AdaCluster: Adaptive Query-Key Clustering for Sparse Attention in Video Generation

谭昊玥

导师：李诚、白有辉

中国科学技术大学

目录

DiT模型的兴起

❑DiT 的核心技术优势

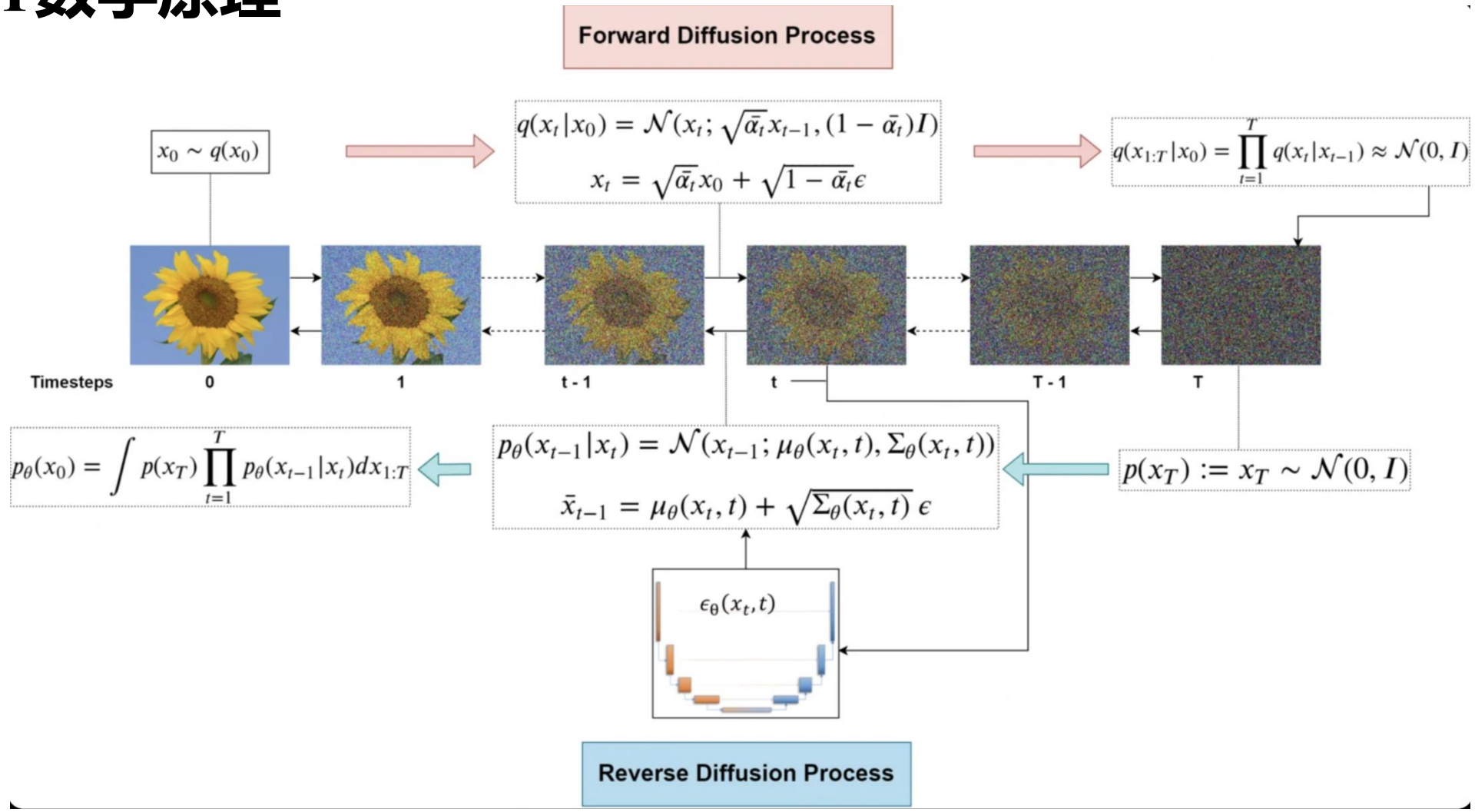
- ❖帧间时空一致性强，无闪烁 / 跳帧问题
- ❖支持高分辨率建模（480p/720p/1080p+）
- ❖适配长序列视频生成（81/128 帧及以上）
- ❖模型可扩展性佳，支持参数量灵活缩放
- ❖文本 - 视频跨模态对齐精度高

