

**Towards the 1P Model:
Feasibility Conditions for Training a Peta-Parameter
Learning System**

PIATRA . INSTITUTE

September 2026

Abstract

The largest documented language models hold about 10^{12} parameters. We ask what training a system with 10^{15} learned parameters would require, and which requirement binds first under sourced hardware and scaling assumptions. Storing the weights takes 2 PB in BF16 and the mixed-precision Adam training state 16 PB, the memory of 200,000 accelerators with 80 GB each. A dense model trained at the compute-optimal ratio of 21.2 tokens per parameter needs 1.27×10^{32} FLOP, about 10,000 years on 10^6 devices at the throughput sustained in a recent 405-billion-parameter run; at an activation fraction of 10^{-4} it needs one year. The data do not exist: that ratio asks for 53 times the estimated stock of public human text. Two further bounds depend on the balance of the hardware more than on its quantity. When parameters and optimizer state stream between memory and compute, each update must route at least $n^* = \beta q/6$ tokens to every parameter, 1,299 on a TPU v4 with Adam; since the global batch is limited by the critical batch size, sparse training costs a factor $1 + n^*/(AB_{crit})$, which exceeds 2 below $A \approx 1.8 \times 10^{-4}$ for a loss-based critical batch. Checkpoint-restart efficiency depends only on $\rho = Nb/(wM)$, restart state over per-device storage bandwidth times mean time between failures; at the per-device values of the same run, $\rho = 0.54$ at 10^{15} parameters and the useful fraction of time falls to 0.17. Weight-stationary memory and independently recovering modules relax these two bounds, and device endurance limits the first.

Introduction

The largest language models with published technical reports hold on the order of 10^{12} parameters. Kimi K2 has 1 trillion parameters, of which 32 billion are activated per token, and was pre-trained on 15.5 trillion tokens (Kimi Team et al. 2025). DeepSeek-V3 has 671 billion parameters, 37 billion activated, and was pre-trained on 14.8 trillion tokens (DeepSeek-AI et al. 2024). Both are mixtures of experts, in which a router sends each token to a few of many feed-forward sub-networks so that the parameter count grows without a proportional growth in computation per token (Shazeer et al. 2017; Lepikhin et al. 2020; Fedus et al. 2021). Dense models stop near 4×10^{11} parameters: Llama 3 405B was pre-trained with 3.8×10^{25} FLOP on 15.6 trillion tokens using up to 16,384 GPUs (Grattafiori et al. 2024). A system with 10^{15} learned parameters, a “1P model”, would sit three orders of magnitude above the largest of these.

Whether such a system can be built is less informative than which of its requirements binds first and which change in hardware or algorithm relaxes it. Some requirements scale with the total parameter count N , some with the parameters activated per token $N_a = AN$, some with the training data D , and some with properties of the hardware that do not change with the size of the cluster. We separate them with a feasibility calculator in which every constant is either copied from a cited source, with its locator recorded, or labelled an assumption, and in which every derived number is checked against a closed form or an independent published figure. The result is a set of conditions under stated assumptions; it contains no forecast of when a 1P model will exist.

Three distinctions organize the analysis. The first separates nominal parameters, stored and trainable, from active parameters, used for a given token, and from effective parameters, those whose values change the model’s predictions. Routed models already have a scaling-law notion of an effective parameter count, along which routed and dense models scale at the same rate (Clark et al. 2022). The second separates quantities that grow with N , such as memory and checkpoint size, from quantities that grow with $N_a D$, such as arithmetic. The third separates bounds set by the size of a cluster from bounds set by its balance: the ratio of arithmetic to memory bandwidth, of storage bandwidth to failure rate, of update count to device endurance. Bounds of the third kind do not yield to buying more devices of the same kind, and they decide whether a peta-parameter system needs different hardware.

Accounting

We write N for the total parameter count, $A = N_a/N$ for the activation fraction, D for the number of training tokens and B for the global batch in tokens per optimizer update. Training a Transformer costs about $6N$ floating-point operations per token (Kaplan et al. 2020); for a routed model we use $6AN$, which ignores the router and treats attention as part of the active parameters. Mixed-precision training with Adam (Kingma and Ba 2014; Micikevicius et al. 2017) keeps 16 bytes per parameter: 2 for half-precision weights, 2 for gradients and 12 for the optimizer’s full-precision copy of the weights and its two moment estimates (Rajbhandari et al. 2020). A device has peak arithmetic rate F and memory bandwidth W , and its balance $\beta = F/W$ is the number of operations it must perform per byte moved to keep its arithmetic busy, the ridge point of the roofline model (Williams et al. 2009). For time and energy we use the throughput sustained in the Llama 3 405B run, 400 TFLOP/s per H100 GPU at 41% of its BF16 peak, and its 700 W thermal design power, which gives 1.75 pJ per sustained operation at the device and excludes cooling, networking and storage (Grattafiori et al. 2024). DeepSeek-V3 reports 180,000 H800 GPU hours per trillion training tokens (DeepSeek-AI et al. 2024), which with 37 billion active parameters implies 343 TFLOP/s per GPU, consistent with the Llama figure.

