Ternary Bonsai 8B

Ternary (1.58-bit) language models
at 8B, 4B, and 1.7B scale

75.5	1.75 GB	9.4×	27 tok/s
average benchmark score	memory footprint	smaller than FP16	on iPhone

PrismML · April 16, 2026

Contents

Category	Summary
Models	Ternary Bonsai — 1.58-bit quantized 8B, 4B, and 1.7B LLMs built from Qwen3 [32]
Format	Ternary g128: $\{-1, 0, +1\}$ weights with FP16 group-wise scaling (group size 128)
Highlights	8B scores 75.5 avg at 1.75 GB; 4B scores 70.7 at 0.86 GB; 1.7B scores 57.5 at 0.37 GB
Focus	Benchmark quality and intelligence density across three model scales
License	Apache License; weights, demos, and integrations

1. Executive Summary

In this short manuscript, we introduce **Ternary Bonsai**, a new family of highly compressed language models that builds on our earlier 1-bit Bonsai work. By moving from binary weights $\{-1, +1\}$ to ternary weights $\{-1, 0, +1\}$, Ternary Bonsai adds an expressive zero state while remaining in the sub-2-bit regime. The result is a stronger quality–efficiency tradeoff: better benchmark performance than our 1-bit models, while preserving the compact footprint and deployment advantages that make Bonsai practical on real hardware.

We present Ternary Bonsai models at **8B, 4B, and 1.7B** parameters, all using group size 128 with FP16 scaling. At the largest scale, **Ternary Bonsai 8B** achieves an average benchmark score of **75.5** at just **1.75 GB**—a $9\times$ **reduction** relative to the **16.38 GB** FP16 baseline, and a **5-point improvement** over its 1-bit counterpart. Despite their compact memory footprint, Ternary Bonsai models outperform nearly all models in their weight class and remain competitive with substantially larger models.

More broadly, these results show that aggressive weight compression need not come with a severe loss in model quality. Instead, Ternary Bonsai advances the deployment frontier for language models, offering a practical path toward capable systems that fit within the memory, bandwidth, and power constraints of real devices.

The Pareto Frontier of Intelligence vs. Size

A natural tradeoff in AI is between model size (measured by the number of bits needed to represent a model) and intelligence (measured by benchmark performance). To study this tradeoff, we evaluated 20 leading instruct models ranging from 1.2 GB (Qwen 3 0.6B [32]) to 18 GB (GLM 4 9B [38]) across six benchmarks covering knowledge, problem solving, math, coding, instruction following, and tool calling. The resulting scatter plot reveals a Pareto frontier defined by Qwen 3 0.6B, 1.7B, 4B, and 8B, together with Ministral 3 3B [25].

Our previous **1-bit Bonsai** release showed that binary-weight models could already shift this frontier substantially to the left [1]. Here we show that **Ternary Bonsai** pushes it further, delivering stronger benchmark performance at 8B, 4B, and 1.7B while preserving the extreme compactness that makes the Bonsai family compelling for deployment.