❑当前主流DiT 模型

- ❖HunyuanVideo,Wan-2.1等

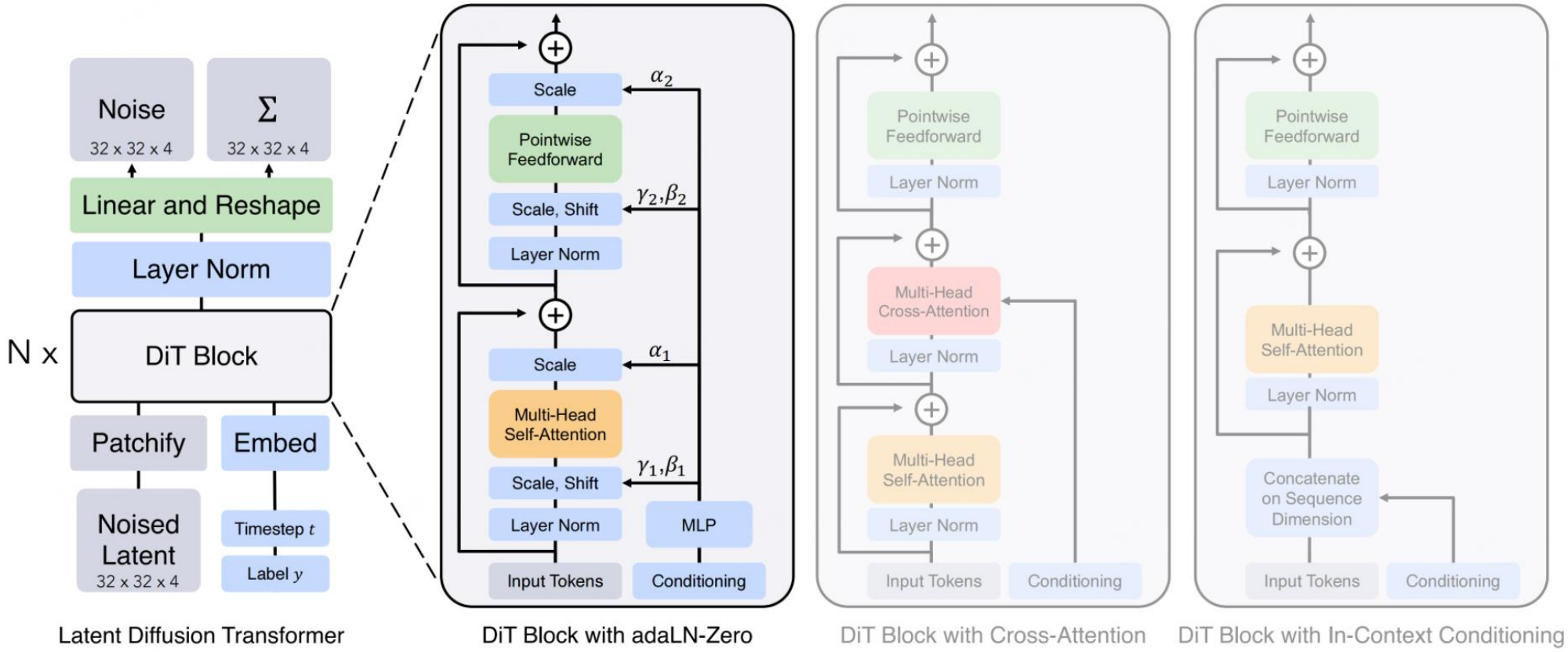
扩散模型回顾

❑DiT数学原理



扩散模型回顾

❑DiT宏观架构



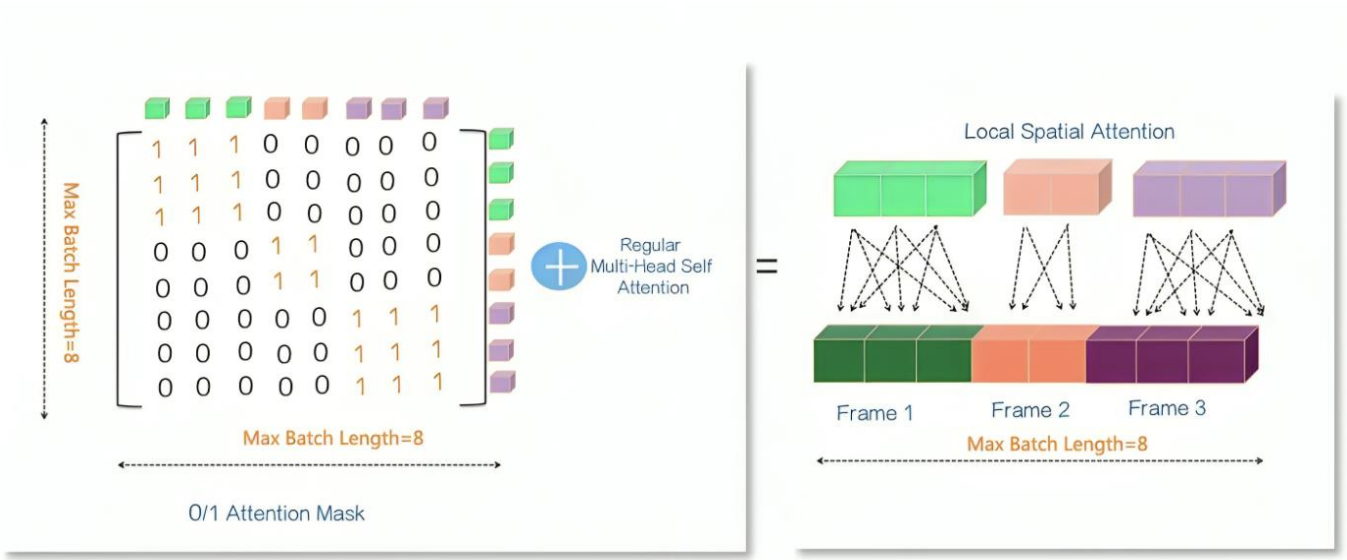
Latent Diffusion Transformer　　DiT Block with adaLN-Zero　　DiT Block with Cross-Attention　　DiT Block with In-Context Conditioning

