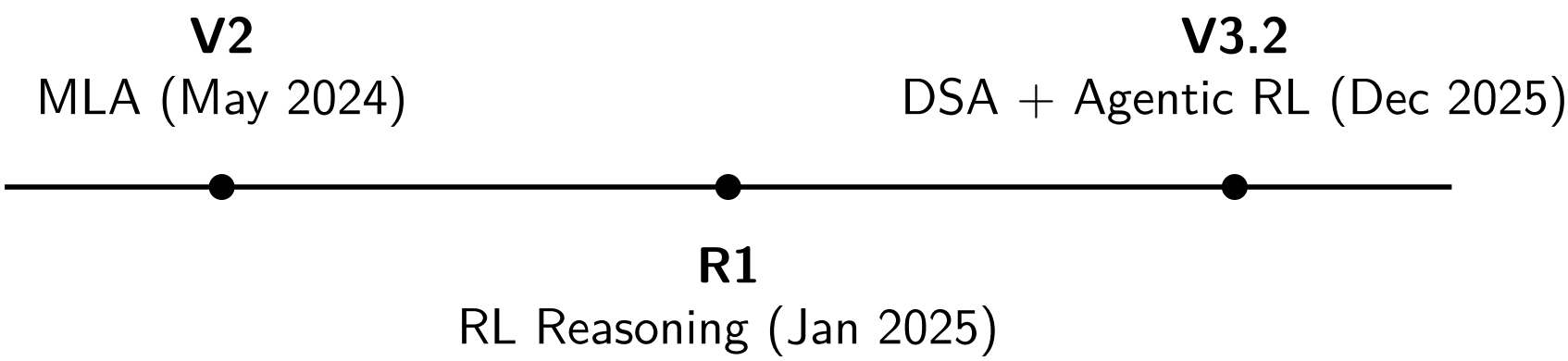
DeepSeek-V3.2: Pushing the Frontier of Open Large Language Models

Vishaal N Jamched, Junbo Zou

Mar 2, 2026

DeepSeek AI

DeepSeek LLM: V2 → R1 → V3.2



Key takeaways:

- **MLA → DSA:** from static attention to structured, scalable attention
- **RL → Agentic RL:** from single-step reasoning to long-horizon decision making

Motivation

Challenges of Traditional LLMs under Long Context

- **Lack of Focus:** Struggle to attend to critical information
- **Inference:** Computational cost and latency grow significantly

