

DEEPSEEK V3
OPEN SOURCE
STATE OF THE ART LLM

PRESENTED BY
MIKE DOAN



WHO IS **DEEPSEEK**


WHAT IS **DEEPSEEK V3**

PERFECT HARDWARE UTILIZATION

RESULTS

WHAT IS **NEXT?**


CHAPTER I - WHO IS DEEPSEEK




Chinese **Open**AI



<https://www.deepseek.com/> [deepseek_ai](#) [deepseek-ai](#)


Activity Feed [Request to join this org](#) [Follow](#) 6,295

 **AI & ML interests**

None defined yet.

 **Recent Activity**

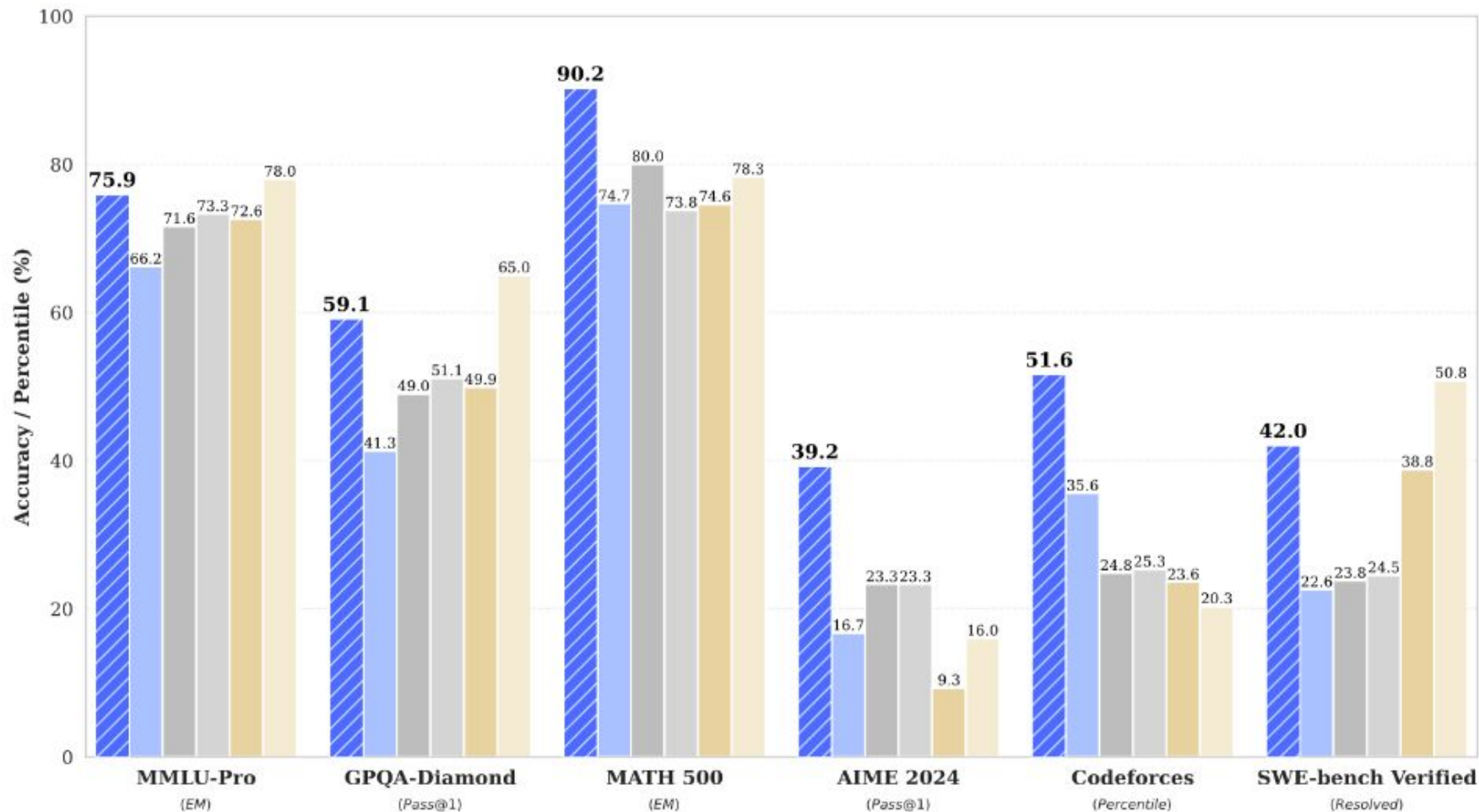
-  [Chester111](#) updated a collection 7 d...
[DeepSeek-V3](#)
-  [GeekExplorer](#) updated a collection 9
[DeepSeek-V2](#)

 **Organization Card** [Community](#) [About org cards](#)

DeepSeek (深度求索), founded in 2023, is a Chinese company dedicated to making AGI a reality.

Unravel the mystery of AGI with curiosity. Answer the essential question with long-termism.

DeepSeek-V3 DeepSeek-V2.5 Qwen2.5-72B-Inst Llama-3.1-405B-Inst GPT-4o-0513 Claude-3.5-Sonnet-1022



Problem

Let \mathcal{B} be the set of rectangular boxes with surface area 54 and volume 23. Let r be the radius of the smallest sphere that can contain each of the rectangular boxes that are elements of \mathcal{B} . The value of r^2 can be written as $\frac{p}{q}$, where p and q are relatively prime positive integers. Find $p + q$.

Problem

Each vertex of a regular octagon is independently colored either red or blue with equal probability. The probability that the octagon can then be rotated so that all of the blue vertices end up at positions where there were originally red vertices is $\frac{m}{n}$, where m and n are relatively prime positive integers. What is $m + n$?