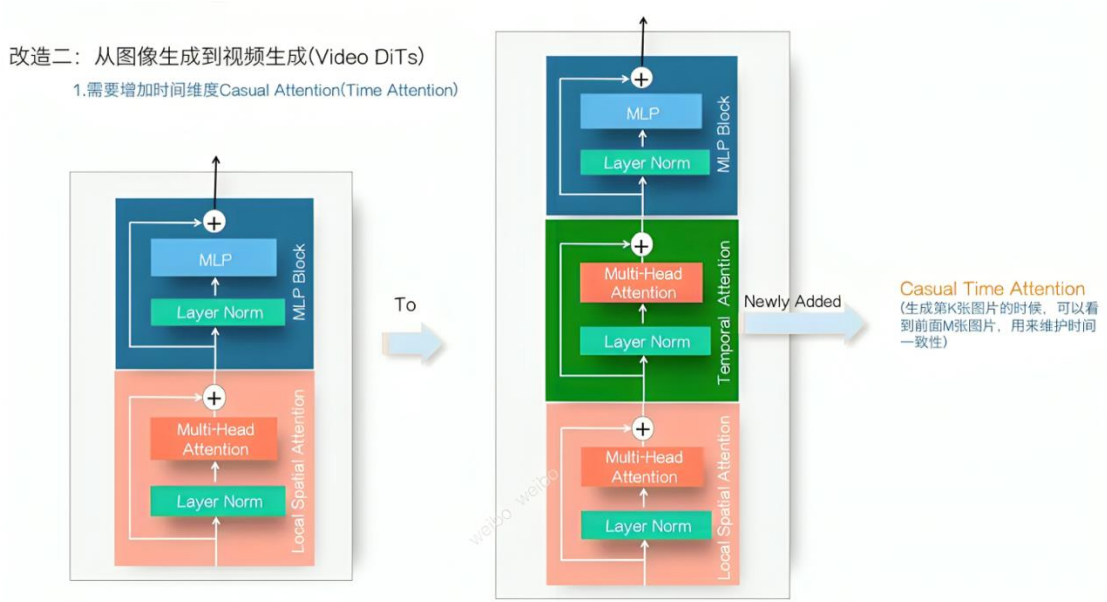
如何改造DiT实现视频生成

❑两项关键改造

❖属于某一帧图像的Patch无法看到其他帧的内容

❖为模型增加时间维度，将图像生成模型转变为视频生成模型



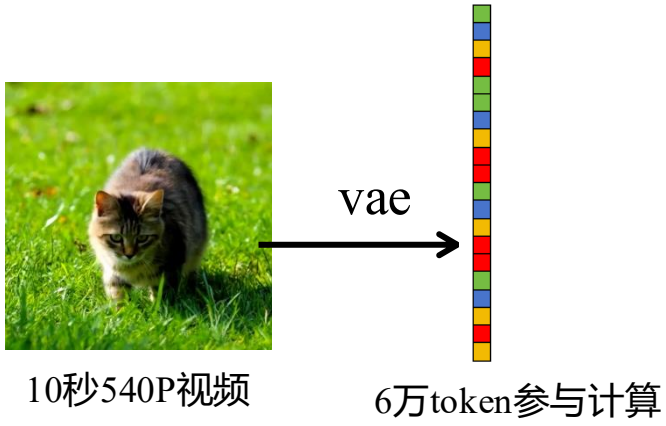


视频扩散模型的计算瓶颈

❑核心计算瓶颈

❖注意力计算复杂度：$O(N^2)$，随 Token 数呈二次增长

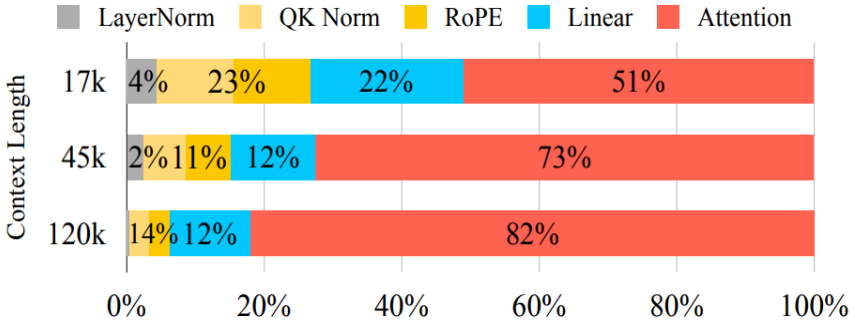
❖Token 数激增本质：Token数 = 帧数 × 分辨率(H×W) / 下采样率



10秒540P视频

6万token参与计算

视频生成任务中
的token数较多



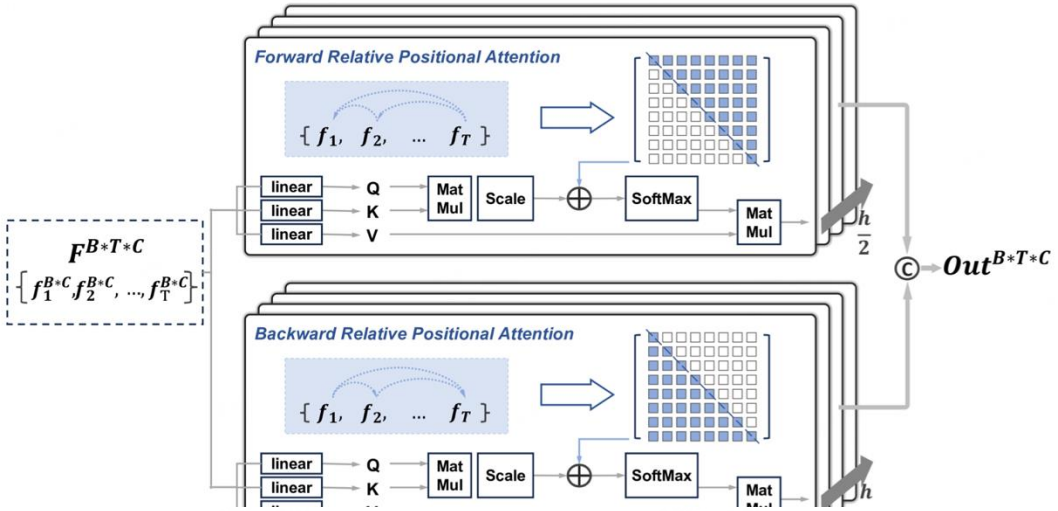
随着输入token变多
注意力占比高达82%

现有解决方案——稀疏注意力

❑核心逻辑： 既然全量计算attention太慢，那就只算重要的部分。

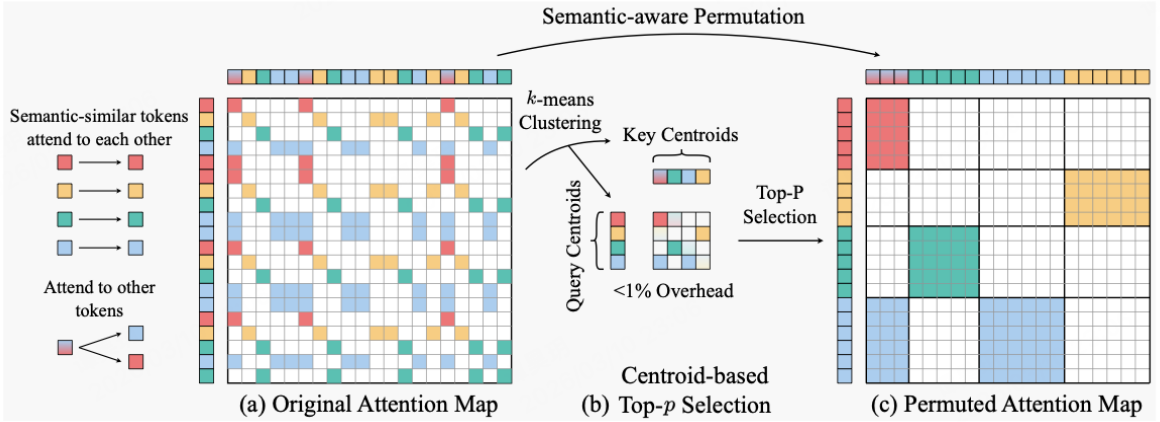
❖静态稀疏性

- ➢特点：提前设定好固定的注意力模式（如只看附近帧或特定区域）。
- ➢优点：简单快速。
- ➢缺点：缺乏通用性，容易漏掉关键信息。



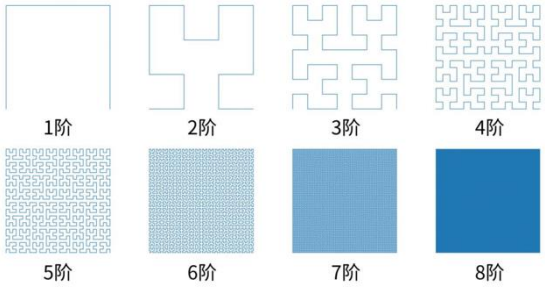
❖动态稀疏性：

- ➢特点：根据输入内容实时计算哪些token重要（如Top-K方法、聚类方法）。
- ➢代表工作：SpargeAttn, SVG2。
- ➢优点：灵活性高，理论上效果更好。

