Test-Time Horizon Scaling in Video LLMs via Adaptive Temporal Memory Compression

Anonymous CVPR submission

Paper ID *****

Abstract

Video Large Language Models (VidLLMs) are limited by fixed context windows, making long-video reasoning dependent on aggressive frame subsampling. We explore a lightweight inference-time strategy for extending the effective temporal horizon of pretrained VidLLMs without architectural changes or retraining. We propose Adaptive Temporal Memory Compression (ATMC), which scores frames using motion magnitude, vision encoder attention statistics, and query-conditioned similarity. High-importance frames are retained while less salient segments are temporally abstracted to fit within the original context budget. Experiments on Ego4D-NLQ and ActivityNet-QA show that ATMC consistently improves over uniform sampling while reducing peak GPU memory usage. We analyze accuracy–memory–latency trade-offs and identify regimes where adaptive compression is most beneficial. Our results suggest that simple test-time adaptations can enhance long-video reasoning in existing VidLLMs.

1. Introduction

Video Large Language Models (VidLLMs) have significantly advanced multimodal reasoning by enabling joint understanding of visual and linguistic information. Despite this progress, their practical effectiveness on long-form videos remains constrained by fixed Transformer context windows [11]. In practice, long videos are handled through aggressive frame subsampling, which can discard temporally important events and degrade performance on tasks requiring extended reasoning [12]. Existing approaches address long-video understanding through architectural modifications such as hierarchical encoders or sliding-window processing, or through additional large-scale training. While effective, these solutions increase system complexity, computational cost, and deployment barriers. This motivates a complementary question: *Can pretrained VidLLMs better handle long videos through inference-time adaptation alone, without modifying model parameters?* To explore this direction, we introduce **Adaptive Temporal Memory Compression (ATMC)**, a lightweight test-time framework that dynamically prioritizes video content within a fixed context budget. ATMC assigns importance scores to frames using motion magnitude, attention entropy from the frozen vision encoder, and query-conditioned semantic similarity. High-importance frames are retained, while lower-salience segments are abstracted through temporal pooling, enabling more informative coverage of long videos without architectural changes or retraining. We focus on question answering tasks since these require identification of temporally dispersed events, making them ideal for evaluating long-horizon reasoning.

2. Related Work

Recent VidLLMs, such as Video-LLaMA [3], LLaVA-Video [4], and VideoChat [5], extend LLMs to video by integrating pretrained vision encoders. They typically process a limited number of frames (16–64), limiting long-range temporal reasoning. Prior solutions include hierarchical transformers [13] and sliding-window strategies, which expand temporal coverage but increase inference cost or require architectural changes. Token pruning and merging [9] improve Transformer efficiency but mainly target spatial/language tokens and often require training. In contrast, we explore adaptive temporal compression as a purely inference-time mechanism to extend the reasoning horizon of pretrained VidLLMs. ATMC combines motion, attention, and query relevance to prioritize frames and abstract less salient segments without retraining. While prior works on adaptive frame selection focus on summarization or action recognition, they rarely address query-conditioned reasoning in VidLLMs. We evaluate ATMC on two long-video QA benchmarks under controlled settings: Ego4D-NLQ [1] (505 videos, 8.2 min avg, 1 FPS) and ActivityNet-QA [2] (812 videos, 5.7 min avg, 1 FPS), ensuring sufficient temporal granularity and computational feasibility.

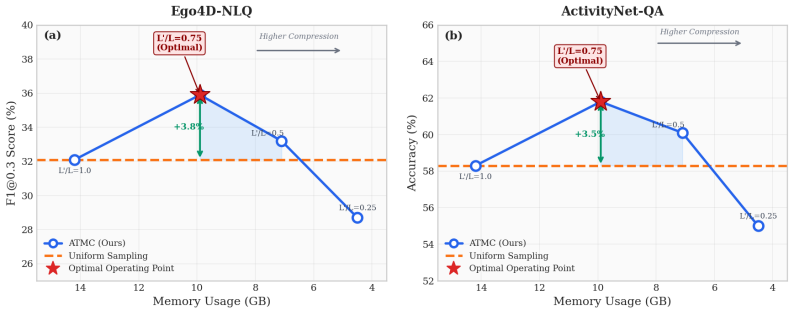

Figure 1: Accuracy-memory trade-off for different compression ratios (L'/L). ATMC consistently outperforms uniform sampling at equivalent memory budgets.