Storage and training state

Storing 10^{15} weights takes 2 PB in BF16, 1 PB in an 8-bit format, 500 TB at 4 bits, 197.5 TB with ternary weights of 1.58 bits (Ma et al. 2024) and 100 TB at the 0.8 bits per parameter to which QMoE compressed the 1.6-trillion-parameter Switch Transformer (Frantar and Alistarh 2023). The mixed-precision Adam state is 16 PB. That is the high-bandwidth memory of 200,000 accelerators with 80 GB each, 12.2 times the 16,384 GPUs of the Llama 3 run, or of 465,662 TPU v4 chips with 32 GiB each (Jouppi et al. 2023). A wafer-scale engine with 40 GB of on-chip memory (Z. Zhang et al. 2024) would need 400,000 wafers. Offloading state to host memory and NVMe storage already lets models of tens to hundreds of trillions of parameters be trained on current GPU clusters (Rajbhandari et al. 2021), and partitioning the state across data-parallel workers removes its replication (Rajbhandari et al. 2020). Capacity is a procurement problem. The bounds that follow concern what happens when that capacity has to be read, written and protected at the rate training demands.

Compute, time and energy

At the compute-optimal ratio, about 21.2 tokens per parameter at 10^{12} parameters in the Chinchilla analysis (Hoffmann et al. 2022), a 1P model would train on 2.12×10^{16} tokens. Two published figures check the accounting: $6ND$ reproduces the Chinchilla table’s compute for 1 and 10 trillion parameters to within 0.22% and Llama 3’s reported compute to within 0.24%. A dense 1P model then needs 1.27×10^{32} FLOP, 3.3 million times the Llama 3 budget. At 400 TFLOP/s per device it would take 10,077 years on 10^6 devices and consume 7,054 GW-years at device power.

Training time and energy scale linearly in A (Figure 1a). At $A = 10^{-3}$ the same token count takes 10.1 years on 10^6 devices and 7.05 GW-years; at $A = 10^{-4}$ it takes 1.01 years on 10^6 devices, or 10.1 years on 100,000, and 0.71 GW-years. Current models activate far larger fractions: 0.055 for DeepSeek-V3 and 0.032 for Kimi K2. Reaching 10^{-3} means a model 32 times sparser than Kimi K2, and reaching 10^{-4} a model 551 times sparser than DeepSeek-V3. Routed models with more than a million small experts selected through product keys (Lample et al. 2019; He 2024) show that very large expert counts can be trained; whether capability keeps improving at such sparsity is a separate question. Fine-grained and joint scaling laws for mixtures of experts find gains from granularity and an optimal sparsity under fixed budgets, measured on models far smaller than 10^{12} parameters (Krajewski et al. 2024; Abnar et al. 2025; Ludziejewski et al. 2025).

Data

The token count that makes compute optimal at 10^{12} parameters does not exist at 10^{15} . The effective stock of public human-generated text on the indexed web is about 4×10^{14} tokens (Villalobos et al. 2024), and 2.12×10^{16} tokens is 53 times that stock. Trained on the whole stock, a 1P model sees 0.4 tokens per parameter. In a sparse model the relevant quantity is the number of tokens each expert parameter sees, AD : at $A = 10^{-3}$ and $D = 4 \times 10^{14}$ that is 4×10^{11} tokens, 53 times fewer than a compute-optimal dense model of 10^{12} parameters gives each of its parameters.

The Chinchilla fit $L(N, D) = E + A_0/N^{0.34} + B_0/D^{0.28}$, with $E = 1.69$, $A_0 = 406.4$ and $B_0 = 410.7$ (Hoffmann et al. 2022), indicates what the extra parameters would buy if a dense model of that size could be trained on the available text (Figure 1b). At $D = 4 \times 10^{14}$ the data term is 0.0335 nats while the parameter term at 10^{15} is 0.0032, a ratio of 10.4, and going from 10^{13} to 10^{15} parameters lowers the predicted loss by 0.012 nats. The fit was measured on dense models up to 16 billion parameters and is extrapolated here by five orders of magnitude; it shows the shape of the trade-off and predicts nothing about a routed model at that scale. Without new sources of tokens, the data term dominates the predicted loss. Synthetic data, transfer from data-rich domains and gains in data efficiency are the routes that the analysis of the stock itself proposes (Villalobos et al. 2024), and domains whose outputs can be checked automatically, such as programs and proofs, are the natural sources of synthetic tokens.