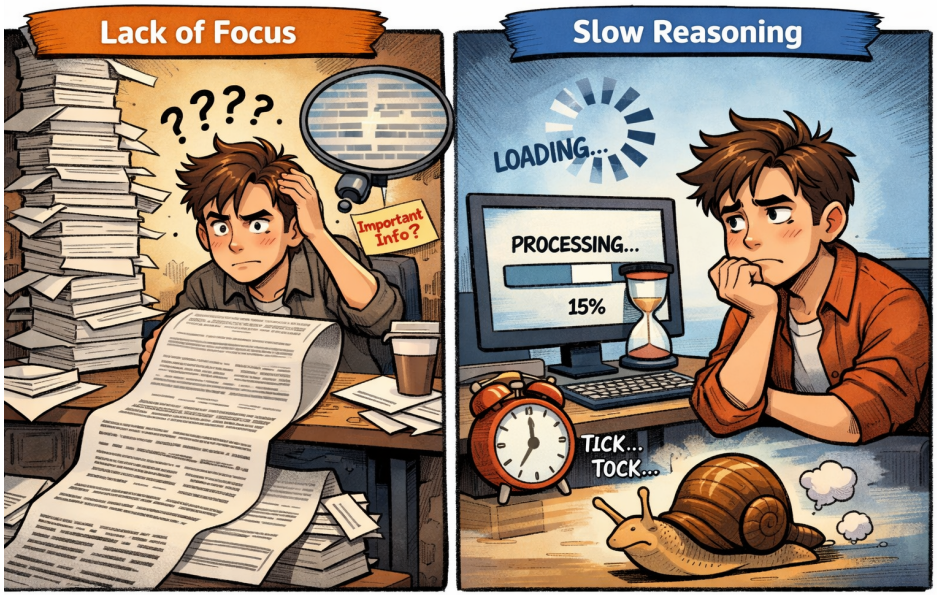


Figure 1 Why Long Context Breaks Traditional LLMs

From MHA to MLA

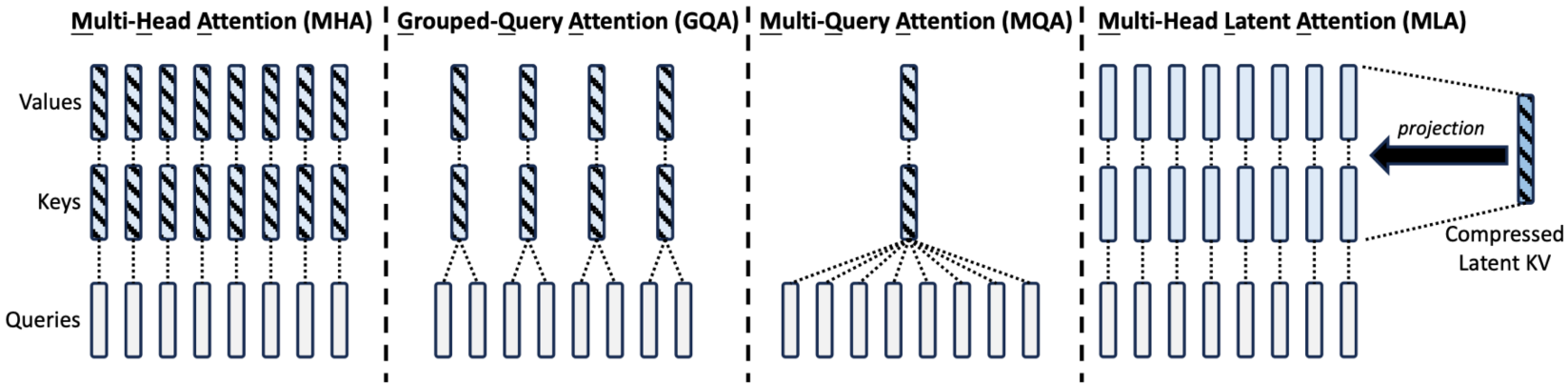


Figure 2 Development of Attention

From MLA to DSA

Limitation of MLA

$$\mathrm{Attn}(Q, K, V) = \mathrm{Softmax}\left(\frac{QK^\top}{\sqrt{d}}\right) V$$

$$Q, K, V \in \mathbb{R}^{L \times d}$$

Time / Memory Complexity: $\mathcal{O}(L^2)$

- MLA compresses KV states into a latent space.
- However, the attention matrix remains dense.
- Quadratic scaling becomes prohibitive for ultra-long contexts.

Mathematical Foundation of DSA

From Dense MLA to Hardware-Aligned Sparsity

1. Lightning Indexer

$$I_{t,s} = \sum_{j=1}^{H^I} w_{t,j}^I \cdot \mathrm{ReLU}(q_{t,j}^I \cdot k_s^I)$$

- $I_{t,s}$: Index score between query t and preceding token s.
- H^I: Number of indexer heads (small for speed).
- $q_{t,j}^I, k_s^I$: Low-rank query/key vectors in FP8.

Mathematical Foundation of DSA

From Dense MLA to Hardware-Aligned Sparsity

1. Lightning Indexer

$$I_{t,s} = \sum_{j=1}^{H^I} w_{t,j}^I \cdot \mathrm{ReLU}(q_{t,j}^I \cdot k_s^I)$$

- $I_{t,s}$: Index score between query t and preceding token s.
- H^I: Number of indexer heads (small for speed).
- $q_{t,j}^I, k_s^I$: Low-rank query/key vectors in FP8.