Chapter II - WHAT IS DEEPSEEK V3

- Top 6 among all models on OpenRouter this month. (SOTA)
- Costed 5 mil to train (OOMs less than other offerings)
- My personal 1.5th favorite model (sometimes better than claude due to better availability)

	Top today	Top this week	Top this month	Trending
1.	Anthropic: Claude 3.5 Sonnet (self-moderated) > New Claude 3.5 Sonnet delivers better - than - Opus capabilities, faste...			390B tokens ↑169%
2.	Anthropic: Claude 3.5 Sonnet > New Claude 3.5 Sonnet delivers better - than - Opus capabilities, faste...			233B tokens ↑149%
3.	Google: Gemini Flash 1.5 > Gemini 1.5 Flash is a foundation model that performs well at a variety...			153B tokens ↑72%
4.	Google: Gemini Flash 1.5 8B > Gemini Flash 1.5 8B is optimized for speed and efficiency, offering en...			120B tokens ↑2%
5.	Mistral: Mistral Nemo > A 12B parameter model with a 128k token context length built by Mis...			59.4B tokens ↑116%
6.	DeepSeek V3 > DeepSeek - V3 is the latest model from the DeepSeek team, building u...			51.4B tokens new
	OpenAI: GPT-4o-mini >			50B tokens 29%

Training Costs	Pre-Training	Context Extension	Post-Training	Total
in H800 GPU Hours	2664K	119K	5K	2788K
in USD	\$5.328M	\$0.238M	\$0.01M	\$5.576M

Llama who?



Claude who?

Pro
For Claude power users

Everything in Free, plus:

- ✓ More usage than Free
- ✓ Access to Projects to organize documents and chats
- ✓ Ability to use more models, like Claude 3.5 Sonnet and Claude 3 Opus

Access to new features

Friendship ended with
Now

 Claude

 deepseek-ai / DeepSeek-V3

Error receiving response
An unknown error has occurred.

✕ Dismiss

\$18

Per month with annual subscription discount;
\$216 billed up front. \$20 if billed monthly.

is my
best friend



deepseek

Architecture

- Multi Latent Attention
- DeepSeekMoE
- Multi Token Prediction
- 671B params
- 37B active !!!
- AdamW
- 14.8T tokens

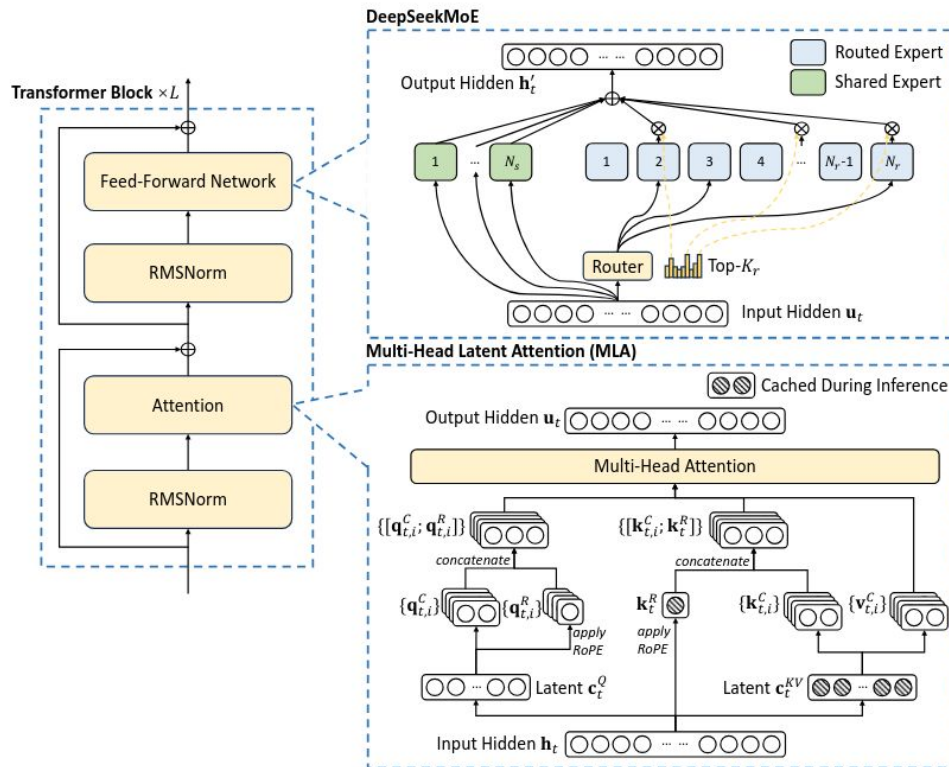


Figure 2 | Illustration of the basic architecture of DeepSeek-V3. Following DeepSeek-V2, we adopt MLA and DeepSeekMoE for efficient inference and economical training.

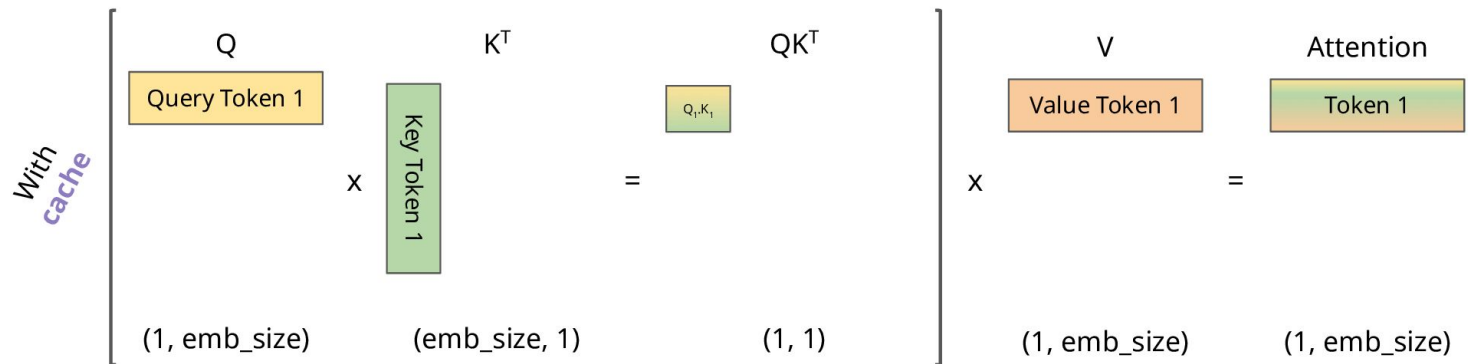
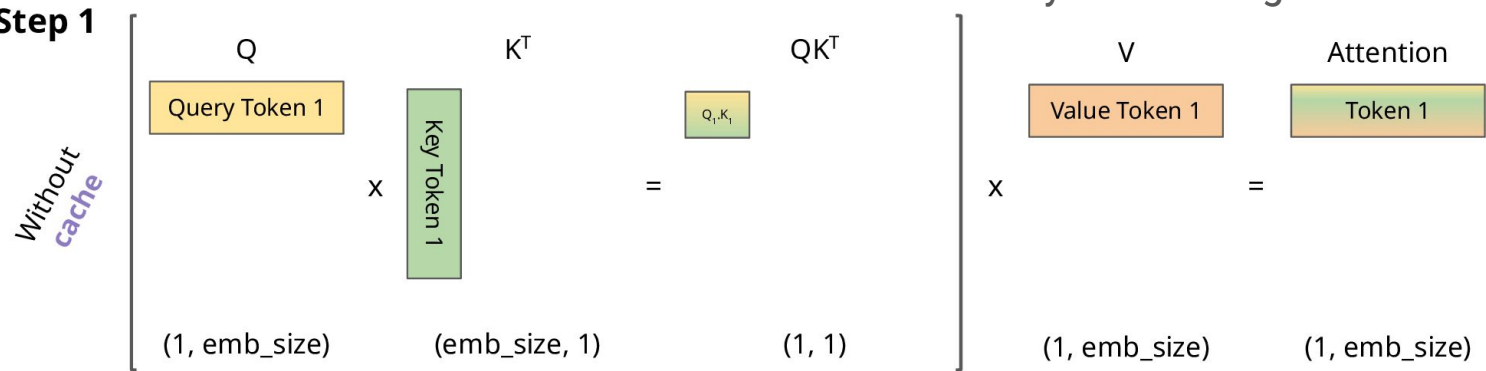
Multi Latent Attention