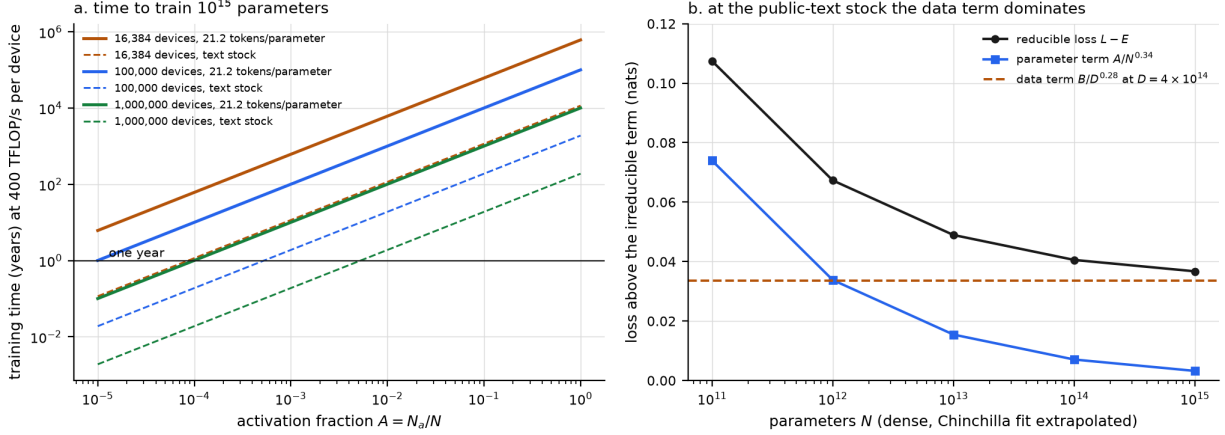


Figure 1: Training time and the role of data at 10^{15} parameters. (a) Training time against activation fraction for 10^{15} parameters at 400 TFLOP/s per device, for three cluster sizes, at the compute-optimal ratio of 21.2 tokens per parameter (solid) and on the estimated stock of public text, 4×10^{14} tokens (dashed). (b) The Chinchilla loss fit for dense models evaluated at $D = 4 \times 10^{14}$: the reducible loss, its parameter term and its data term, which does not change with N .

A bound on sparsity when parameters are streamed

A conventional accelerator keeps parameters and optimizer state in off-chip memory and streams them through its arithmetic units. Per optimizer update, each parameter is read twice for the forward and backward passes, its gradient is written and read again, its optimizer state is read and written, and the new weight is written. With 2-byte weights and gradients and 12 bytes of state this traffic is $q = 34$ bytes per parameter per update. If routing is balanced, each parameter takes part in AB of the B tokens of an update and performs $6AB$ operations on them. The update is compute-bound only if its arithmetic intensity reaches the device balance, $6AB/q \geq \beta$, that is if

$$AB \geq n^* = \frac{\beta q}{6}.$$

The bound does not involve N . On a TPU v4, with 275 TFLOP/s and 1,200 GB/s of memory bandwidth, $\beta = 229$ operations per byte and $n^* = 1,299$ tokens; on an A100, $\beta = 153$ and $n^* = 867$ (Jouppi et al. 2023). Dense training satisfies it with room to spare: the 16 million tokens of a Llama 3 batch are 12,321 times n^* . A sparse model needs a global batch of at least n^*/A tokens, and that requirement grows as the model becomes sparser (Figure 2a).

The global batch cannot grow without cost. Beyond the critical batch size B_{crit} , larger batches buy fewer steps at the price of more data. An empirical model of large-batch training puts the data needed to reach a given loss at $D_{min}(1 + B/B_{crit})$ (McCandlish et al. 2018), and for language models B_{crit} follows a power law in the loss, about $2 \times 10^8/L^{1/0.21}$ tokens (Kaplan et al. 2020), or in the training data (H. Zhang et al. 2024; Bergsma et al. 2025).

Proposition. With utilization $\min(1, AB/n^*)$ and data multiplier $1 + B/B_{crit}$, training time is minimized at $B = n^*/A$, where it exceeds the time of a compute-bound run at minimum data by the factor

$$1 + \frac{n^*}{AB_{crit}}.$$

Proof. Time is proportional to $(1 + B/B_{crit})/\min(1, AB/n^*)$. Above $B = n^*/A$ it increases with B ; below, it equals $n^*/(AB) + n^*/(AB_{crit})$ and decreases with B . $\square$

A numerical minimization agrees with the closed form to 3.8×10^{-8} . Sparse training of streamed parameters therefore costs at most twice the compute-bound minimum only if $A \geq n^*/B_{crit}$ (Figure 2b). With the loss law, B_{crit} is 7.4×10^6 tokens at a loss of 2.0 and 2.9×10^7 at 1.5, so the activation fraction must exceed 1.8×10^{-4} or 4.5×10^{-5} . At $A = 10^{-4}$ and a loss of 1.5 the penalty is 1.45; at $A = 10^{-5}$ it is 5.48. A law in which B_{crit} grows with the data, fitted on models up to 3.3 billion parameters (Bergsma et al. 2025), gives 5.4×10^8 tokens at $D = 4 \times 10^{14}$ and moves the threshold to 2.4×10^{-6} . The two laws were fitted on dense models of at most a few billion parameters, they disagree by two orders of magnitude when extrapolated, and no measurement says whether the critical batch of a routed model is set by its total or its active parameters. The activation fractions that make a 1P model affordable in compute, 10^{-3} to 10^{-4} , lie inside the range where that unmeasured quantity decides whether streamed training is efficient.

The bound can be relaxed in two ways. The first lowers q . Eight-bit moments with a full-precision master copy (Dettmers et al. 2021) give $q = 22$ and $n^* = 840$ on a TPU v4; a factored second moment without momentum (Shazeer and Stern 2018) gives $q = 18$; an optimizer without state gives $q = 10$. Accumulating gradients over many steps and updating rarely cannot go below the 8 bytes per step needed to read the weights twice and maintain a gradient accumulator, so with streamed weights $n^* \geq 306$ on a TPU v4 and the threshold at a loss of 2.0 stays above 4.1×10^{-5} . Optimizer state accounts for most of q : keeping weights and gradients next to the arithmetic while streaming the state still leaves $q = 24$, a saving of only 29%.

