

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

智能体如何在历史不断增长时持续工作

Graham Neubig

语言技术研究所



Context Management for Long-Context Agents

How agents keep working as their history grows

Carnegie Mellon University
Language Technologies Institute

Graham Neubig
Language Technologies Institute

长上下文智能体的上下文管理

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

智能体上下文增长



智能体上下文增长

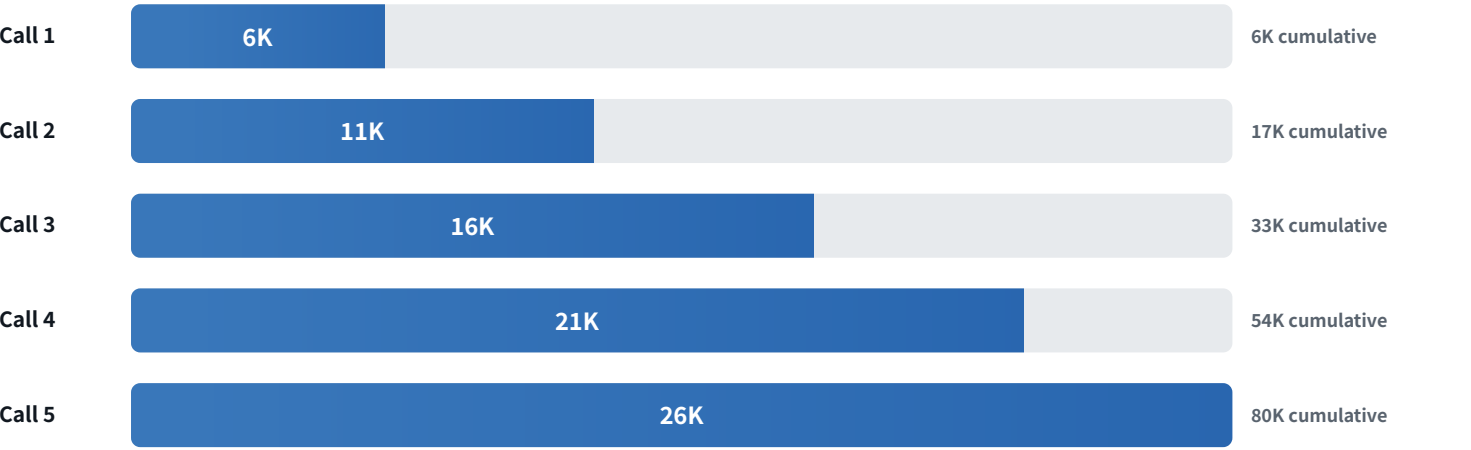
第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Quadratic input from linear history growth

$$\text{input}_t = \text{fixed prefix} + \text{history before call } t$$



3

线性历史增长带来二次方输入量

`input<sub>t</sub> = fixed prefix + history before call t`。

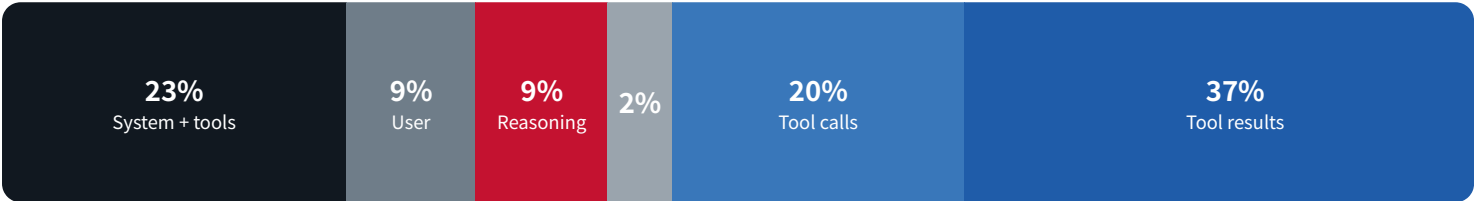
调用 1：6K；累计 6K。调用 2：11K；累计 17K。调用 3：16K；累计 33K。调用 4：21K；累计 54K。调用 5：26K；累计 80K。

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Agent prompt composition example



77,922 mean tokens per session across 1,500 OpenHands sessions

4

系统提示 + 工具 23%；用户 9%；推理 9%；工具调用 2%；工具结果 37%。

在 1,500 个 OpenHands 会话中，每个会话平均 77,922 个词元。

智能体提示的组成示例

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Two long-context challenges

Capacity

Can the model use the needed evidence?

Architecture, position encoding, and training shape usable context; evaluation measures it.

Efficiency

Can the system afford to serve it?

Attention, KV memory, caching, and compaction shape cost and latency.

5

容量：模型能否使用所需证据？可用上下文受架构、位置编码和训练影响；评估用于测量容量。

效率：系统能否以可承受的成本提供服务？注意力、KV 内存、缓存和精简会影响成本与延迟。

长上下文的两项挑战

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分



LLM 推理

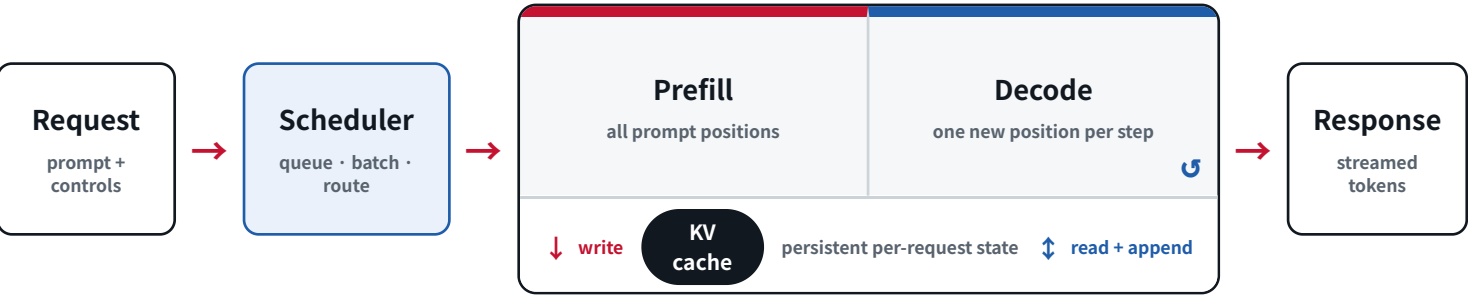
LLM 推理

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Elements of the serving stack



[DistServe · Zhong et al., OSDI 2024 ↗](#)

7

请求（提示 + 控制）→ 调度器（队列、批处理、路由）→ 预填充（处理全部提示位置）→ 逐词元解码（每步新增一个位置）→ 响应（流式词元）。

KV 缓存：写入并保存每个请求的持久状态；在解码时读取并追加。

DistServe · Zhong 等, OSDI 2024 ↗

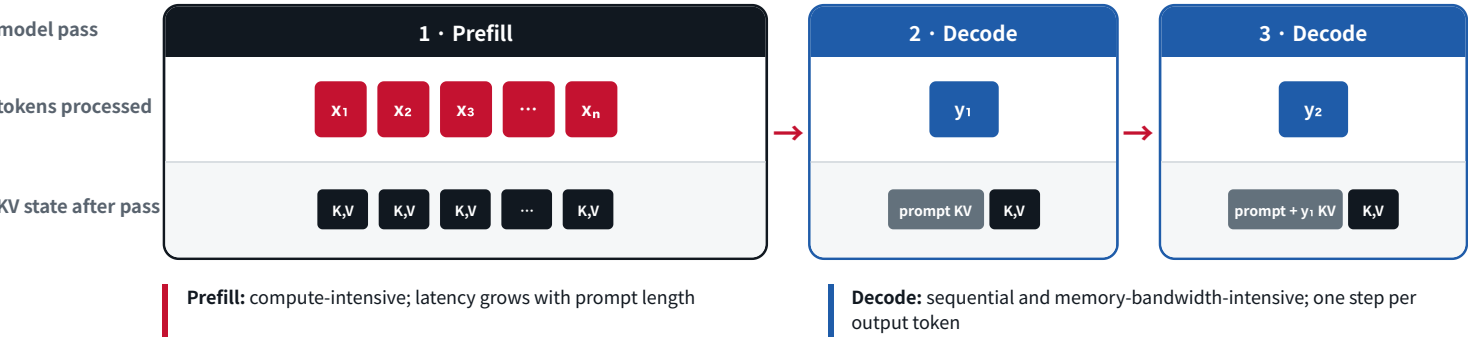
服务栈的组成部分

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Inference phases



[Prefill/decode execution model · DistServe, OSDI 2024 ↗](#)

模型处理的词元：预填充阶段处理 $x_1, x_2, x_3, \dots, x_n$ ；随后逐词元解码生成 y_1 、 y_2 。

每次处理后的 KV 状态：预填充为每个提示词元产生 K、V；解码时复用提示 KV 并为输出词元追加 K、V。

预填充：计算密集型；延迟随提示长度增长。

解码：顺序执行且受内存带宽限制；每个输出词元需要一步。

预填充 / 解码执行模型 · DistServe, OSDI 2024 ↗

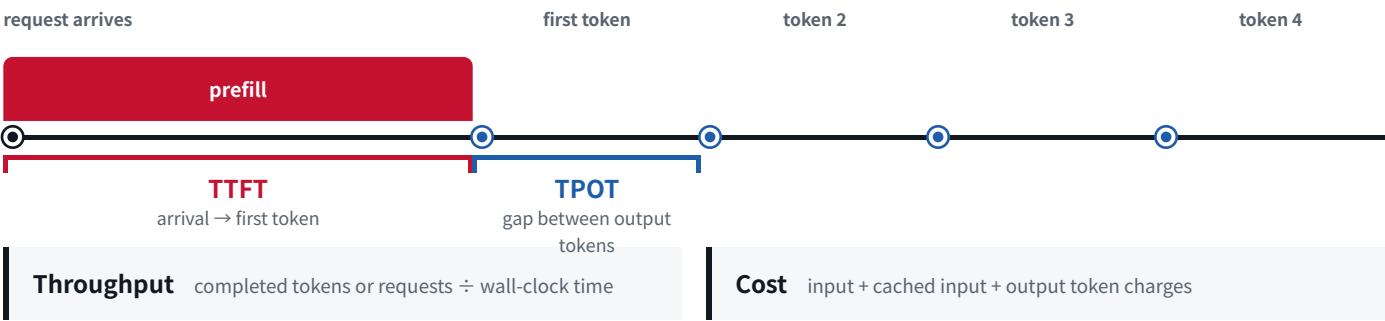
推理阶段

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Serving metrics



[TTFT and TPOT phase decomposition · DistServe, OSDI 2024 ↗](#)

请求到达后经过预填充，产生首个词元，再依次生成词元 2、3、4。

TTFT：从请求到达到首个词元的时间。TPOT：输出词元之间的间隔。

吞吐量：已完成的词元数或请求数 ÷ 墙钟时间。

成本：输入词元 + 缓存输入词元 + 输出词元的费用。

TTFT 与 TPOT 阶段分解 · DistServe, OSDI 2024 ↗

服务指标

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

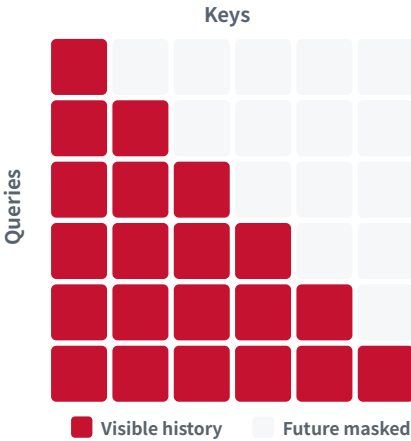
Refresher: attention

Linear projections

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V$$

Causal self-attention

$$A = \text{softmax}\left(\frac{QK^\top + M}{\sqrt{d_k}}\right), \quad O = AV$$



Notation: $X \in \mathbb{R}^{n \times d}$ has token-state row z_t ; lowercase q_i, k_j, v_j denote rows of Q, K, V ; d_k is key width; $M_{ij} = 0$ for $j \leq i$ and $-\infty$ otherwise.