Key Differences Between MHA and MHLA		
Feature	Multi-Head Attention (MHA)	Multi-Head Latent Attention (MHLA)
Attention Mechanism	Direct attention between input tokens (Q and K).	Attention mediated through a latent space.
Complexity	Scales quadratically with sequence length.	Scales linearly with the number of latent vectors.
Latent Space	No explicit latent space.	Uses a learned latent space for attention.
Interpretability	Harder to interpret attention patterns.	More interpretable due to abstract latent vectors.
Use Case	Standard in Transformers for NLP and other tasks.	Useful for reducing computational cost or improving interpretability.

Intermission - KV Caching

This is bottlenecked by embedding dimension!

Step 1



Values that will be masked Values that will be taken from cache

Multi Latent Attention

2.1.1. Multi-Head Latent Attention

For attention, DeepSeek-V3 adopts the MLA architecture. Let d denote the embedding dimension, n_h denote the number of attention heads, d_h denote the dimension per head, and $\mathbf{h}_t \in \mathbb{R}^d$ denote the attention input for the t -th token at a given attention layer. The core of MLA is the low-rank joint compression for attention keys and values to reduce Key-Value (KV) cache during inference:

$$\mathbf{c}_t^{KV} = W^{DKV} \mathbf{h}_t, \quad (1)$$

$$[\mathbf{k}_{t,1}^C; \mathbf{k}_{t,2}^C; \dots; \mathbf{k}_{t,n_h}^C] = \mathbf{k}_t^C = W^{UK} \mathbf{c}_t^{KV}, \quad (2)$$

$$\mathbf{k}_t^R = \text{RoPE}(W^{KR} \mathbf{h}_t), \quad (3)$$

$$\mathbf{k}_{t,i} = [\mathbf{k}_{t,i}^C; \mathbf{k}_t^R], \quad (4)$$

$$[\mathbf{v}_{t,1}^C; \mathbf{v}_{t,2}^C; \dots; \mathbf{v}_{t,n_h}^C] = \mathbf{v}_t^C = W^{UV} \mathbf{c}_t^{KV}, \quad (5)$$

Mixture of Experts

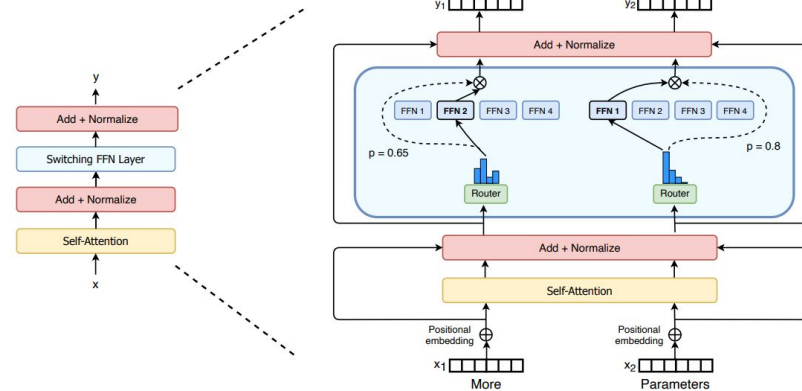
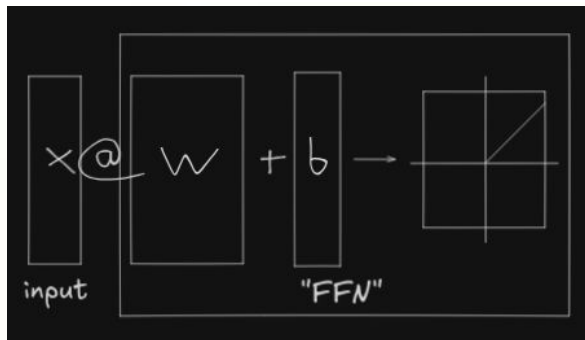
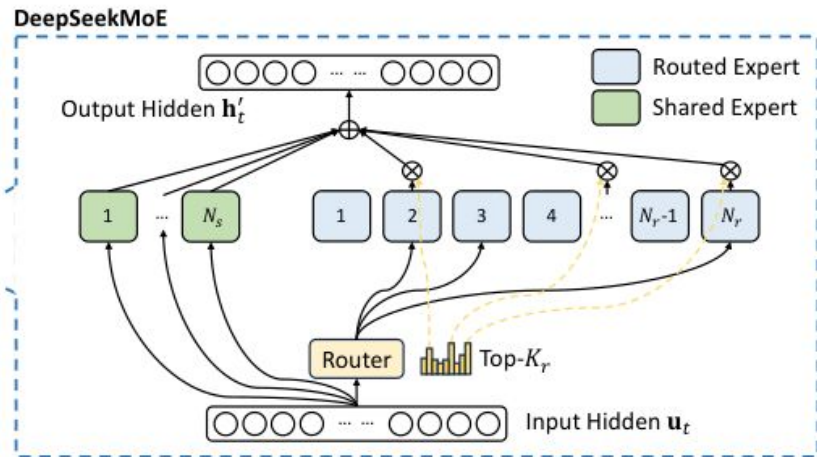


Figure 2: Illustration of a Switch Transformer encoder block. We replace the dense feed forward network (FFN) layer present in the Transformer with a sparse Switch FFN layer (light blue). The layer operates independently on the tokens in the sequence. We diagram two tokens (x_1 = "More" and x_2 = "Parameters" below) being routed (solid lines) across four FFN experts, where the router independently routes each token. The switch FFN layer returns the output of the selected FFN multiplied by the router gate value (dotted-line).