The second lowers β by holding the training state where the arithmetic happens. A wafer-scale engine with 7.5 PFLOP/s and 20 PB/s of on-chip memory bandwidth has $\beta = 0.375$ and $n^* = 2.1$ (Z. Zhang et al. 2024), which removes the bound for any practical activation fraction, at a capacity of 40 GB per wafer. Weight-stationary dataflows (Chen et al. 2016; Sze et al. 2017) and analog in-memory computing, in which a crossbar of programmable conductances performs a matrix-vector product where the weights are stored (Ielmini and Wong 2018; Sebastian et al. 2020), go further. For a peta-parameter model this is the quantitative content of moving activations to the weights: with streamed state, sparsity raises the batch the hardware needs while the optimization caps the batch it can use.

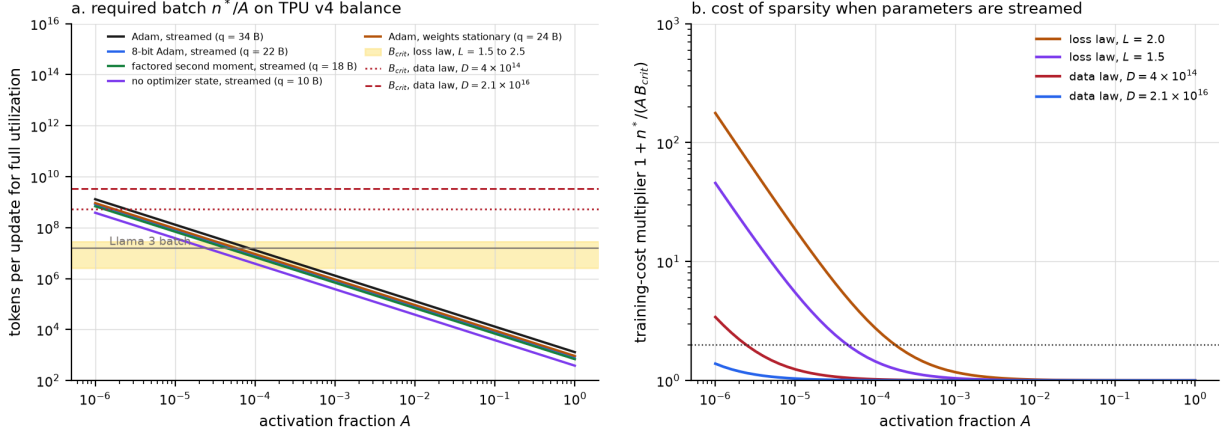


Figure 2: The sparsity-batch bound. (a) Tokens per update needed for compute-bound training, n^*/A , on the balance of a TPU v4, for five accounts of the bytes moved per parameter per update, against the critical batch size from a loss-based law (L from 1.5 to 2.5), from a data-based law at two data sizes, and the Llama 3 batch. (b) Training-cost multiplier at the optimal batch, $1 + n^*/(AB_{crit})$, with $n^* = 1,299$; the dotted line marks a factor of 2.

Expert communication

Routing adds traffic between devices. Per token and expert layer, dispatching the activation to k experts and combining their outputs moves $2kdb_a$ bytes forward and as much backward, for hidden dimension d and b_a bytes per activation, while the layer performs about $6 \times 12d^2$ operations. If all routed traffic leaves the device, the network keeps up when $d \geq \beta_{net}kb_a/18$, where β_{net} is the ratio of sustained arithmetic to network bandwidth. With 343 TFLOP/s per GPU and 50 GB/s of InfiniBand per GPU, as in DeepSeek-V3’s cluster, $\beta_{net} = 6,852$ and the bound is $d \geq 6,091$ for 8 experts with BF16 activations, or 3,045 with 8-bit activations. DeepSeek-V3’s hidden dimension of 7,168 clears it by a factor of 1.18, and the model also sends each token to at most 4 nodes and overlaps communication with computation (DeepSeek-AI et al. 2024). The bound depends on k , d and the network, and it does not grow with N ; a peta-parameter model inherits the communication problem of current mixtures of experts and adds the problem of routing among far more experts. Optically reconfigurable interconnects (Jouppi et al. 2023) and optical links whose energy per bit approaches the attojoule range (Miller 2017) address the bandwidth term.

Reliability and checkpointing

Large training runs fail often. In a 54-day period of Llama 3 405B pre-training on 16,384 GPUs there were 419 unexpected interruptions, about 78% attributed to confirmed or suspected hardware problems, and the storage fabric sustained 2 TB/s of checkpoint throughput, with peaks of 7 TB/s (Grattafiori et al. 2024). Per GPU, that is one interruption every 5.78 years and 122 MB/s of checkpoint bandwidth.

Proposition. Let a cluster of G devices fail independently with per-device mean time between failures M , write and read checkpoints at per-device bandwidth w , and hold b bytes of restart state per parameter. With exponential failures and checkpoint and restart times equal to the time to write the state, the largest achievable fraction of time spent on useful training depends only on

$$\rho = \frac{Nb}{wM},$$

and equals $\max_x x e^{-\rho} / (e^{x+\rho} - 1)$, where x is the checkpoint interval in units of the system's mean time between failures.