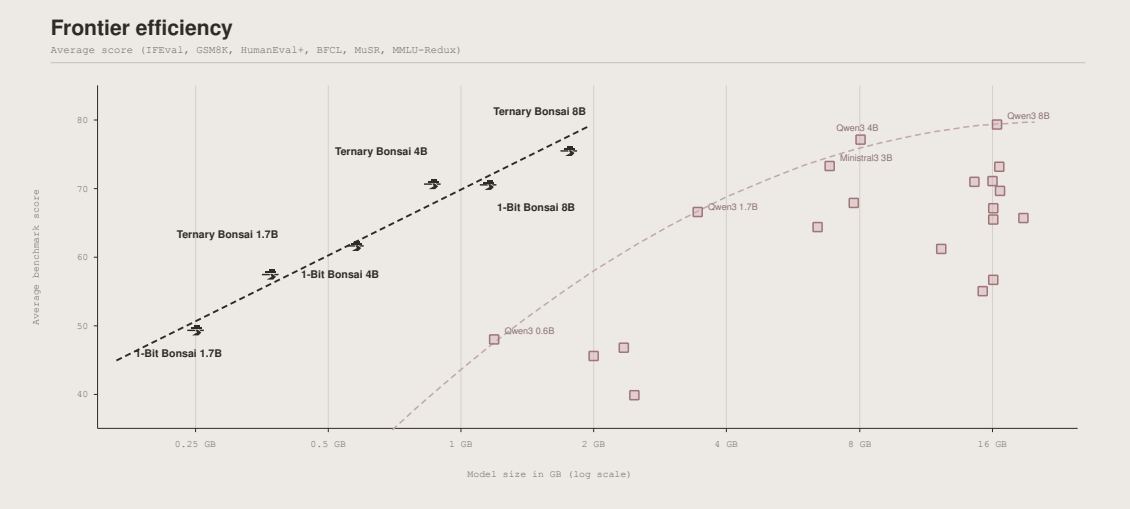

Figure 1. Benchmark score vs. model size (GB, log scale). The Ternary Bonsai family shifts the Pareto frontier of intelligence vs. size decisively to the left.

2. Ternary Bonsai Model Summary

Ternary Bonsai extends our earlier 1-bit Bonsai framework [1] using the Qwen3 [32] family of dense decoder-only causal language models (8B, 4B, and 1.7B). The underlying architectures are unchanged: the novelty lies entirely in the weight representation. Specifically, ternary weights in $\{-1, 0, +1\}$ are applied across all major matrix-heavy components using group size 128 with shared FP16 scaling.

Table 1. System specification.

Item	Specification
Base models	Qwen3-8B, Qwen3-4B, and Qwen3-1.7B [32] dense causal language models
Parameters (8B)	8.19 B (∼6.95 B non-embedding); 36 Transformer blocks
Architecture	GQA [3] (32 query / 8 KV heads), SwiGLU [35] MLP, RoPE [34], RMSNorm [33]
Context length	65,536 tokens
Weight format	Ternary g128: $\{-1, 0, +1\}$ weights with shared FP16 group-wise scaling
Quantized layers	Embeddings, attention projections, MLP projections, and LM head
Backends	MLX (Python, Swift)
License	Apache License

2.1 Ternary Weight Format

As in our earlier 1-bit release [1], Ternary Bonsai uses a group-wise low-bit weight format. Each weight takes a value from $\{-1, 0, +1\}$, with one shared FP16 scale for each group of 128 weights. The effective weight is

$$w_i = s_g \cdot t_i, \qquad t_i \in \{-1,\, 0,\, +1\},$$

where s_g is the shared scale for group g.

Ternary code values carry $\log_2 3 \approx 1.585$ bits of information per weight. With one FP16 scale per group of 128 weights, the effective storage cost is approximately

$$b_{\text{eff}} \approx 1.585 + \frac{16}{128} = 1.71 \text{ bits/weight},$$

which yields an idealized raw-weight compression of $16/1.71 \approx 9.4\times$ relative to FP16, before container overhead and alignment. Relative to binary quantization, the additional zero state provides a more expressive representation and allows better preservation of model quality under extreme compression.

As 1-bit Bonsai models, the ternary format is applied uniformly across embeddings, attention projections, MLP projections, and the LM head. Normalization parameters and scale metadata remain in higher precision for numerical stability, but they account for a negligible share of memory traffic relative to the large weight tensors that dominate bandwidth during decoding.

2.2 Storage Footprint

The deployed package size is a first-order consequence of ternary quantization. For the 8B model, FP16 safetensors occupy 16.38 GB, while the ternary representation reduces this to approximately 1.75 GB. Comparable reductions hold at the 4B and 1.7B scales. Since MLX does not yet provide an efficient native implementation for ternary (1.58-bit) weights, we use current

2-bit kernels for deployment on Apple platforms to avoid computational overhead and potential inference slowdown. Table 2 summarizes both the theoretical storage footprint implied by the bitwidth and the actual deployed memory footprint of the Ternary Bonsai models.

