PURETENSOR
INTELLIGENCE DISTILLED

Most of the Slowdown Was Not the Weights

A Vendor 4-Bit Build of DeepSeek-V4.1-Flash-1017 Measured With a Same-Weights Control Arm

Paper: PT-R-2026-061 v1.0

Author: PureTensor Ltd

Date: 31 October 2026

Status: Published

Abstract

At the end of October 2026 NVIDIA published an NVFP4 build of DeepSeek-V4.1-Flash-1017, the model we serve as a resident seat on one node of four workstation Blackwell GPUs. NVFP4 is a 4-bit floating-point format with a scale for every 16 values; the build quantises the routed experts (the mixture-of-experts layers holding most of the parameters) to 4-bit weights and activations. We asked whether the seat should switch. The obvious comparison mixes two effects: the production engine has a fused decode kernel that only understands the source checkpoint's expert layout, and the vendor build cannot use it. We added a control arm, the source weights on the same patched engine the vendor build needs, with that kernel off.

On batch-1 code decode the production seat ran at 580.7 tokens per second, the control arm at 464.3 and the NVFP4 build at 407.1. The engine path alone cost 20.0% and the weights a further 12.3%, 29.9% in all. In tokens per second lost (173.6), 67% belongs to the engine path and 33% to the weights, on that prompt. The share is not constant: across five workloads it runs from 52% to 86% engine, and the weights penalty tends to shrink as concurrency rises. All of it comes from one unrandomised window, with the production seat measured last.

Quality was compared between the production seat and the NVFP4 build only; the control arm was not run on the quality suites. Six suites and 93 paired items from our internal benchmark showed no broad difference, apart from one post hoc gap in counting among distractors in long documents (25 against 17 of 40, $p = 0.039$ uncorrected, about 0.19 corrected), which we treat as a hypothesis. Perplexity rose 1.9%, but identical runs on the same arm agree on only about 94% of top-1 tokens, so every fidelity number needs that floor. The engine change left the output distribution within the noise; the NVFP4 weights moved it by about twice the source arm's floor (about 1.3 times the build's own, noisier floor). Resident expert memory grew 5.9% and the KV-cache pool shrank 30.5%. The seat stays on the source weights.

The build was validated by NVIDIA on datacenter Blackwell without speculative decoding. These results describe workstation Blackwell, this engine path and batch sizes of 1 to 16 only, not the build where it was meant to run.

1. Why a Third Arm

DeepSeek-V4.1-Flash-1017 was released on 17 October 2026 and we serve it on one compute node with four RTX PRO 6000 Blackwell Max-Q cards (96 GB each, 325 W cap, workstation class), tensor parallel across all four. A 4-bit build promises faster decode, so the question was fair.

The obvious comparison is hard to interpret. The production seat runs an engine with a fused decode kernel (one GPU kernel doing several expert-computation steps together), hard-wired to the source checkpoint's expert layout, which the vendor checkpoint does not share. Any speed gap between the production seat and the vendor build is the sum of what the weights do and what losing the kernel does.

So we ran three arms. The new arm, C, uses the source checkpoint on the patched engine the vendor build needs, with the fused kernel off. A to C changes only the engine; C to B only the weights. Speed and fidelity (how far the output distribution moves) have all three arms; the quality suites have two, so the quality comparison carries the engine as a confound, which Section 5 argues is small. Our October post on this seat's decode speed reported lower code-prompt figures than the 580.7 tokens per second here; we did not reconcile the two.

2. What the Checkpoint Was Built For

The model card describes routed experts converted from the source MXFP4 format to NVFP4 for weights and activations (group size 16). It reports that all 16,986,931,200 expert weight blocks passed a lossless conversion check, while adding that lossless weights do not imply identical outputs. The Engram lookup tables (a large memory of learned embeddings consulted per token) are converted from FP8 to MXFP4, which the card says is lossy. Its validated setup is four NVIDIA GB300 GPUs (datacenter Blackwell) on vLLM with an upstream pull request open at the time; speculative decoding was not exercised; its reported accuracy is within about 0.7 points of the source on all six benchmarks it lists. Our hardware, engine, speculative decoding and batch sizes of 1 to 16 are all outside that; nothing below says the build is slow where it was made to run.