Proof. The system fails at rate G/M , so its mean time between failures is $M_s = M/G$, and it writes its state in time $\delta = Nb/(Gw)$; hence $\delta/M_s = \rho$ and G cancels. In the standard renewal argument, the expected time to complete a segment of useful length τ followed by a checkpoint, with a restart of length R after each failure, is $M_s e^{R/M_s} (e^{(\tau+\delta)/M_s} - 1)$. Setting $R = \delta = \rho M_s$ and $x = \tau/M_s$ and dividing τ by this time gives the expression. $\square$

For small ρ the efficiency is close to $1 - \sqrt{2\rho}$, the first-order result behind Young's optimal checkpoint interval (Young 1974; Daly 2006). A Monte Carlo simulation of failures, checkpoints and interrupted restarts agrees with the exact expression to within 0.12% for ρ from 10^{-3} to 0.54 (Figure 3a). Restart state is the 12 bytes per parameter of full-precision weights and moments. At the Llama 3 per-device values, ρ is 2.2×10^{-4} for 405 billion parameters and the efficiency is 0.979, consistent with the more than 90% effective training time the run reports. The efficiency is 0.967 at 10^{12} parameters, 0.895 at 10^{13} , 0.670 at 10^{14} and 0.167 at 10^{15} , where $\rho = 0.54$. Adding devices does not help, because each added device brings its own failures and its own bandwidth. Writing at the fabric's peak rate raises the efficiency at 10^{15} to 0.465; reaching 0.9 requires the product wM to grow 110-fold.

A system whose modules checkpoint and restart independently, so that one module's failure does not roll back the others, divides ρ by the number of modules K : the efficiency at 10^{15} parameters becomes 0.670 with 10 modules, 0.895 with 100 and 0.967 with 1,000 (Figure 3b). Independent recovery requires that modules can make progress without waiting for one another. Training methods in which separately trained experts are later merged or mixed (Li et al. 2022; Sukhbaatar et al. 2024), and low-communication schemes in which islands of devices synchronize rarely (Douillard et al. 2023), are the existing steps in that direction; at 10^{15} parameters they become a reliability requirement as well as a communication saving.

Endurance, precision and energy

Weight-stationary hardware stores weights in the devices that compute with them, so training rewrites those devices at every update. The number of updates is D/B : 1.3×10^9 at the compute-optimal token count and the Llama 3 batch, and 2.5×10^7 on the stock of public text. Differential oxide memories programmed at low voltage have reached an endurance of billions of cycles (Bocquet et al. 2018), the order of the update count at the compute-optimal token count. Train-

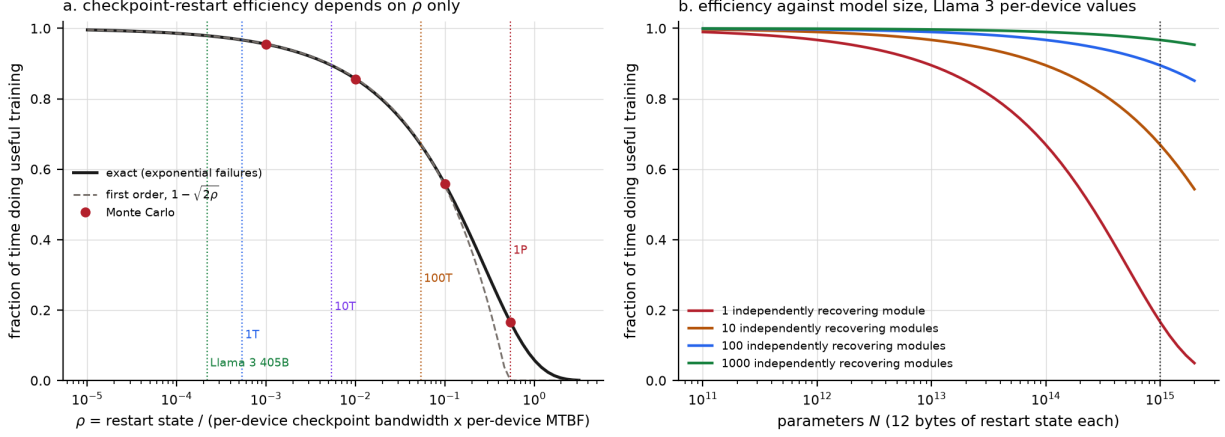


Figure 3: Checkpoint-restart efficiency. (a) Fraction of time spent on useful training against $\rho = Nb/(wM)$: the exact optimum for exponential failures, the first-order approximation $1 - \sqrt{2\rho}$ and a Monte Carlo simulation, with the values of ρ for five model sizes at the per-device storage bandwidth and failure rate of the Llama 3 405B run. (b) Efficiency against parameter count at those per-device values, for a single synchronous job and for 10, 100 and 1,000 modules that recover independently.

ing with analogue memory has reached accuracy equivalent to software (Ambrogio et al. 2018), and schemes that accumulate updates in low-precision digital memory and write the analog devices rarely keep the number of writes a small fraction of endurance (Joshi et al. 2021). Precision is the second constraint: a resistive compute-in-memory chip reached software-comparable accuracy for models quantized to 4-bit weights (Wan et al. 2022), and an analog chip with 35 million phase-change devices reached near-software accuracy on a speech model with 45 million weights (Ambrogio et al. 2023; see also Le Gallo et al. 2023). Both demonstrations are inference on models of tens of millions of weights.