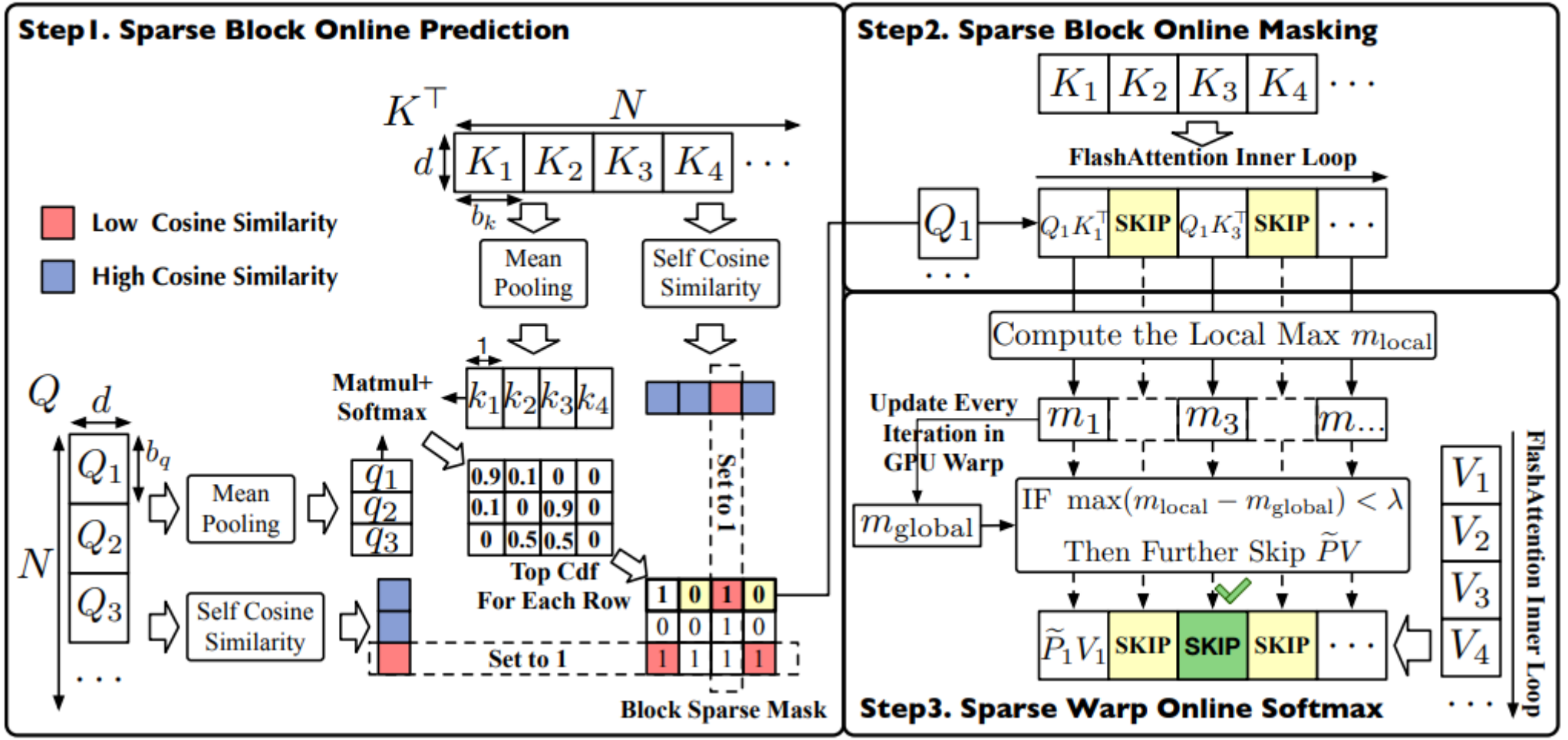


SpargeAttn: 通用且无需训练的稀疏注意力

- 目标：设计一个通用、准确且无需训练的稀疏注意力算子，能够加速各种模型（语言、图像、视频生成）的推理，同时不损失模型性能。
- 核心方法——两阶段在线过滤
  - 第一阶段：选择性Token压缩预测稀疏块
    - 利用相邻token高度相似的观察，将块内token压缩为单个代表token
    - 快速计算压缩注意力图，仅对累积贡献达阈值τ的块执行完整计算
    - 非自相似块（相似度<θ）强制保留，避免信息丢失
  - 第二阶段：稀疏Warp在线Softmax
    - 在FlashAttention的online softmax过程中检测可忽略的块
    - 若块内最大值与全局最大值的差距超过阈值λ，则跳过该块的 P~ijVjP~ijVj 计算
    - 零额外开销，进一步增加稀疏度



