison

As shown in Tab. 4, VMonarch outperforms VSA and VMoBA in PSNR and SSIM under the training-free setting, demonstrating that Monarch's block-diagonal structure effectively aligns with the spatio-temporal priors of video.

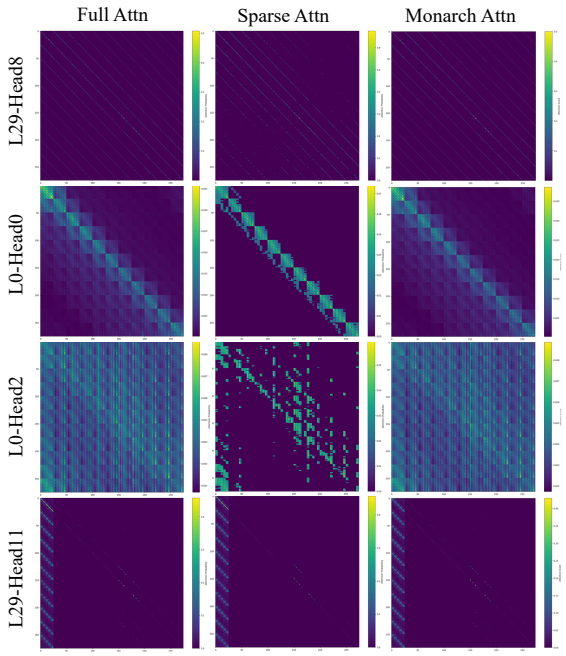


Figure 11. Attention map comparison among Full Attention, Sparse Attention, and Monarch Attention.

I. Comparison of Attention Maps

We visualize the attention maps of Full Attention [8], Sparse Attention(corresponding to VSA [69]), and Monarch Attention(corresponding to VMonarch) of different heads on Wanx1.3B [47]. We downsample the attention maps to 300×300 for better visualization. As shown in Fig. 11, Full Attention produces a dense attention map, while Sparse Attention generates a highly sparse attention map with many zero values. Our VMonarch does not produce an overly sparse distribution and is much more continuous compared to VSA.

J. Manual Control of Monarch Parameters

To better understand the behavior of Monarch Attention, we conduct a detailed ablation study on its key components, with results presented in Tab. 5. We analyze the impact of the number of iterations (t) and the manual control of the intermediate optimization terms, $\boldsymbol{c_L}$ and $\boldsymbol{c_R}$. For $\boldsymbol{c_L}$ and $\boldsymbol{c_R}$, we explore three strategies applied after each iteration: no control (using the updated value), fixing the value for the first frame only, and fixing the value for all frames.

Our findings reveal several key insights. First, the number of iterations is crucial; increasing from one ($Exp.1$, Score: 2.19) to two ($Exp.8$, Score: 3.29) yields a substantial performance improvement. Second, while manually controlling $\boldsymbol{c_L}$ and $\boldsymbol{c_R}$ can boost the score in the single-iteration setting (e.g., $Exp.2$ achieves a score of 3.11), this benefit vanishes with two iterations. In the cases of two iterations (Exp[8-14]), none of the manual control strategies outperform the baseline. This suggests that intervening in the optimization process may disrupt the stability of the alternating updates, leading to suboptimal performance.

In contrast, our proposed recomputation mechanism ($Exp.15$) provides a simple yet effective solution. It achieves the highest overall score of 3.44, outperforming all manual control variants and underscoring its effectiveness in enhancing generation quality without destabilizing the core optimization.

K. Different Monarch Factorization methods

Beyond the temporal-axis Monarch factorization method, we also explored alternative factorization strategies, such as spatial dimension factorization. Specifically, we grouped the video's spatial dimensions H or W with a grouping unit of 4, referred to as $(4, HWT/4)$ and $(4, WHT/4)$ respectively. We conducted a simple comparison of different grouping methods in a zero-shot setting, with the results shown in Fig. 12. Grouping along the H or W dimensions results in significant spatial discontinuities. Additionally, we experimented with larger temporal blocks for grouping, such as dividing the sequence into four temporal segments, referred to as $(4, THW/4)$. We observed that this grouping method leads to substantial temporal jumps between blocks, although temporal consistency within blocks remains stable. Our default setting (T, HW) provides the best balance between spatial and temporal coherence.

Figure 12. Comparison of different Monarch factorizations.

L. More Visual Results

We provide more visual comparisons in Fig. 13, comparing Full Attention, VSA, and our VMonarch on various prompts. The results demonstrate that VMonarch consistently produces higher-quality videos that compare favorably to Full Attention, while VSA sometimes exhibits semantic inconsistencies.