3. Arms, Patches and Settings

The arms were A (source checkpoint, production engine build with the fused decode kernel), C (the same source bytes on the patched engine build with the kernel off, isolating the engine) and B (the NVIDIA NVFP4 checkpoint on the same patched build as C). They ran on the same four GPUs and never at the same time.

Three local engine changes were needed for B and C. The 40 expert layers listed as NVFP4 in the manifest are routed to the engine's native NVFP4 mixture-of-experts backend, with the speculative draft experts left MXFP4 as shipped. A switch turns the fused decode kernel off, the only difference between A's engine and C's. The engine had no MXFP4 Engram path, so we expanded the NVIDIA tables to FP8 without loss (every E2M1 value is exactly representable in E4M3, and 20,000 random rows per table verified identical). B therefore runs NVIDIA's exact Engram values without the memory saving. All 50 large files were checksum-verified.

Batch-1 speed used temperature 0, streaming, the model's own draft head, an idle node, arm order B, C, A, and five runs per no-thinking prompt or three per thinking prompt after a warm-up. Sustained decode, prefill and retrieval used an open-source benchmark with 30-second cells. Quality used our six-suite bake-off (extended budget, temperature 1.0 to match the vendor) and our internal frontier benchmark (94 original items, ten dimensions, code-produced verdicts, three rollouts per item, all required to pass). Fidelity used 120 excerpts of our code, notes and reports (207,735 teacher-forced tokens, top-5 log-probabilities), each arm scored twice, plus 50 greedy completions generated twice per arm.

Caveats apply throughout. A's quality suites ran on the live production seat, which also carried other traffic, while B's ran alone on the test endpoint. A's speed was measured last, after a fresh restart, and arm order was not randomised. The 13-probe seat check failed the same one probe on every arm, and the frontier benchmark's serving-path pre-flight reported the same six blocking findings on both arms and was run warn-only; neither separates the arms.

4. Speed: How the Loss Divides

workload	A production	C source, patched engine	B NVFP4, patched engine	A to C (engine)	C to B (weights)
code prompt, no thinking (n 5)	580.7	464.3	407.1	-20.0%	-12.3%
reasoning prompt, no thinking (n 5)	316.8	258.4	230.3	-18.4%	-10.9%
2.7k-token code prompt, no thinking (n 5)	595.8	495.4	402.0	-16.9%	-18.9%
code prompt, thinking high (n 3)	462.3	352.3	334.1	-23.8%	-5.2%
reasoning prompt, thinking high (n 3)	348.8	290.9	259.8	-16.6%	-10.7%

Table 1. Batch-1 decode in tokens per second (mean of runs, speculative decoding on), with the change at each step.

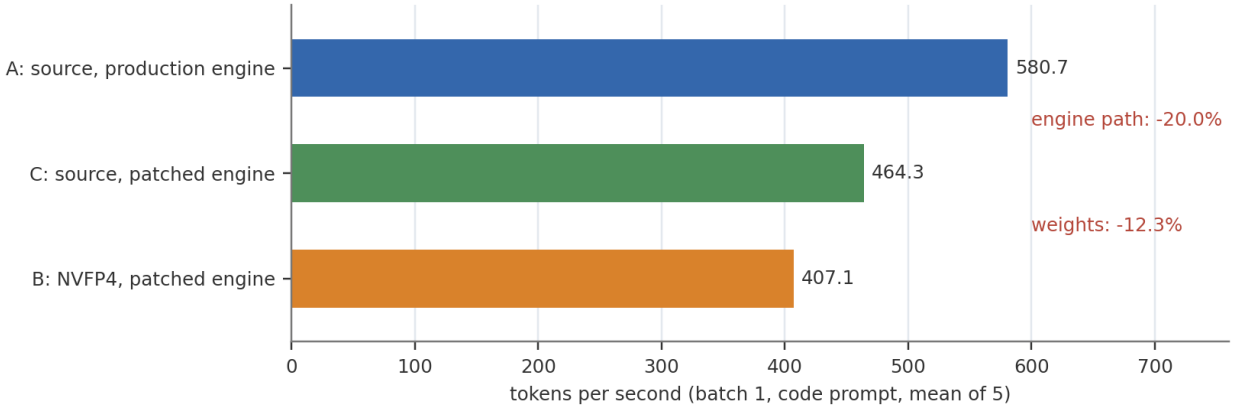


Figure 1. Batch-1 code-prompt decode in tokens per second for the three arms. Values are in Table 1.