SpargeAttn 核心技术



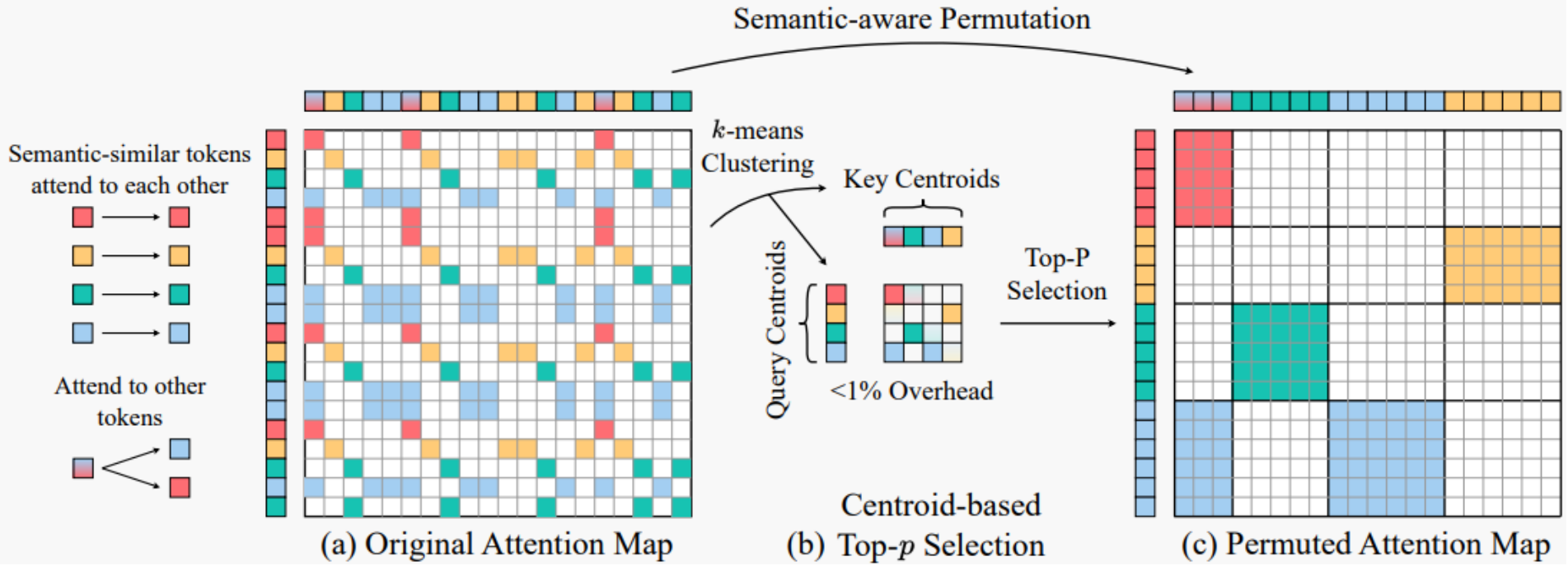
SVG2: Sparse VideoGen2 - 方法概述

- **SVG2 是一个无训练的稀疏注意力框架，旨在解决现有动态稀疏方法在视频生成中面临的两大核心挑战**
  - 关键 Token 识别不准确: 现有方法基于位置而非语义对 Token 进行分块聚类，导致聚合表示不准确。
  - 计算资源浪费: 关键 Token 分布零散，GPU 处理稀疏数据时会因填充而造成大量计算浪费。

SVG2 的核心解决方案: 语义感知置换 (Semantic-Aware Permutation)。

SVG2 核心技术

- 语义感知置换: 在识别关键 Token 前，对每一层、每个注意力头的 Q/K/V 向量执行 k-means 聚类，并将语义相似的 Token 重排在一起。
- 动态预算控制: 采用 Top-p 策略，使用聚类中心近似注意力分数，动态选择累计分数最高的前 p% 的簇作为关键簇。



SVG2 聚类

- 突破 “基于位置聚类” 的局限，按 Token 语义相似性聚类 + 重排
  - 1. 分层分 Head，独立处理
  - 2. 欧氏距离 k-means 聚类
  - 3. 簇内 Token 物理重排
  - 4. 注意力输出等价性证明

$$
\begin{aligned}
O' =& \pi_q^\top \text{Attention}(Q', K', V') = \pi_q^\top \text{softmax}\left(\frac{(\pi_q Q)(\pi_k K)^\top}{\sqrt{d}}\right)\pi_k V \\
=& (\pi_q^\top \pi_q)\text{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right)(\pi_k^\top \pi_k)V = \text{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right)V = O
\end{aligned}
$$

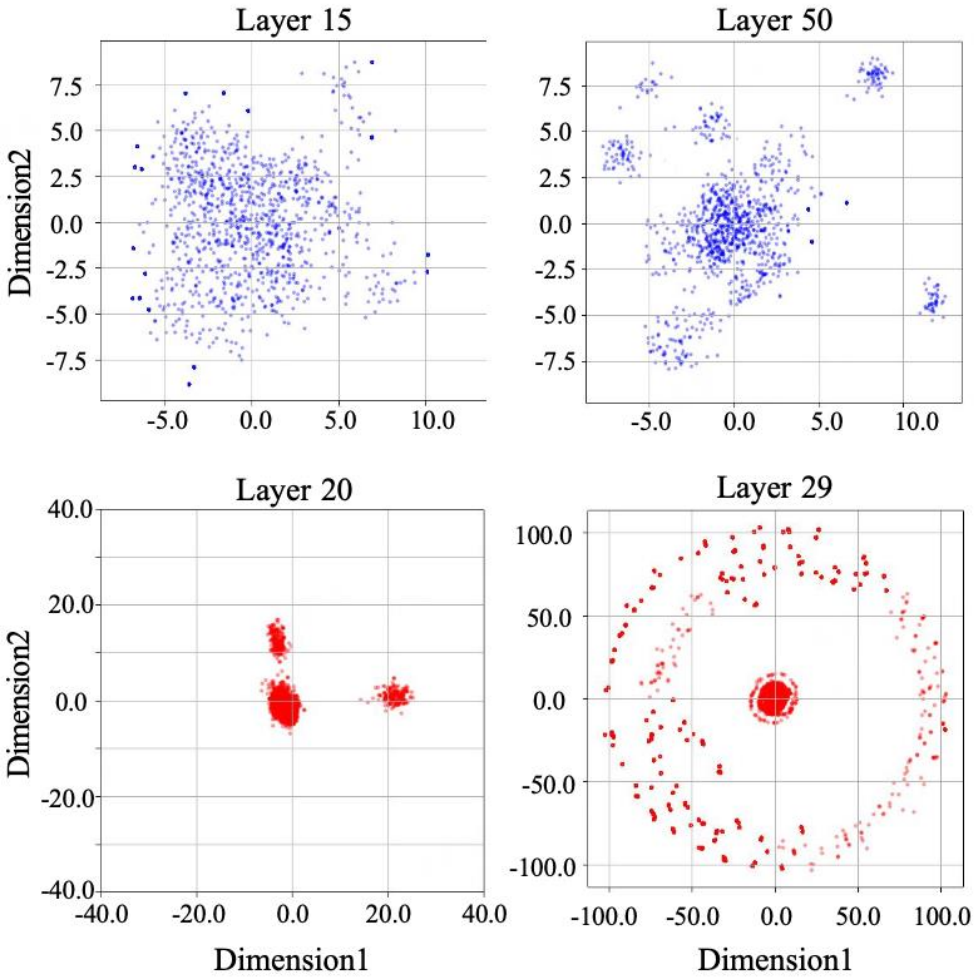
目前动态稀疏的两大局限

❑局限一：忽略向量分布差异。

- ❖对Query和Key向量一视同仁，忽略了Q和K的本质差异，使用完全相同的聚类策略。
- ❖固定簇数的聚类导致分散层压缩不足（信息丢失），集中层压缩效率低（计算浪费），无法在全局达到最优的精度-速度平衡。