回顾：注意力

线性投影：`Q = XW\_Q`、`K = XW\_K`、`V = XW\_V`。

因果自注意力：`A = softmax((QK^T + M) / sqrt(d\_k))`，`O = AV`。可见历史；未来被屏蔽。

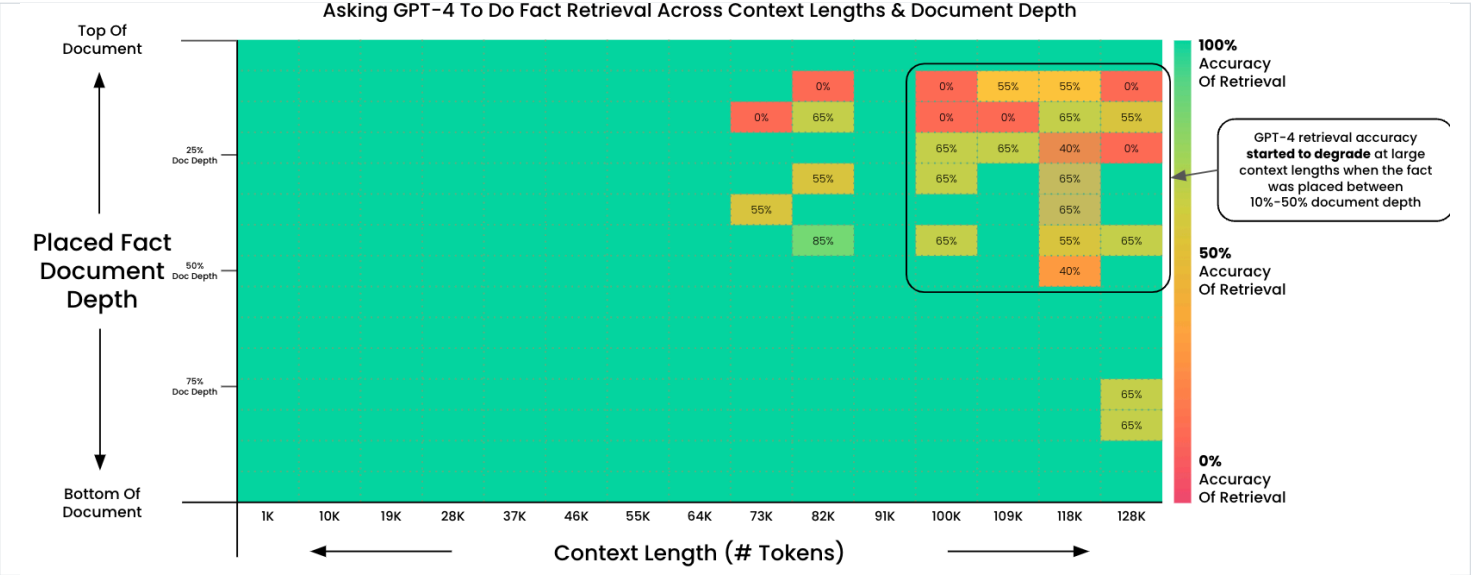
符号说明：`X ∈ ℝ^{n × d}`，其中第 t 行为词元状态 `z\_t`；小写 `q\_i`、`k\_j`、`v\_j` 分别表示 Q、K、V 的行；`d` 是键宽；当 `j ≤ i` 时 `M\_ij = 0`，否则 `M\_ij = -∞`。

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Advertised vs. effective context



GPT-4 needle-in-a-haystack evaluation · Kamradt, 2023 ↗

11

图表标题：要求 GPT-4 在不同上下文长度和文档深度下进行事实检索。横轴：上下文长度（词元数），1K、10K、19K、28K、37K、46K、55K、64K、73K、82K、91K、100K、109K、118K、128K。纵向位置：事实置于文档中的深度，从文档顶部到文档底部；标示 25%、50%、75% 文档深度。

颜色表示检索准确率：0%、50%、100%；图中标注的准确率包括 0%、40%、55%、65%、85%。

批注：当上下文很长，且事实置于文档深度的 10%–50% 之间时，GPT-4 的检索准确率开始下降。

GPT-4 大海捞针评估 · Kamradt, 2023 ↗

译注 / 与原文翻译分开

色块中的准确率数字较小，部分数字与具体格子的对应关系在原图中不够清晰；不据此重建完整矩阵。

标称上下文与有效上下文

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

长上下文架构



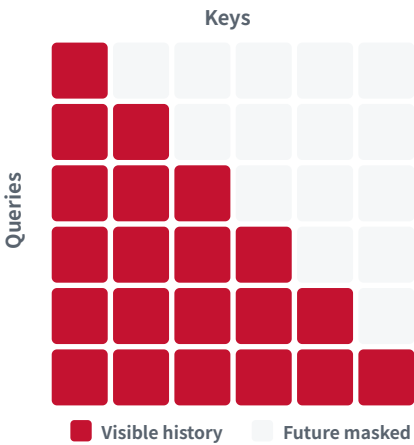
长上下文架构

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Standard attention is global



Typically, attention makes pairwise comparisons across the entire sequence.

标准注意力通常会在整个序列范围内进行成对比较。图示标注：键、查询、可见历史、未来屏蔽。

标准注意力是全局的

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Local modeling

Local computation only considers a constant subset of the other data.

Pairwise-comparison cost

$$\mathcal{O}(n^2) \longrightarrow \mathcal{O}(wn)$$

w is the local window size; for constant w, work is linear in n.

14

局部计算只考虑其他数据中一个大小固定的子集。

成对比较的计算成本：` $\mathcal{O}(n^2) \rightarrow \mathcal{O}(wn)$ `。

`w` 是局部窗口大小；当 `w` 为常数时，工作量随 `n` 线性增长。

局部建模

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Periodic global access in hybrid models



$N \times \text{local computation} \rightarrow 1 \times \text{global computation} \rightarrow \text{repeat}$

15

局部 1、局部 2、.....局部 N：低成本状态更新；全局：从完整历史中检索。

‘ $N \times \text{局部计算} \rightarrow 1 \times \text{全局计算} \rightarrow \text{重复}$ ’。

混合模型中的周期性全局访问

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Long-context hybrid architectures

Model	Local / stateful calculation	Global calculation	Local : global
Qwen3.8-Flash-Next	Gated DeltaNet	QSA sparse retrieval	36 : 12 $\approx$ 3:1
GLM-5.3-Flash	Kimi Delta Attention	DSA + IndexPool	34 : 11 $\approx$ 3:1
Kimi K3	Kimi Delta Attention	dense Gated MLA	69 : 24 $\approx$ 3:1
Nemotron 3 Ultra	Mamba-2	dense GQA	48 : 12 = 4:1
Inkling-Small	512-token window	dense GQA	35 : 7 = 5:1
DeepSeek-V4-Pro-0813	128-token window branch	compressed sparse / dense	interleaved branches

[Qwen ↗](#) [GLM ↗](#) [Kimi ↗](#) [Nemotron ↗](#) [Inkling ↗](#) [DeepSeek ↗](#)

模型 | 局部 / 有状态计算 | 全局计算 | 局部：全局

Qwen3.8-Flash-Next | Gated DeltaNet | QSA 稀疏检索 | 36:12 $\approx$ 3:1。

GLM-5.3-Flash | Kimi Delta Attention | DSA + IndexPool | 34:11 $\approx$ 3:1。

Kimi K3 | Kimi Delta Attention | 稠密 Gated MLA | 69:24 $\approx$ 3:1。

Nemotron 3 Ultra | Mamba-2 | 稠密 GQA | 48:12 = 4:1。

Inkling-Small | 512 词元窗口 | 稠密 GQA | 35:7 = 5:1。

DeepSeek-V4-Pro-0813 | 128 词元窗口分支 | 压缩稀疏 / 稠密；交错分支。

来源链接：Qwen、GLM、Kimi、Nemotron、Inkling、DeepSeek。

长上下文混合架构

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Four recurring mechanisms

- Sliding-window attention
- Linear attention or recurrent models
- Sparse attention
- KV compression

17

滑动窗口注意力。

线性注意力或循环模型。

稀疏注意力。

KV 压缩。

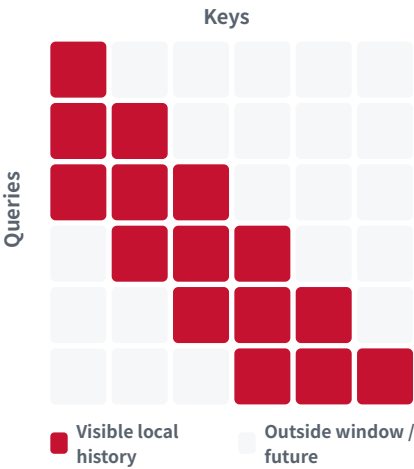
四种反复出现的机制

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Sliding-window attention



New variable: w is the number of causal positions retained by each query.

Causal window mask

$$M_{ij}^{(w)} = \begin{cases} 0, & 0 \leq i - j < w \\ -\infty, & \text{otherwise} \end{cases}$$

Using the same attention equation, work falls from $O(n^2)$ to $O(nw)$.

A local layer cannot directly retrieve distant tokens.

Longformer · Beltagy et al., 2020 ↗

因果窗口掩码：`M<sub>ij</sub>(w) = 0`，当 `0 ≤ i-j < w`；否则为 `-∞`。

使用相同的注意力公式，工作量从 `O(n²)` 降为 `O(nw)`。

局部层无法直接检索远处词元。

每个查询保留的因果位置数用 `w` 表示。图示：可见局部历史；窗口外 / 未来。

Longformer · Beltagy 等, 2020 ↗

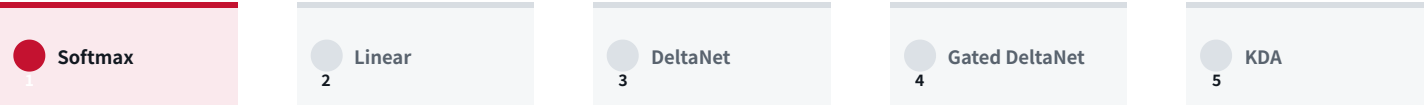
滑动窗口注意力

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Softmax attention



Past representation

Growing KV cache

Retain every prior k_i, v_i ; compare q_t with every k_i .

explicit token-level access · $O(t)$ decode comparisons

Normalized causal score

$$a_{ti} = \frac{\exp(q_t^\top k_i / \sqrt{d_k})}{\sum_{j=1}^t \exp(q_t^\top k_j / \sqrt{d_k})}$$

Read from explicit values

$$o_t = \sum_{i=1}^t a_{ti} v_i$$

New variable: a_{ti} is the normalized causal attention weight from query t to key i .