On the code prompt the engine path costs 20.0% and the weights a further 12.3%, 29.9% in all; the two multiply, since 0.800 times 0.877 is 0.701. In absolute terms, of the 173.6 tokens per second lost between A and B, 116.4 (67%) belongs to the engine path and 57.2 (33%) to the weights. That split depends on the workload and on the order in which the steps are taken. Computed from Table 1, the engine share of tokens lost is 67% on the code prompt, 68% on the reasoning prompt, 52% on the 2.7k-token prompt (where the weights step is the larger), 86% on thinking-high code and 65% on thinking-high reasoning. Across the five workloads the engine step costs 16.6 to 23.8% and the weights step 5.2 to 18.9%.

On the code prompt the single-run ranges do not overlap between arms; on thinking-high code, C and B do, so that 5.2% step is not resolved at three runs.

Speculative decoding throughput is tokens accepted per step divided by time per step. On the code prompt, accepted tokens per step were 5.15 (A), 5.03 (C) and 4.86 (B), and the implied step times (our derivation, not a measured quantity) 8.86, 10.84 and 11.93 ms. From A to C acceptance falls 2% and step time rises 22%, so the engine penalty is almost all a longer step: the fused kernel removes work per step. From C to B acceptance falls 3.4% and step time rises 10%. On this prompt a quarter to a third of the weights penalty is lower draft acceptance, which

fits the draft experts remaining MXFP4 while the main model's activations differ. Not generally: on thinking-high code B accepts more tokens per step than C (3.91 against 3.75), and on the 2.7k-token prompt acceptance is 10.7% lower.

concurrency	A	C	B	B vs C
1	305.7	253.8	215.7	-15.0%
2	404.7	331.4	298.5	-9.9%
4	577.7	495.3	439.5	-11.3%
8	795.7	639.1	598.6	-6.3%
16	1045.9	911.6	851.0	-6.6%

Table 2. Sustained aggregate decode in tokens per second at zero context (30-second cells).

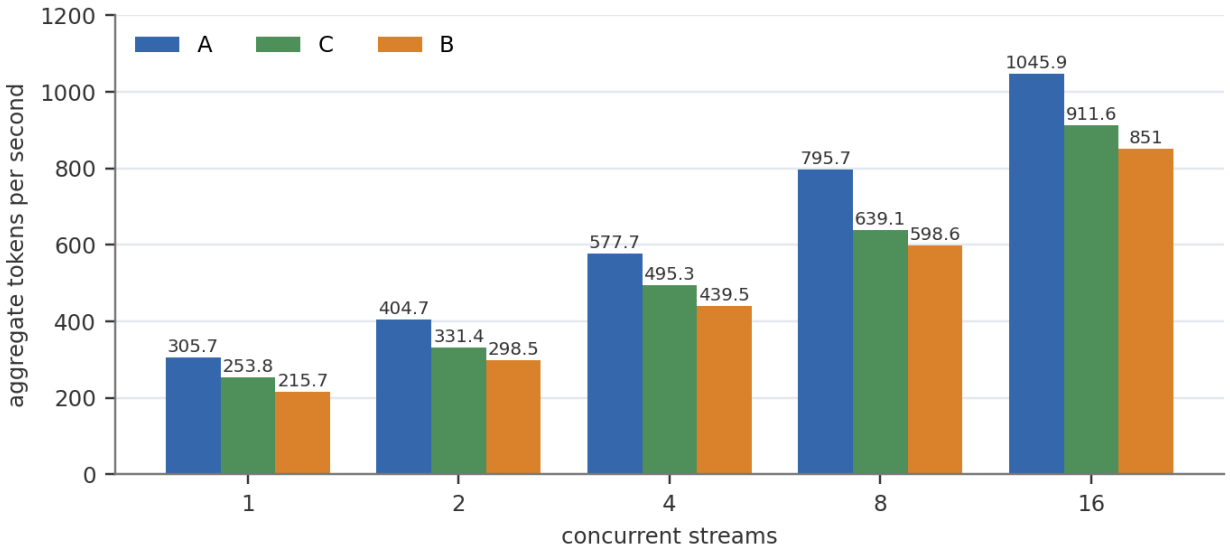


Figure 2. Sustained decode against concurrency. Values are in Table 2.