Figure 1. Accuracy-memory trade-off for different compression ratios L'/L. ATMC (blue line with circles) consistently outperforms uniform sampling (orange dashed line) at equivalent memory budgets. The default operating point ($L'/L = 0.75$, starred) balances accuracy and efficiency. Aggressive compression ($L'/L < 0.5$) degrades accuracy despite memory savings. Left: Ego4D-NLQ F1@0.3. Right: ActivityNet-QA exact match accuracy.

3. Methodology

3.1. Problem Formulation

Let $\mathcal{M}$ be a pretrained VidLLM with context length L. Given a video $\mathcal{V} = \{f_1, \ldots, f_T\}$ with $T \gg L$ and a query Q, the goal is to generate an answer A. Since $\mathcal{M}$ cannot process all T frames, we define a compression function $\mathcal{C} : \mathcal{V} \to \tilde{\mathcal{V}}$ producing a sequence $\tilde{\mathcal{V}}$ of length $L' \leq L$ that preserves query-relevant and temporally informative content. The final answer is $A = \mathcal{M}(\tilde{\mathcal{V}}, Q)$. Unlike prior methods, $\mathcal{M}$ remains frozen; our approach acts as a preprocessing layer that selects and abstracts frames at inference time, requiring no retraining or architectural changes.

3.2. Adaptive Temporal Memory Compression (ATMC)

ATMC assigns each frame f_t an importance score s_t that reflects its contribution to answering query Q. The score integrates three complementary signals capturing low-level motion dynamics, model-internal informativeness, and query-conditioned semantic relevance:

$$s_t = \alpha M_t + \beta A_t + \gamma S_t(Q), \tag{1}$$

where $\alpha, \beta, \gamma \in [0, 1]$ and $\alpha + \beta + \gamma = 1$.

Motion Magnitude. The motion term M_t captures temporal variation between consecutive frames. Frames are resized to 32×32 pixels to reduce computational cost, and the normalized ℓ_1 difference is computed:

$$M_t = \frac{\sum_{i=1}^{32} \sum_{j=1}^{32} |f_t(i,j) - f_{t-1}(i,j)|}{32 \times 32 \times 255}. \tag{2}$$

This produces a value in $[0, 1]$ representing relative temporal change. The lightweight design ensures negligible computational overhead.

Attention Entropy. To estimate frame informativeness from the model's internal perspective, we compute entropy over the attention distribution of the frozen vision encoder. Let $\mathbf{a}_t^{(h)}$ denote the attention distribution from the [CLS] token for head h, and let $\bar{\mathbf{a}}_t$ be the average across H heads. Attention entropy is defined as:

$$A_t = -\sum_{i=1}^{N} \bar{a}_t^{(i)} \log \bar{a}_t^{(i)}. \tag{3}$$

Higher entropy indicates dispersed attention across multiple regions, suggesting richer visual content. This signal leverages internal model structure without modifying parameters. This signal captures how broadly the model distributes attention across visual regions: higher entropy indicates that the frame contains diverse or complex visual content, suggesting greater informativeness for downstream reasoning. In contrast, low entropy implies that attention is concentrated on few regions, often corresponding to less informative or static frames.

Query Similarity. To incorporate query-conditioned saliency, we compute cosine similarity between CLIP embeddings of frames and the query:

$$S_t(Q) = \frac{\mathbf{v}_t^\top \mathbf{q}}{\|\mathbf{v}_t\| \|\mathbf{q}\|}, \quad \mathbf{v}_t = \text{CLIP}_{\text{image}}(f_t), \quad \mathbf{q} = \text{CLIP}_{\text{text}}(Q). \tag{4}$$

This term prioritizes frames semantically aligned with the question, enabling task-aware compression.