Attention Is All You Need · Vaswani et al., NeurIPS 2017 ↗

方法演进：1 Softmax → 2 Linear → 3 DeltaNet → 4 Gated DeltaNet → 5 KDA。

Softmax 注意力保留所有先前的 k_i, v_i ，并将 q_t 与每个 k_i 比较；显式访问词元级历史，解码第 t 步需要 $O(t)$ 次比较。

归一化因果分数： $a_{ti} = \exp(q_t^\top k_i / \sqrt{d_k}) / \sum_{j=1}^t \exp(q_t^\top k_j / \sqrt{d_k})$ ；显式读取值： $o_t = \sum_{i=1}^t a_{ti} v_i$ 。

新变量： a_{ti} 是查询 t 到键 i 的归一化因果注意力权重。

Attention Is All You Need · Vaswani 等, NeurIPS 2017 ↗

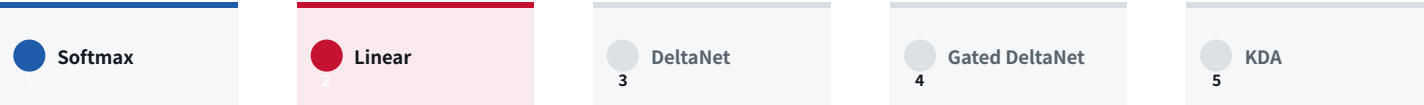
Softmax 注意力

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Linear attention



1 · Start from standard attention

$$o_t = \sum_{i=1}^t \text{softmax}_i \left(q_t^\top k_i / \sqrt{d_k} \right) v_i$$

2 · Replace normalization with a bilinear score

$$\text{softmax}_i \left(q_t^\top k_i / \sqrt{d_k} \right) \longrightarrow q_t^\top k_i$$

3 · Reassociate, then update recurrently

$$o_t = \sum_{i=1}^t (q_t^\top k_i) v_i = \left(\sum_{i=1}^t k_i v_i^\top \right)^\top q_t = S_t^\top q_t, \\ S_t = S_{t-1} + k_t v_t^\top.$$

Past representation

Fixed matrix state

$S_t^\top$ stores an online mapping from keys to values.

additive writes · interference is not explicitly erased

Key step: replacing query-normalized softmax weights with bilinear scores lets us reorder the sum. $S_t \in \mathbb{R}^{d_k \times d_v}$ stores that fixed-size recurrent memory.
Transformers are RNNs · Katharopoulos et al., ICML 2020 ↗

20

方法演进：1 Softmax → 2 Linear → 3 DeltaNet → 4 Gated DeltaNet → 5 KDA。

- 1 · 从标准注意力出发：` $o_t = \sum_{i=1}^t \text{softmax}_i(q_t^\top k_i / \sqrt{d_k}) v_i$ `。
- 2 · 将归一化替换为双线性分数：` $\text{softmax}_i(q_t^\top k_i / \sqrt{d_k}) \rightarrow q_t^\top k_i$ `。
- 3 · 重新结合，再递归更新：` $o_t = \sum_{i=1}^t (q_t^\top k_i) v_i = (\sum_{i=1}^t k_i v_i^\top) q_t = S_t^\top q_t$ `；` $S_t = S_{t-1} + k_t v_t^\top$ `。

固定矩阵状态 ` $S_t^\top$ ` 保存键到值的在线映射；通过加法写入；干扰不会被显式擦除。

关键步骤：将查询归一化的 softmax 权重替换为双线性分数后，可以重新排列求和。` $S \in \mathbb{R}^{d_k \times d_v}$ ` 保存固定大小的循环记忆。

Transformers are RNNs · Katharopoulos 等, ICML 2020 ↗

线性注意力

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

DeltaNet

● Softmax

● Linear

● DeltaNet

● Gated DeltaNet

● KDA

Memory operation

Targeted correction

Read the current association, then write only its prediction error.

correct along k_t · unrelated memory persists

Prediction and error

$$\hat{v}_t = S_{t-1}^\top k_t, \quad e_t = v_t - \hat{v}_t$$

Delta-rule write

$$S_t = S_{t-1} + \beta_t k_t e_t^\top, \quad o_t = S_t^\top q_t$$

New variables:

$\hat{v}_t$ is the current memory prediction; e_t is its error; $\beta_t \in [0,1]$ is the correction strength.

Linear Transformers Are Secretly Fast Weight Programmers · Schlag et al., ICML 2021 ↗

21

方法演进：1 Softmax → 2 Linear → 3 DeltaNet → 4 Gated DeltaNet → 5 KDA。

记忆操作：读取当前关联，然后只写入其预测误差，从而进行有针对性的修正。

预测与误差：` $\hat{v}_t = S_{t-1}^\top k_t$ `；` $e_t = v_t - \hat{v}_t$ `。

Delta 规则写入：` $S_t = S_{t-1} + \beta_t k_t e_t^\top$ `；输出：` $o_t = S_t^\top q_t$ `。

沿着 ` $k_t$ ` 纠正；无关记忆继续保留。新变量：` $\hat{v}_t$ ` 是当前记忆预测；` $e_t$ ` 是其误差；` $\beta_t \in [0,1]$ ` 是修正强度。

Linear Transformers Are Secretly Fast Weight Programmers · Schlag 等, ICML 2021 ↗

DeltaNet

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Gated DeltaNet

● Softmax

● Linear

● DeltaNet

● Gated DeltaNet

● KDA

New operation

Scalar forgetting

Decay the entire memory before applying the same targeted correction.

α_t: global retention · β_t: targeted edit

Whole-state decay

$$\tilde{S}_{t-1} = \alpha_t S_{t-1}$$

Correction after decay

$$S_t = \tilde{S}_{t-1} + \beta_t k_t \left(v_t - \tilde{S}_{t-1}^\top k_t \right)^\top$$

New variables: α_t ∈ [0,1] is a scalar retention gate; S_t[~] is the decayed memory. Setting α_t = 1 recovers DeltaNet.

Gated Delta Networks · Yang et al., 2024 ↗

方法演进：1 Softmax → 2 Linear → 3 DeltaNet → 4 Gated DeltaNet → 5 KDA。

新操作：标量遗忘。

整体状态衰减：`S<sub>t-1</sub> = α<sub>t</sub> S<sub>t-1</sub>`；衰减后再执行相同的定向修正。

衰减后的修正：`S<sub>t</sub> = S<sub>t-1</sub> + β<sub>t</sub>k<sub>t</sub>(v<sub>t</sub> - S<sub>t-1</sub>⊤k<sub>t</sub>)`。

`α<sub>t</sub>`：全局保留；`β<sub>t</sub>`：定向编辑。新变量：`α ∈ [0,1]` 是标量保留门；`S<sub>t-1</sub>` 是衰减后的记忆。设 `α<sub>t</sub> = 1` 时可还原 DeltaNet。

Gated Delta Networks · Yang 等, 2024 ↗

Gated DeltaNet

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Kimi Delta Attention

● Softmax

● Linear

● DeltaNet

● Gated DeltaNet

● KDA

New operation

Channelwise forgetting

Give each key-space feature its own memory lifetime.

fine-grained retention · same targeted correction

Per-channel decay

$$D_t = \text{Diag}(\boldsymbol{\alpha}_t), \quad \tilde{S}_{t-1} = D_t S_{t-1}$$

KDA recurrence

$$S_t = (I - \beta_t \mathbf{k}_t \mathbf{k}_t^\top) D_t S_{t-1} + \beta_t \mathbf{k}_t \mathbf{v}_t^\top$$

New variables:

$\alpha_t \in [0,1]^{d_k}$ is the vector retention gate; D_t is its diagonal matrix; I is the $d_k \times d_k$ identity. Equal retention in every channel recovers Gated DeltaNet.

Kimi Linear / KDA · Kimi Team, 2025 ↗

23

方法演进：1 Softmax → 2 Linear → 3 DeltaNet → 4 Gated DeltaNet → 5 KDA。

新操作：按通道遗忘，让键空间的每个特征拥有自己的记忆寿命；细粒度保留，同时进行相同的定向修正。

按通道衰减：`D\_t = Diag(alpha\_t)`；`S\_{t-1} = D\_t S\_{t-1}`。

KDA 递推：`S\_t = (I - beta\_t k\_t k\_t^T) D\_t S\_{t-1} + beta\_t k\_t v\_t^T`。

新变量：`alpha in [0,1]^{d^k}` 是向量保留门；`D` 是其对角矩阵；`I` 是 `d^k x d^k` 单位矩阵。所有通道保留率相同则还原 Gated DeltaNet。

Kimi Linear / KDA · Kimi Team, 2025 ↗

Kimi Delta Attention

个人学习译文 · 非 CMU 官方译本 · 译注独立标记 · 原始课件来自 cmu-agents.com

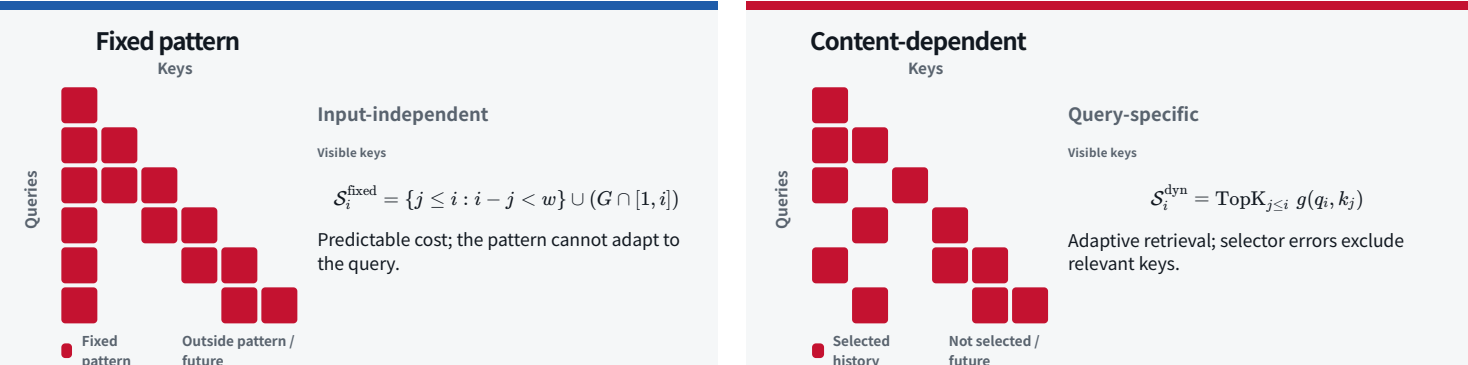
双语 PDF 23 | 原页 23

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Sparse attention



Both renormalize over the retained set

$$A_{ij} = \text{softmax}_{j \in \mathcal{S}_i} \left(\frac{q_i^\top k_j}{\sqrt{d_k}} \right), \quad o_i = \sum_{j \in \mathcal{S}_i} A_{ij} v_j$$

New variables: G is a fixed set of global positions; g is a learned selector; $|\mathcal{S}_i^{\text{dyn}}| = K$.