Table 2. Ideal and deployed memory footprints of the Ternary Bonsai family.

Model	FP16 Size	Ternary Size	Ratio	Deployed Size
Ternary Bonsai 8B	16.38 GB	∼1.75 GB	**9.36**×	2.16 GiB
Ternary Bonsai 4B	8.04 GB	∼0.86 GB	**9.35**×	1.05 GiB
Ternary Bonsai 1.7B	3.44 GB	∼0.37 GB	**9.3**×	0.45 GiB

2.3 Throughput Performance on Apple Silicon

We evaluated throughput of Ternary Bonsai models on Apple hardware using an M4 Pro 48 GB system with MLX Python and an iPhone 17 Pro Max with MLX Swift. Because MLX does not yet provide an efficient native implementation for ternary (1.58-bit) weights, these measurements use the current 2-bit deployment path on Apple platforms. We report standardized tg128 / pp512 deployment measurements, where tg128 measures token generation throughput over 128 generated tokens and pp512 measures prompt processing throughput over 512 input tokens. These measurements are intended to capture practical deployment performance rather than long sustained-run workloads. On the M4 Pro 48 GB system, we compare Ternary Bonsai models against the corresponding FP16 baselines, whereas on the iPhone 17 Pro Max the baseline is 4-bit, as reported in Tables 3 and 4, respectively.

Table 3. Throughput of Ternary Bonsai using 2-bit MLX deployment path compared to FP16 baseline on Apple M4 Pro 48GB system.

Model	Platform	Backend	TG128 (tok/s)	PP512 (tok/s)	Base TG (tok/s)	Speedup
Ternary Bonsai 8B	M4 Pro 48 GB	MLX (Python)	83	434	16	**5.2**×
Ternary Bonsai 4B	M4 Pro 48 GB	MLX (Python)	133	728	28	**4.8**×
Ternary Bonsai 1.7B	M4 Pro 48 GB	MLX (Python)	235	1585	62	**3.8**×

Table 4. Throughput of Ternary Bonsai using 2-bit MLX deployment path compared to 4-bit baseline on iPhone 17 Pro Max.

Model	Platform	Backend	TG128 (tok/s)	PP512 (tok/s)	Base TG (tok/s)	Speedup
Ternary Bonsai 8B	iPhone 17 Pro Max	MLX Swift	27	356	14	**1.9**×
Ternary Bonsai 4B	iPhone 17 Pro Max	MLX Swift	50	659	27	**1.8**×
Ternary Bonsai 1.7B	iPhone 17 Pro Max	MLX Swift	103	1456	60	**1.7**×

These results show that Ternary Bonsai retains strong deployment efficiency on Apple hardware even under the current 2-bit MLX implementation path. Native ternary kernels should further improve the realized footprint advantage relative to the measurements reported here.

2.4 Energy per Token

We evaluate the energy efficiency of Ternary Bonsai models on an M4 Pro system with 48 GB of memory. For Mac measurements, reported power includes CPU, GPU, ANE, and DRAM, while excluding overall system overhead.

Table 5 summarizes the measured energy per generated token on the Mac M4 Pro 48 GB system. On the Mac M4 Pro, Ternary Bonsai models achieve a **3–4× reduction** in energy per generated token relative to the baseline.

Table 5. Measured energy per generated token (E_{tg}) on the Mac M4 Pro 48 GB system.

Model	Platform	Bonsai E_{tg} (mWh/tok)	Baseline E_{tg} (mWh/tok)	Improvement
Ternary Bonsai 8B	Mac M4 Pro	0.105	0.415	**4.0**×
Ternary Bonsai 4B	Mac M4 Pro	0.061	0.211	**3.5**×
Ternary Bonsai 1.7B	Mac M4 Pro	0.031	0.093	**3.0**×

For deployment on the iPhone 17 Pro Max, energy is estimated using Xcode Power Profiler together with battery-drain observations, so these measurements are less precise than those on Mac. We evaluate only Ternary Bonsai 8B in this setting. The ternary 8B model (MLX 2-bit) yields an estimated E_{tg} of approximately 0.132 mWh/tok, which is about 1.9× higher than the 1-bit Bonsai 8B model at 0.068 mWh/tok. We attribute this gap primarily to the use of MLX 2-bit kernels in the deployment of Ternary Bonsai 8B.

