PAPER WALKTHROUGH

HubKV

从 Top-K 的局部冗余问题，到可并行的 Submodular Marginal Discounting

KV Cache Compression

Modular vs. Submodular Selection

Final: head_nms_gated_k5_p0.5_mild

先看问题：KV cache 为什么会成为瓶颈

长上下文推理时，模型需要保存历史 token 的 Key/Value。

这样生成下一个 token 会更快，但 cache 大小随上下文长度线性增长，显存和吞吐都会被拖住。

HubKV 的切入点：不重写 cache 系统，不重新训练 scorer，而是在 Top-K 之前修正分数。

核心概念：Modular vs. Submodular

Modular Top-K

$$F(S) = \Sigma\, s_i$$

每个 token 的价值固定，和已经选了什么无关。

问题：如果一片局部区域全都高分，Top-K 会连续保留很多相似 token。

Submodular Coverage

$$F(S) = \Sigma \max w_{i,j}$$

新增收益取决于当前集合；附近已有代表 token 时，邻居的边际收益下降。

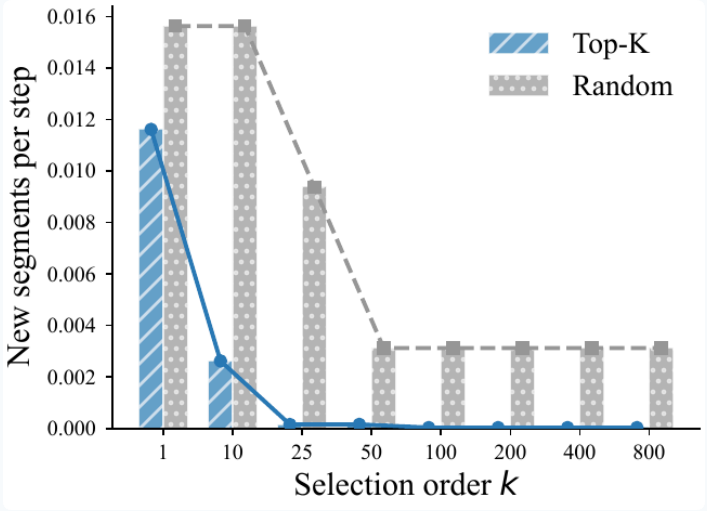
直觉：好的 cache 像摘要，要覆盖更多信息区域，而不是重复同一段。

> *HubKV 的论文主张：KV cache retention 应该从“捡最高分 token”转向“保留高分且覆盖互补的代表 token”。*

OBSERVATION

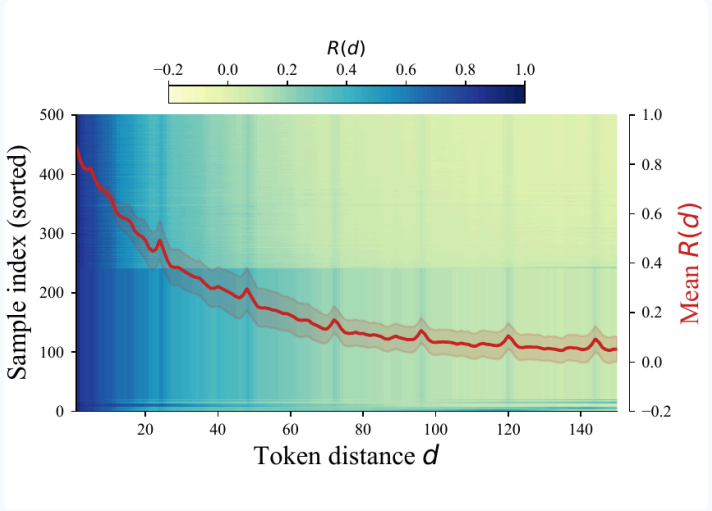
为什么 Top-K 会浪费预算

三条观察支撑论文动机



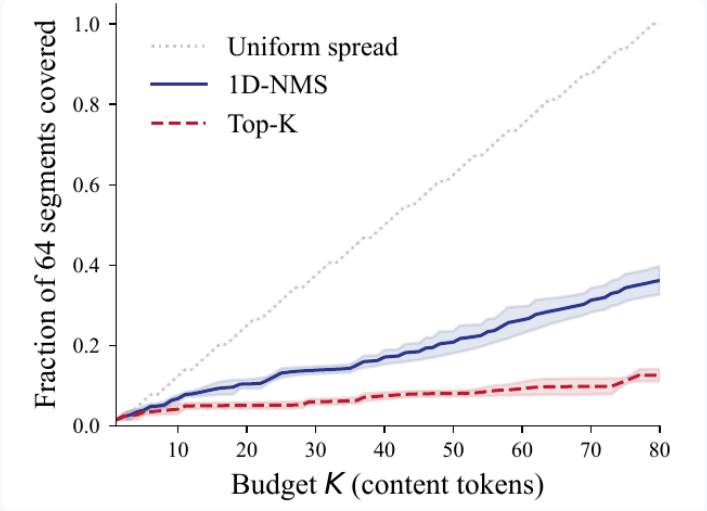
覆盖增益下降

Top-K 很快开始重复覆盖同一局部区域。



分数有局部相关

高分 token 往往成簇出现，不是孤立点。

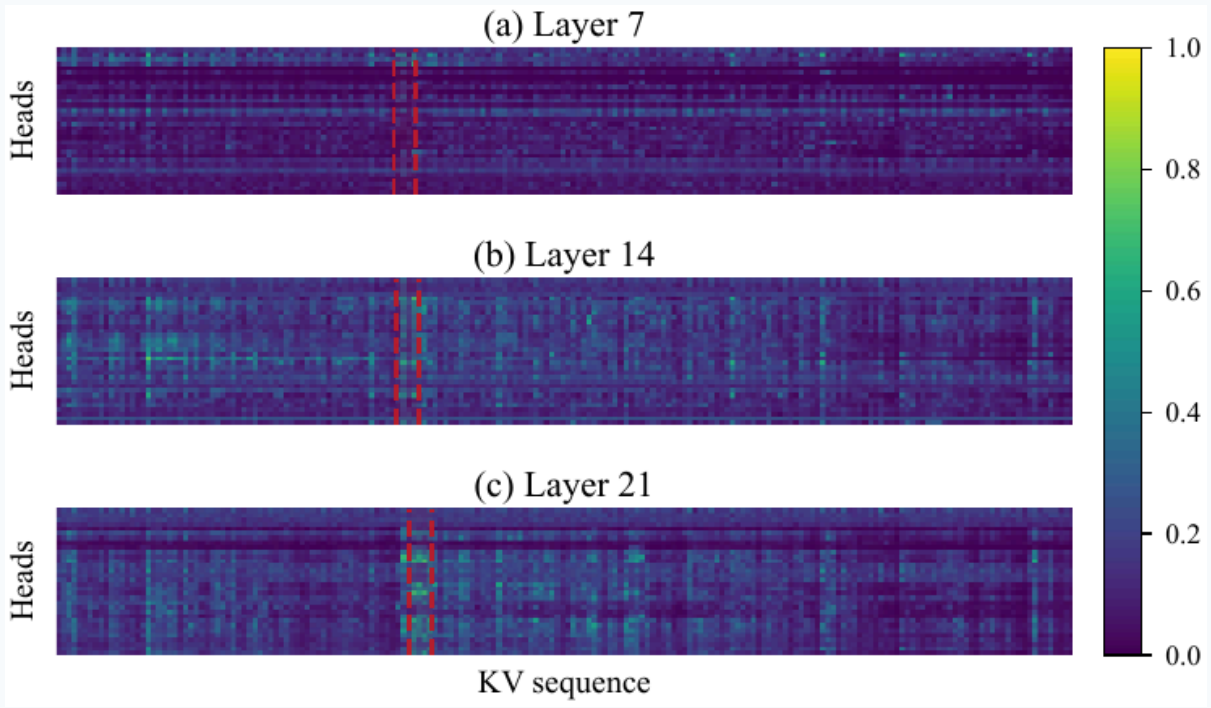


NMS 提升覆盖

局部抑制可以把预算推向更多 segment。

Figures from the current HubKV paper draft: empirical motivation panels.