The energy argument for keeping data in place is old. In a 45 nm process a 32-bit read from DRAM costs 640 pJ, against 3.7 pJ for a floating-point multiply and 5 pJ for a read from a 32 KB SRAM (Horowitz 2014; Han et al. 2016), so a byte from DRAM costs about 70 times an operation. Under that table the energy of moving $q = 34$ bytes per parameter per update equals the energy of the arithmetic when each parameter sees 394 tokens per update, an energy counterpart of n^* . The analog chip’s 12.4 TOPS/W corresponds to 81 fJ per operation (Ambrogio et al. 2023); a forward pass through 10^{12} active parameters would then cost 0.16 J, against 3.5 J at the H100’s power per sustained operation, a factor of 21.7 that compares low-precision analog operations with BF16 arithmetic and omits system overheads. Photonic matrix multiplication (Shen et al. 2017) has been projected to reach an energy advantage of more than 8,000 times over digital processors for Transformers at the quadrillion-parameter scale, a projection from small optical experiments and simulation (Anderson et al. 2023). Three-dimensional integration of memory and logic shortens the distances over which data move (Shulaker et al. 2017).

Three routes to 10^{15} parameters

The calculator places three ways of building a 1P model against the bounds above. A dense model on conventional accelerators fails on compute: 1.27×10^{32} FLOP, 10,077 years on 10^6 devices and 7,054 GW-years at the compute-optimal ratio, and still 190 years on 10^6 devices on the stock of public text. A sparse model on conventional accelerators fits a one-year budget on 10^6 devices at $A = 10^{-4}$, inside an energy budget of 0.71 GW-years; it is then constrained by the data, by the sparsity-batch bound whose position depends on the unmeasured critical batch of routed models, and by checkpoint-restart efficiency, which falls to 0.17 for a synchronous job at today's per-device reliability. A weight-stationary or in-memory substrate removes the batch bound and most of the movement energy and inherits limits of endurance, precision and capacity per device. None of the three routes is excluded by physics. The second is buildable from existing components if routed models keep improving at activation fractions near 10^{-4} and if training can be organized into modules that recover independently; the third becomes necessary if the critical batch of routed models turns out to be small.

Open measurements

The engineering bounds are computable; the deciding quantities are not yet measured. The first is the scaling of capability with total parameters at fixed active parameters, at activation fractions of 10^{-3} and below, which effective-parameter analyses of routed models (Clark et al. 2022) and sparsity scaling laws (Abnar et al. 2025) have begun to measure at much smaller scale. The second is the critical batch size of routed models, and whether it is governed by total parameters, active parameters or the tokens each expert sees. The third is whether experts trained with stale or asynchronous updates, as independent recovery requires, lose sample efficiency. The fourth is the supply of tokens beyond the stock of human text. Each can be measured on families of models from 10^9 to 10^{12} parameters with the activation fraction varied over three orders of magnitude.

A 1P model also has to beat the alternatives for the same resources. A retrieval-enhanced model with a 2-trillion-token database matched models with 25 times more parameters (Borgeaud et al. 2021), and additional computation at inference time can substitute for a larger model on some tasks (Snell et al. 2024). Parameters, retrieval and inference-time computation buy capability at different costs, and the case for 10^{15} learned parameters rests on parameters remaining the cheapest of the three for some class of capabilities.

Limitations

The calculator uses published figures for specific devices and one training run; other hardware generations change the constants and leave the forms of the bounds unchanged. Compute is counted as $6AN$ per token, which omits attention cost at long context, the router and load imbalance. Time and energy use the sustained throughput and device power of one run and exclude cooling, networking and storage. The critical-batch laws and the Chinchilla fit are extrapolated far beyond the models on which they were measured. The reliability model assumes independent exponential

failures and restart times equal to checkpoint times; correlated failures and silent data corruption make it optimistic. The analog and photonic figures are chip-level or projected and refer to inference. Hardware announced but not yet documented in sources we could check is not used.

Conclusion

Storage, compute and energy for a peta-parameter model become manageable at activation fractions of 10^{-3} to 10^{-4} on 10^6 current accelerators, and data do not: the compute-optimal token count is 53 times the public text stock. Two bounds do not yield to cluster size. When state is streamed, sparse training needs $AB_{crit} \geq n^* = \beta q/6$, and the activation fractions that make the compute affordable sit where the unmeasured critical batch of routed models decides the matter. Checkpoint-restart efficiency depends on $\rho = Nb/(wM)$ alone and falls to 0.17 at 10^{15} parameters at today’s per-device storage bandwidth and failure rate, unless the system recovers in independent modules. Weight-stationary memory, low-state optimizers and modular training address these terms, which gives each candidate technology a measurable target.

Reproducibility

The calculator in `simulation/` takes every constant from `params.py`, where each value carries its source and locator, and regenerates every number and figure: `uv run run_all.py` writes `output/results.json` and the three figures and evaluates 20 internal checks, including closed forms against numerical optimization, the checkpoint model against Monte Carlo simulation, and the accounting against published compute figures. The claims ledger `claims.yaml` binds each reported number to its location in `results.json` through a recorded execution.