❑局限二：聚类中心无法代表边界Token。

- ❖现有方法通过固定的kmeans聚类算出的聚类中心来判断一个簇是否重要。如果一个重要的token落在簇的边界（离中心远），仅仅因为簇中心得分低而被忽略，导致关键信息遗漏。



方法概述——AdaCluster整体框架

❑针对局限一：忽略向量分布差异。

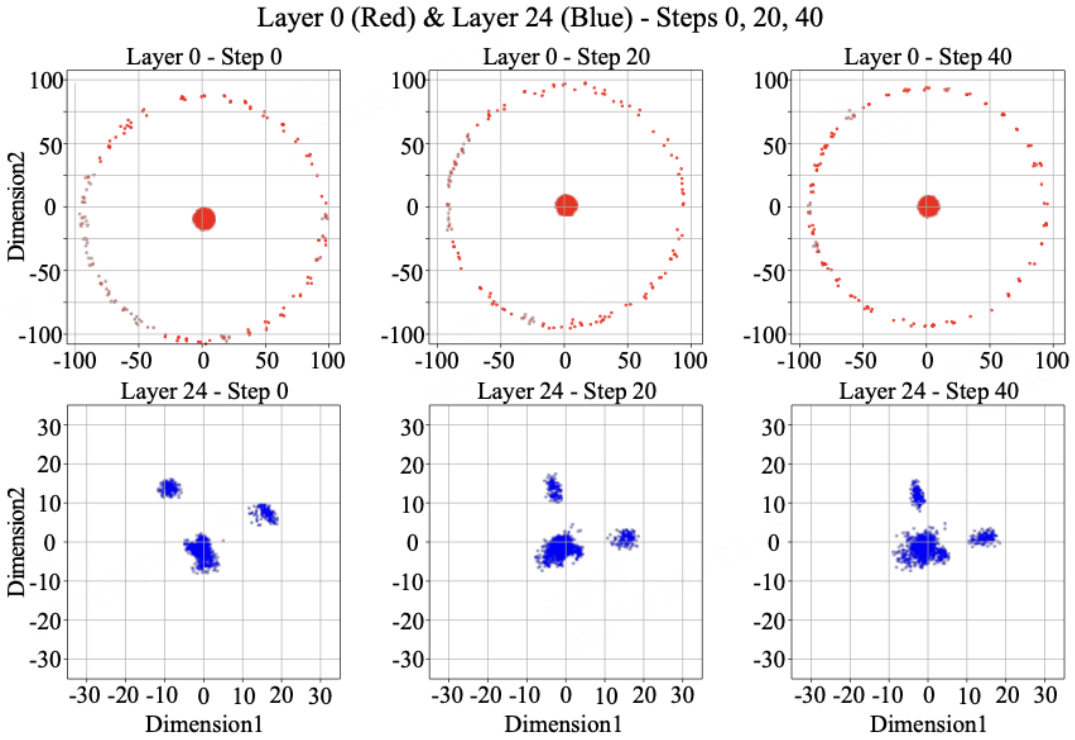
❖洞察： q的聚类只需要角度相似性，而k需要欧氏距离相似性。

❖解决方案： AdaCluster为q和k别设计聚类方法，在保证精度的同时最大化效率

❑针对局限二：聚类中心无法代表边界Token。

❖洞察： 不同的step/prompt在相同的layer之内，q、k的分布相似

❖解决方案： AdaCluster视频生成的第一个step 动态确定每层的聚类数，并利用相邻去噪步骤的分布相似性加速。



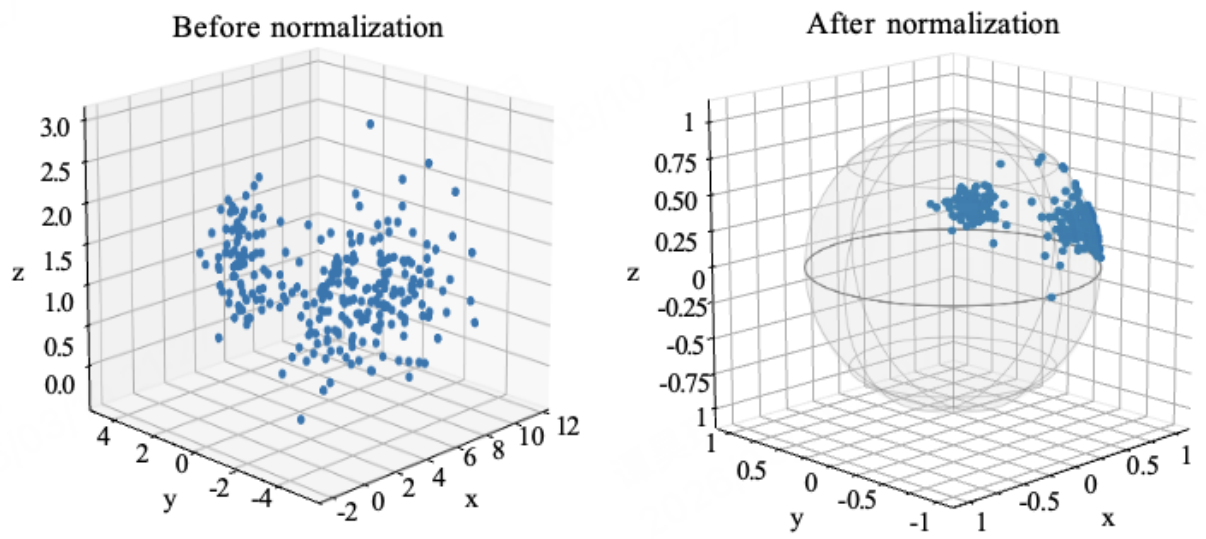
Query 聚类：基于角度相似性

- 动机： 原始Query向量长度差异大，直接聚类困难，压缩比有限。
- 关键观察： Query-Key分数的相对大小与向量的长度无关，只与方向有关。
- 设计策略：