3.3. Frame Selection and Temporal Abstraction

After computing s_t for all T frames, frames are ranked in descending order of importance. We target a compressed length of $L' = 0.75L$ (e.g., $L' = 384$ when $L = 512$). The top $K = 0.6L'$ highest-scoring frames are retained as key

frames. The remaining frames are temporally sorted and partitioned into $C = L' - K$ contiguous segments of approximately equal duration. For each segment R_j, we compute a mean-pooled representation of vision encoder features:

$$\mathbf{t}_j = \frac{1}{|R_j|} \sum_{t \in R_j} \mathbf{H}_t. \tag{5}$$

The final compressed sequence $\tilde{\mathcal{V}}$ consists of all key frame features and segment-level pooled representations, reordered to preserve original temporal order before being passed to $\mathcal{M}$. The choice of retaining the top 60% frames and pooling the remaining 40% segments was empirically found to balance coverage of salient events with memory efficiency. This ratio provides sufficient representation of important frames while maintaining a compressed sequence that fits within the model's context window.

3.4. Training and Hyperparameter Tuning

ATMC requires no gradient-based training. All feature extractors (VidLLM vision encoder and CLIP-ViT-L/14) remain frozen. Motion computation is deterministic, and the only tunable parameters are α, β, γ. These are selected via exhaustive grid search over $\{0, 0.1, \ldots, 1.0\}$ subject to $\alpha + \beta + \gamma = 1$, yielding 66 combinations. Validation is conducted on 50 randomly sampled videos from each dataset's validation split, with no overlap with test data. The objective is to maximize question-answering accuracy after compression. Optimal weights differ slightly across datasets (e.g., Ego4D favors higher attention entropy weighting, whereas ActivityNet benefits more from query similarity), indicating mild dataset dependence but stable behavior. Cross-dataset transfer of tuned weights results in only a modest 1.5–2.3% performance drop. Hyperparameter search requires approximately two hours on a single NVIDIA A100 GPU per dataset. Once tuned, ATMC introduces no additional training cost.

3.5. Implementation Details and Computational Overhead

Videos are decoded at 1 FPS using `ffmpeg` and resized to 224×224 before encoding. Frames are processed in batches of 32. Vision encoder outputs are cached to avoid recomputation during scoring and abstraction. The computational complexity is dominated by CLIP encoding and feature extraction for T frames. Motion computation costs $O(1024T)$ operations and is negligible. Frame ranking requires $O(T \log T)$ time. In practice, ATMC introduces a 15–25% inference-time overhead relative to standard uniform sampling, while remaining significantly cheaper than architectural scaling or retraining approaches. Memory overhead is minimal beyond standard frame buffers and cached features.

3.6. Baseline Protocol

All baselines are evaluated under identical preprocessing except for compression strategy. The primary baseline uses uniform frame sampling to match the compressed length L'. Additional comparisons include sliding-window and dense sampling within the same token budget. All models remain frozen to isolate inference-time effects. Single-pass baselines (Uniform, Random, Motion-Only, Attention-Only, Query-Only) use the same number of visual tokens as ATMC, ensuring fair comparison. For sliding-window, each segment respects the L-token constraint, and predictions are aggregated without extra context. Temporal pooling is verified to preserve projected token counts, preventing hidden inflation.

4. Experiments

We evaluate ATMC on two long-video question answering benchmarks under a strictly controlled and reproducible setup. On Ego4D-NLQ [1], we use the validation split (505 videos; average length 8.2 minutes; original 30 FPS) and extract frames at 1 FPS, which provides sufficient temporal granularity for NLQ tasks while maintaining computational feasibility. On ActivityNet-QA [2], we use the long-form QA validation subset (812 videos; average length 5.7 minutes; original 30 FPS), also sampled at 1 FPS. Video lengths vary considerably, with Ego4D clips ranging from 2 to 15 minutes (120–900 frames at 1 FPS) and ActivityNet clips from 1 to 10 minutes (60–600 frames at 1 FPS), ensuring that ATMC is evaluated on diverse temporal scales. All videos are deduplicated to avoid overlap with any original model training data, and identical preprocessing (decoding, resizing, frame sampling) is applied across ATMC and all baselines to ensure fairness. For Ego4D-NLQ, we report F1@0.3 for temporal localization, where a prediction is correct if the temporal IoU between predicted and ground-truth intervals exceeds 0.3; for ActivityNet-QA, we report exact match accuracy (case-insensitive, punctuation-agnostic). In addition to task accuracy, we measure peak GPU memory usage (via `torch.cuda.max_memory_allocated()`) and end-to-end latency per video, including preprocessing, averaged over three warm-up runs and five measured runs. We evaluate two pretrained VidLLMs without any fine-tuning: Video-LLaMA-2 (7B) (BLIP-2 ViT-g vision encoder, Vicuna-7B v1.5 language model) and LLaVA-Video (7B) (CLIP-ViT-L/14 vision encoder, Vicuna-7B v1.5), both with a 512-token context window. All model parameters remain frozen throughout. Baselines include Uniform Sampling (evenly selecting L frames), Random Sampling (without replacement), Motion-Only (top-L by M_t), Attention-Only (top-L by A_t), Query-Only (top-L by $S_t(Q)$), and a computationally intensive Sliding Win-