2. Token Selection & Attn

$$u_t = \mathrm{Attn}(h_t, \{c_s \mid I_{t,s} \in \text{Top-}k(I_{t,:})\})$$

- u_t: Final attention output for query token t.
- **Top-k**: The selection mechanism picking the k highest $I_{t,s}$ scores.
- c_s: The latent KV entries (MLA compressed states).

From MLA to DSA

DeepSeek Sparse Attention (DSA)

$$\mathcal{S}_i = \text{Top-}k\left(Q_i K^\top\right)$$

$$\text{Attn}_i = \sum_{j \in \mathcal{S}_i} \alpha_{ij} V_j$$

Complexity: $\mathcal{O}(Lk), \quad k \ll L$

- Each query attends only to its most relevant k tokens.
- The attention matrix becomes sparse instead of dense.
- Enables efficient long-context reasoning with minimal quality loss.

Attention Mechanism Complexity Overview

Method	Attention	Time Complexity	KV Cache
MHA	Dense $L \times L$	$\mathcal{O}(L^2)$	$\mathcal{O}(LHd)$
MQA	Dense $L \times L$	$\mathcal{O}(L^2)$	$\mathcal{O}(Ld)$
GQA	Dense $L \times L$	$\mathcal{O}(L^2)$	$\mathcal{O}(LGd)$
MLA	Dense $L \times L$	$\mathcal{O}(L^2)$	$\mathcal{O}(Ld_{\text{lat}})$
DSA	Sparse $L \times k$	$\mathcal{O}(Lk)$	$\mathcal{O}(Lk)$

- MQA, GQA, and MLA reduce **KV memory** but retain quadratic attention computation.
- DSA introduces structured sparsity, reducing computation from $\mathcal{O}(L^2)$ to $\mathcal{O}(Lk)$.
- Only sparse attention mechanisms fundamentally mitigate the quadratic scaling bottleneck.

From MLA to DSA

DeepSeek Sparse Attention (DSA)