  归一化 query 到单位球面，使用角度相似性聚类

- 效果：

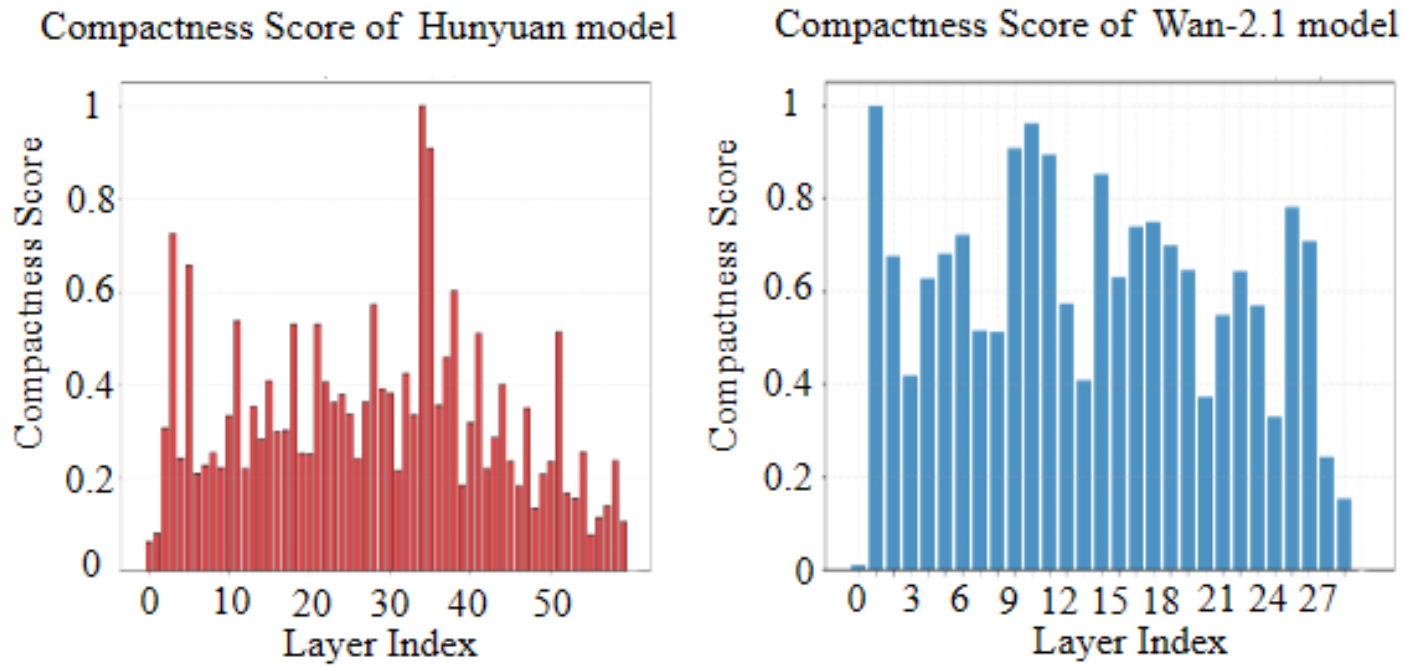
  压缩比提升 3.6 倍



Key 聚类：多层自适应策略

❑层级差异性分析：

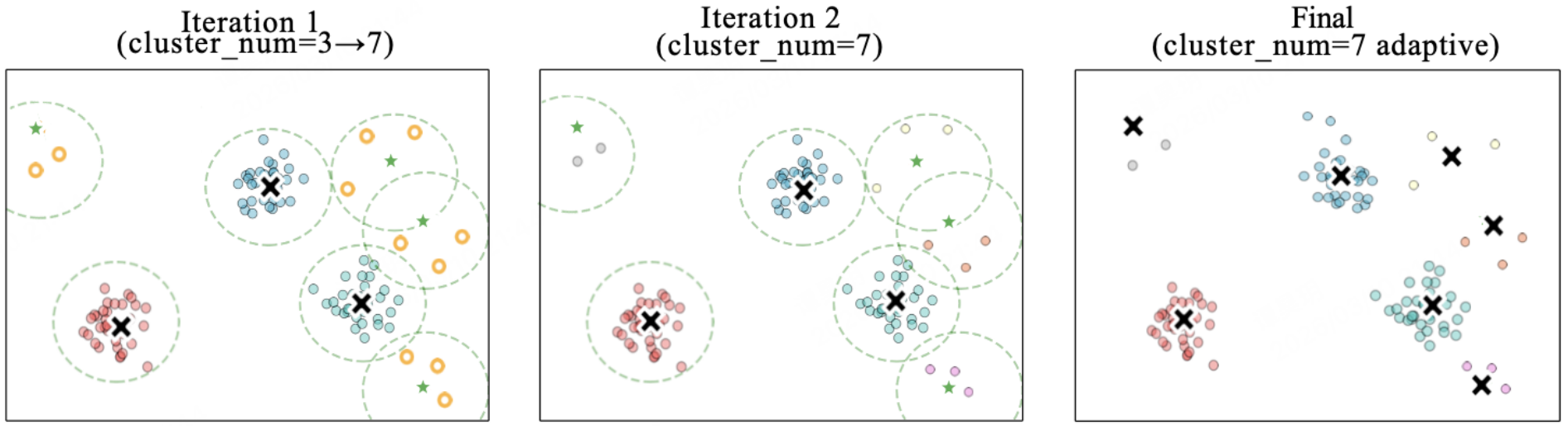
- ❖不同层的Key向量分布差异显著，需要欧氏聚类保证精度
- ❖聚类数量应自适应调整



Key 聚类：多阶段 K-means 算法

❑多阶段 K-means 算法：

- ❖用标准的kmeans进行进行初始聚类
- ❖找出距离簇中心超过阈值 τ 的token
- ❖对离群点用少量新簇再次聚类



TensorQuest: 高效关键簇选择

问题：如何快速识别重要的聚类?