3. Benchmarks and Intelligence Density

We evaluated the Ternary Bonsai family at 8B, 4B, and 1.7B scale across six skill categories: knowledge, reasoning, math, coding, instruction following, and tool calling. As in our earlier 1-bit Bonsai white paper [1], all benchmarks were run under matched infrastructure, decoding, scoring, and judge settings using EvalScope [2] with the vLLM [37] backend on NVIDIA H100 GPUs. Please refer to Appendix B of [1] for further details of our full benchmark suite.

Table 6 summarizes the 8B-scale results on one benchmark per skill category: MMLU-Redux [27, 15] for knowledge, MuSR [28] for reasoning, GSM8K [8] for math, HumanEval+ [9, 20] for coding, IFEval [36] for instruction following, and BFCLv3 [5] for tool calling. The corresponding 1-bit Bonsai 8B results from our earlier white paper are also shown for reference [1]. Additional results for these categories, including Ternary Bonsai 4B and 1.7B, are provided in Appendix A.

Table 6. Benchmark comparison. Ternary Bonsai 8B is compared against 11 leading conventional models of comparable scale.

Model	Size	Average	MMLU Redux	MuSR	GSM8K	Human Eval+	IFEval	BFCLv3
Qwen 3 8B [32]	16.38 GB	**79.3**	83	55	93	82.3	81.5	81
Ternary Bonsai 8B	**1.75 GB**	**75.5**	72.6	56.2	91	77.4	81.8	73.9
RNJ 8B [12]	16.63 GB	**73.1**	75.5	50.4	93.7	84.2	73.8	61.1
Ministral3 8B [24]	16.04 GB	**71.0**	68.9	53.8	87.9	72.6	67.4	75.4
Olmo 3 7B [30]	14.60 GB	**70.9**	72	56.1	92.5	79.3	87.1	38.4
1-bit Bonsai 8B	**1.15 GB**	**70.5**	65.7	50	88	73.8	79.8	65.7
LFM2 8B [22]	16.68 GB	**69.6**	72.7	49.5	90.1	61	82.2	62.0
Llama 3.1 8B [18]	16.06 GB	**67.1**	72.9	51.3	87	63.4	76.4	51.5
GLM 4 9B [38]	18.80 GB	**65.7**	81.9	53.2	89.4	78.7	69.3	21.9
Hermes 3 8B [29]	16.06 GB	**65.4**	67.4	52.2	82.9	51.2	69.3	69.6
Trinity Nano 6B [4]	12.24 GB	**61.2**	66.8	52.6	81.1	54	50	62.5
Marin 8B [23]	16.06 GB	**56.6**	64.8	42.6	86.1	55.9	63	27.9
DeepSeek R1 Qwen 7B [11]	15.23 GB	**55.0**	62.5	29.1	92.7	81.7	48.8	15.4

Ternary Bonsai 8B delivers a particularly strong result: at just 1.75 GB, it reaches an average benchmark score of 75.5, versus 79.3 for the 16.38 GB FP16 Qwen 3 8B baseline. Thus, a **9.36× reduction in size** costs only **3.8 average-score points**, i.e. **less than 5%** relative. Put differently, Ternary Bonsai retains more than **95%** of the benchmark quality of the full-precision model while using only about **one-ninth** of the memory. Compared with 1-bit Bonsai 8B, it gains **5.0 points** at an additional footprint of just **0.60 GB**, demonstrating that ternary weights achieve substantial improvement without giving up the deployment advantages of extreme low-bit compression.