Mixture of Experts - No Auxiliary Loss for Load Balancing

Auxiliary-Loss-Free Load Balancing. For MoE models, an unbalanced expert load will lead to routing collapse (Shazeer et al., 2017) and diminish computational efficiency in scenarios with expert parallelism. Conventional solutions usually rely on the auxiliary loss (Fedus et al., 2021; Lepikhin et al., 2021) to avoid unbalanced load. However, too large an auxiliary loss will impair the model performance (Wang et al., 2024a). To achieve a better trade-off between load balance and model performance, we pioneer an auxiliary-loss-free load balancing strategy (Wang et al., 2024a) to ensure load balance. To be specific, we introduce a bias term b_i for each expert and add it to the corresponding affinity scores $s_{i,t}$ to determine the top-K routing:

$$g'_{i,t} = \begin{cases} s_{i,t}, & s_{i,t} + b_i \in \text{Topk}(\{s_{j,t} + b_j | 1 \leq j \leq N_r\}, K_r), \\ 0, & \text{otherwise.} \end{cases} \quad (16)$$

Note that the bias term is only used for routing. The gating value, which will be multiplied with

Mixture of Experts

Complementary Sequence-Wise Auxiliary Loss. Although DeepSeek-V3 mainly relies on the auxiliary-loss-free strategy for load balance, to prevent extreme imbalance within any single sequence, we also employ a complementary sequence-wise balance loss:

Node-Limited Routing. Like the device-limited routing used by DeepSeek-V2, DeepSeek-V3 also uses a restricted routing mechanism to limit communication costs during training. In short, we ensure that each token will be sent to at most M nodes, which are selected according to the sum of the highest $\frac{K_r}{M}$ affinity scores of the experts distributed on each node. Under this constraint, our MoE training framework can nearly achieve full computation-communication overlap.

No Token-Dropping. Due to the effective load balancing strategy, DeepSeek-V3 keeps a good load balance during its full training. Therefore, DeepSeek-V3 does not drop any tokens during training. In addition, we also implement specific deployment strategies to ensure inference load balance, so DeepSeek-V3 also does not drop tokens during inference.

Multi Token Prediction (MTP)

2.2. Multi-Token Prediction

Inspired by Gloeckle et al. (2024), we investigate and set a Multi-Token Prediction (MTP) objective for DeepSeek-V3, which extends the prediction scope to multiple future tokens at each position. On the one hand, an MTP objective densifies the training signals and may improve data efficiency. On the other hand, MTP may enable the model to pre-plan its representations for better prediction of future tokens. Figure 3 illustrates our implementation of MTP. Different from Gloeckle et al. (2024), which parallelly predicts D additional tokens using independent output heads, we **sequentially predict additional tokens** and keep the complete causal chain at each prediction depth. We introduce the details of our MTP implementation in this section.

Multi Token Prediction (MTP)

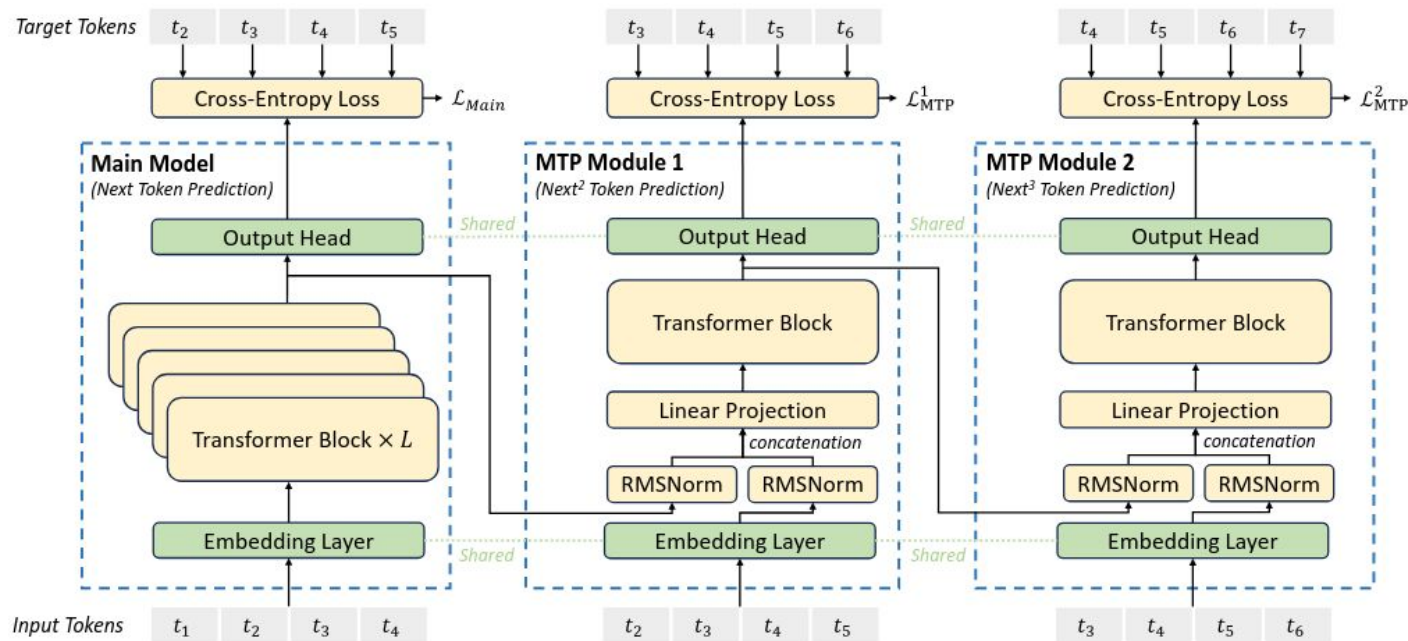


Figure 3 | Illustration of our Multi-Token Prediction (MTP) implementation. We keep the complete causal chain for the prediction of each token at each depth.