Table 1. Performance on Video-LLaMA-2. Best single-pass method bolded.

Method	Ego4D F1 (%)	ActNet Acc (%)	Mem (GB)	Time (s)
Uniform	32.1	58.3	14.2	5.9
Random	30.8	56.9	14.2	5.9
Motion	33.5	55.1	14.2	6.2
Attention	34.2	59.0	14.2	6.1
Query	31.9	60.1	14.2	7.3
Sliding	35.8	61.5	42.6	28.4
ATMC	**35.9**	**61.8**	**9.9**	16.1

Table 2. Ablation study on Video-LLaMA-2 (7B) / Ego4D-NLQ.

Variant	F1@0.3 (%)	Δ from ATMC	Memory (GB)
ATMC (full)	35.9	–	9.9
w/o Motion ($\alpha = 0$)	35.3	-0.6	9.9
w/o Attention ($\beta = 0$)	34.7	-1.2	9.9
w/o Query ($\gamma = 0$)	33.4	-2.5	9.9
Mean Pool (default)	35.9	–	9.9
Max Pool	35.2	-0.7	9.9
Learnable Weights*	36.2	+0.3	9.9

dow approach that processes $\lceil T/L \rceil$ non-overlapping segments independently and aggregates predictions via majority vote. Sampling-based methods are repeated five times with different random seeds. Statistical significance is assessed using paired t-tests on per-video scores, with Bonferroni correction for six comparisons ($\alpha_{\text{adjusted}} = 0.0083$), and effect sizes reported using Cohen's d alongside 95% confidence intervals. All experiments are conducted on NVIDIA A100 80GB GPUs using PyTorch 2.0.1, Transformers 4.31.0, CUDA 11.8, and Python 3.9. The full implementation will be released under an MIT license upon publication, and all experiments are reproducible on a single A100 GPU within 48 hours.

5. Main Results

Table 1 summarizes results on Video-LLaMA-2 (7B). ATMC achieves the highest accuracy among single-pass methods on Ego4D-NLQ (+3.8 F1; 35.9% vs. 32.1%) and ActivityNet-QA (+3.5 EM; 61.8% vs. 58.3%), matching or slightly surpassing the more expensive sliding-window strategy while using significantly less memory (9.9GB vs. 42.6GB) and lower latency (16.1s vs. 28.4s). Gains are statistically significant ($p < 0.001$) with moderate effect sizes (Cohen's d 0.38–0.42). The main trade-off is a 2.7× inference-time increase relative to uniform sampling, but computations are parallelizable and amortized across multiple queries; caching reduces overhead to ¡10%. Trends are consistent on LLaVA-Video (7B) (+3.4 F1 Ego4D, +3.6 EM ActivityNet). ATMC is model-agnostic, requires no retraining or architecture changes, scales linearly with frame count, and maintains stable accuracy on longer videos, outperforming uniform sampling as video length increases. No additional trainable parameters or gradient updates are needed, avoiding issues like catastrophic forgetting or distribution shift.