3.1 Intelligence Density

As in our earlier 1-bit Bonsai white paper [1], we summarize the tradeoff between capability and model size using *intelligence density*: the amount of intelligence a model delivers per unit of memory. Raw average benchmark score is not an ideal measure of intelligence, since equal score gains do not correspond to equal improvements in capability across the scale. Following our earlier treatment [1], we define $P_e = 1 - \frac{\text{average benchmark score}}{100}$, as the model's probability of error, and measure intelligence as

$$-\log(P_e) \;=\; -\log\Big(1 - \frac{\text{average benchmark score}}{100}\Big). \tag{1}$$

We then define *intelligence density* as intelligence per unit model size:

$$D = \frac{-\log(P_e)}{N}, \tag{2}$$

where N denotes model size, measured here in GB.

We compare intelligence density of 18 instruct models in the 1.2B–9B parameter range together with Ternary Bonsai 8B, 4B, and 1.7B, as well as 1-bit Bonsai models from our prior release for reference. In particular, Figure 1 visualizes the tradeoff between benchmark performance and model size directly, while Table 7 reports the corresponding intelligence density values.

Table 7. Intelligence density (1/GB). Average score is computed over MMLU-Redux, MuSR, GSM8K, HumanEval+, IFEval, and BFCLv3. Intelligence metric is computed as shown in Equation (1) and intelligence density is computed via Equation (2), where GB is used as the unit of model size.

Model	Intelligence Density (1/GB)	Size	Average Score
1-bit Bonsai 1.7B	*2.832*	*0.24 GB*	*49.60*
Ternary Bonsai 1.7B	**2.389**	**0.37 GB**	**58.47**
1-bit Bonsai 4B	*1.744*	*0.57 GB*	*62.72*
Ternary Bonsai 4B	**1.426**	**0.86 GB**	**70.65**
1-bit Bonsai 8B	*1.060*	*1.15 GB*	*70.50*
Ternary Bonsai 8B	**0.803**	**1.75 GB**	**75.48**
Qwen 3 0.6B [32]	0.549	1.19 GB	48.02
Qwen 3 1.7B [32]	0.318	3.44 GB	66.57
Gemma 3 1B [16]	0.304	2.00 GB	45.53
LFM2 1.2B [21]	0.269	2.34 GB	46.73
Llama 3.2 1B [18]	0.206	2.47 GB	39.88
Ministral3 3B [25]	0.192	6.86 GB	73.22
Qwen 3 4B [32]	0.183	8.04 GB	77.10
Llama 3.2 3B [18]	0.161	6.43 GB	64.35
Gemma 3 4B [16]	0.146	7.76 GB	67.88
Qwen 3 8B [32]	0.096	16.38 GB	79.30
Olmo 3 7B [30]	0.085	14.60 GB	70.90
RNJ 8B [12]	0.079	16.62 GB	73.12
Trinity Nano 6B [4]	0.077	12.24 GB	61.17
Ministral3 8B [24]	0.077	16.04 GB	71.00
LFM2 8B [22]	0.071	16.68 GB	69.58
Llama 3.1 8B [18]	0.069	16.06 GB	67.08
Hermes 3 8B [29]	0.066	16.06 GB	65.43
GLM 4 9B [38]	0.057	18.80 GB	65.73

As expected, smaller models tend to exhibit higher intelligence density. The more important result, however, is that the Ternary Bonsai models outperform all conventional models by a wide margin, indicating that their advantage is not merely a consequence of smaller size. Rather, they achieve a fundamentally better efficiency–performance tradeoff, delivering substantially more intelligence per GB than full-precision models. Relative to the 1-bit Bonsai models, the ternary variants trade some density for materially higher benchmark scores—for example, Ternary Bonsai 8B scores 75.5 versus 70.5 for 1-bit Bonsai 8B—showing that the additional expressiveness of the zero weight value translates into meaningful performance improvement.

A. All Benchmark Results

In evaluating model performance, we use the full benchmark suite described in our 1-bit Bonsai release [1]; for further details, please refer to Appendix B of [1]. In this section, we present the extended benchmark results and the corresponding intelligence density results with 10 benchmarks we consider. In particular, Tables 8, 9 and 10 report comparisons with reference models for the Ternary Bonsai family on the 8B, 4B and 1.7B scales, respectively.