References

- Abnar, Samira, Harshay Shah, Dan Busbridge, Alaaeldin Mohamed Elnouby Ali, Josh Susskind, and Vimal Thilak. 2025. “Parameters Vs FLOPs: Scaling Laws for Optimal Sparsity for Mixture-of-Experts Language Models.” Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2501.12370>.
- Ambrogio, S., P. Narayanan, A. Okazaki, et al. 2023. “An Analog-AI Chip for Energy-Efficient Speech Recognition and Transcription.” *Nature* 620 (7975): 768–75. <https://doi.org/10.1038/s41586-023-06337-5>.
- Ambrogio, Stefano, Pritish Narayanan, Hsin-yu Tsai, et al. 2018. “Equivalent-Accuracy Accelerated Neural-Network Training Using Analogue Memory.” *Nature* 558 (7708): 60–67. <https://doi.org/10.1038/s41586-018-0180-5>.
- Anderson, Maxwell G., Shi-Yuan Ma, Tianyu Wang, Logan G. Wright, and Peter L. McMahon. 2023. “Optical Transformers.” Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2302.10360>.
- Bergsma, Shane, Nolan Dey, Gurpreet Gosal, Gavia Gray, Daria Soboleva, and Joel Hestness. 2025. “Power Lines: Scaling Laws for Weight Decay and Batch Size in LLM Pre-Training.” Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2505.13738>.

- Bocquet, M., T. Hirztl, J.-O. Klein, et al. 2018. "In-Memory and Error-Immune Differential RRAM Implementation of Binarized Deep Neural Networks." *2018 IEEE International Electron Devices Meeting (IEDM)*, 20.6.1–4. <https://doi.org/10.1109/iedm.2018.8614639>.
- Borgeaud, Sebastian, Arthur Mensch, Jordan Hoffmann, et al. 2021. "Improving Language Models by Retrieving from Trillions of Tokens." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2112.04426>.
- Chen, Yu-Hsin, Joel Emer, and Vivienne Sze. 2016. "Eyeriss: A Spatial Architecture for Energy-Efficient Dataflow for Convolutional Neural Networks." *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, 367–79. <https://doi.org/10.1109/isca.2016.40>.
- Clark, Aidan, Diego de las Casas, Aurelia Guy, et al. 2022. "Unified Scaling Laws for Routed Language Models." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2202.01169>.
- Daly, J.T. 2006. "A Higher Order Estimate of the Optimum Checkpoint Interval for Restart Dumps." *Future Generation Computer Systems* 22 (3): 303–12. <https://doi.org/10.1016/j.future.2004.11.016>.
- DeepSeek-AI, Aixin Liu, Bei Feng, et al. 2024. "DeepSeek-V3 Technical Report." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2412.19437>.
- Dettmers, Tim, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 2021. "8-Bit Optimizers via Block-Wise Quantization." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2110.02861>.
- Douillard, Arthur, Qixuan Feng, Andrei A. Rusu, et al. 2023. "DiLoCo: Distributed Low-Communication Training of Language Models." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2311.08105>.
- Fedus, William, Barret Zoph, and Noam Shazeer. 2021. "Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2101.03961>.
- Frantar, Elias, and Dan Alistarh. 2023. "QMoE: Practical Sub-1-Bit Compression of Trillion-Parameter Models." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2310.16795>.
- Grattafiori, Aaron, Abhimanyu Dubey, Abhinav Jauhri, et al. 2024. "The Llama 3 Herd of Models." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2407.21783>.
- Han, Song, Xingyu Liu, Huizi Mao, et al. 2016. "EIE: Efficient Inference Engine on Compressed Deep Neural Network." *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, 243–54. <https://doi.org/10.1109/isca.2016.30>.
- He, Xu Owen. 2024. "Mixture of A Million Experts." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2407.04153>.

- Hoffmann, Jordan, Sebastian Borgeaud, Arthur Mensch, et al. 2022. "Training Compute-Optimal Large Language Models." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2203.15556>.
- Horowitz, Mark. 2014. "Computing's Energy Problem (and What We Can Do about It)." 2014 *IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, 10–14. <https://doi.org/10.1109/isscc.2014.6757323>.
- Ielmini, Daniele, and H.-S. Philip Wong. 2018. "In-Memory Computing with Resistive Switching Devices." *Nature Electronics* 1 (6): 333–43. <https://doi.org/10.1038/s41928-018-0092-2>.
- Joshi, Vinay, Wangxin He, Jae-sun Seo, and Bipin Rajendran. 2021. "Hybrid In-Memory Computing Architecture for the Training of Deep Neural Networks." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2102.05271>.
- Jouppi, Norm, George Kurian, Sheng Li, et al. 2023. "TPU v4: An Optically Reconfigurable Supercomputer for Machine Learning with Hardware Support for Embeddings." *Proceedings of the 50th Annual International Symposium on Computer Architecture*, 1–14. <https://doi.org/10.1145/3579371.3589350>.
- Kaplan, Jared, Sam McCandlish, Tom Henighan, et al. 2020. "Scaling Laws for Neural Language Models." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2001.08361>.
- Kimi Team, Yifan Bai, Yiping Bao, et al. 2025. "Kimi K2: Open Agentic Intelligence." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2507.20534>.
- Kingma, Diederik P., and Jimmy Ba. 2014. "Adam: A Method for Stochastic Optimization." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.1412.6980>.
- Krajewski, Jakub, Jan Ludziejewski, Kamil Adamczewski, et al. 2024. "Scaling Laws for Fine-Grained Mixture of Experts." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2402.07871>.
- Lample, Guillaume, Alexandre Sablayrolles, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2019. "Large Memory Layers with Product Keys." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.1907.05242>.
- Le Gallo, Manuel, Riduan Khaddam-Aljameh, Milos Stanisavljevic, et al. 2023. "A 64-Core Mixed-Signal in-Memory Compute Chip Based on Phase-Change Memory for Deep Neural Network Inference." *Nature Electronics* 6 (9): 680–93. <https://doi.org/10.1038/s41928-023-01010-1>.
- Lepikhin, Dmitry, HyoukJoong Lee, Yuanzhong Xu, et al. 2020. "GShard: Scaling Giant Models with Conditional Computation and Automatic Sharding." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2006.16668>.
- Li, Margaret, Suchin Gururangan, Tim Dettmers, et al. 2022. "Branch-Train-Merge: Embarrassingly Parallel Training of Expert Language Models." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2208.03306>.