Multi Token Prediction (MTP)

Basically just cross entropy over pre-set number of tokens

MTP Training Objective. For each prediction depth, we compute a cross-entropy loss $\mathcal{L}_{\text{MTP}}^k$:

$$\mathcal{L}_{\text{MTP}}^k = \text{CrossEntropy}(P_{2+k:T+1}^k, t_{2+k:T+1}) = -\frac{1}{T} \sum_{i=2+k}^{T+1} \log P_i^k[t_i], \quad (24)$$

CHAPTER III - PERFECT INFRASTRUCTURE

Training Costs	Pre-Training	Context Extension	Post-Training	Total
in H800 GPU Hours	2664K	119K	5K	2788K
in USD	\$5.328M	\$0.238M	\$0.01M	\$5.576M

Table 1 | Training costs of DeepSeek-V3, assuming the rental price of H800 is \$2 per GPU hour.

Compute. Llama 3 405B is trained on up to 16K H100 GPUs, each running at 700W TDP with 80GB HBM3, using Meta’s Grand Teton AI server platform ([Matt Bowman, 2022](#)). Each server is equipped with eight GPUs and two CPUs. Within a server, the eight GPUs are connected via NVLink. Training jobs are scheduled using MAST ([Choudhury et al., 2024](#)), Meta’s global-scale training scheduler.

✦ Answer

The cost of training Llama 3.1 405B is estimated to be between \$60 million and \$640 million. Meta used over 16,000 Nvidia H100 GPUs to train the model on more than 15 trillion tokens ² ⁴. The wide range in cost estimates is due to different factors considered:

“Oh yea, we wrote a new framework just for this model”

Pre-Training: Towards Ultimate Training Efficiency

- We design an FP8 mixed precision training framework and, for the first time, validate the feasibility and effectiveness of FP8 training on an extremely large-scale model.
- Through the co-design of algorithms, frameworks, and hardware, we overcome the communication bottleneck in cross-node MoE training, achieving near-full computation-communication overlap. This significantly enhances our training efficiency and reduces the training costs, enabling us to further scale up the model size without additional overhead.
- At an economical cost of only 2.664M H800 GPU hours, we complete the pre-training of DeepSeek-V3 on 14.8T tokens, producing the currently strongest open-source base model. The subsequent training stages after pre-training require only 0.1M GPU hours.

Pipeline parallelism - DualPipe

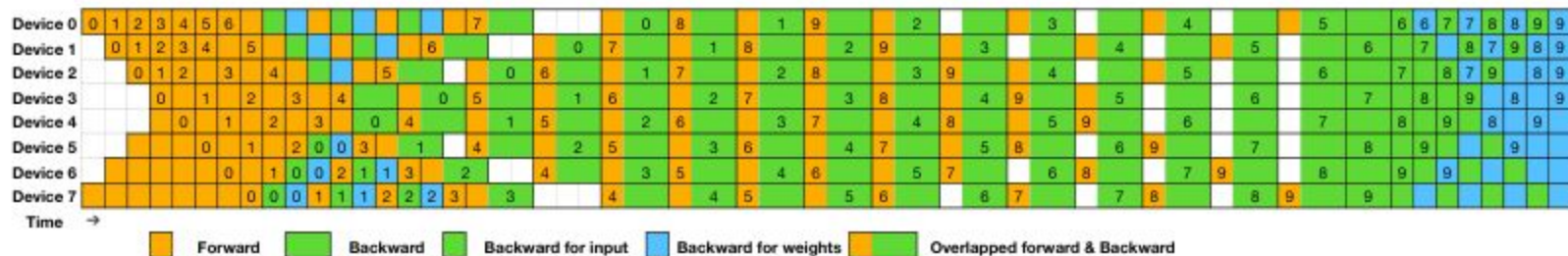
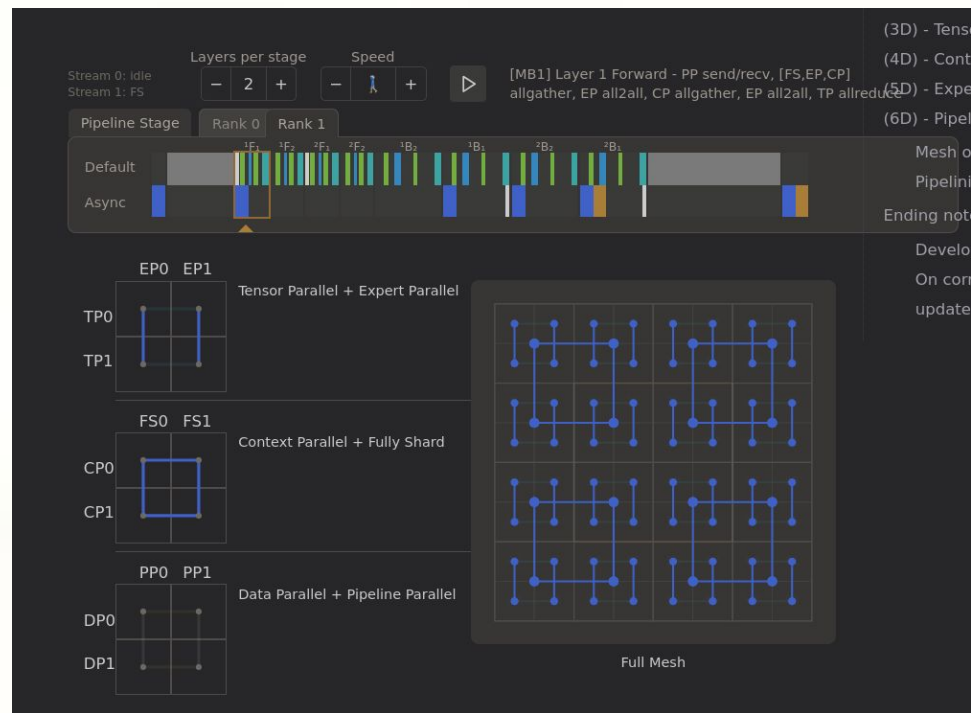


Figure 5 | Example DualPipe scheduling for 8 PP ranks and 20 micro-batches in two directions. The micro-batches in the reverse direction are symmetric to those in the forward direction, so we omit their batch ID for illustration simplicity. Two cells enclosed by a shared black border have mutually overlapped computation and communication.

