NVFP4 PRECISION LAB • PUBLIC RESULTS

4-bit Nemotron matches BF16 on code

until it starts reasoning

NVFP4 is NVIDIA's 4-bit floating-point format: E2M1 values in blocks of 16 with an FP8 scale per block, decoded natively by Blackwell tensor cores. We wanted to know what it costs a small model in the place that matters, **hidden test cases**, not perplexity. So we quantized **Llama-3.1-Nemotron-Nano-4B-v1.1** with one standard post-training recipe and scored it against its own BF16 weights on 542 HumanEval+ and MBPP+ tasks, with thinking off, thinking on, and sampled.

Headline result

With thinking off, NVFP4 matches BF16 on HumanEval+/MBPP+ (−0.5 pt, 95% CI ±4.5) and runs 1.9× faster at batch 1 from a 3.7 GB checkpoint. With reasoning on it loses 4–6 points. Best-of-8 with a selector that only sees the prompt's own examples closes that gap: −0.4 pt pooled over three independent runs (95% CI −2.1 to +1.4).

THINKING OFF	BATCH-1 SPEED	CHECKPOINT	BEST-OF-8, REASONING
−0.5 pt	**1.90×**	**3.7 GB**	**−0.4 pt**
NVFP4 − BF16, CI ±4.5	268 vs 141 tok/s in vLLM	vs ~9 GB BF16	pooled, 3 runs, 2 providers

Setup

	What we used
Model	nvidia/Llama-3.1-Nemotron-Nano-4B-v1.1, revision d552708a
Quantization	ModelOpt 0.47.0 PTQ, NVFP4_DEFAULT_CFG, group 16, 128 wikitext calibration samples; lm_head, embeddings and KV cache left in BF16
Hardware	NVIDIA RTX PRO 6000 Blackwell (sm_120, 96 GB), Colab G4 and Brev
Serving	vLLM 0.32.0, modelopt_fp4 kernels
Benchmark	EvalPlus 0.3.1: 164 HumanEval+ and 378 MBPP+ tasks, scored on the hidden plus tests
Statistics	Paired per task: bootstrap 95% CI resampling tasks within each dataset, exact McNemar p

Each headline comparison has a same-model repeat in a separate process. Batched decoding in vLLM is not batch-invariant, so a single-run comparison means nothing until it sits next to a same-model repeat. Results below are on the plus tests across all 542 tasks unless stated.

Thinking off: indistinguishable from BF16

System prompt `detailed thinking off`, greedy, up to 1,024 new tokens.

Run	HumanEval+ plus	MBPP+ plus	Truncated	Batched tok/s
BF16 r1	0.616	0.386	37	12,379
BF16 r2	0.622	0.373	33	12,377
NVFP4 r1	0.585	0.392	62	15,858
NVFP4 r2	0.592	0.397	61	15,852

Pairing	b − a	95% CI	Only a passes	Only b passes	p
BF16 r1 vs BF16 r2	−0.7 pt	[−3.1, +1.8]	26	22	0.67
BF16 r1 vs NVFP4 r1	**−0.5 pt**	[−5.0, +3.9]	75	72	0.87

Same pass rate, not the same model. 147 of 542 tasks flip between BF16 and NVFP4, in near-equal numbers each way. In a later session where two BF16 runs were byte-identical, NVFP4 still flipped 135, so those are precision effects, not scheduling noise. NVFP4 also hits the 1,024-token cap more often, but only 6 of the 75 tasks it loses were truncated.

The 1.9× is at batch 1 (268 vs 141 tok/s). Fully batched, the gain shrinks to 1.28×.

Thinking on: a 4–6 point gap

Same 542 tasks and the same checkpoint, `detailed thinking on`, greedy, up to 8,192 new tokens. The `<think>` block is stripped before scoring, and a trace that never closes counts as no answer.

Run	HumanEval+ plus	MBPP+ plus	All plus	Unclosed traces	Median tokens
BF16 think r1	0.829	0.645	0.701	72	2,022
BF16 think r2	0.835	0.661	0.714	63	2,041
NVFP4 think r1	0.780	0.606	0.659	80	2,431
NVFP4 think r2	0.787	0.595	0.653	77	2,409

All four cross pairings favour BF16: **−4.2, −4.8, −5.5 and −6.1 points**, $p \le 0.018$ each. In the same session, the thinking-off pair landed within noise again (−1.5, CI ±4.5). Each model's own repeat moved by at most 1.3 points, and 28 tasks passed in both BF16 runs while failing in both NVFP4 runs, against 12 the other way.