- Ludziejewski, Jan, Maciej Pióro, Jakub Krajewski, et al. 2025. "Joint MoE Scaling Laws: Mixture of Experts Can Be Memory Efficient." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2502.05172>.
- Ma, Shuming, Hongyu Wang, Lingxiao Ma, et al. 2024. "The Era of 1-Bit LLMs: All Large Language Models Are in 1.58 Bits." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2402.17764>.
- McCandlish, Sam, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. 2018. "An Empirical Model of Large-Batch Training." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.1812.06162>.
- Micikevicius, Paulius, Sharan Narang, Jonah Alben, et al. 2017. "Mixed Precision Training." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.1710.03740>.
- Miller, David A. B. 2017. "Attojoule Optoelectronics for Low-Energy Information Processing and Communications." *Journal of Lightwave Technology* 35 (3): 346–96. <https://doi.org/10.1109/jlt.2017.2647779>.
- Rajbhandari, Samyam, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. "ZeRO: Memory Optimizations Toward Training Trillion Parameter Models." *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–16. <https://doi.org/10.1109/sc41405.2020.00024>.
- Rajbhandari, Samyam, Olatunji Ruwase, Jeff Rasley, Shaden Smith, and Yuxiong He. 2021. "ZeRO-Infinity: Breaking the GPU Memory Wall for Extreme Scale Deep Learning." *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–14. <https://doi.org/10.1145/3458817.3476205>.
- Sebastian, Abu, Manuel Le Gallo, Riduan Khaddam-Aljameh, and Evangelos Eleftheriou. 2020. "Memory Devices and Applications for in-Memory Computing." *Nature Nanotechnology* 15 (7): 529–44. <https://doi.org/10.1038/s41565-020-0655-z>.
- Shazeer, Noam, Azalia Mirhoseini, Krzysztof Maziarsz, et al. 2017. "Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.1701.06538>.
- Shazeer, Noam, and Mitchell Stern. 2018. "Adafactor: Adaptive Learning Rates with Sublinear Memory Cost." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.1804.04235>.
- Shen, Yichen, Nicholas C. Harris, Scott Skirlo, et al. 2017. "Deep Learning with Coherent Nanophotonic Circuits." *Nature Photonics* 11 (7): 441–46. <https://doi.org/10.1038/nphoton.2017.93>.
- Shulaker, Max M., Gage Hills, Rebecca S. Park, et al. 2017. "Three-Dimensional Integration of Nanotechnologies for Computing and Data Storage on a Single Chip." *Nature* 547 (7661): 74–78. <https://doi.org/10.1038/nature22994>.

- Snell, Charlie, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. "Scaling LLM Test-Time Compute Optimally Can Be More Effective Than Scaling Model Parameters." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2408.03314>.
- Sukhbaatar, Sainbayar, Olga Golovneva, Vasu Sharma, et al. 2024. "Branch-Train-MiX: Mixing Expert LLMs into a Mixture-of-Experts LLM." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2403.07816>.
- Sze, Vivienne, Yu-Hsin Chen, Tien-Ju Yang, and Joel S. Emer. 2017. "Efficient Processing of Deep Neural Networks: A Tutorial and Survey." *Proceedings of the IEEE* 105 (12): 2295–329. <https://doi.org/10.1109/jproc.2017.2761740>.
- Villalobos, Pablo, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. 2024. "Will We Run Out of Data? Limits of LLM Scaling Based on Human-Generated Data." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2211.04325>.
- Wan, Weier, Rajkumar Kubendran, Clemens Schaefer, et al. 2022. "A Compute-in-Memory Chip Based on Resistive Random-Access Memory." *Nature* 608 (7923): 504–12. <https://doi.org/10.1038/s41586-022-04992-8>.
- Williams, Samuel, Andrew Waterman, and David Patterson. 2009. "Roofline: An Insightful Visual Performance Model for Multicore Architectures." *Communications of the ACM* 52 (4): 65–76. <https://doi.org/10.1145/1498765.1498785>.
- Young, John W. 1974. "A First Order Approximation to the Optimum Checkpoint Interval." *Communications of the ACM* 17 (9): 530–31. <https://doi.org/10.1145/361147.361115>.
- Zhang, Hanlin, Depen Morwani, Nikhil Vyas, et al. 2024. "How Does Critical Batch Size Scale in Pre-Training?" Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2410.21676>.
- Zhang, Zuoning, Dhruv Parikh, Youning Zhang, and Viktor Prasanna. 2024. "Benchmarking the Performance of Large Language Models on the Cerebras Wafer Scale Engine." Preprint, arXiv. <https://doi.org/10.48550/ARXIV.2409.00287>.