The weights penalty tends to be smaller at higher concurrency: 15.0% at one stream, 6.3 and 6.6% at 8 and 16, though not monotonically (9.9% at 2, 11.3% at 4). The engine penalty, A to C, is 17.0, 18.1, 14.3, 19.7 and 12.8% at 1, 2, 4, 8 and 16 streams, with no clean trend. Each cell is a single 30-second window, and the 32-stream cell was capacity-limited to 16 running sequences, so we do not quote it.

B draws less board power at one stream (about 1,040 W against 1,210 W for A and 1,224 W for C) but decodes slower, so its energy per token is no lower than C's. At 128k context and one stream B is 15.0% below C and 25.9% below A. Prefill goes the other way: B is 2 to 7% faster than A from 8k tokens upward and 3 to 4% slower at 1k and 4k. A 134k-token retrieval test scored 10 of 10 on all three arms.

5. Fidelity: A Noise Floor Before Any Effect

Each arm was teacher-forced over the 120 texts and scored twice, the second pass being the noise-floor repeat. KL divergence is computed over the union of the two top-5 sets, an approximation in which a token missing from one side gets that side's fifth-best log-probability.

pair	perplexity	top-1 agreement	KL (top-5 union)	greedy exact match (50 prompts)
A vs A (repeat)	5.0066 / 5.0086	93.92%	0.01424	16 of 50 (32%)
C vs C (repeat)	5.0081 / 5.0028	93.86%	0.01431	15 of 50 (30%)
A vs C (engine only, same weights)	5.0066 / 5.0081	93.91%	0.01432	14 of 50 (28%)
B vs B (repeat)	5.1004 / 5.0995	92.38%	0.02179	7 of 50 (14%)
A vs B	5.0066 / 5.1004 (+1.9%)	91.50%	0.02797	2 of 50 (4%)
C vs B (weights only, same engine)	5.0081 / 5.1004 (+1.8%)	91.40%	0.02800	0 of 50 (0%)

Table 3. Fidelity on 120 texts and greedy agreement on 50 prompts; same-arm repeats are the noise floor.

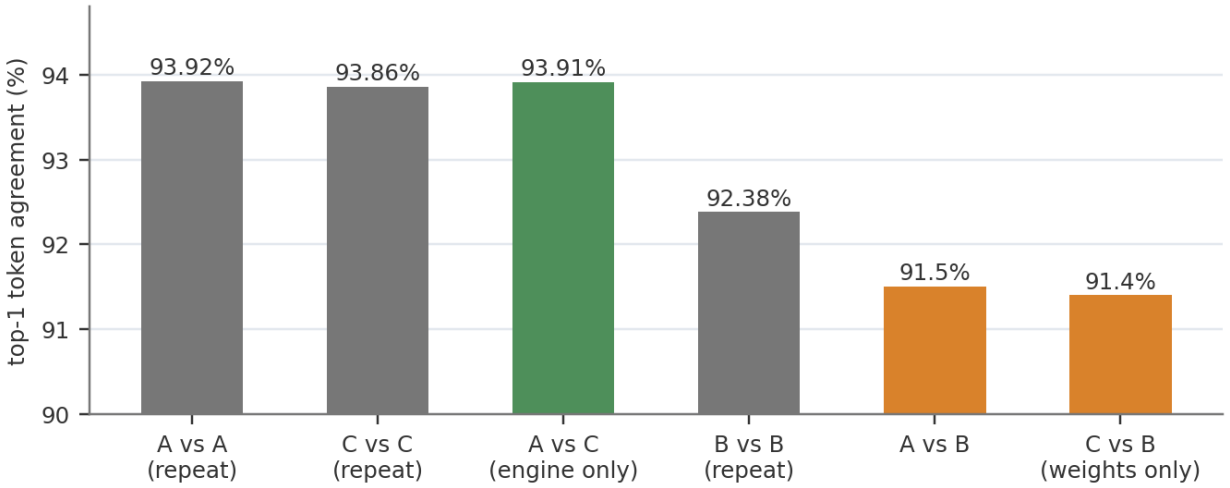


Figure 3. Top-1 agreement for repeats and cross-arm pairs; the axis starts at 90%. Values are in Table 3.