> **The equivalence holds for direct answers only**
> Quantization error that washes out over a 400-token answer appears to compound over a 2,000-token reasoning trace. NVFP4 traces run ~20% longer and close less often, yet most of its lost tasks are traces that closed with the wrong answer.

That is not a case against reasoning. Thinking is worth **+21 to +23 points** to both models. NVFP4 with thinking on (0.65) beats BF16 with it off (0.47) by a wide margin. The precision gap is real, but it is a fifth of the size of what reasoning buys.

Best-of-8 closes it

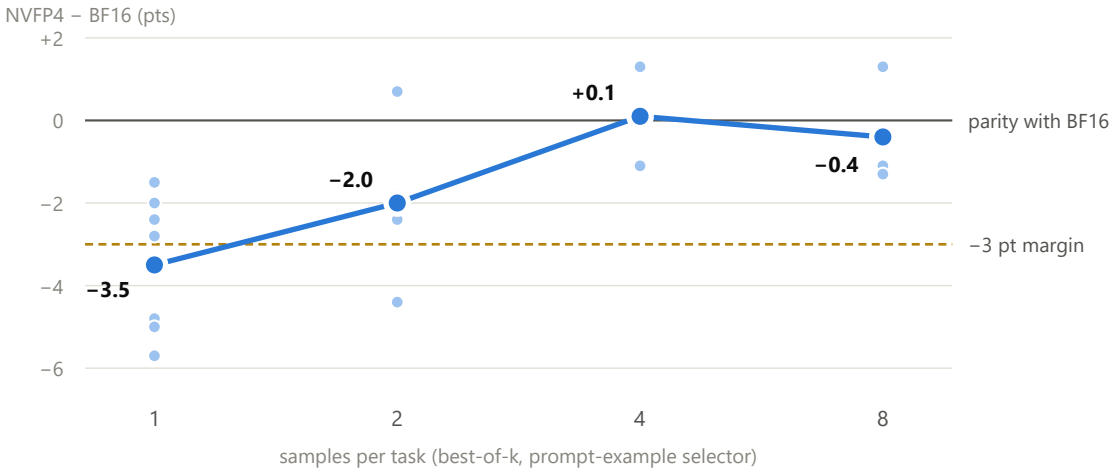
We switched to the model card's sampling (T 0.6, top-p 0.95), using the same seeds for both models, and drew 8 samples per task. To pick one answer, a selector runs each sample against the examples *written in the prompt*: HumanEval docstring lines (126 of 164 tasks) and MBPP's visible assert (378 of 378). The sample that passes the most examples wins, and a closed trace breaks ties. **No hidden test reaches the selector.**

	Greedy	pass@1 (sampled)	Selected@8	Oracle@8	Output tokens
BF16	0.701	0.706	0.775	0.825	13.3M
NVFP4	0.659	0.672	**0.788**	0.832	15.1M

One sampled answer still trails by ~3.5 points. Eight close the gap, because NVFP4's **oracle@8 is as high as BF16's**. A correct answer appears in its sample set just as often, only less often per sample. When at least one of the 8 samples is correct, the selector picks a passing one 94–95% of the time.

How the gap closes as samples are added

Light dots are disjoint best-of-k replicates from sampling run 1. At k = 8 they are the three independent runs, and the line follows their mean (pooled at k = 8).



Reproduced on new seeds and a second provider. We stated the claim before re-running: *NVFP4 best-of-8 falls no more than 3 points below BF16 best-of-8*. Two fresh sampling runs on Brev used new seeds and the same pinned stack. ModelOpt's export is deterministic, so all three runs served the same checkpoint.

Run	BF16 selected@8	NVFP4 selected@8	NVFP4 − BF16	95% CI	Sample-0 gap
1 (Colab G4)	0.775	0.788	+1.3	[−1.1, +3.7]	−2.4
2 (Brev)	0.790	0.779	−1.1	[−3.7, +1.3]	−3.3
3 (Brev)	0.792	0.779	−1.3	[−4.1, +1.1]	−3.7
Pooled			**−0.4**	**[−2.1, +1.4]**	

Non-inferiority holds

Every run sits above −3, and the pooled lower bound is −2.1. Run 1's +1.3 was the favourable draw, so the number to quote is −0.4. The single-answer gap reproduced in every run.