不只是 token：head 也有不同功能

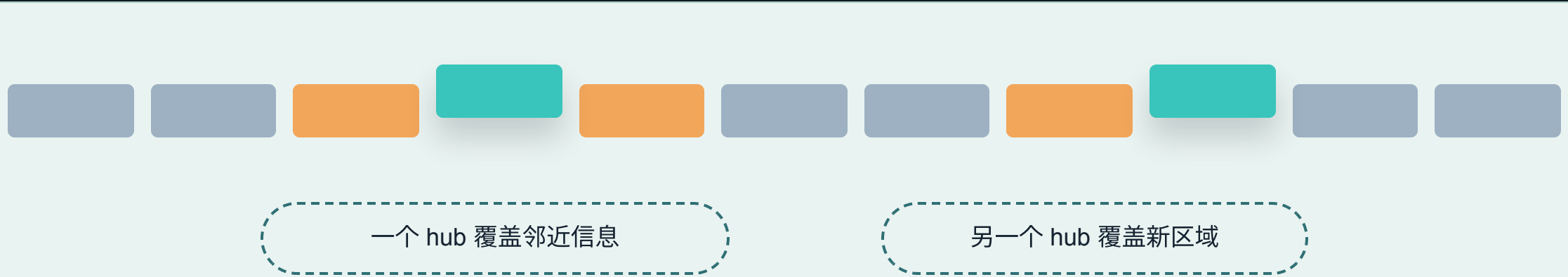


尖锐 head 像定位器，分数集中在少量 token。

平坦 head 像背景信号，对很多 token 给相似分数。

方法启发：不能只做 token-level NMS，还要做 bounded head-wise calibration。

理论桥梁：把 cache retention 看成局部覆盖



理论 oracle

Submodular greedy 会逐步选择边际收益最大的 token，但它是顺序过程，推理时太贵。

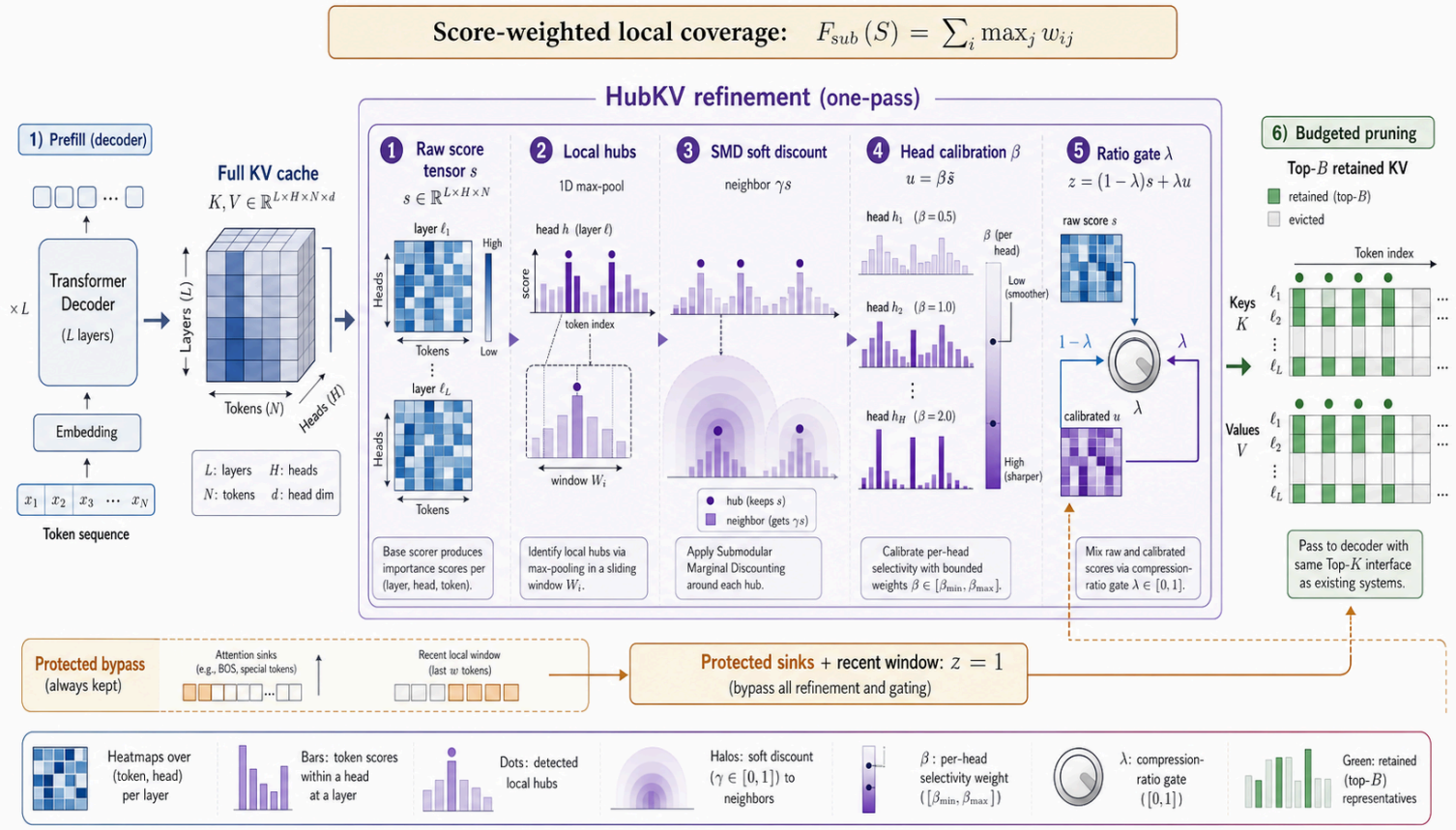
HubKV 选择

用一维局部 hub detection + neighbor discount 做 first-order proxy，保持并行 Top-K 接口。

METHOD

HubKV 怎么落地

HubKV 是一层 score refinement



Base scorer 负责产生分数；HubKV 只在排序前做局部冗余修正；最后仍交给 Top-K-style pruning。

最终算法：四步修正原始分数

1

Local hubs

用 kernel width 5 的 max-pooling 找局部峰值。

2

SMD discount

hub 保持原分；non-hub 邻居乘以 0.5。

3

Head calibration

按 selectivity 做温和权重，限制在 [0.8, 1.2]。

4

Ratio gate