The first finding is the floor: two scorings of the same arm agree on only about 94% of top-1 tokens and differ by a KL of about 0.014. We did not isolate the cause; no fidelity number is interpretable without a repeat.

The second is the control. A against C sits on the floor: 93.91% top-1 agreement against 93.92% for the A repeat, KL 0.01432 against 0.01424. The engine difference is a speed difference and not a numerical one, which is why we do not regard the engine as a large confound in the quality comparison, though without a quality run on C we cannot rule out a small one.

The third is the weights. B moves the distribution by about twice the source arm's floor in KL (0.0280 against 0.0142), but B is itself less repeatable (own-repeat KL 0.0218, 92.38% top-1), so against its own floor the shift is about 1.3 times in KL and 0.9 points in top-1. The shift is real but modest, not a clean twofold signal. B adds 1.8 to 1.9% to perplexity, between 1.7% (code) and 2.0% (notes, reports) by source.

Greedy completions are noisy. Same-arm repeats of the source weights are identical in only 30 to 32% of prompts, and B's own repeat in 14%; A against B matches 4% and C against B 0%, so the build visibly lowers greedy stability, but we did not isolate how much of any of this is speculative-decoding or kernel numerics and how much quantisation.

NVIDIA states the expert weight conversion is lossless, so the shift should come from the 4-bit activations and the lossy MXFP4 Engram tables. We did not run a mode keeping 8-bit activations with NVFP4 weights, so this attribution is by elimination and untested.

6. Quality: No Broad Difference, With One Gap We Do Not Trust

suite	n	A source	B NVFP4	B-only / A-only discordant	p
reasoning (AIME 2026, HMMT 2026, GPQA)	261	90.0% (235)	89.7% (234)	7 / 8	1.0
competition maths, average of 4 samples	252 samples	92.9%	94.0%	not paired	CIs overlap
long context 32k to 240k	180	91.1% (164)	86.7% (156)	3 / 11	0.057
vision (charts, tables, logs, pages)	116	90.5% (105)	95.7% (111)	7 / 1	0.07

Table 4. Six-suite bake-off at extended budget, exact McNemar on paired items where items are paired.

Knowledge (300 items, 26.3% against 27.0%, p 0.89) and coding (95 real defects, 97.9% against 98.9%, p 1.0) were level. Writing (blind pairwise, 30 prompts) was A 15 wins, B 13, two ties. No suite is significant before the post hoc look below. B is numerically lower on reasoning, long context and writing, and higher on vision (7 to 1 discordant, although the vision tower is identical), which with six uncorrected paired tests we treat as sampling noise. At these sample sizes we see no broad quality loss.

The long-context gap sits in one family of five, which asks how often a planted item occurs in a long document of distractors. Counting scored 25 of 40 (62%) for A and 17 of 40 (42%) for B; the other four families were level or nearly so (40 and 40, 40 and 39, 39 and 40, 20 and 20). Paired, A alone is right on 10 counting items and B alone on 2, exact McNemar p = 0.0386. We found the family after seeing the results; Bonferroni-corrected over five families, p is about 0.19. Both arms undercount (12 of A's 15 wrong answers, 19 of B's 23), and truncation does not explain it. The vendor's long-context test shows no loss and we did not repeat the family, so this is a hypothesis, not a finding.

On our internal frontier benchmark (low effort, temperature 0, three rollouts per item with all three required to pass) A passed 82 of 93 scored items and B 81 of 94. One grounding item was unscorable for A, so the harness marks A's run partial (coverage 99%) and renormalises its composites. On the 93 items scored in both it is 82 against 81, exact McNemar p = 1.0, with three discordant items (one B passes, two A passes).