Table 5. Ablation study on VMonarch Attention. We evaluate several variants by adjusting its iterative parameters, including the number of iterations and the manual control of intermediate terms. Metrics with a superscript '1' (e.g., $PSNR^1$) are calculated on the first frame only. The score is calculated as Score $=$ (PSNR $+$ PSNR^1)$/200+$ (SSIM $+$ SSIM^1)$/2+$ (1 $-$ LPIPS $+$ 1 $-$ LPIPS^1)$/2+$ IQ $+$ SC $+$ OC, where IQ, SC, and OC denote Imaging Quality, Subject Consistency, and Overall Consistency, respectively. In the Method column, M$^{(t)}$ denotes the number of iterations t. +Re represents our recomputation mechanism. For the control of $\boldsymbol{c_R}$, +$\boldsymbol{c_R^*}$ indicates fixing it to 1 for all frames, +$\boldsymbol{c_R^1}$ for the first frame only, and +$\boldsymbol{c_R^0}$ means no fixing. For the control of $\boldsymbol{c_L}$, +$\boldsymbol{c_L^*}$ indicates fixing it to the initial expected value $\log(M)$ for all frames, where M is the monarch factor size, +$\boldsymbol{c_L^1}$ for the first frame only, and +$\boldsymbol{c_L^0}$ means no fixing.

Exp.	Method	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR^1 ↑	SSIM^1 ↑	LPIPS^1 ↓	IQ ↑	SC ↑	OC ↑	**Score** ↑
1	M$^{(1)}$	11.01	0.21	96.41%	10.21	0.12	90.44%	48.54%	87.00%	16.28%	2.19
2	M$^{(1)}$+$\boldsymbol{c_R^*}$+$\boldsymbol{c_L^1}$	10.66	0.23	80.39%	11.21	0.30	51.54%	62.22%	87.53%	18.85%	3.11
3	M$^{(1)}$+$\boldsymbol{c_R^1}$+$\boldsymbol{c_L^1}$	10.66	0.23	80.39%	11.21	0.30	51.54%	62.22%	87.53%	18.85%	3.11
4	M$^{(1)}$+$\boldsymbol{c_R^1}$+$\boldsymbol{c_L^*}$	11.01	0.18	99.78%	10.29	0.11	95.63%	41.32%	85.87%	15.69%	1.98
5	M$^{(1)}$+$\boldsymbol{c_R^*}$+$\boldsymbol{c_L^*}$	11.01	0.18	99.78%	10.29	0.11	95.63%	41.32%	85.87%	15.69%	1.98
6	M$^{(1)}$+$\boldsymbol{c_R^1}$+$\boldsymbol{c_L^0}$	11.01	0.21	96.41%	10.21	0.12	90.44%	48.54%	87.00%	16.28%	2.19
7	M$^{(1)}$+$\boldsymbol{c_R^0}$+$\boldsymbol{c_L^1}$	10.66	0.23	80.39%	11.21	0.30	51.54%	62.22%	87.53%	18.85%	3.11
8	M$^{(2)}$	11.19	0.28	52.44%	9.39	0.21	55.53%	72.17%	75.96%	19.47%	3.29
9	M$^{(2)}$+$\boldsymbol{c_R^*}$+$\boldsymbol{c_L^1}$	6.93	0.11	78.38%	7.90	0.16	71.14%	48.53%	97.68%	3.02%	2.42
10	M$^{(2)}$+$\boldsymbol{c_R^1}$+$\boldsymbol{c_L^1}$	6.94	0.11	78.59%	7.78	0.16	71.70%	49.00%	97.88%	2.97%	2.41
11	M$^{(2)}$+$\boldsymbol{c_R^1}$+$\boldsymbol{c_L^*}$	10.71	0.24	64.26%	8.68	0.12	58.73%	75.60%	90.48%	14.03%	3.12
12	M$^{(2)}$+$\boldsymbol{c_R^*}$+$\boldsymbol{c_L^*}$	8.12	0.17	64.85%	6.96	0.07	59.14%	52.85%	96.08%	8.66%	2.73
13	M$^{(2)}$+$\boldsymbol{c_R^1}$+$\boldsymbol{c_L^0}$	11.13	0.27	55.81%	9.14	0.19	55.01%	69.62%	76.55%	19.72%	3.21
14	M$^{(2)}$+$\boldsymbol{c_R^0}$+$\boldsymbol{c_L^1}$	6.51	0.13	80.88%	7.16	0.18	71.79%	49.18%	97.23%	1.51%	2.40
15	M$^{(2)}$+Re	11.55	0.29	56.96%	10.13	0.27	58.18%	73.06%	91.26%	17.19%	3.44

M. Discussion on Monarch Attention

Monarch Attention [60] is a general form of normal self-attention with Monarch matrix [11] representation, which is suitable for sparse distribution modeling. As shown in Eq. (4), Eq. (5), Eq. (6), Eq. (7) and Eq. (8), Monarch Attention has many parameters and can derive many variants that deserve to be discussed.