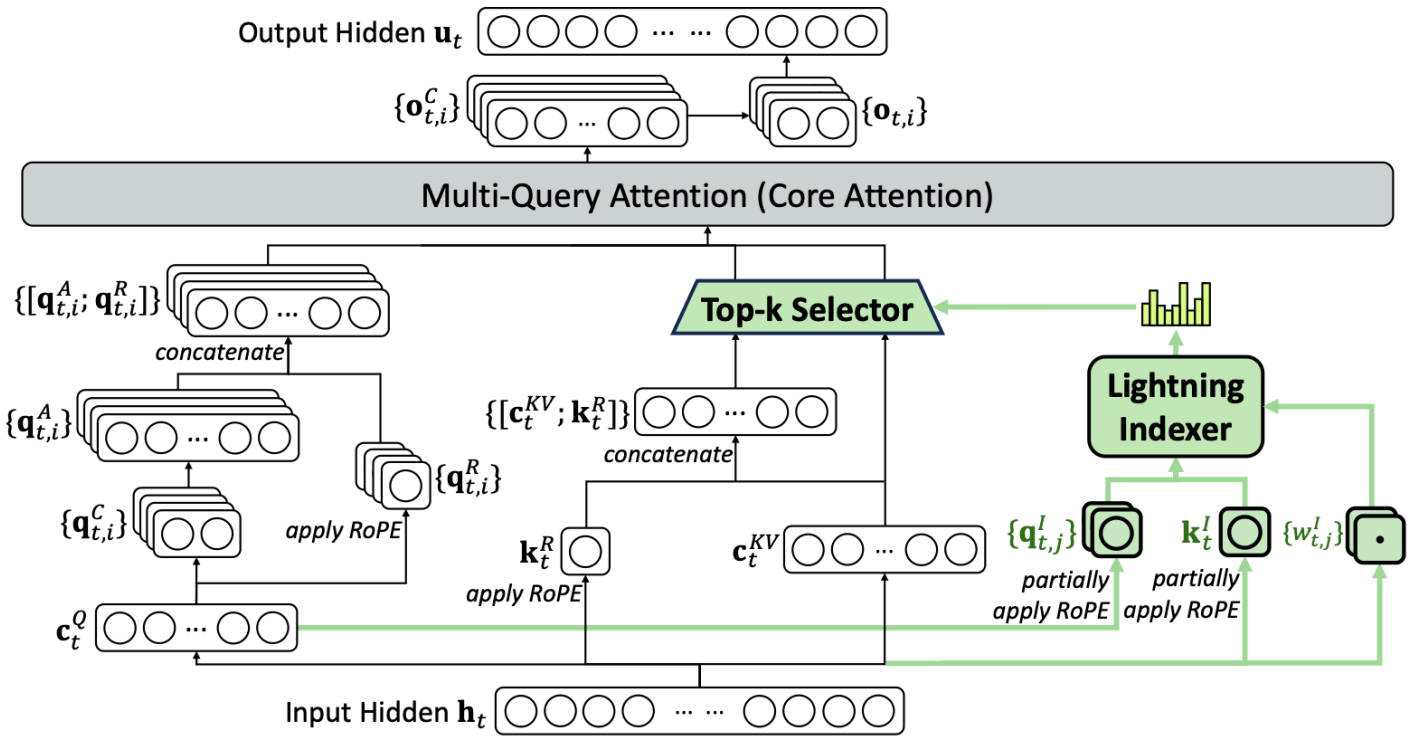


Figure 3 Attention architecture of DeepSeek-V3.2, where DSA is instantiated under MLA. The green part illustrates how DSA selects the top-k key-value entries according to the indexer.

Pre-Training: Stage 1 —Dense Warm-up

Goal: Initialize the Lightning Indexer before sparse training.

Setup:

- Start from DeepSeek-V3.1-Terminus (128K context)
- Keep **dense attention**; freeze all parameters **except** the Lightning Indexer
- Align Indexer output distribution with main attention distribution

Training Objective — KL-divergence loss:

$$\mathcal{L}^I = \sum_t D_{\mathrm{KL}}\left(p_{t,:} \,\middle\|\, \mathrm{Softmax}(I_{t,:})\right)$$

where $p_{t,:} \in \mathbb{R}^t$ is the L1-normalized sum of main attention scores across all heads for query token t.

Pre-Training: Stage 2 —Sparse Training

Goal: Adapt the full model to the sparse attention pattern of DSA.

Setup:

- Introduce **fine-grained token selection**: each query attends to top-k tokens only
- Optimize **all** model parameters; Indexer input is **detached** from the main graph

Training Objective — KL loss restricted to selected token set $\mathcal{S}_t$:

$$\mathcal{L}^I = \sum_t D_{\mathrm{KL}}\left(p_{t,\mathcal{S}_t} \,\middle\|\, \mathrm{Softmax}(I_{t,\mathcal{S}_t})\right), \quad \mathcal{S}_t = \{s \mid I_{t,s} \in \text{Top-}k(I_{t,:})\}$$

Decoupled optimization:

- Indexer optimized by $\mathcal{L}^I$ only
- Main model optimized by language modeling loss only

GRPO

GRPO (Group Relative Policy Optimization)

Sample a *group* of responses $\{y_j\}_{j=1}^{G}$ for each prompt x, score them with a reward $r(x, y_j)$, and form a group-relative advantage:

$$A_j = \frac{r(x, y_j) - \mu_r}{\sigma_r + \epsilon}, \mu_r = \frac{1}{G}\sum_{j=1}^{G} r(x, y_j), \sigma_r = \sqrt{\frac{1}{G}\sum_{j=1}^{G}(r(x, y_j) - \mu_r)^2}.$$