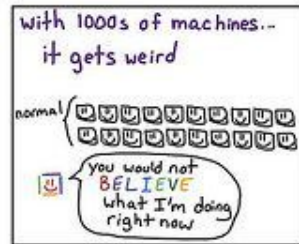
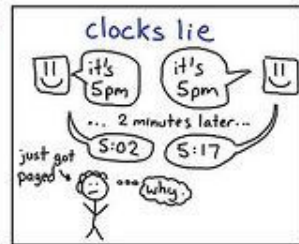
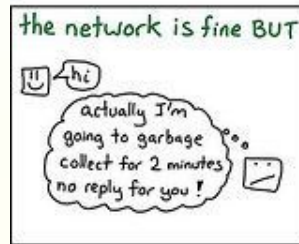
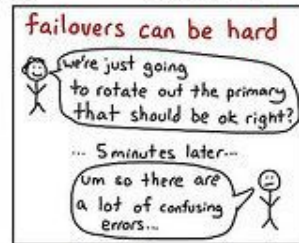
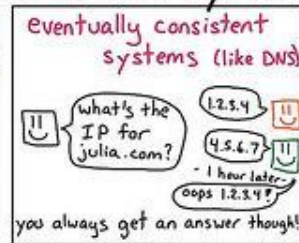
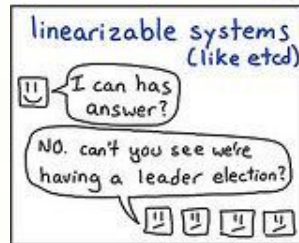
Intermission - Distributed Computing

<https://main-horse.github.io/posts/visualizing-6d/>



scenes from distributed systems

JULIA EVANS
@bork



Custom Kernels For Everything

Scaling too memory intensive -> use quantized training

Backprop makes pipelining not ideal -> GET RID OF BACKPROP

Summary of Custom Kernels

Kernel Type	Purpose
Cross-Node All-to-All Communication	Efficient token dispatch and combine across nodes for MoE routing.
FP8 GEMM	Accelerate matrix multiplications in FP8 precision.
FP8 Quantization/Dequantization	Quantize and dequantize activations and weights for FP8 training.
MoE Gating and Routing	Compute routing decisions and ensure balanced expert utilization.
Memory-Efficient RMSNorm/MLA	Recompute operations during backpropagation to save memory.
Low-Precision Activation Storage	Cache activations in FP8 format for backward pass.
Inference-Specific Kernels	Optimize MoE routing and load balancing during inference.
Custom PTX Instructions	Fine-tune communication kernels for IB and NVLink.
Warp-Level Cast	Speed up format conversion between FP8 and higher precision.
Transposed GEMM	Handle matrix transposition efficiently during FP8 training.

Example: FP8(?) GEMM

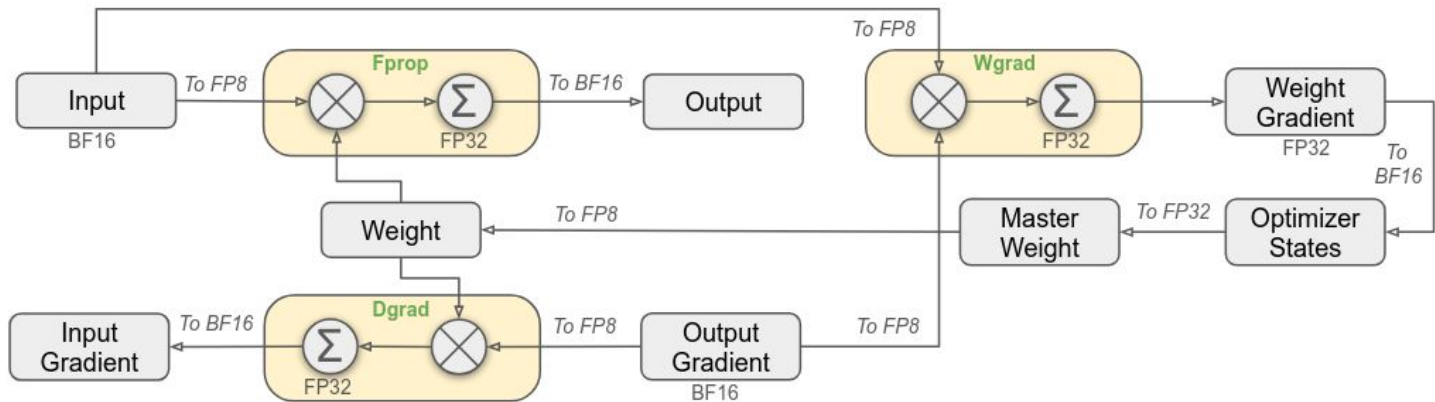


Figure 6 | The overall mixed precision framework with FP8 data format. For clarification, only the Linear operator is illustrated.

Example: FP8(?) GEMM

Increasing Accumulation Precision. Low-precision GEMM operations often suffer from underflow issues, and their accuracy largely depends on high-precision accumulation, which is commonly performed in an FP32 precision (Kalamkar et al., 2019; Narang et al., 2017). However, we observe that the accumulation precision of FP8 GEMM on NVIDIA H800 GPUs is limited to retaining around 14 bits, which is significantly lower than FP32 accumulation precision. This problem will become more pronounced when the inner dimension K is large (Wortsman et al., 2023), a typical scenario in large-scale model training where the batch size and model width are increased. Taking GEMM operations of two random matrices with $K = 4096$ for example, in our preliminary test, the limited accumulation precision in Tensor Cores results in a maximum relative error of nearly 2%. Despite these problems, the limited accumulation precision is still the default option in a few FP8 frameworks (NVIDIA, 2024b), severely constraining the training accuracy.