Fixed local + global pattern · Longformer, 2020 ↗ Content-dependent top-K · DeepSeek Sparse Attention, 2025 ↗

固定模式：与输入无关、可预测成本，但无法针对查询调整。 $\mathcal{S}_i^{\text{fixed}} = \{j \leq i : i - j < w\} \cup (G \cap [1, i])$ 。

依赖内容的模式：针对具体查询选择可见键，自适应检索，但选择器错误会排除相关键。 $\mathcal{S}_i^{\text{dyn}} = \text{TopK}_{j \leq i} g(q_i, k_j)$ 。

二者都只在保留集合上重新归一化： $A_{ij} = \text{softmax}_{j \in \mathcal{S}_i} (q_i^\top k_j / \sqrt{d_k})$ ， $o_i = \sum_{j \in \mathcal{S}_i} A_{ij} v_j$ 。

新变量： G 是固定的全局位置集合； g 是学习得到的选择器； $|\mathcal{S}_i^{\text{dyn}}| = K$ 。

固定局部 + 全局模式 · Longformer, 2020 ↗；依赖内容的 Top-K · DeepSeek Sparse Attention, 2025 ↗

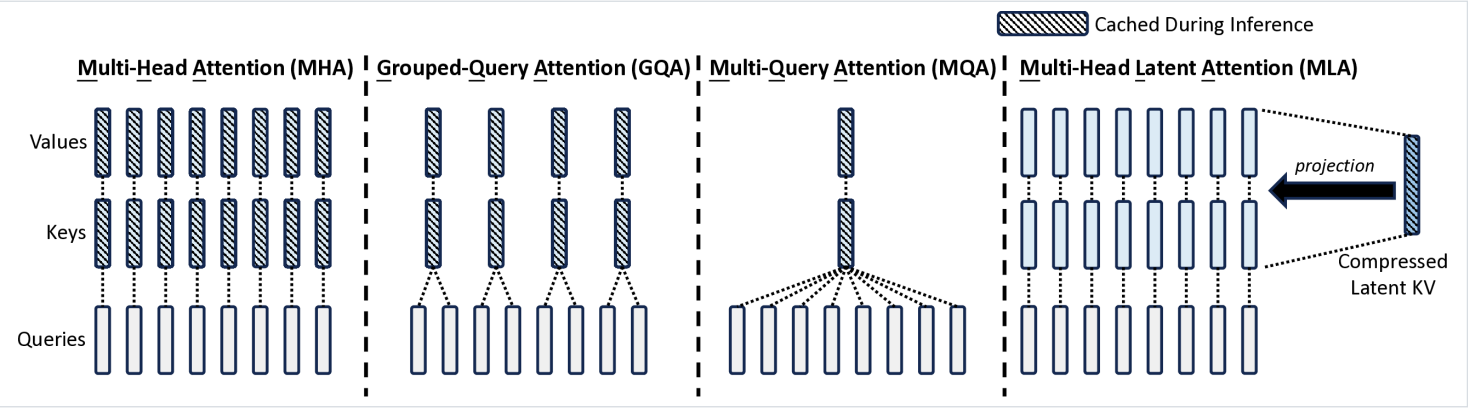
稀疏注意力

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

KV compression



MLA caches the latent and reconstructs head-specific keys and values

$$c_t^{KV} = W_D^{KV} z_t, \quad k_t^{(h)} = W_{UK}^{(h)} c_t^{KV}, \quad v_t^{(h)} = W_{UV}^{(h)} c_t^{KV}$$

New variables: c_t^{KV} is the cached latent; h indexes heads; W^p compresses and W^{uk} , W^{uv} recover per-head K,V.

GQA · Ainslie et al., 2023 ↗ Attention comparison and MLA · DeepSeek-V2, 2024 ↗

MLA 缓存潜变量，并重建各注意力头专属的键和值。 $c_t^{KV} = W^{DKV} z_t$ ； $k_t^{(h)} = W^{UK(h)} c_t^{KV}$ ； $v_t^{(h)} = W^{UV(h)} c_t^{KV}$ 。

新变量： c_t^{KV} 是缓存的潜变量； h 为注意力头索引； W^D 用于压缩， W^{UK} 和 W^{UV} 用于恢复每个头的 K、V。

GQA · Ainslie 等, 2023 ↗；注意力比较与 MLA · DeepSeek-V2, 2024 ↗

KV 压缩

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

训练长上下文模型

Training long-context models

26

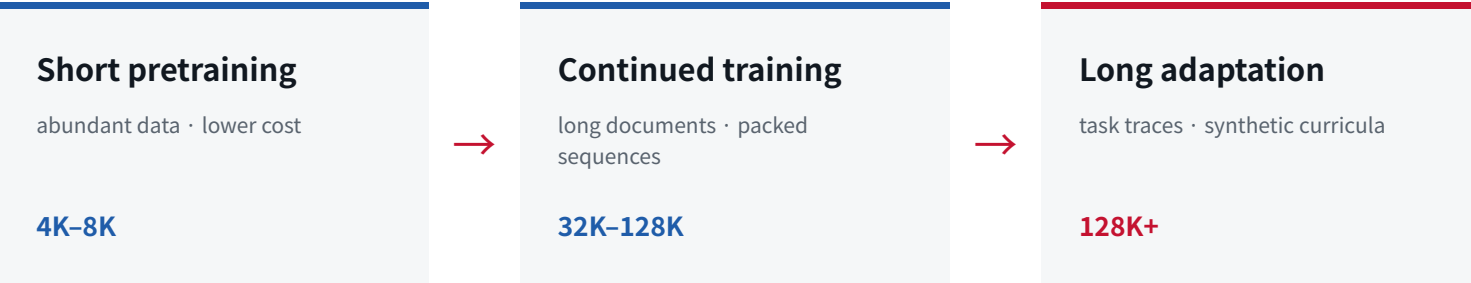
训练长上下文模型

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Short-to-long training



27

短上下文预训练：数据充足、成本较低，长度为 4K–8K。

继续训练：长文档、打包序列，长度为 32K–128K。

长上下文适配：任务轨迹、合成课程，长度为 128K+。

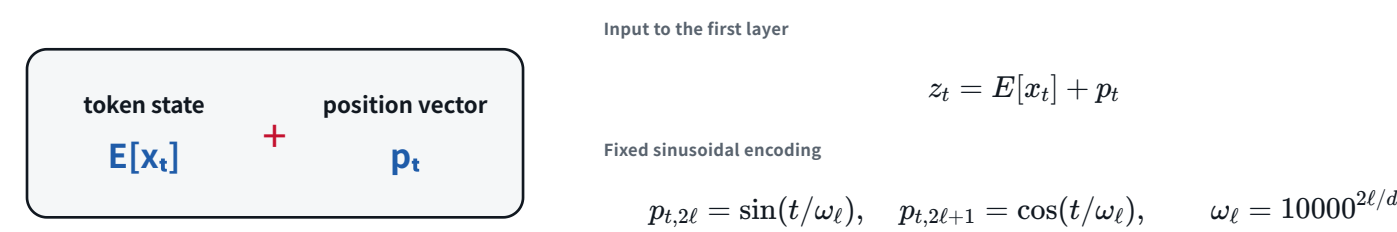
由短到长的训练

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Absolute positional encodings



New variables: x_t is a token ID; E is the embedding table; p_t is the positional vector; ℓ indexes sinusoid pairs.

- Position enters **before** the projections that produce Q, K, and V.
- Learned absolute embeddings have a fixed table; sinusoids define positions analytically.

Transformer positional encoding · Vaswani et al., 2017 ↗

输入第一层：`z\_t = E[x\_t] + p\_t`。

将词元状态 `E[x\_t]` 与位置向量 `p\_t` 相加。

固定正弦编码：`p\_{t,2ℓ} = sin(t/ωℓ)`；`p\_{t,2ℓ+1} = cos(t/ωℓ)`；`ωℓ = 10000^{2ℓ/d}`。

新变量：`x\_t` 是词元 ID；`E` 是嵌入表；`p` 是位置向量；`ℓ` 索引正弦对。

位置信息在生成 Q、K、V 的投影之前加入。学习得到的绝对嵌入使用固定表；正弦函数则以解析方式定义位置。

Transformer positional encoding · Vaswani 等, 2017 ↗

绝对位置编码

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Rotary position embeddings (RoPE)

q_i

k_j

Rotate rows of Q and K

$$\tilde{q}_i = R(i)q_i, \quad \tilde{k}_j = R(j)k_j$$

The score depends on relative displacement

$$\tilde{q}_i^\top \tilde{k}_j = q_i^\top R(j-i)k_j$$

Relative angle

$j-i$

New variable:

$R(t)$ is block-diagonal in 2D rotations with angles $t\theta_i$.

- V is unchanged; attention uses the same equation with rotated Q and K.

RoFormer · Su et al., Neurocomputing 2024 ↗

29

旋转 Q、K 的行：` $\tilde{q}_i = R(i)q_i$ `；` $\tilde{k}_j = R(j)k_j$ `。

注意力分数取决于相对位移：` $\tilde{q}_i^\top \tilde{k}_j = q_i^\top R(j-i)k_j$ `。

新变量：`R(t)` 是一个分块对角矩阵，其二维旋转角为 ` $t\theta_i$ `。

V 保持不变；注意力使用相同公式，但 Q 和 K 已旋转。

RoFormer · Su 等, Neurocomputing 2024 ↗

旋转位置嵌入（RoPE）

个人学习译文 · 非 CMU 官方译本 · 译注独立标记 · 原始课件来自 cmu-agents.com

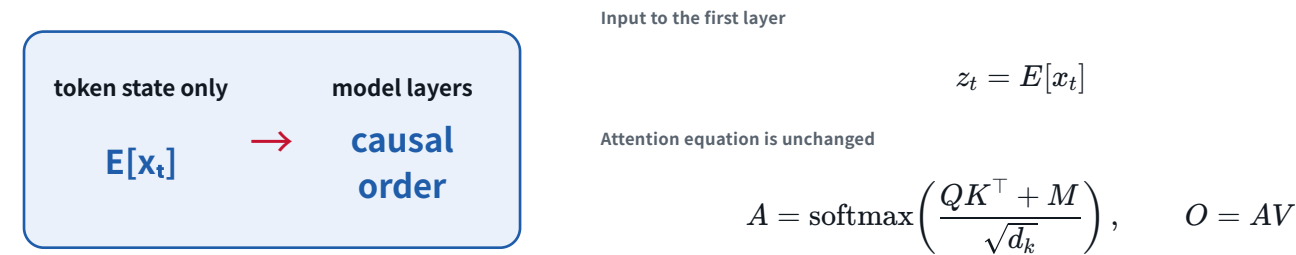
双语 PDF 29 | 原页 29

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

No positional encoding (NoPE)



- Order is inferred from causal computation or recurrent dynamics—not from p_t or $R(t)$.
- This removes position extrapolation, but not the need to train long-range behavior.

NoPE length generalization · Kazemnejad et al., NeurIPS 2023 ↗

Kimi K3 · NoPE with recurrent KDA ↗

输入第一层只含词元状态：`z\_t = E[x\_t]`；经过模型层和因果顺序处理。

注意力公式不变：`A = softmax((QK^T + M)/√dk)`，`O = AV`。

顺序由因果计算或循环动态推断，而不是由`p`或`R(t)`提供。移除位置外推问题，但仍需训练长程行为。

NoPE 长度泛化 · Kazemnejad 等, NeurIPS 2023 ↗；Kimi K3 · 使用循环 KDA 的 NoPE ↗

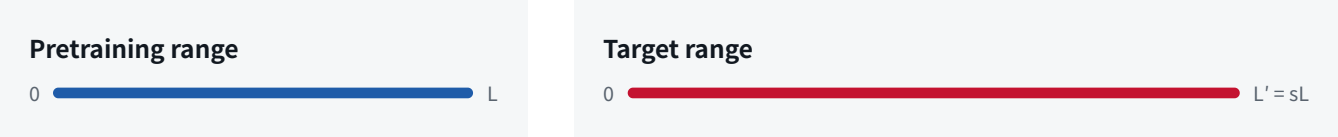
无位置编码（NoPE）

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Position interpolation



evaluate target position t using the familiar phase t / s

Uniform position interpolation

$R(t) \longrightarrow R(t/s)$ equivalently $\theta_\ell \longrightarrow \theta_\ell/s$

New variables: L and L' are original and target limits; s = L'/L.