Define the likelihood ratio

$$\rho_j(\theta) = \frac{\pi_\theta(y_j \mid x)}{\pi_{\theta_{\text{old}}}(y_j \mid x)}.$$

Then optimize a PPO-style clipped objective:

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{x,\{y_j\}}\left[\frac{1}{G}\sum_{j=1}^{G}\min\Big(\rho_j(\theta)A_j,\ \text{clip}(\rho_j(\theta), 1-\varepsilon, 1+\varepsilon)A_j\Big)\right].$$

RL Experiment

Figure 4 GRPO GPU Memory Usage

Setup: 8× H20 (NVLink), veRL (Framework), GRPO
Backend: vLLM (infer), LLM-as-Judge (reward), FSDP (train)
Base: Qwen3-VL-8B-Ins, multi-turn interaction
Para: Batch = 32, Rollout = 8, Sample / step = 256

From RL to Agentic RL

Traditional RL: Optimize final answers.
Agentic RL: Train the model to plan, call tools, and reason.
Teaching an LLM to use agents is better than giving it an agent.

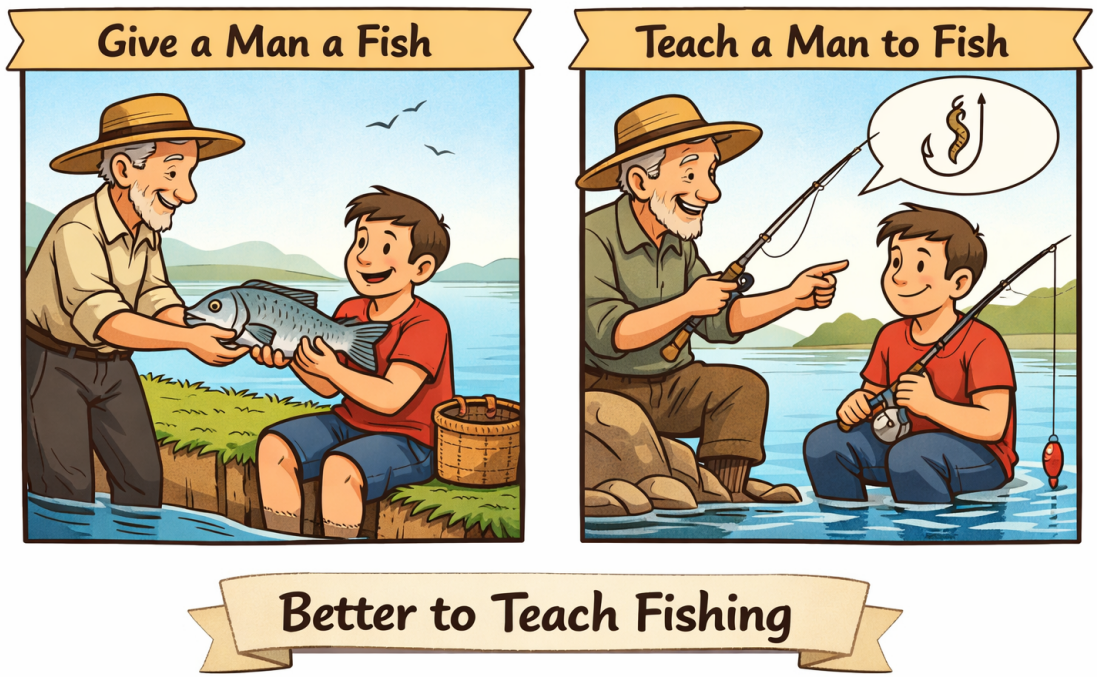
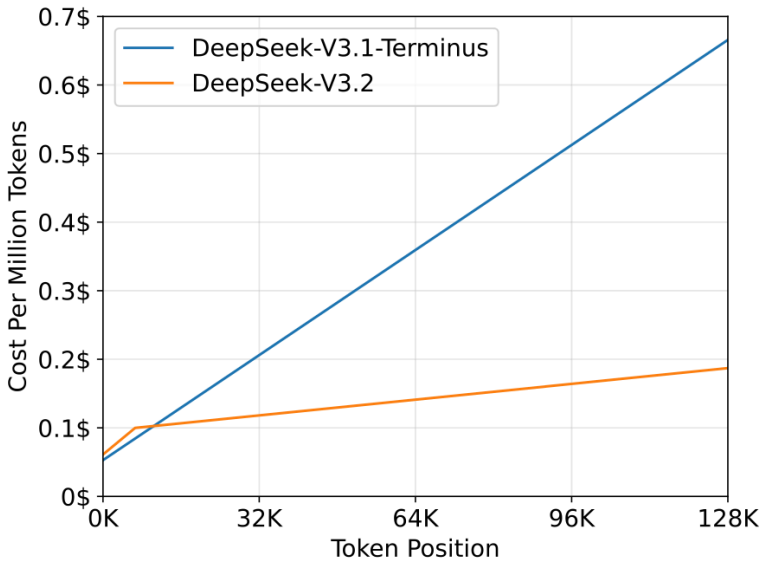
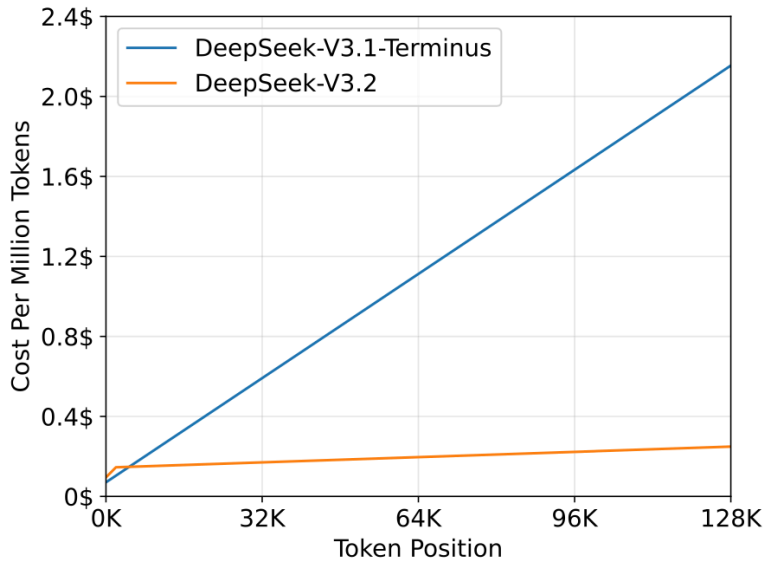