Monarch Approximation Error Analysis. We formulate the Monarch approximation error as $D_{KL}(S||M) \approx D_{KL}(S||M^*) + D_{KL}(M^*||M)$. The structural error $D_{KL}(S||M^*)$ represents the intrinsic gap between full attention S and the optimal Monarch factorization M^*, which is minimized when S aligns with block-diagonal structures; VMonarch effectively reduces this via Spatio-Temporal factorization that matches video priors. The iteration error $D_{KL}(M^*||M)$ stems from the iterative solver, and we find that it converges after 2 iterations.

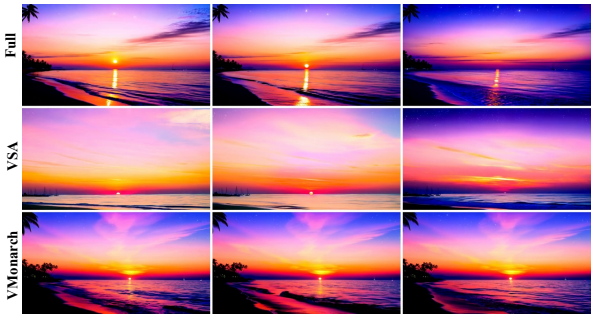
Understanding iteration terms. $\boldsymbol{c_R}$ and $\boldsymbol{c_L}$ can be interpreted as temperature adjustment factors for the softmax function based on entropy statistics within each monarch factor L and R. They are the intermediate terms under the alternating maximization optimization process. A better optimization algorithm or a suitable optimization object target may bring simpler and more efficient intermediate terms.

Correlation with full attention. As $N = m \times b$, if $b = 1$ or $m = 1$, MonarchAttention reduces to standard attention. From this point of view, full attention can be viewed as a special case of Monarch attention, where only a single monarch factor is applied.

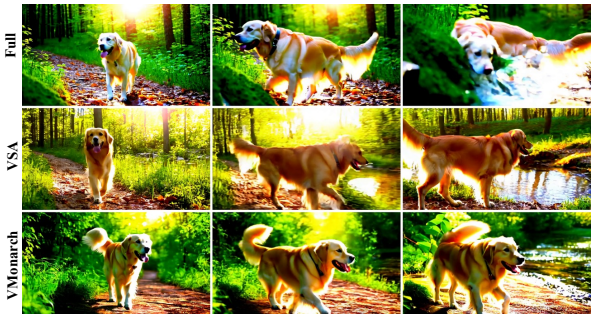
Correlation with spatial-temporal attention. When no iteration is performed, Monarch Attention degenerates to Spatial/Temporal Attention. When only R updating Eq. (4) is applied and $\mathbf{c_R}$ is set to the all-ones matrix and L is set to the identity matrix, it is equal to the temporal-attention. Although Monarch Attention [60] provides an iterative approximation algorithm to reduce $D_{KL}(M^*||M)$, its inefficient entropy calculation hinders scaling to long sequences. Our proposed Online-Entropy Flash Attention overcomes the OOM bottleneck via algorithm-system co-design, which is critical for the video modality. Furthermore, we leverage Spatio-Temporal Factorization aligned with video priors to directly minimize $D_{KL}(S||M^*)$, and introduce a general Recomputation strategy to help model convergence by losslessly approximating specific tokens.

N. Limitations and Future Work

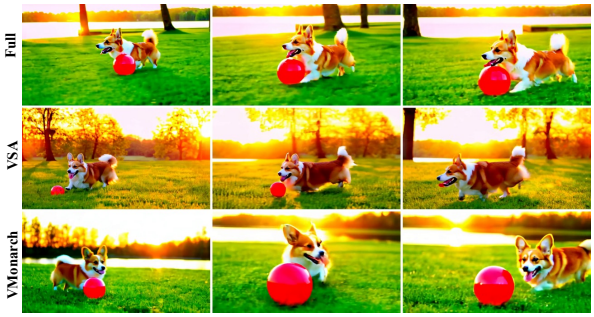
VMonarch has several limitations that present opportunities for future research. First, we employ a static sparsification strategy with uniform Monarch parameters across all layers and heads, whereas a dynamic strategy that adapts to the distinct distribution priors of different layers could potentially improve performance. Second, although VMonarch achieves a great theoretical sparsity, it still relies on dense attention within each Monarch factor, and sparsifying this internal computation could further boost efficiency. Third, a more balanced Monarch factorization (e.g., via spatial downsampling) could further enhance efficiency.