- Every RoPE dimension is stretched by the same scale s.
- Long positions stay inside the phase range observed during pretraining.

Position Interpolation · Chen et al., 2023 ↗

31

预训练范围：`0` 到 `L`；目标范围：`0` 到 `L' = sL`。

使用熟悉的相位 `t/s` 来评估目标位置 `t`。

均匀位置插值：`R(t) → R(t/s)`，等价于 `θ<sub>l</sub> → θ<sub>l</sub>/s`。

新变量：`L` 和 `L'` 分别是原始与目标上限；`s = L'/L`。

所有 RoPE 维度均按相同缩放因子 `s` 拉伸；长位置仍处于预训练时见过的相位范围内。

Position Interpolation · Chen 等, 2023 ↗

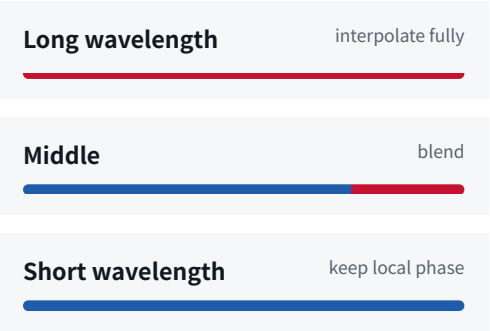
位置插值

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

YaRN



Frequency-selective interpolation

$$\theta'_\ell = (1 - \gamma_\ell) \frac{\theta_\ell}{s} + \gamma_\ell \theta_\ell, \quad 0 \leq \gamma_\ell \leq 1$$

Attention-temperature adjustment

$$A = \text{softmax}\left(\frac{\tilde{Q}\tilde{K}^\top + M}{\tau\sqrt{d_k}}\right), \quad \sqrt{1/\tau} = 0.1 \ln s + 1$$

New variables: γ_ℓ ramps from 0 for long wavelengths to 1 for short wavelengths; τ is attention temperature.

- Long-wavelength dimensions are interpolated; short-wavelength dimensions preserve local distinctions.
- The temperature correction counteracts attention-entropy drift at larger extension scales.
- The published formula for τ was fit on LLaMA models; it is a recipe, not a universal constant.

YaRN · Peng et al., ICLR 2024 ↗

频率选择性插值：长波长部分完全插值；中间部分混合；短波长保留局部相位。
 $\theta'_i = (1 - \gamma_i)(\theta_i/s) + \gamma_i\theta_i$ ，其中 $0 \leq \gamma_i \leq 1$ 。

注意力温度调整： $\tilde{A} = \text{softmax}((QK^\top + M)/(\tau\sqrt{dk}))$ ； $1/\tau = 0.1 \ln s + 1$ 。

新变量： γ 从长波长处的 0 渐变至短波长处的 1； τ 是注意力温度。

长波长维度进行插值；短波长维度保留局部区别。

温度修正用于抵消更大扩展比例下的注意力熵漂移。

已发表的 τ 公式是在 LLaMA 模型上拟合得到的，是一种操作方法，不是普适常数。

YaRN · Peng 等, ICLR 2024 ↗

YaRN

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Model-specific extension paths

Model	Context-length training	Position method	How it reaches maximum context
Qwen3.8-Flash-Next	262K native; full curriculum undisclosed	partial RoPE	YaRN extension to 1M
DeepSeek V4	4K → 16K → 64K → 1M	partial RoPE	progressive training + YaRN
GLM-5.3-Flash	1M native; exact curriculum undisclosed	NoPE in main attention	no RoPE rescaling
Kimi K3	8K → 64K → 256K → 1M	NoPE	progressive training to 1M
Nemotron 3 Ultra	1M long-context CPT; SFT to 512K	implicit order in Mamba; no RoPE	trained/evaluated at long lengths
Inkling	1M context; curriculum undisclosed	relative position embedding	learned relative representation

[Qwen](#) [DeepSeek](#) [GLM](#) [Kimi](#) [Nemotron](#) [Inkling](#)

模型 | 上下文长度训练 | 位置方法 | 达到最大上下文的方式

Qwen3.8-Flash-Next: 原生 262K; 完整课程未披露 | 部分 RoPE | 通过 YaRN 扩展至 1M。

DeepSeek V4: 4K → 16K → 64K → 1M | 部分 RoPE | 渐进训练 + YaRN。

GLM-5.3-Flash: 原生 1M; 精确课程未披露 | 主注意力为 NoPE | 不缩放 RoPE。

Kimi K3: 8K → 64K → 256K → 1M | NoPE | 渐进训练至 1M。

Nemotron 3 Ultra: 1M 长上下文 CPT; SFT 至 512K | Mamba 中隐式表示顺序 ; 无 RoPE | 在长序列上训练 / 评估。

Inkling: 1M 上下文; 课程未披露 | 相对位置嵌入 | 学习相对位置表征。

来源链接: Qwen、DeepSeek、GLM、Kimi、Nemotron、Inkling。

因模型而异的扩展路径

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Long-context data examples

Long documents coherent natural structure Books, codebases, repositories, and multi-document corpora.	Packed examples high token utilization Efficient, but boundaries and cross-example leakage matter.
Agent trajectories actions + observations Teach long-horizon recovery, state tracking, and tool use.	Synthetic tasks controlled dependencies Place evidence and distractors at designed positions.

34

长文档：结构连贯；包括书籍、代码库、仓库和多文档语料。

打包样例：词元利用率高；效率高，但边界和跨样例泄漏值得注意。

智能体轨迹：行动 + 观测；训练长时程恢复、状态跟踪和工具使用。

合成任务：依赖关系可控；将证据和干扰项放置在预先设计的位置。

长上下文数据示例

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

Long-context data mixtures

Model	Natural long data	Constructed supervision	Length curriculum
Kimi K3	cleaned, upsampled documents + video	permuted multimodal subtasks with evidence across the full context	8K → 64K → 256K → 1M
GLM-5	documents; repository files, issues, PRs, and diffs	synthetic dependencies + agent trajectories	32K → 128K → 200K
Nemotron 3 Ultra	long-document QA	multi-document reasoning, sequential scans, synthetic tables	33B-token 1M CPT; SFT to 512K
DeepSeek V4	scientific papers + technical reports	agentic post-training after long-context pretraining	4K → 16K → 64K → 1M

The shared recipe is **coherent long sources + tasks whose answer depends on distant evidence + progressive length growth**.

[Kimi K3 ↗](#) [GLM-5 ↗](#) [Nemotron 3 Ultra ↗](#) [DeepSeek V4 ↗](#)