Viewed through raw benchmark scores alone, the Ternary Bonsai models are competitive across all three scales. Viewed through intelligence density, however, they emerge as clear outliers. Across the 8B, 4B, and 1.7B regimes, Ternary Bonsai delivers substantially more intelligence per GB than nearby conventional baselines, demonstrating that ternary quantization can dramatically reduce model footprint while preserving practical capability.

Table 8. Full Benchmark Suite Comparison. Ternary Bonsai 8B is compared against 11 leading conventional models of comparable scale. 1-bit Bonsai 8B is shown for reference. Intelligence density is reported in 1/GB using 10 benchmarks using (2).

Model	Intelligence Density	Avg.	MMLU Redux	GPQA Diamond	MuSR	IFEval	IFBench	GSM8K	MATH 500	Human Eval+	MBPP+	BFCLv3
Ternary Bonsai 8B	**0.602**	**65.14**	**72.6**	**34.9**	**56.2**	**81.8**	**21.9**	**91.0**	**73.2**	**77.4**	**68.5**	**73.9**
1-bit Bonsai 8B	*0.792*	*59.86*	*65.7*	*30.0*	*50.0*	*79.8*	*19.8*	*88.0*	*66.0*	*73.8*	*59.8*	*65.7*
Olmo 3 7B	0.077	67.34	72.0	40.4	56.1	87.1	45.1	92.5	91.6	79.3	70.9	38.4
Qwen 3 8B	0.076	71.02	83.0	49.3	55.0	81.5	27.2	93.0	84.4	82.3	73.5	81.0
RNJ 8B	0.065	66.19	75.5	34.0	50.4	73.8	30.7	93.7	82.6	84.2	75.9	61.1
LFM2 8B	0.061	63.89	72.7	31.6	49.5	82.2	40.7	90.1	84.8	61.0	64.3	62.0
Trinity Nano 6B	0.060	51.79	66.8	22.0	52.6	54.0	15.3	81.1	61.2	50.0	52.4	62.5
Mistral3 8B	0.058	60.51	68.9	29.2	53.8	67.4	28.8	87.9	56.0	72.6	65.1	75.4
DS R1 Qwen 7B	0.053	55.39	62.5	41.5	29.1	48.8	25.7	92.7	89.6	81.7	66.9	15.4
Llama 3.1 8B	0.053	56.97	72.9	27.2	51.3	76.4	25.8	87.0	50.2	63.4	64.0	51.5
Hermes 3 8B	0.049	54.13	67.4	29.5	52.2	69.3	29.6	82.9	32.2	51.2	57.4	69.6
GLM 4 9B	0.047	58.88	81.9	37.3	53.2	69.3	23.8	89.4	68.2	78.7	65.1	21.9
Marin 8B	0.042	49.04	64.8	24.8	42.6	63.0	15.3	86.1	56.0	54.9	55.0	27.9

Table 9. Full Benchmark Suite Comparison. Ternary Bonsai 4B is compared against 4 leading conventional models of comparable scale. 1-bit Bonsai 4B is shown for reference. Intelligence density is reported in 1/GB using 10 benchmarks using (2).

Model	Intelligence Density	Avg.	MMLU Redux	GPQA Diamond	MuSR	IFEval	IFBench	GSM8K	MATH 500	Human Eval+	MBPP+	BFCLv3
Ternary Bonsai 4B	**1.151**	**62.83**	**69.7**	**34.1**	**45.1**	**72.1**	**27.6**	**90.5**	**76.8**	**78.7**	**65.9**	**67.8**
1-bit Bonsai 4B	*1.427*	*55.39*	*58.7*	*28.7*	*41.4*	*69.6*	*25.2*	*87.3*	*65.8*	*71.3*	*57.9*	*48.0*
Ministral3 3B	0.162	66.99	77.5	41.7	56.5	73.1	37.9	91.4	84.6	69.5	66.4	71.3
Qwen 3 4B	0.143	68.31	79.8	42.9	57.4	80.0	24.3	92.1	83.2	74.4	70.1	78.9
Llama 3.2 3B	0.125	55.15	65.5	27.2	48.9	78.3	31.5	80.1	49.0	52.4	57.7	60.9
Gemma 3 4B	0.120	60.47	66.0	28.7	46.3	73.0	23.7	89.8	75.2	67.1	69.8	65.1