A breathtaking time-lapse captures the sun setting over a tranquil beach, where the sky transforms from a soft orange to deep purples and pinks. Wispy clouds drift gracefully across the horizon, reflecting the changing hues of the sky. The golden sun slowly dips below the water, casting a shimmering path of light on the gentle waves. Silhouettes of distant sailboats and palm trees add to the serene ambiance. As the sky darkens, stars begin to twinkle, and the last remnants of daylight fade, leaving a peaceful, starlit night over the calm, rhythmic ocean.



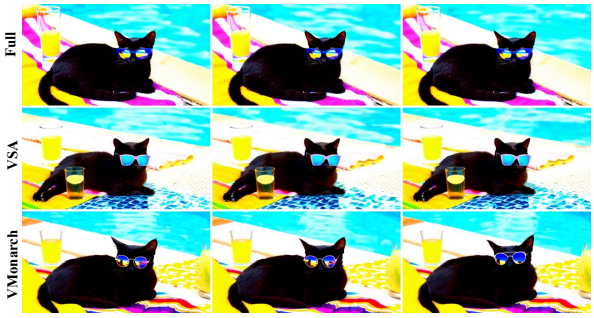
A golden retriever with a shiny coat strolls leisurely through a sun-dappled forest path, the morning light filtering through the trees casting a warm glow. The dog's tail wags gently as it sniffs the air, ears perked up, taking in the serene surroundings. The camera captures close-ups of its joyful expression, tongue lolling out, and eyes sparkling with contentment. As it walks, the soft crunch of leaves under its paws adds to the tranquil ambiance. The scene transitions to the dog pausing by a clear, babbling brook, lapping up the cool water, before continuing its peaceful journey through the picturesque woodland.



A joyful Corgi with a fluffy coat and perky ears bounds through a sunlit park, the golden hues of sunset casting a warm glow on the scene. The dog's playful energy is evident as it chases after a bright red ball, its short legs moving swiftly across the lush green grass. The Corgi pauses momentarily to look back at the camera, its tongue lolling out in a happy grin, before darting off again, its tail wagging furiously. The backdrop of tall trees and a serene lake reflects the soft, amber light of the setting sun, creating a picturesque and heartwarming moment.

A serene cow with a glossy brown coat lies comfortably on a bed of fresh straw inside a rustic, sunlit barn. The gentle rays of the afternoon sun filter through the wooden slats, casting a warm, golden glow over the scene. The cow's large, expressive eyes blink slowly as it rhythmically chews its cud, creating a sense of calm and contentment. Surrounding the cow are various farm tools and bales of hay, adding to the authentic, tranquil atmosphere. The soft sounds of the barn—occasional rustling of straw and distant chirping of birds—enhance the peaceful ambiance, making it a perfect moment of rural serenity.

A sleek motorcycle, gleaming under the midday sun, cruises effortlessly along a winding coastal highway. The rider, clad in a black leather jacket, helmet, and jeans, leans into the curves with precision, the ocean's azure waves crashing against rugged cliffs below. The bike's engine purrs smoothly, harmonizing with the rhythmic sound of the waves. As the motorcycle glides past tall, swaying palm trees and sun-drenched sandy beaches, the horizon stretches endlessly, blending the sky's deep blue with the sea's shimmering surface. The scene captures the essence of freedom and adventure, with the coastal breeze adding a sense of exhilaration to the journey.



A sleek, black cat lounges on a sunlit poolside deck, wearing stylish, tiny sunglasses that reflect the shimmering water. The cat's fur glistens under the bright sun, and its relaxed posture exudes cool confidence. Nearby, a colorful beach towel and a half-empty glass of lemonade add to the summery vibe. The cat occasionally stretches, its sunglasses staying perfectly in place, while the gentle ripples in the pool create a soothing background. The scene captures a perfect blend of feline elegance and laid-back summer fun.

A sleek, black sedan glides slowly down a deserted, rain-soaked street, its headlights cutting through the misty evening air. The streetlights cast a warm, golden glow on the wet pavement, reflecting the car's silhouette as it moves. Raindrops gently patter on the car's roof and windows, creating a soothing rhythm. The surrounding buildings, with their darkened windows and muted colors, stand silent and still, adding to the serene, almost melancholic atmosphere. The car's windshield wipers sweep rhythmically, clearing the view ahead as it continues its unhurried journey through the tranquil, rain-drenched night.

A towering Bigfoot trudges through a fierce snowstorm, its massive, fur-covered form barely visible against the swirling white. The creature's powerful strides leave deep footprints in the snow, each step echoing its immense weight and strength. Snow clings to its thick, matted fur, and its eyes, glowing faintly, peer through the blizzard with an almost human-like intensity. The wind howls around it, whipping up flurries that obscure its path, but Bigfoot moves with purpose, undeterred by the harsh elements. The scene captures the raw, untamed wilderness, with the mythical creature embodying the mystery and majesty of nature's most elusive legends.

Figure 13. More visual results comparing Full Attention, VSA, and VMonarch on various prompts.