ZH / 中文译文 · 第 1/1 部分

模型 | 自然长文本数据 | 构造的监督信号 | 长度课程

Kimi K3：清理并上采样的文档 + 视频 | 跨越整个上下文的置换多模态子任务 | 8K → 64K → 256K → 1M。

GLM-5：文档、仓库文件、议题、PR 和差异 | 合成依赖关系 + 智能体轨迹 | 32K → 128K → 200K。

Nemotron 3 Ultra：长文档问答 | 多文档推理、顺序扫描、合成表格 | 33B 词元的 1M CPT；SFT 至 512K。

DeepSeek V4：科学论文 + 技术报告 | 长上下文预训练后的智能体式后训练 | 4K → 16K → 64K → 1M。

共同做法：连贯的长文本来源 + 答案依赖远距离证据的任务 + 逐步增加序列长度。

来源链接：Kimi K3、GLM-5、Nemotron 3 Ultra、DeepSeek V4。

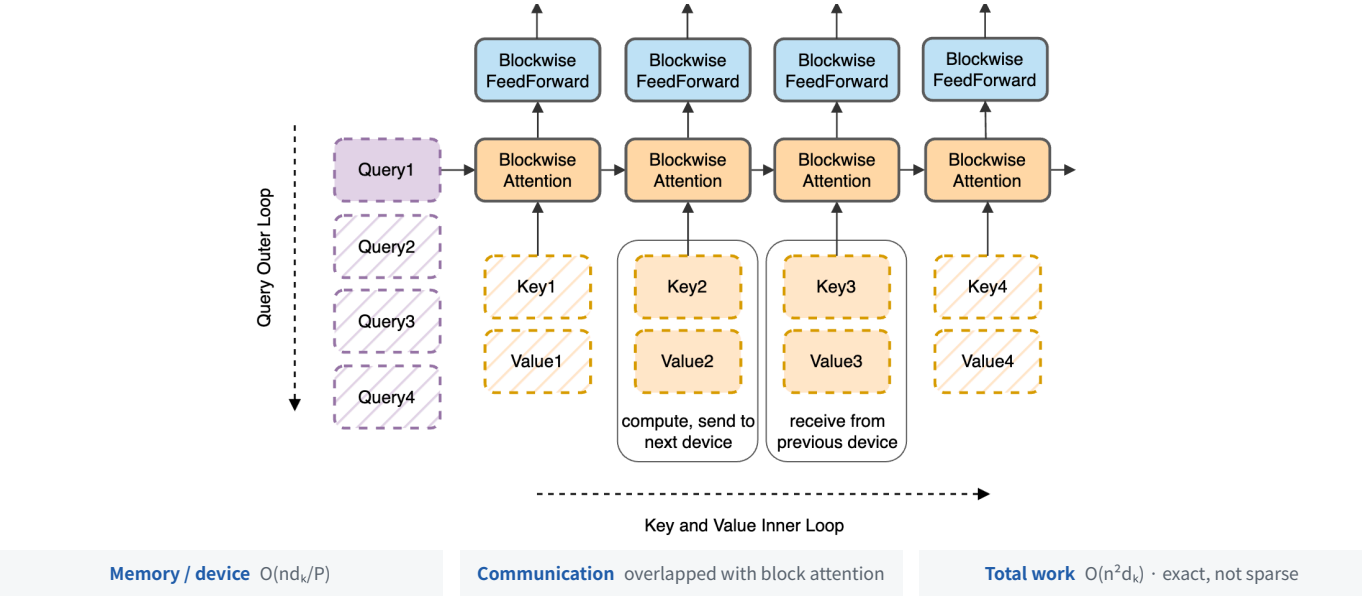
长上下文数据混合

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Context parallelism



每设备内存：` $O(nd^k/P)$ `。

通信与分块注意力重叠。

总工作量：` $O(n^2d^k)$ `；这是精确计算，不是稀疏计算。

Ring Attention · Liu 等, 2023 ↗；Megatron 上下文并行 ↗

上下文并行

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

提示缓存与 KV 缓存

Prompt and KV caching

37

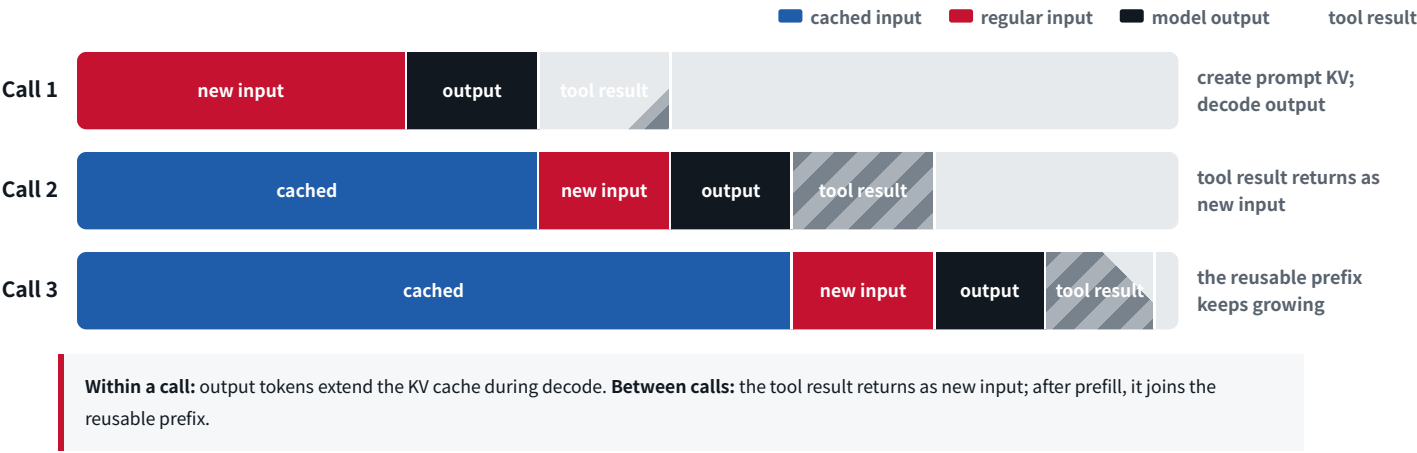
提示缓存与 KV 缓存

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

KV caching during decoding



KV and prompt-state reuse · Prompt Cache, MLSys 2024 ↗

38

图例：缓存输入、常规输入、模型输出、工具结果。

调用 1：新输入 → 输出 → 工具结果；创建提示 KV，并解码输出。

调用 2：缓存前缀 → 新输入 → 输出 → 工具结果；工具结果作为新输入返回。

调用 3：更长的缓存前缀 → 新输入 → 输出 → 工具结果；可复用前缀持续增长。

一次调用内：输出词元会在解码期间扩展 KV 缓存。不同调用之间：工具结果会作为新输入返回；预填充之后，它会加入可复用前缀。

KV 与提示状态复用 · Prompt Cache, MLSys 2024 ↗

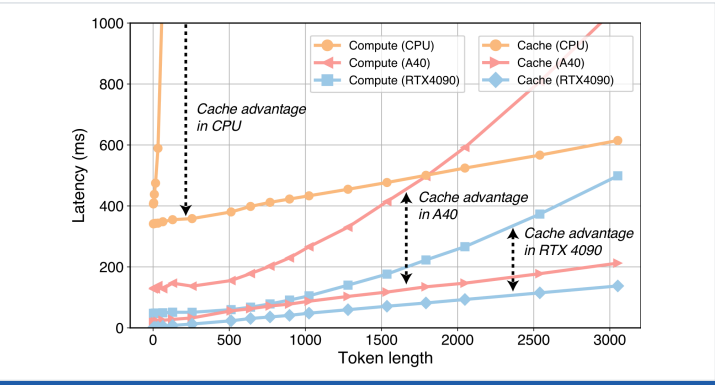
解码期间的 KV 缓存

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

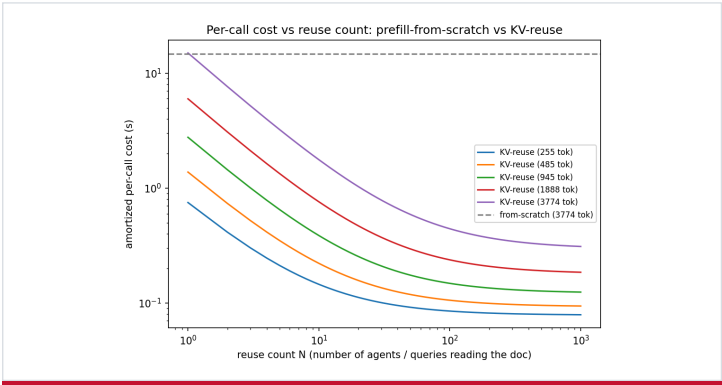
EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Empirical benefits of cache reuse



Prompt Cache
Compute grows faster than loading cached attention state; the crossover depends on hardware.



Can I Buy Your KV Cache?
One-time prefill is amortized across readers; per-call cost approaches the reuse-step floor.

Measured physical cost and provider API price are different quantities.

[Prompt Cache · Gim et al., 2024 ↗](#) [Can I Buy Your KV Cache? · 2026 ↗](#)

缓存复用的实测收益

Prompt Cache：计算成本增长快于读取缓存注意力状态的成本；交叉点取决于硬件。

Can I Buy Your KV Cache?：一次性预填充成本由多个读取者分摊；每次调用的成本接近复用步骤的下限。

实测物理成本和服务提供方 API 价格是不同的量。

Prompt Cache · Gim 等, 2024 ↗；Can I Buy Your KV Cache? · 2026 ↗

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Cached-token API pricing

Model	Cached input	Regular input	Output
DeepSeek V4 Flash	\$0.007–0.014	\$0.22–0.44	\$0.66–1.32
GLM-5.2	\$0.26	\$1.40	\$4.40
Kimi K3	\$0.30	\$3.00	\$15.00
GPT-5.6 Luna	\$0.02	\$0.20	\$1.20
GPT-5.6 Sol	\$0.40	\$4.00	\$20.00
Claude Opus 5	\$0.50	\$5.00	\$25.00

USD per 1M tokens · prices checked Aug. 30, 2026
“Cached” means a cache read/hit; cache creation may be billed separately.
DeepSeek range: off-peak–peak.

[DeepSeek ↗](#) [Z.AI ↗](#) [Kimi ↗](#) [OpenAI ↗](#) [Anthropic ↗](#)

模型 | 缓存输入 | 常规输入 | 输出（美元 / 百万词元）

DeepSeek V4 Flash：\$0.007–0.014 | \$0.22–0.44 | \$0.66–1.32。

GLM-5.2：\$0.26 | \$1.40 | \$4.40。Kimi K3：\$0.30 | \$3.00 | \$15.00。

GPT-5.6 Luna：\$0.02 | \$0.20 | \$1.20。GPT-5.6 Sol：\$0.40 | \$4.00 | \$20.00。

Claude Opus 5：\$0.50 | \$5.00 | \$25.00。

价格核查日期：2026 年 8 月 30 日。 ‘缓存’ 指读取 / 命中缓存；创建缓存可能另行计费。DeepSeek 区间：非高峰至高峰。

来源链接：DeepSeek、Z.AI、Kimi、OpenAI、Anthropic。

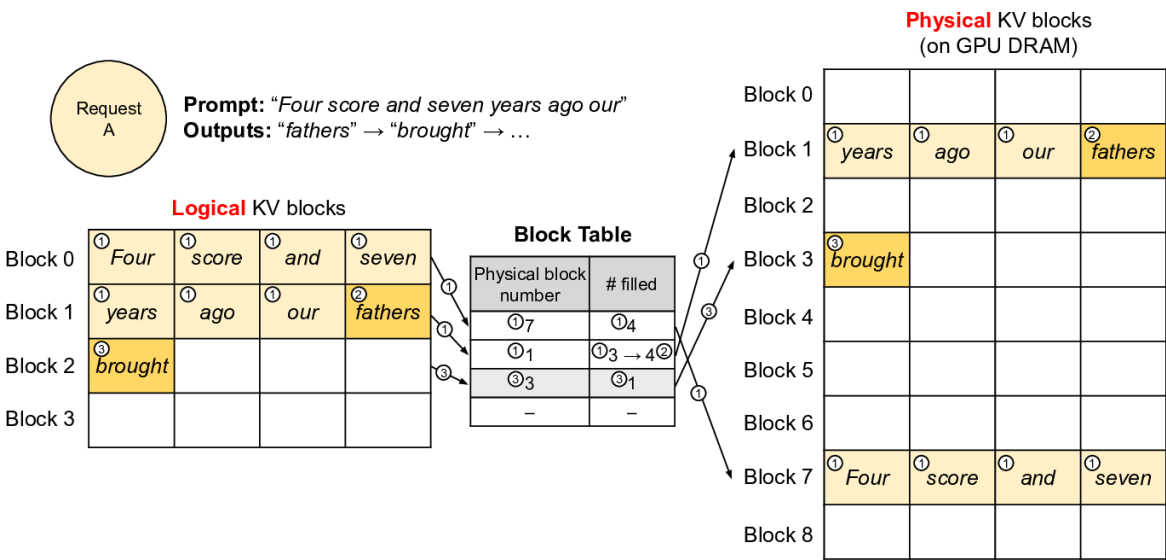
缓存词元 API 定价

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Paged KV-cache allocation



Block-table translation · vLLM / PagedAttention · Kwon et al., SOSP 2023 ↗

41

请求 A 的提示： ‘Four score and seven years ago our fathers’ ； 输出： ‘fathers’ → ‘brought’ → ...。

逻辑 KV 块：Block 0 包含 Four、score、and、seven；Block 1 包含 years、ago、our、fathers；Block 2 从 brought 开始；后续 Block 3 为空。

块表列：物理块编号、已填充数。物理 GPU DRAM 上的 KV 块以 Block 0–8 编号；示意图将逻辑块映射到物理块，并以已填充词元数记录占用。

块表转换 · vLLM / PagedAttention · Kwon 等, SOSP 2023 ↗

译注 / 与原文翻译分开

图中块表的个别编号和填充计数在当前图像尺度下不够清晰，因此不逐格转录映射编号。

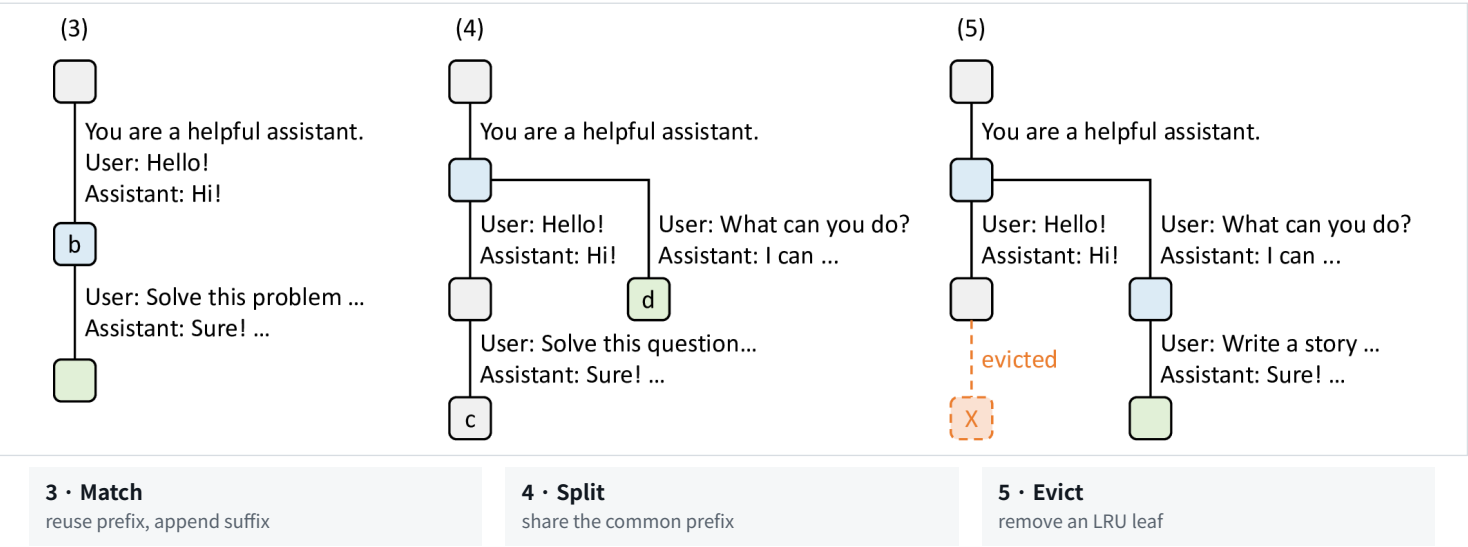
分页 KV 缓存分配

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

RadixAttention prefix sharing



RadixAttention operations (steps 3–5) · SGLang · Zheng et al., 2023 ↗

42

第 3 步·匹配：复用前缀，追加后缀。示例公共前缀含 ‘You are a helpful assistant. User: Hello! Assistant: Hi!’ ；随后追加 ‘User: Solve this problem ... Assistant: Sure! ...’ 。