Example: FP8(?) GEMM

In order to address this issue, we adopt the strategy of **promotion to CUDA Cores** for higher precision (Thakkar et al., 2023). The process is illustrated in Figure 7 (b). To be specific, during MMA (Matrix Multiply-Accumulate) execution on Tensor Cores, intermediate results are accumulated using the limited bit width. Once an interval of N_C is reached, these partial results will be copied to FP32 registers on CUDA Cores, where full-precision FP32 accumulation is performed. As mentioned before, our fine-grained quantization applies per-group scaling factors along the inner dimension K . These scaling factors can be efficiently multiplied on the CUDA Cores as the dequantization process with minimal additional computational cost.

Intermission - Imfao

3.5. Suggestions on Hardware Design

Based on our implementation of the all-to-all communication and FP8 training scheme, we propose the following suggestions on chip design to AI hardware vendors.

Higher FP8 GEMM Accumulation Precision in Tensor Cores. In the current Tensor Core implementation of the NVIDIA Hopper architecture, FP8 GEMM (General Matrix Multiply) employs fixed-point accumulation, aligning the mantissa products by right-shifting based on the maximum exponent before addition. Our experiments reveal that it only uses the highest 14

Support for Tile- and Block-Wise Quantization. Current GPUs only support per-tensor quantization, lacking the native support for fine-grained quantization like our tile- and block-wise quantization. In the current implementation, when the N_C interval is reached, the partial results will be copied from Tensor Cores to CUDA cores, multiplied by the scaling factors, and

Support for Online Quantization. The current implementations struggle to effectively support online quantization, despite its effectiveness demonstrated in our research. In the existing process, we need to read 128 BF16 activation values (the output of the previous computation) from HBM (High Bandwidth Memory) for quantization, and the quantized FP8 values are then written back to HBM only to be read again for MMA. To address this inefficiency, we

Deepseek v3 \approx Claude

5.4.2. Self-Rewarding

Rewards play a pivotal role in RL, steering the optimization process. In domains where verification through external tools is straightforward, such as some coding or mathematics scenarios, RL demonstrates exceptional efficacy. However, in more general scenarios, constructing a feedback

mechanism through hard coding is impractical. During the development of DeepSeek-V3, for these broader contexts, we employ the constitutional AI approach (Bai et al., 2022), leveraging the voting evaluation results of DeepSeek-V3 itself as a feedback source. This method has produced notable alignment effects, significantly enhancing the performance of DeepSeek-V3 in subjective evaluations. By integrating additional constitutional inputs, DeepSeek-V3 can

Results - SOTA

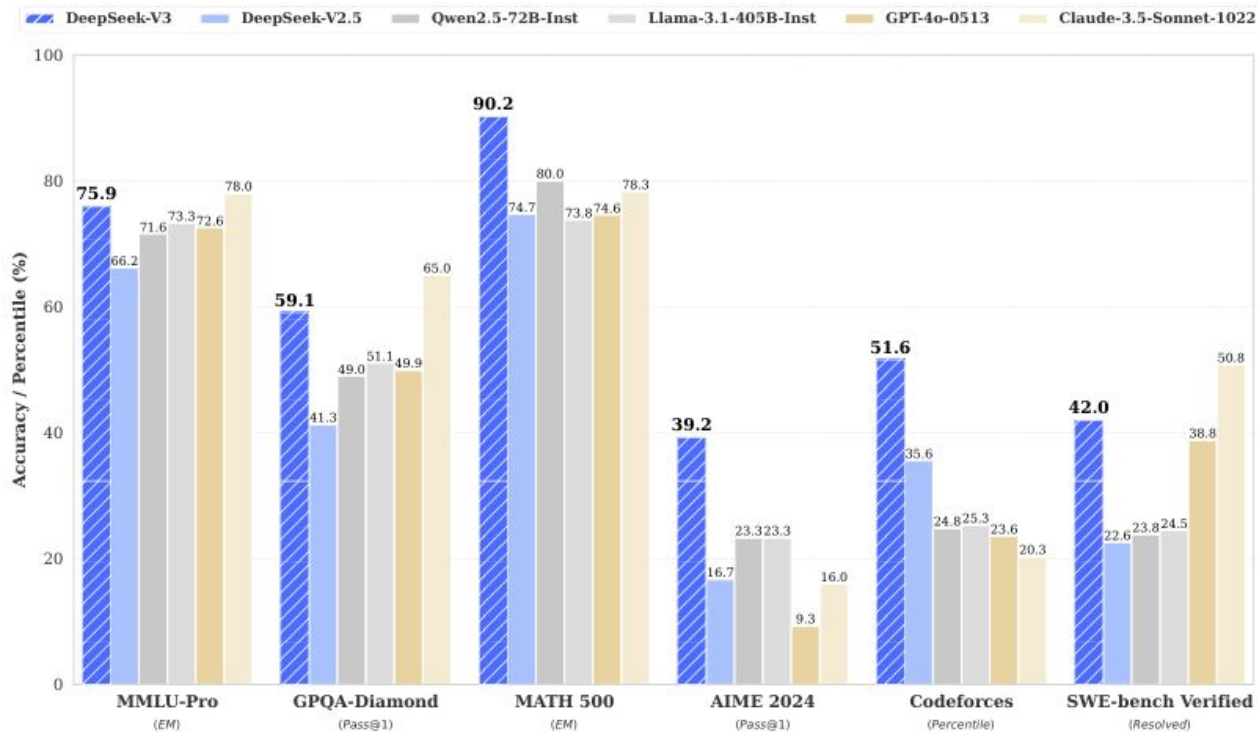


Figure 1 | Benchmark performance of DeepSeek-V3 and its counterparts.