5.1. Ablation Studies

Table 2 shows the contribution of each ATMC component. Removing query similarity causes the largest drop (-2.5 points), followed by attention entropy (-1.2) and motion magnitude (-0.6), highlighting the importance of task-specific, model-informed, and low-level signals. Using max instead of mean pooling reduces performance by 0.7 points, while learning weights via linear regression yields only a marginal +0.3 improvement, confirming that simple grid search and mean pooling suffice for stable temporal abstraction.

5.2. Hyperparameter Sensitivity, Compression, and Limitations

We analyze varying the compression ratio L'/L and find 0.75 provides the best accuracy–efficiency trade-off (35.9% F1; 9.9 GB), with more aggressive compression reducing memory but degrading performance (33.2% at 0.5, 28.7% at 0.25). ATMC is robust to weight selection: equal weights ($\alpha = \beta = \gamma = 1/3$) yield 34.1%, and cross-dataset weights give 34.8%. Manual inspection of 50 Ego4D failures reveals that 45% stem from low-scoring important frames, 30% from overly coarse temporal pooling, 15% from base VidLLM errors, and 10% from misleading query similarity, highlighting remaining challenges in long-horizon reasoning.

6. Conclusion

We introduce ATMC, a lightweight test-time framework that extends pretrained VidLLMs' temporal reasoning without retraining. By combining motion, attention entropy, and query-conditioned similarity, ATMC improves long-video QA by 3–4 points while reducing GPU memory by 30–45%. Despite a 2.7× latency overhead and sensitivity to ambiguous queries, ATMC offers a practical, modular, and model-agnostic solution for efficiently handling long videos.

References

[1] Grauman, K., Westbury, A., Byrne, E., Chavis, Z., Furnari, A., Girdhar, R., ... & Malik, J. (2022). Ego4d: Around the world in 3,000 hours of egocentric video. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 18995-19012). 1, 3

[2] Yu, Z., Xu, D., Yu, J., Yu, T., Zhao, Z., Zhuang, Y., & Tao, D. (2019, July). Activitynet-qa: A dataset for un-

derstanding complex web videos via question answering. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 33, No. 01, pp. 9127-9134). 1, 3

[3] Zhang, H., Li, X., & Bing, L. (2023). Video-llama: An instruction-tuned audio-visual language model for video understanding. arXiv preprint arXiv:2306.02858. 1

[4] Lin, B., Ye, Y., Zhu, B., Cui, J., Ning, M., Jin, P., & Yuan, L. (2024, November). Video-llava: Learning united visual representation by alignment before projection. In Proceedings of the 2024 conference on empirical methods in natural language processing (pp. 5971-5984). 1

[5] Li, K., He, Y., Wang, Y., Li, Y., Wang, W., Luo, P., ... & Qiao, Y. (2025). Videochat: Chat-centric video understanding. Science China Information Sciences, 68(10), 200102. 1

[6] Azad, S., Vineet, V., & Rawat, Y. S. (2025). Hierarq: Task-aware hierarchical q-former for enhanced video understanding. In Proceedings of the Computer Vision and Pattern Recognition Conference (pp. 8545-8556).

[7] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., ... & Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 10012-10022).

[8] Li, L., Ma, R., Guo, Q., Xue, X., & Qiu, X. (2020). Bert-attack: Adversarial attack against bert using bert. arXiv preprint arXiv:2004.09984.

[9] Bolya, D., Fu, C. Y., Dai, X., Zhang, P., Feichtenhofer, C., & Hoffman, J. (2022). Token merging: Your vit but faster. arXiv preprint arXiv:2210.09461. 1

[10] Rao, Y., Liu, Z., Zhao, W., Zhou, J., & Lu, J. (2023). Dynamic spatial sparsification for efficient vision transformers and convolutional neural networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 45(9), 10883-10897.

[11] Beltagy, I., Peters, M. E., & Cohan, A. (2020). Longformer: The long-document transformer. arXiv preprint arXiv:2004.05150. 1

[12] Feichtenhofer, C., Fan, H., Xiong, B., Girshick, R., & He, K. (2021). A large-scale study on unsupervised spatiotemporal representation learning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 3299-3309). 1

[13] Bertasius, G., Wang, H., & Torresani, L. (2021, July). Is space-time attention all you need for video understanding?. In Icml (Vol. 2, No. 3, p. 4). 1