用 α = compression_ratio² 控制修正强度。

$$z_i = s_i + r^2 \cdot (\beta_h \cdot q_i - s_i)$$

尝试路线：效果一般的可能原因

不要扩大局部簇

Avg-pool 平滑效果一般，可能因为它让高分区域更厚，和去冗余目标相反。

CommunitySmooth

不要过度重写 scorer

SVD、IB、固定 head weight 效果一般，可能因为它们会抹掉关键 anchor。

SVD / IB / Head NMS

不要引入顺序瓶颈

随机贪心更接近理论 oracle，但运行路径更重，整体收益不如当前方案。

Stochastic Greedy

> *最终方案之所以保留，是因为它目前效果最好；机制上，它保留 NMS 的有效内核，并用 mild calibration 与 ratio gate 控制扰动。*

整篇 paper 的 walkthrough

1. Bottleneck
KV cache 太大，必须删。

2. Flaw
Top-K 把 token utility 当成独立。

3. Evidence
高分 token 局部成簇，head 选择性不同。

4. Model
用 submodular local coverage 表达 diminishing returns。

5. Method
SMD + head calibration + ratio gate。

6. Claim
不是全局优化器，而是可并行的 marginal-gain proxy。

最终选择当前方案，直接原因是实验效果最好；理论部分提供的是理解它为何合理的解释框架。