The composites are less comfortable. The agentic seat composite is 84.1 for A and 84.3 for B, but the scorer composite is 92.1 against 88.0 and the flat composite 87.8 against 85.8: B is lower by 4.1 and 2.0 points (2.7 and 0.9 against A's floors of 90.7 and 86.7, which count unmeasured weight as unproven). With 8 to 12 items per dimension, the item-level test cannot separate the arms and B is still a little lower on the dimensions weighted into those seats. B also emitted 10.3% more completion tokens here, against 0.7% fewer on the six suites.

Calibration (a stated probability of being right, scored by Brier score) was identical: 0.0565 on both arms against a random floor of 0.1875, with per-item probabilities within 0.01. With ten items this is a small-sample contrast to a different model's 4-bit build we published in September, which lost calibration; model, hardware and format all differ.

7. Memory: Smaller on Disk, Larger on the GPU

On disk the checkpoint shrinks from 475.3 to 399.5 GiB (-16%), with the Engram tables falling from 188.8 to 97.3 GiB (-48%). On the GPU it goes the other way. The routed experts grow from 268.9 to 284.8 GiB (+5.9%) because NVFP4 carries a scale for every 16 values where the source carries one for every 32, and weights per GPU at load rise from 74.09 to 78.04 GiB. Every arm gives the engine the same GPU memory fraction, so the engine's start-up report of KV-cache capacity falls from 12,538,594 tokens (C) to 8,710,580 (B), a 30.5% squeeze (A's figure was 10,160,282). Our runs expanded the Engram tables to FP8, so the Engram saving was not realised on the GPU path.

8. What These Numbers Will Not Carry

This is one window on one node on one day, with five runs per speed cell (three for thinking prompts) and the thinking-high weights step unresolved. The engine and weights split holds for this engine, this fused kernel and batch sizes 1 to 16, and varies from about half to most of the loss across workloads. The engine share is also definitional: it is the fused kernel that exists only for the source expert layout, so part of the engine penalty is a consequence of the NVFP4 layout being unable to use it. That is accurate for what we changed, but it does not make the format blameless.

On quality, the comparison is A against B only, with one sample per item at temperature 1.0, six uncorrected paired tests and one post hoc subgroup. What survives is the absence of a broad quality loss at these sample sizes, and item-level parity on the internal benchmark. What does not survive is any claim that the counting deficit is real, that vision improved, or that the scorer and flat composites are equal. Whether C matches A on the quality suites is unknown. The writing suite was graded by blind language-model graders, and the KL is a top-5 approximation, but the measured floors keep the A to C conclusion sound.

9. What We Changed

The resident seat stays on the source weights and production engine. The test endpoint was removed and the production seat was started again on the source weights and passed its probe; the service's restart counter stayed at zero. Two methods changed: every fidelity comparison now carries a same-arm repeat as its floor, and every candidate speed comparison now includes the candidate's engine on the incumbent's weights. The next experiment is the 8-bit-activation mode with NVFP4 weights, which tests the one attribution that rests on elimination, followed by a repeat of the counting family with more items and a run on the hardware and engine the build was validated on.

10. Methods and Artefacts

The engine is a vLLM-derived build with a community native-kernel package for this GPU generation; speculative decoding uses the model's draft head, configured for 7 speculative tokens. Sustained decode, prefill and retrieval used an open-source decode benchmark, version 0.6.2; batch-1 speed used our single-stream client with the public leaderboard's code and reasoning prompts plus a 2.7k-token code prompt; quality used our six-suite harness and our internal frontier benchmark, version 1.1. Statistics are exact two-sided McNemar tests for paired items, Wilson 95% intervals, a bootstrap for the maths average, and a Bonferroni correction over five families for the counting test. The vendor checkpoint (nvidia/DeepSeek-V4.1-Flash-1017-NVFP4) and the source (deepseek-ai/DeepSeek-V4.1-Flash-1017) are public. We retain the raw speed records, benchmark output, power logs, scored responses, teacher-forced score files and engine start-up

logs.

PureTensor operates a sovereign AI infrastructure fleet (on-premises GPU compute, storage, and serving) run day-to-day by autonomous agents under human direction. This paper is PT-R-2026-061. All measurements were taken on 31 October 2026, the last finishing at about 13:07 UTC; raw artefacts are retained.