Table 10. Full Benchmark Suite Comparison. Ternary Bonsai 1.7B is compared against 5 leading conventional models of comparable scale. 1-bit Bonsai 1.7B is shown for reference. Intelligence density is reported in 1/GB using 10 benchmarks using (2).

Model	Intelligence Density	Avg.	MMLU Redux	GPQA Diamond	MuSR	IFEval	IFBench	GSM8K	MATH 500	Human Eval+	MBPP+	BFCLv3
Ternary Bonsai 1.7B	**1.862**	**49.58**	**52.9**	**23.1**	**50.8**	**70.1**	**18.0**	**74.2**	**54.4**	**51.8**	**49.5**	**51.0**
1-bit Bonsai 1.7B	*2.172*	*40.88*	*43.2*	*20.7*	*45.1*	*63.0*	*13.8*	*66.3*	*34.4*	*45.1*	*42.3*	*34.9*
Qwen3 0.6B	0.477	43.34	47.5	26.0	41.5	62.8	17.9	64.1	55.6	30.5	45.8	41.7
Gemma3 1B	0.268	41.49	43.2	24.0	37.0	61.9	14.6	64.4	46.2	40.2	56.9	26.5
Qwen3 1.7B	0.254	58.24	66.8	32.5	50.1	70.3	18.3	83.1	74.0	57.3	58.2	71.8
LFM2 1.2B	0.226	41.08	52.9	23.5	25.4	77.5	27.7	62.2	42.2	36.0	37.0	26.4
Llama 3.2 1B	0.182	36.15	47.2	22.5	29.2	47.7	18.0	49.0	34.4	35.4	47.3	30.8

References

[1] Prism ML. 1-bit Bonsai 8B. White paper. https://github.com/PrismML-Eng/Bonsai-demo/blob/main/1-bit-bonsai-8b-whitepaper.pdf, 2026.

[2] Alibaba ModelScope. EvalScope: Evaluation framework for large language models. https://github.com/modelscope/evalscope, 2024.

[3] J. Ainslie, J. Lee-Thorp, M. de Jong, Y. Zemlyanskiy, F. Lebrón, and S. Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of EMNLP*, 2023.

[4] Arcee AI. Trinity Nano (6B). https://docs.arcee.ai/language-models/trinity-nano-6b, 2025.

[5] S. G. Patil, H. Mao, F. Yan, C. C. Ji, V. Suresh, I. Stoica, and J. E. Gonzalez. The Berkeley Function Calling Leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Proceedings of ICML*, 2025.

[6] S. Ma, H. Wang, L. Ma, L. Wang, W. Wang, S. Huang, L. Dong, R. Wang, J. Xue, and F. Wei. The Era of 1-bit LLMs: All large language models are in 1.58 bits. *arXiv preprint arXiv:2402.17764*, 2024.

[7] H. Wang, S. Ma, L. Dong, S. Huang, H. Wang, L. Ma, F. Yang, R. Wang, Y. Wu, and F. Wei. BitNet: Scaling 1-bit Transformers for large language models. *arXiv preprint arXiv:2310.11453*, 2023.

[8] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[9] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. Pinto de Oliveira, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[10] T. Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

[11] DeepSeek AI. DeepSeek-R1-Distill-Qwen-7B. https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-7B, 2025.

[12] Essential AI. Rnj-1. https://essential.ai/research/rnj-1, 2025.

[13] G. Gerganov. GGUF: GPT-Generated Unified Format specification. https://github.com/ggml-org/ggml/blob/master/docs/gguf.md, 2023.