Figure 5 Teaching a man to fish is better than giving him a fish.

Inference Cost



(a) Prefilling



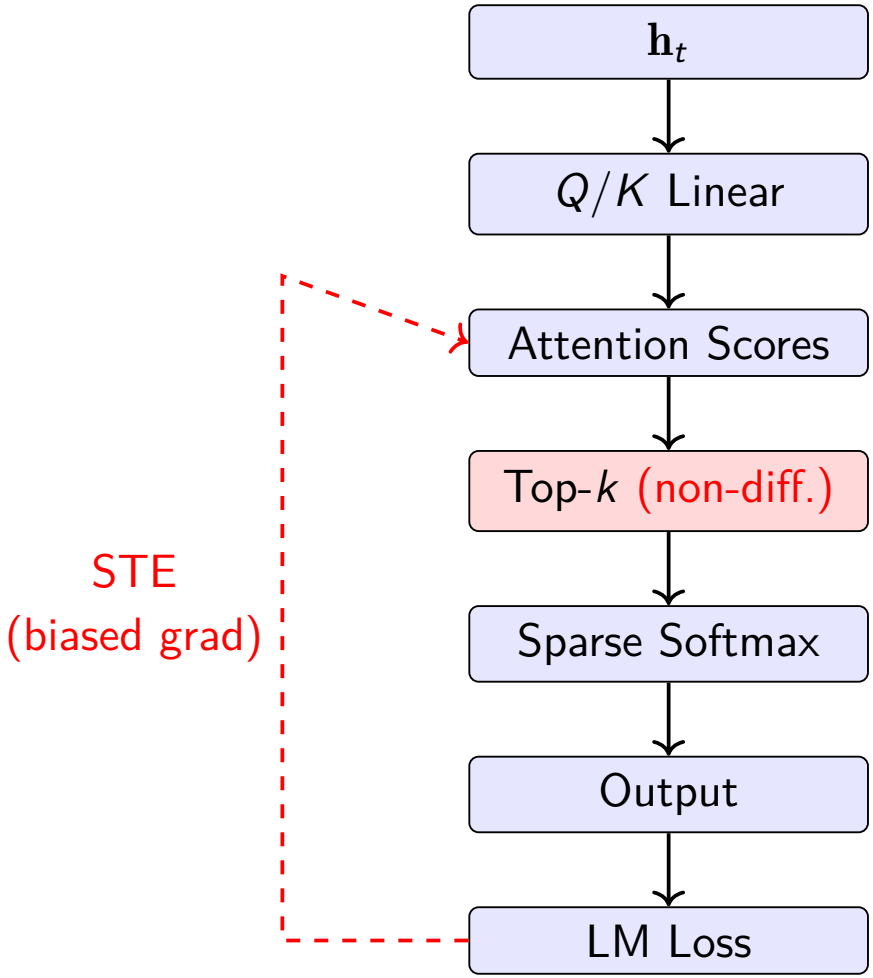
(b) Decoding

Figure 6 Inference costs of DeepSeek-V3.1-Terminus and DeepSeek-V3.2 on H800 clusters