What didn't close it

Before reaching for more samples, we tried five cheaper fixes on the same 542 tasks. For QAD, the success criterion was written down before the run.

Intervention	What it did	NVFP4 − BF16 after
Chat-template calibration	Calibrated on Magicoder + UltraChat instead of wikitext	−5.2 pt (worse, p 0.028)
Quantization-aware distillation	Held-out KL −37%, top-1 agreement +2.4 pt	+1.1 pt, n.s.: 136 flips vs 135
16k reasoning budget	Both models a little better	−3.7 pt (p 0.02)
Forced answer at the cap	+4.5 pt to both models, 0 regressions	−3.7 to −6.8 pt
Triad loop (rule-based critic)	+6.6 (BF16) and +7.0 (NVFP4) at the same token budget	−3.9 pt (p 0.01)

QAD worked at what it optimises and missed what we measure. Distilling NVFP4 toward the BF16 teacher cut KL by 37%, but at 86% per-token agreement over ~400-token answers, almost every answer still diverges somewhere. The set of passing tasks just gets re-drawn: QAD fixed 67 of PTQ's flips and introduced 68 new ones. Task-level equivalence needs agreement much closer to 100%.

More room doesn't help. Doubling the reasoning budget left the gap where it was, and the traces still unclosed at 16k are loops. Forcing an answer out of a looped trace recovered about a third of them, but NVFP4's loops were no more salvageable than BF16's. The gap is in traces that close with the wrong answer.

The Triad helps both models equally. It reasons in 1k-token chunks with a model-free critic. The critic closes repeating traces and runs the prompt's examples. A failing answer is rewound, with the failure written back into the reasoning. That makes it the best single-sample method we tested: it adds ~7 points to either model and matches or beats plain sampling at every N up to 4. BF16 Triad best-of-4 (0.795) beats plain best-of-8 (0.785) at half the tokens. But because it lifts both precisions, the gap under it stays at −2.2 to −3.9.

Measuring it honestly

- **The noise floor moves between sessions.** Two identical BF16 runs disagreed on 48 tasks in one session and on zero in another, with the same vLLM version and settings. We still don't know why. The headline comparisons above each carry a same-session repeat.
- **Greedy is reproducible only under a fixed config.** Changing `max_model_len` changes batching, and almost every reasoning trace drifts. Only 2 of 464 closed BF16 traces stayed byte-identical between 8k and 16k.
- **The selector's ground truth is imperfect.** On 5 HumanEval tasks the canonical solution fails its own docstring examples, because the docstrings are wrong. We left them in.
- **Teacher-forced agreement didn't predict task accuracy.** Chat calibration scored higher top-1 agreement (87.1% vs 86.7%) and still lost 5 points on hidden tests.

> **Evaluation guardrail**
> Never report a single-run A-vs-B number without the A-vs-A repeat from the same session beside it. A 147-task disagreement means little until you know two identical runs can disagree on 48, or on none.

What we're honest about

Best-of-8 is not free. NVFP4 writes 13% more tokens, because its traces are longer. In saturated, long-context batched decoding it ran at about BF16's rate (3,293 vs 3,141 tok/s), so its 8 samples took 76 GPU-minutes against BF16's 71. NVFP4's advantage is batch-1 latency and memory (3.7 vs ~9 GB). It does not make large-batch reasoning cheaper. Closing the gap costs eight samples, which BF16 could also spend.

One model, one recipe, one benchmark family. These are function-level Python tasks, which are likely in pre-training data. A contamination-resistant set (recent LiveCodeBench) is the next test. The Triad best-of-N rows rest on one seed set per model, and its two arms ran on different sampler backends: vLLM's PyTorch sampler vs FlashInfer's, which sample the same distribution with different code.

The selector depends on the prompt containing examples. These benchmarks provide them. For the 38 HumanEval tasks that have none, it falls back to the first closed trace. A deployment without examples would need tests or a verifier to play the same role.

The open problem

Why does 4-bit precision cost accuracy only when the model reasons, and is there a fix cheaper than eight samples? The right answer is already in NVFP4's sample set as often as in BF16's.

Something in long reasoning makes it come up less often per sample. Precision that tracks trace length, or a verifier that steers mid-trace instead of after it, would be the next things to try.

All numbers come from the run artefacts in research/nvfp4_lab/results/, scored locally with EvalPlus 0.3.1. Compute: about five Colab G4 sessions plus about $22 of Brev RTX PRO 6000 time. Run dates: 2026-11-01.