[14] G. Gerganov et al. llama.cpp: LLM inference in C/C++. https://github.com/ggml-org/llama.cpp, 2023–2026.

[15] A. P. Gema, J. O. J. Leang, G. Hong, A. Devoto, A. C. M. Mancino, R. Saxena, X. He, Y. Zhao, X. Du, M. R. G. Madani, C. Barale, R. McHardy, J. Harris, J. Kaddour, E. van Krieken, and P. Minervini. Are we done with MMLU? *arXiv preprint arXiv:2406.04127*, 2024.

[16] Google. Gemma 3 Release. https://huggingface.co/collections/google/gemma-3-release, 2025.

[17] D. Rein, B. Hou, A. Stickland, J. Petty, R. Pang, J. Dirani, J. Michael, and S. R. Bowman. GPQA: A graduate-level Google-proof Q&A benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

[18] A. Grattafiori, A. Dubey, A. Jauhri, et al. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*, 2024.

[19] H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024.

[20] J. Liu, C. S. Xia, Y. Wang, and L. Zhang. Is your code generated by ChatGPT really correct? Rigorous evaluation of large language models for code generation. In *Proceedings of NeurIPS*, 2023.

[21] Liquid AI. LFM2-1.2B. https://huggingface.co/LiquidAI/LFM2-1.2B, 2025.

[22] Liquid AI. LFM2-8B-A1B: An Efficient On-device Mixture-of-Experts. https://www.liquid.ai/blog/lfm2-8b-a1b-an-efficient-on-device-mixture-of-experts, 2025.

[23] Marin Project (Stanford CRFM). Marin 8B Instruct. https://huggingface.co/marin-community/marin-8b-instruct, 2025.

[24] Mistral AI. Ministral-8B-Instruct-2410. https://huggingface.co/mistralai/Ministral-8B-Instruct-2410, 2024.

[25] Mistral AI. Ministral 3 3B Instruct 2512. https://huggingface.co/mistralai/Ministral-3-3B-Instruct-2512, 2025.

[26] A. Hannun, J. Digani, A. Katharopoulos, and R. Collobert. MLX: An array framework for Apple silicon. https://github.com/ml-explore/mlx, 2023.

[27] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. Measuring massive multitask language understanding. In *Proceedings of ICLR*, 2021.

[28] Z. Sprague, X. Ye, K. Bostrom, S. Chaudhuri, and G. Durrett. MuSR: Testing the limits of chain-of-thought with multistep soft reasoning. In *Proceedings of ICLR*, 2024.

[29] Nous Research. Hermes 3: Llama-3.1-8B. https://huggingface.co/NousResearch/Hermes-3-Llama-3.1-8B, 2024.

[30] Team OLMo, A. Ettinger, A. Bertsch, B. Kuehl, et al. OLMo 3. *arXiv preprint arXiv:2512.13961*, 2025.

[31] V. Pyatkin, S. Malik, V. Graf, H. Ivison, S. Huang, P. Dasigi, N. Lambert, and H. Hajishirzi. Generalizing Verifiable Instruction Following. *arXiv preprint arXiv:2507.02833*, 2025.

[32] Qwen Team. Qwen3 Technical Report. *arXiv preprint arXiv:2505.09388*, 2025.

[33] B. Zhang and R. Sennrich. Root mean square layer normalization. In *Proceedings of NeurIPS*, 2019.

[34] J. Su, Y. Lu, S. Pan, A. Murtadha, B. Wen, and Y. Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[35] N. Shazeer. GLU variants improve Transformer. *arXiv preprint arXiv:2002.05202*, 2020.

[36] J. Zhou, T. Lu, S. Mishra, S. Brahma, S. Basu, Y. Luan, D. Zhou, and L. Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

[37] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. Gonzalez, H. Zhang, and I. Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of SOSP*, 2023.

[38] Z.ai. GLM-4-9B-0414. https://huggingface.co/zai-org/GLM-4-9B-0414, 2025.