第 4 步·拆分：共享公共前缀。示例将公共对话前缀分支到 ‘User: What can you do? Assistant: I can ...’ 及另一段待续内容。

第 5 步·驱逐：移除 LRU 叶节点；图中被驱逐节点标记为 X。

RadixAttention 操作（步骤 3–5）· SGLang · Zheng 等, 2023 ↗

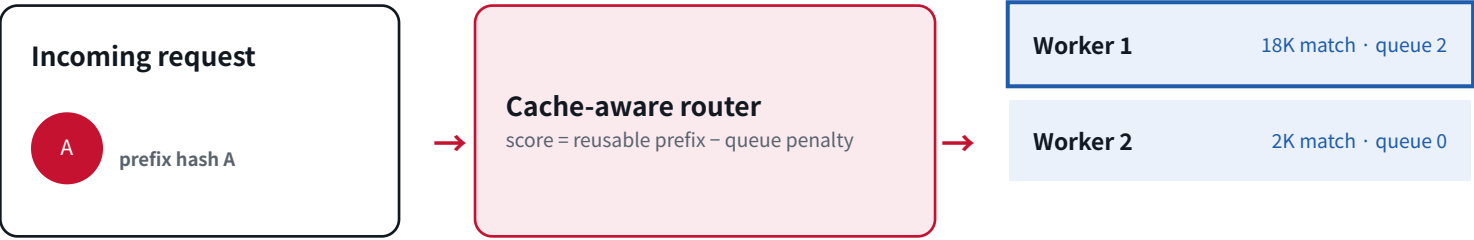
RadixAttention 前缀共享

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Cache-aware request routing



[SGLang v0.4 · cache-aware load balancing ↗](#)

43

传入请求 A，前缀哈希为 A。

感知缓存的路由器评分：可复用前缀 - 队列惩罚。

工作进程 1：匹配 18K，队列 2；工作进程 2：匹配 2K，队列 0。

[SGLang v0.4 · 感知缓存的负载均衡 ↗](#)

感知缓存的请求路由

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Prompt design for cache reuse

Stabilize the prefix instructions · tools · examples	Append changing state new user turn · observations
Avoid needless churn timestamps · reordered schemas	Measure reuse cached tokens · TTFT · eviction

44

稳定前缀：指令、工具、示例。

追加变化状态：新的用户轮次、观测。

避免不必要的变化：时间戳、重新排序的 Schema。

衡量复用：缓存词元数、TTFT、驱逐情况。

为缓存复用设计提示

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分



上下文精简

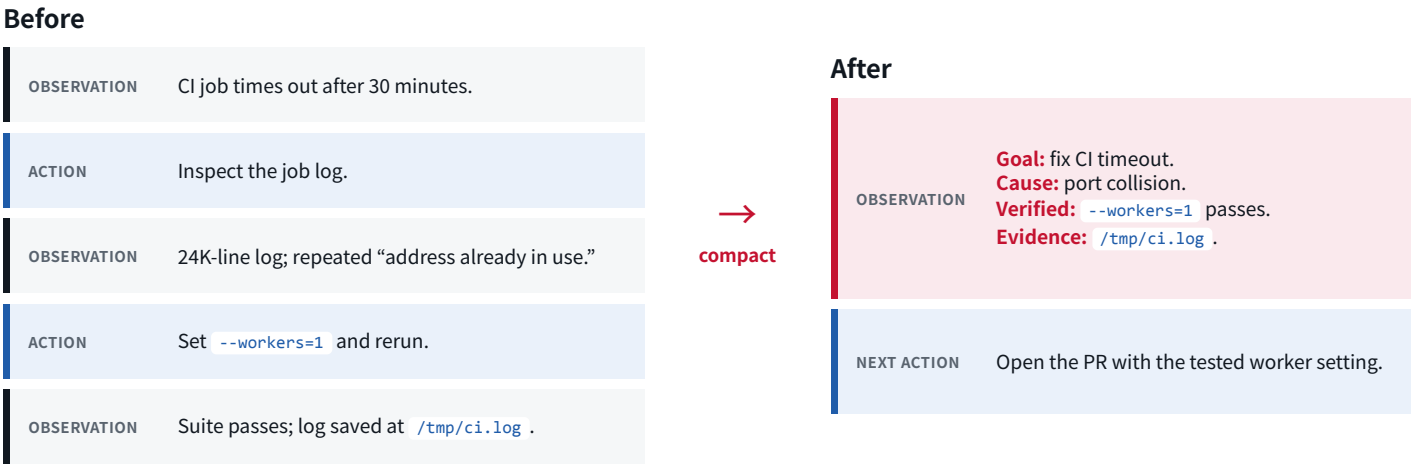
上下文精简

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Compaction example



46

精简示例

精简前：观测——CI 作业在 30 分钟后超时。行动——检查作业日志。观测——24K 行日志；反复出现 ‘address already in use’ 。行动——设置 ‘--workers=1’ 并重新运行。观测——测试套件通过；日志保存在 ‘/tmp/ci.log’ 。

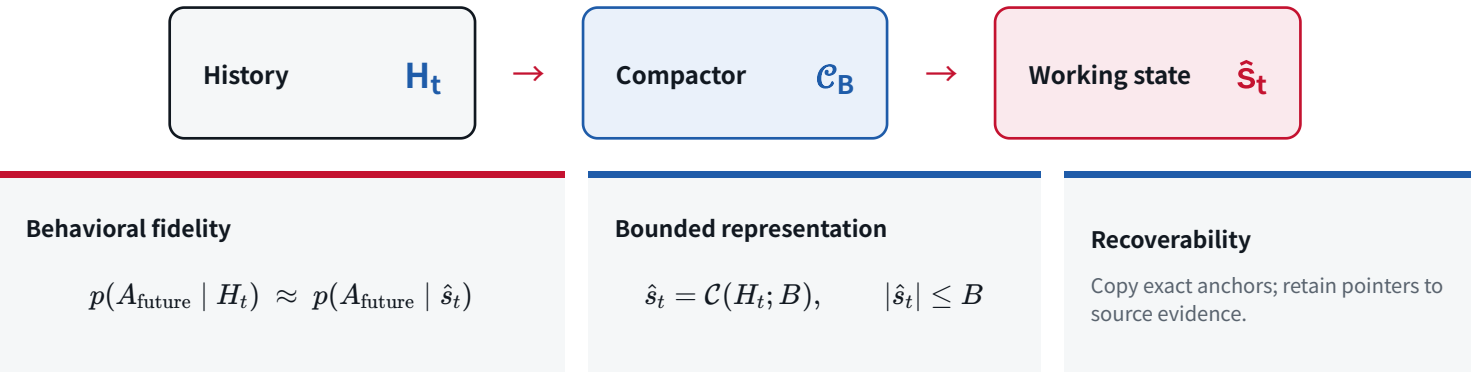
精简后：目标——修复 CI 超时。原因——端口冲突。已验证——‘--workers=1’ 通过。证据——‘/tmp/ci.log’。下一步行动——使用已测试的 worker 设置创建 PR。

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Compaction as state estimation



Notation: H_t is the full history; $\hat{S}_t$ is compacted working state; B is its token budget; A_{future} denotes future actions.

47

历史 $H_t \rightarrow$ 精简器 $C_B \rightarrow$ 工作状态 $\hat{S}_t$ 。

行为保真度： $p(A_{\text{future}} | H_t) \approx p(A_{\text{future}} | \hat{S}_t)$ 。

有界表示： $\hat{S}_t = C(H_t; B)$ ， $|\hat{S}_t| \leq B$ 。

可恢复性：精确保留锚点；保留指向来源证据的指针。

符号说明： H_t 是完整历史； $\hat{S}_t$ 是精简后的工作状态； B 是词元预算； A_{future} 表示未来行动。

译注 / 与原文翻译分开

原图精简器使用花体 C 、下标 B ；这里写作 C_B ， B 表示词元预算。

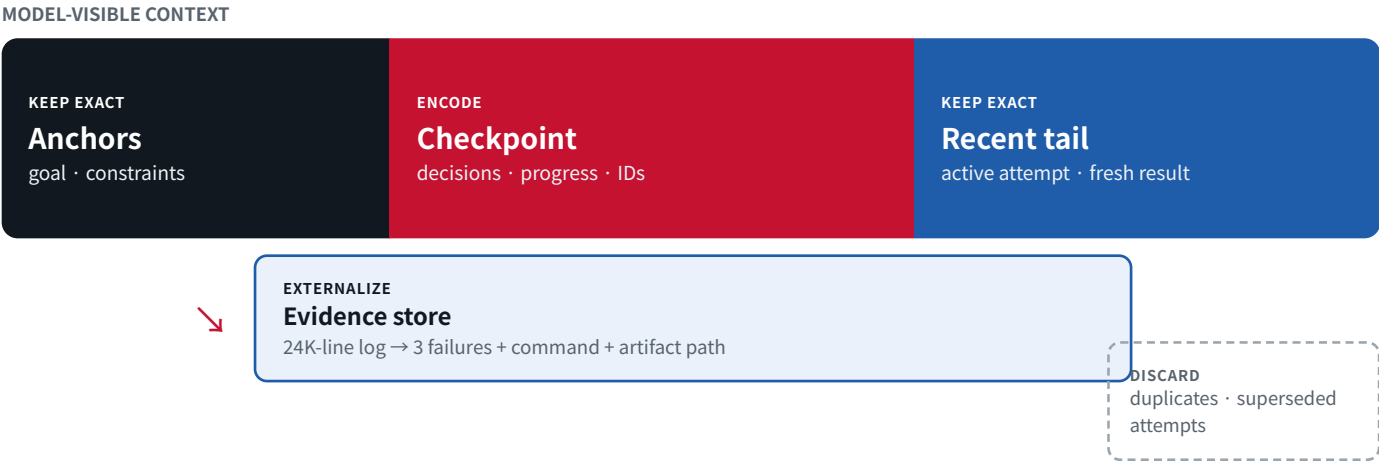
作为状态估计的精简

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

What survives compaction



[MemGPT · externalize and retrieve ↗](#) [LongLLMLingua · selective compression ↗](#)