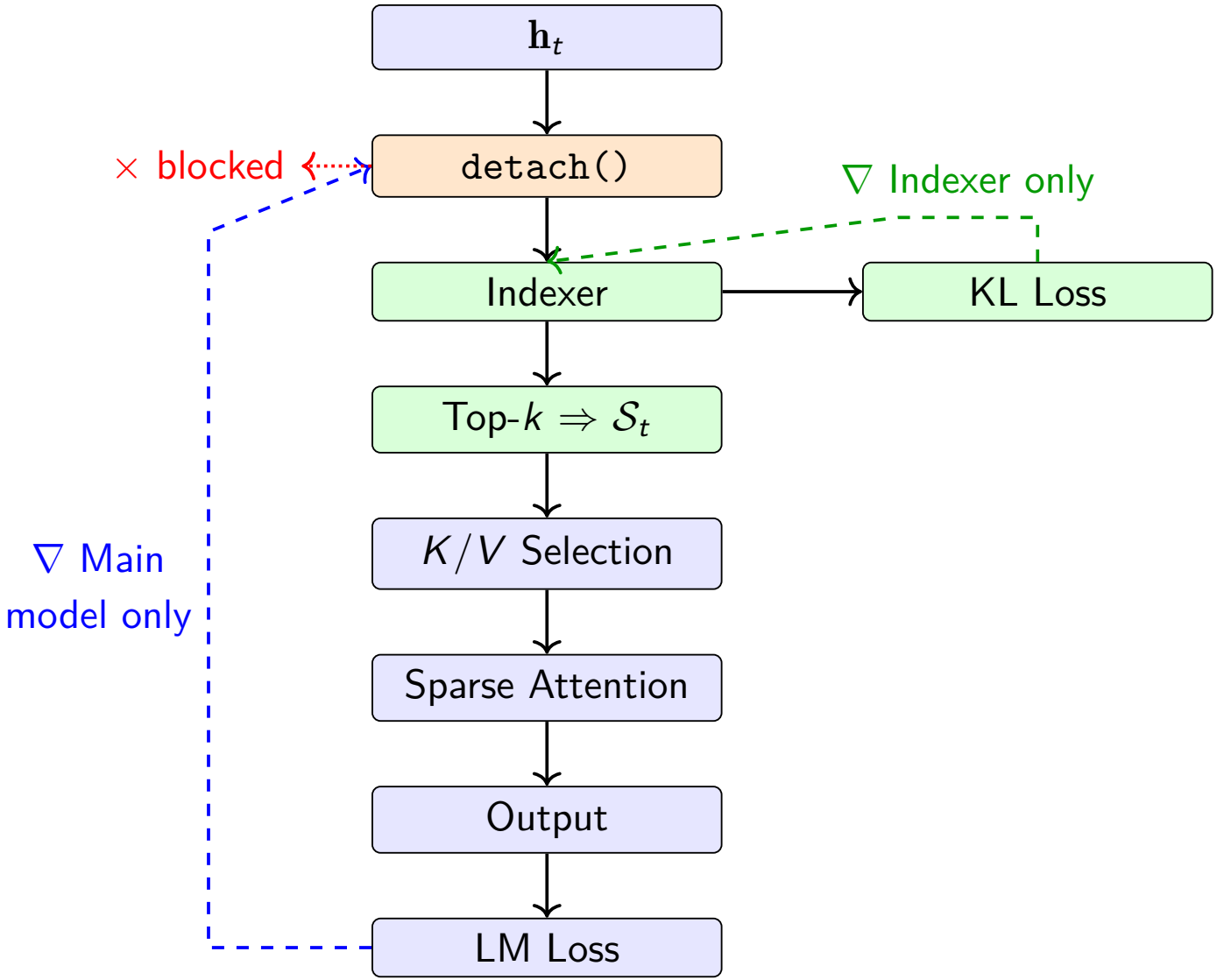
Benchmark Results

Benchmark	Claude-4.5	GPT-5	Gemini-3.0	**DeepSeek-V3.2**
English				
MMLU-Pro	88.2	87.5	**90.1**	85.0
GPQA Diamond	83.4	85.7	**91.9**	82.4
HLE	13.7	26.3	**37.7**	25.1
Code				
LiveCodeBench	64.0	84.5	**90.7**	83.3
Codeforces	1480	2537	**2708**	2386
Math				
AIME 2025	87.0	94.6	**95.0**	93.1
HMMT Feb 2025	88.3	**97.5**	81.7	92.5
IMOAnswerBench	79.2	89.2	**93.3**	90.2
Agent				
Terminal Bench 2.0	42.8	**77.2**	68.0	73.1
BrowseComp	24.1	42.4	35.2	**65.0**

Traditional Top-k: STE Backward (Straight-Through Estimator)



DeepSeek DSA: Detached Indexer



Detached vs. End-to-End Training

	Detached	**End-to-End**
Indexer aware errors	×	✓
Unified Target	× (separate losses)	✓ (LM loss only)
Training stability	✓ (high)	× (discrete gradient)
Performance upper	Limited by Indexer	Theoretically higher

This question is also a challenge in MoE Training.

Other competitive Advantages

API Price Table (25 Feb 2026)

Model	**Input / 1M**	**Output / 1M**
Google Gemini 3.1 Pro	\$2.00 (7.1×)	\$12.00 (28.6×)
Anthropic Claude 4.6 Opus	\$5.00 (17.9×)	\$25.00 (59.5×)
OpenAI GPT 5.2	\$1.75 (6.3×)	\$14.00 (33.3×)
DeepSeek V3.2	\$0.28 (1.0×)	\$0.42 (1.0×)
DeepSeek V3.2 (Cache hit)	\$0.028 (0.1×)	–

We sincerely thank DeepSeek for open-sourcing their models and providing highly competitive token pricing. Their openness and cost efficiency represent one of the company's core strengths.
Moreover, their open-source efforts and affordable API access enable many academic institutions to further explore and research on LLMs.

Thanks!