Results - Perfect needle in haystack

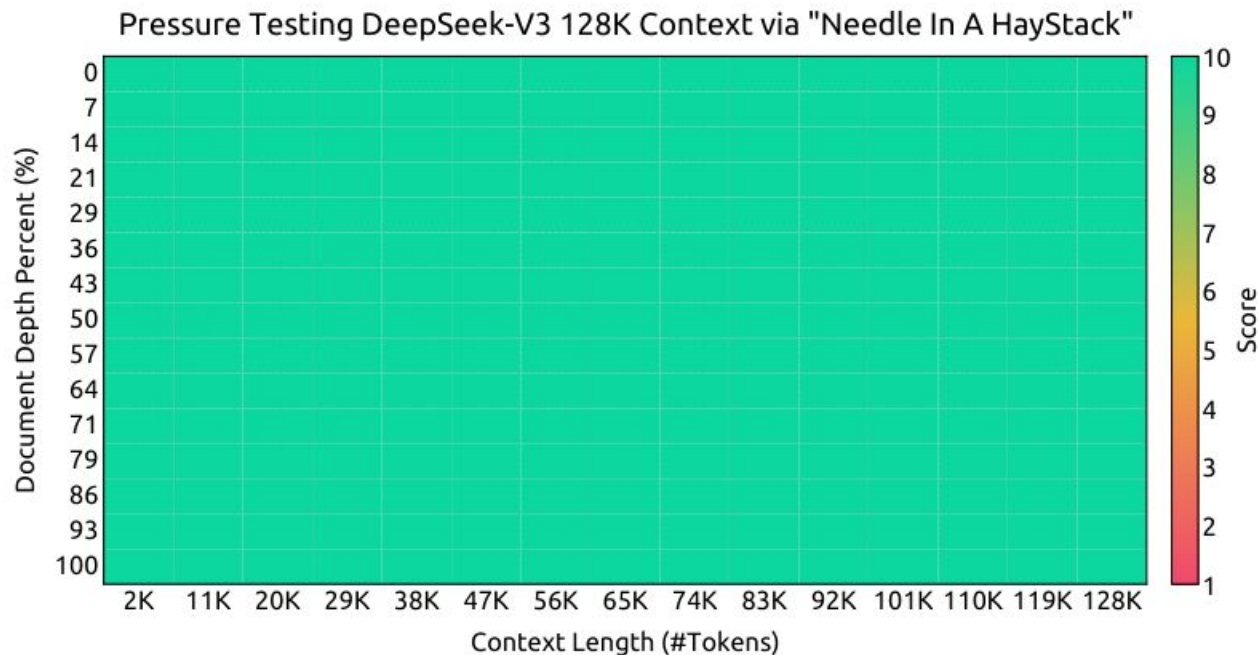


Figure 8 | Evaluation results on the "Needle In A Haystack" (NIAH) tests. DeepSeek-V3 performs well across all context window lengths up to 128K.

Results - It's free software (both as in FREE beer and FREEdom)

deepseek-ai/DeepSeek-V3 like 1.94k Follow

Safetensors deepseek_v3 custom_code fp8 arxiv:2412.19437

Model card Files Community 58

main DeepSeek-V3

DeepSeekDDM Update README.md 4c1f24c

figures

inference

.gitattributes	1.52 kB	
LICENSE-CODE	1.06 kB	
LICENSE-MODEL	13.8 kB	
README.md	22.6 kB	
README_WEIGHTS.md	3.66 kB	
config.json	1.73 kB	
configuration_deepseek.py	10.6 kB	
model-0001-of-000163.safetensors	5.23 GB	LFS

DeepSeek-V3 Multi-Token Prediction Analysis

3. Averaging the losses across depths with a weighting factor λ .

This approach densifies the training signal, improves the model's ability to plan ahead, and can be repurposed for speculative decoding during inference. While it adds complexity to the training process, it significantly enhances the model's performance and coherence.

say hi, DeepseekV3! i'm putting you in the slides for my talk

< 2 / 2 >



Hello! I'm DeepSeek-V3, and I'm thrilled to be part of your talk slides! Let me know if you need any help explaining my architecture, training process, or anything else—I'm here to make your presentation shine! 🚀



New chat

Message DeepSeek

DeepThink

Search



Results - R(1)easoning

Reasoning Data. For reasoning-related datasets, including those focused on mathematics, code competition problems, and logic puzzles, we generate the data by leveraging an internal DeepSeek-R1 model. Specifically, while the R1-generated data demonstrates strong accuracy, it suffers from issues such as overthinking, poor formatting, and excessive length. Our objective is to balance the high accuracy of R1-generated reasoning data and the clarity and conciseness of

Model	LiveCodeBench-CoT		MATH-500	
	Pass@1	Length	Pass@1	Length
DeepSeek-V2.5 Baseline	31.1	718	74.6	769
DeepSeek-V2.5 +R1 Distill	37.4	783	83.2	1510

THEOREM PROVER

Results - Cost to performance

Providers for DeepSeek V3

OpenRouter routes requests to the top-ranked providers able to handle your prompts. ⓘ

● DeepSeek	Context	Max Output	Input	Output	Latency	Throughput	⌵
ⓘ ⓘ fp8 ⓘ ⓘ	64K	8K	\$0.14	\$0.28	3.78s	42.39t/s	
● DeepInfra	Context	Max Output	Input	Output	Latency	Throughput	⌵
ⓘ ⓘ ⓘ ⓘ ⓘ	16K	16K	\$0.85	\$0.9	5.49s	4.77t/s	
● NovitaAI	Context	Max Output	Input	Output	Latency	Throughput	⌵
ⓘ ⓘ ⓘ ⓘ ⓘ	66K	4K	\$0.89	\$0.89	11.55s	13.69t/s	
● Together	Context	Max Output	Input	Output	Latency	Throughput	⌵
ⓘ ⓘ ⓘ ⓘ ⓘ	131K	2K	\$1.25	\$1.25	6.39s	15.38t/s	

Providers for Claude 3.5 Sonnet

OpenRouter routes requests to the top-ranked providers able to handle your prompts. ⓘ

Standard ⓘ

● Amazon BedRock	Context	Max Output	Input	Output	Latency	Throughput	⌵
ⓘ ⓘ ⓘ ⓘ ⓘ	200K	8K	\$3	\$15	2.41s	36.55t/s	
● Google Vertex	Context	Max Output	Input	Output	Latency	Throughput	⌵
ⓘ ⓘ ⓘ ⓘ ⓘ	200K	8K	\$3	\$15	1.73s	48.23t/s	
● Anthropic	Context	Max Output	Input	Output	Latency	Throughput	⌵
ⓘ ⓘ ⓘ ⓘ ⓘ	200K	8K	\$3	\$15	2.49s	61.50t/s	

Results - Who cares about cost?

- Test time compute
- Cheaper inference -> cheaper data -> cheaper training