- ❖问题：如果只看簇中心的attn weight，容易忽略簇内别的attn weight高的token。
- ❖灵感来源： Quest方法（用于LLM），通过估计注意力权重的上界来选出关键KV页。

提出TensorQuest

- ❖将查询和键拆分为正负部分（q^+, q^-, k^+, k^-）。
- ❖利用矩阵乘法在Tensor Core上高效计算Quest分数，避免低效的逐元素操作

AdaCluster总体流程

Algorithm 3 AdaCluster

Require: Query Q, Key K, Value V, distance threshold τ, Top-K parameter $topk$, max cluster count $N_{\max}$

Ensure: AttnOut

$A_q, C_q \leftarrow \text{KMeans}(\text{Normalize}(Q))$
$\mathcal{A}_k, \mathcal{C}_k, F \leftarrow \text{MultiStageClustering}(K, \tau, N_{\max})$
if F is True **then**
    $\text{AttnOut} \leftarrow \text{Attention}(Q, K, V)$
else
    $S \leftarrow \text{TensorQuest}(Q, K)$
    $\text{Index} \leftarrow \text{argtopk}(S, topk)$
    $K^* \leftarrow K[\text{Index}], V^* \leftarrow V[\text{Index}]$
    $\text{AttnOut} \leftarrow \text{Attention}(Q, K^*, V^*)$
end if
return AttnOut

实验设置与基本结果

❑测试模型

- ❖CogVideoX-2B
- ❖Wan-2.1-1.3B
- ❖HunyuanVideo

❑实验配置

- ❖推理步数： 30 步
- ❖硬件： NVIDIA A40 (48GB)
- ❖数据集： PenguinVideoBenchmark

❑评估指标

- ❖相似度： PSNR↑, SSIM↑, LPIPS↓
- ❖视频质量： Vbench指标
- ❖效率：端到端加速比

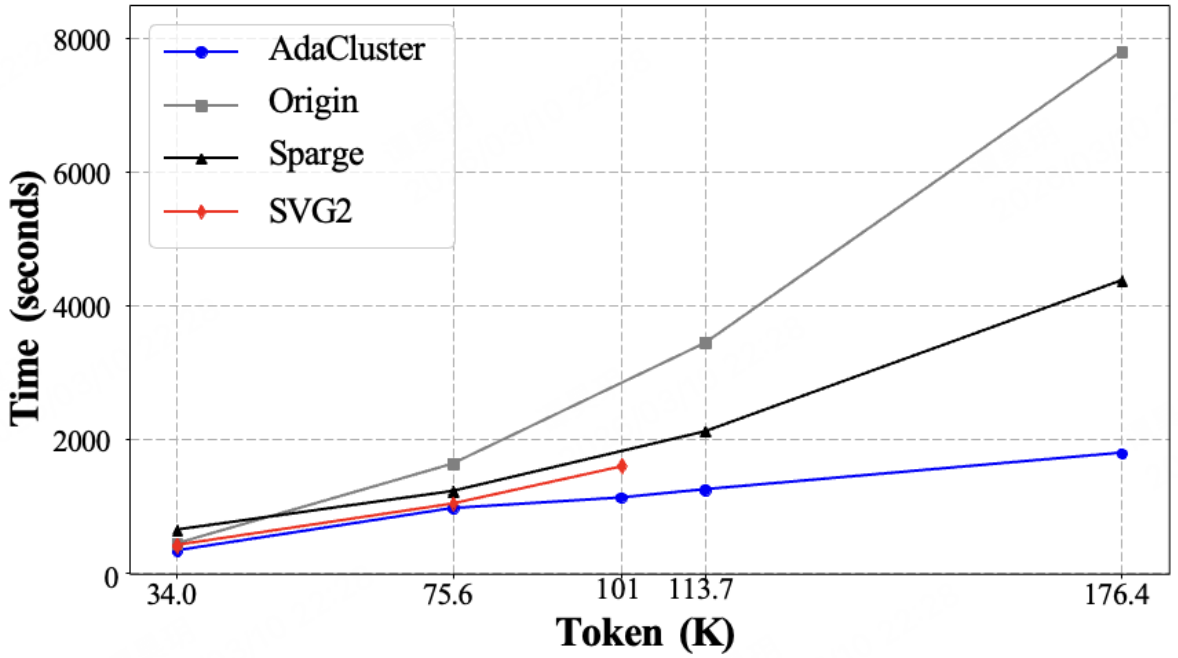
Model	Config	PSNR↑	SSIM↑	LPIPS↓	ImgeQual↑	BgConsis↑	SubConsis↑	TmpFlick↑	Speedup↑
CogVideoX-2B	720×480	-	-	-	61.15%	96.06%	92.71%	97.31%	1×
	SpargeAttn	28.189	0.517	0.618	55.77%	**95.56%**	**94.35%**	**98.14%**	1.23×
	AdaCluster	**30.989**	**0.767**	**0.231**	**56.76%**	93.95%	89.26%	95.18%	**1.67**×
Wan-2.1-1.3B	832×480	-	-	-	67.53%	96.72%	95.08%	98.04%	1×
	SpargeAttn	28.292	0.437	0.599	64.59%	**97.24%**	**96.29%**	**98.36%**	1.81×
	SVG2	28.230	0.358	0.679	66.43%	96.05%	94.36%	97.96%	1.61×
	AdaCluster	**29.083**	**0.571**	**0.393**	**66.45%**	96.35%	94.58%	97.84%	**1.85**×
HunyuanVideo	1280×720	-	-	-	62.64%	95.84%	93.05%	98.64%	1×
	SpargeAttn	28.155	0.490	0.596	61.76%	95.89%	**92.92%**	98.69%	1.33×
	SVG2	29.319	0.794	0.308	**65.42%**	**96.58%**	92.14%	98.73%	1.57×
	AdaCluster	**30.580**	**0.835**	**0.203**	65.11%	95.58%	92.56%	**98.85%**	**1.68**×

Full Attention | latency = 1639s

AdaCluster | latency = 973s | PSNR = 30.58dB

HunyuanVideo：可扩展性分析

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×



序列长度扩展

Token 数	Sparge	AdaCluster
34.0K	1.28×	**1.53×**
75.6K	1.33×	**1.68×**
101.1K	1.45×	**2.31×**
113.7K	1.57×	**3.12×**
176.4K	**1.78×**	**4.31×**

在序列长度达到176.4K token时，AdaCluster达到 4.31× 加速，是SpargeAttn的2.4倍。由于A40显存限制，SVG2所能达到的最大序列长度为101K。

视频生成效果

Prompt 1：海洋中的海豚

Prompt 2：低角度拍摄的飞雁

总结

❑主要贡献

- ❖Query聚类方法：提出基于角度相似性的归一化聚类
- ❖Key聚类方法：提出K的多阶段自适应 K-means 聚类
- ❖TensorQuest 优化：确保关键 token 不遗漏

❑未来工作方向

- ❖扩展到更多模型：支持更广泛的视频生成模型
- ❖扩展到更多型号的机器：使用不同型号的显卡运行AdaCluster，确保可用
- ❖开源发布：代码将开源https://github.com/USTC-MLSys-Team/Adacluster

感谢聆听！

Q & A