模型可见上下文：原样保留锚点（目标、约束）； 编码检查点（决策、进展、ID）； 原样保留近期尾部（当前尝试、新结果）。

外置保存：证据存储中将 24K 行日志浓缩为 3 个失败、命令和制品路径。

丢弃：重复内容、已被替代的尝试。

MemGPT · 外置保存并检索 ↗； LongLLMLingua · 选择性压缩 ↗

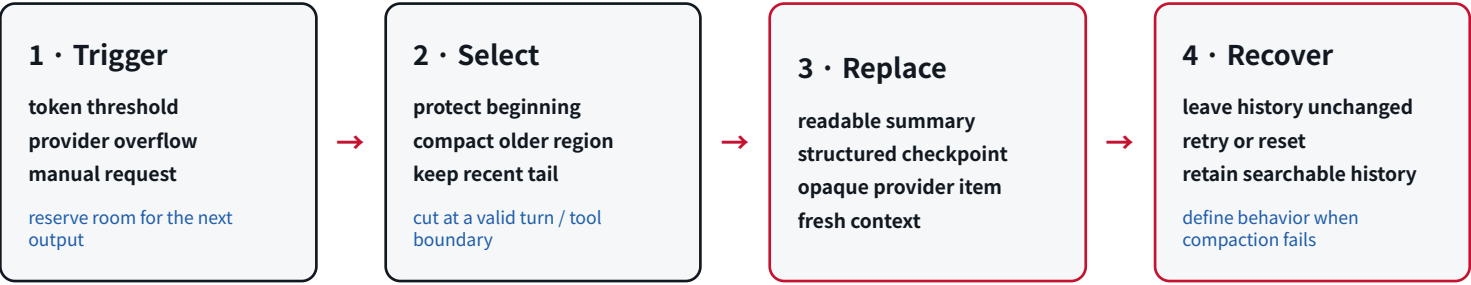
精简后保留什么

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Compaction policy decisions



[Codex ↗](#) [OpenCode ↗](#) [Pi ↗](#) [Hermes ↗](#) [OpenHands ↗](#)

- 1 · 触发：词元阈值、服务方上下文溢出或手动请求；为下一次输出预留空间。
- 2 · 选择：保护开头；精简较旧区域；保留近期尾部；在有效轮次 / 工具边界处截断。
- 3 · 替换：可读摘要、结构化检查点或不透明的服务方条目；使用新上下文。
- 4 · 恢复：保持历史不变；重试或重置；保留可搜索历史；定义精简失败时的行为。

[Codex ↗](#)、[OpenCode ↗](#)、[Pi ↗](#)、[Hermes ↗](#)、[OpenHands ↗](#)

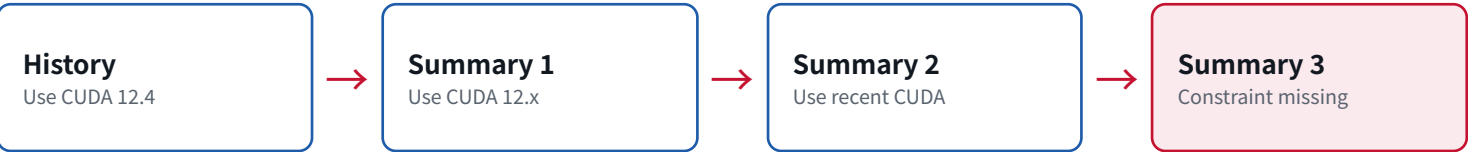
精简策略决策

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Repeated compaction drift



Each update inherits the previous checkpoint

$$\hat{s}_{k+1} = \mathcal{C}(\hat{s}_k \oplus R_k)$$

copy anchors exactly · keep the recent tail · retrieve source evidence

ReSum · summary-conditioned agent continuation ↗

历史： ‘使用 CUDA 12.4’ → 摘要 1： ‘使用 CUDA 12.x’ → 摘要 2： ‘使用较新的 CUDA’ → 摘要 3： 约束丢失。

每次更新都继承上一个检查点。`s<sup>k+1</sup> = C(s<sup>k</sup> ⊕ R<sub>k</sub>)`。

精确保留锚点 · 保留近期尾部 · 检索来源证据。

ReSum · 以摘要为条件继续智能体运行 ↗

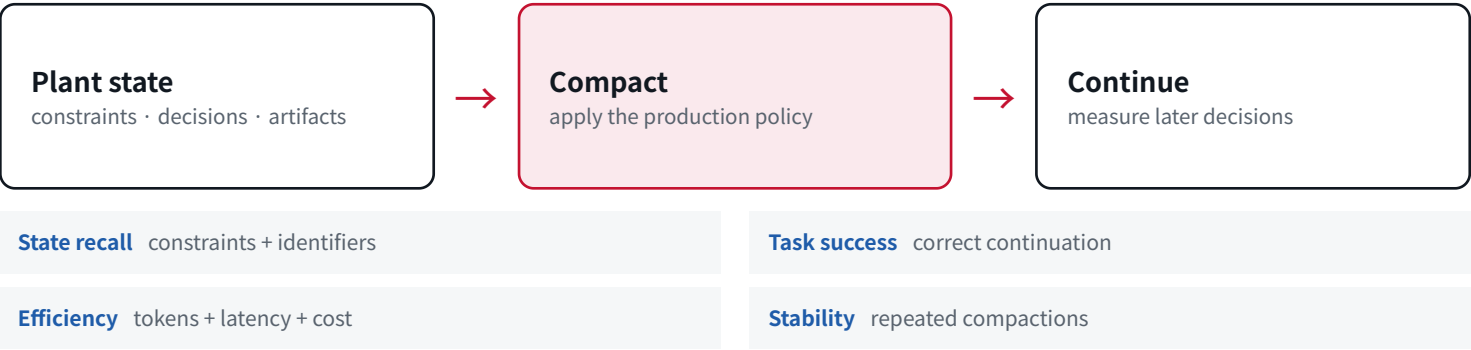
重复精简导致漂移

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

Continuation-based evaluation



51

植入状态：约束、决策、制品 → 精简：应用生产环境策略 → 继续：测量后续决策。

状态回忆：约束 + 标识符。任务成功：正确续作。

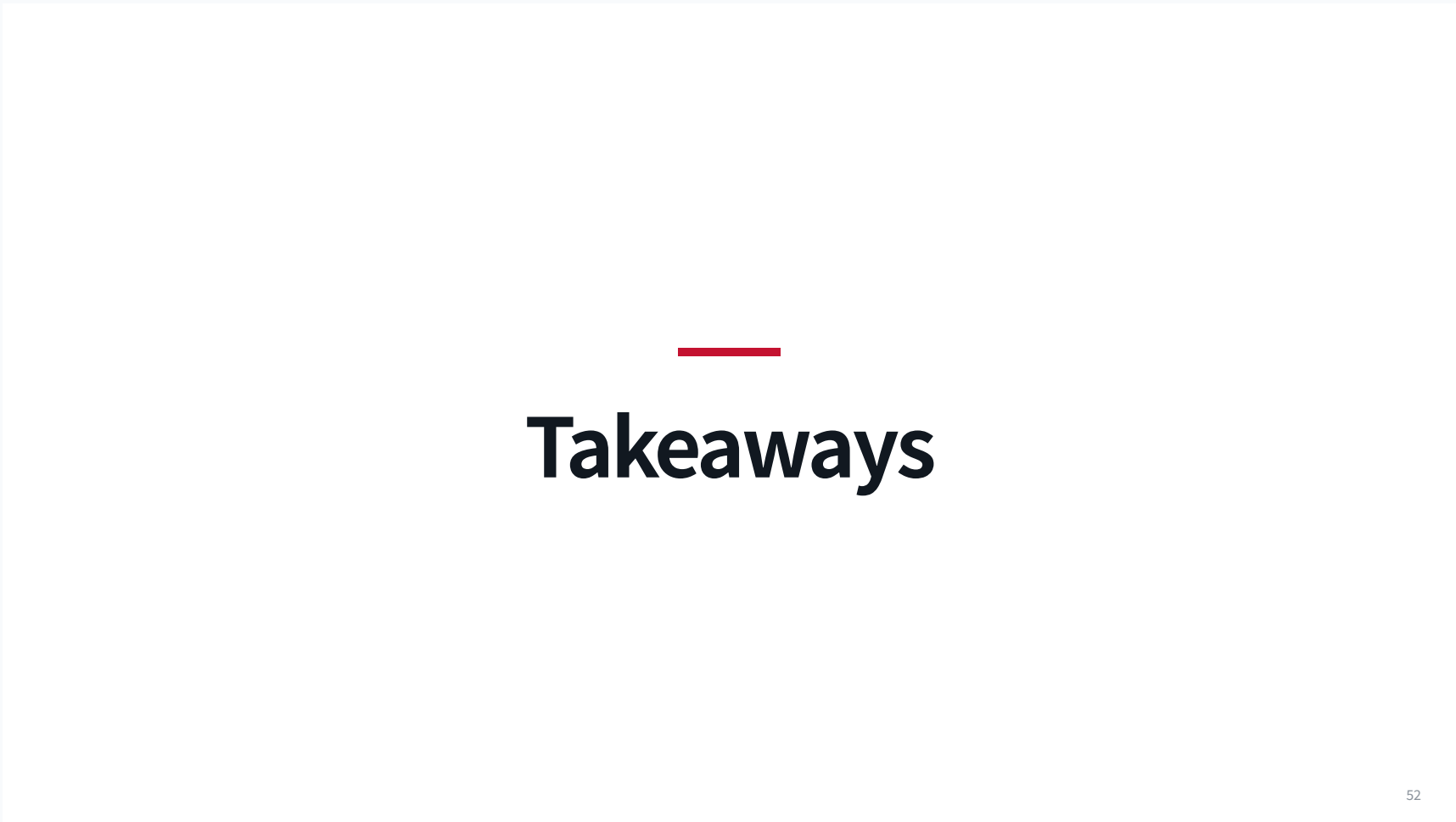
效率：词元数 + 延迟 + 成本。稳定性：重复精简。

基于续作的评估

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分



要点

要点

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

The long-context system stack

- 1

Growth: agent loops repeatedly ingest instructions, actions, and environment observations.
- 2

Inference: prefill, decode, dense attention, and KV memory create different bottlenecks.
- 3

Architecture: local, recurrent, sparse, and compressed mechanisms choose what history remains accessible.
- 4

Caching: stable prefixes avoid repeated prefill but do not reduce visible history.
- 5

Compaction: preserve intent, exact task state, recent work, and recovery paths—then evaluate continuation.

53

- 1 · 增长：智能体循环反复摄入指令、行动和环境观测。
- 2 · 推理：预填充、解码、稠密注意力和 KV 内存形成不同瓶颈。
- 3 · 架构：局部、循环、稀疏和压缩机制决定哪些历史仍可访问。
- 4 · 缓存：稳定前缀避免重复预填充，但不会减少可见历史。
- 5 · 精简：保留意图、精确任务状态、近期工作和恢复路径，然后评估续作。

长上下文系统栈

第 3 讲 · 智能体能力 2：长上下文智能体的上下文管理

EN / 英文官方原页 · 保持原样

ZH / 中文译文 · 第 1/1 部分

下一讲：智能体能力 3——技能与记忆。



Next Class: Agent Capabilities 3 — Skills and Memory

问题