OPEN LLM PROJECTS SHOULD ALLOCATE MORE COMPUTE FOR DATA THAN TRAINING

Maximilian Idahl[1,2]*

[1]ellamind [2]Leibniz University Hannover

ABSTRACT

Open LLM projects aim to build the best possible open language models under constrained compute budgets. Currently, most allocate the vast majority of their GPU compute to training runs rather than better data. This position paper argues that these efforts should invest the majority of their compute in data, not training. Reported efficiency gains of 6-9x from data curation, filtering, and synthetic generation justify allocating 80% or more of development compute to data work. Beyond producing better models, data investments compound across model generations while individual models are often superseded within months. We discuss allocation strategies and call for open LLM projects to adopt explicitly data-centric compute accounting.

1 INTRODUCTION

Open LLM efforts aim to build the best possible models under a fixed development compute budget.[1] For these efforts, the central question is not whether to train a model, but how to allocate compute to maximize capability.

Current practice often answers this question implicitly: most development compute goes to the final training run, with data work receiving only a small fraction of the budget. As a result, projects spend the bulk of their compute budget on producing a single set of weights rather than on improving the data.

Recent results suggest that heavily investing in data work can match or exceed frontier performance with substantially less training compute. AI2's OLMo 3 matches Qwen-3 32B while training on approximately 6× fewer tokens (Team OLMo et al., 2025). FineWeb-Edu achieves ∼8× token efficiency through quality filtering (Penedo et al., 2024). Maini et al. (2025) report 7.7× faster training through systematic rephrasing of web documents. These are multiplicative improvements in capability per compute dollar, not marginal gains.

Beyond immediate efficiency, data investments also exhibit a durability that training investments lack: a curated dataset from 2024 can train models in 2024, 2025, and 2026, while a model trained in early 2024 is likely superseded within the same year. Data investments persist across model generations; training investments depreciate within months.

Our position is that **open LLM efforts should invest the large majority of their compute in data, and not in model training**. Efficiency multipliers of 6–9× imply that the majority of compute, approximately 80%, can profitably go to data work. This compute funds three modes: selection (annotation and quality filtering), transformation (rephrasing and restructuring), and generation (synthetic data at scale). Training-only scaling faces hard limits when high-quality data is exhausted or repeated; data compute offers a path to continued capability gains by producing higher-utility tokens. We complement the position of Kandpal & Raffel (2025), who argue that data should be the

*The positions expressed in this paper are the author's own and do not necessarily reflect the views of affiliated institutions or projects.

[1]We refer to projects releasing open-weight LLMs with transparency about training recipes and data provenance. Our analysis focuses on pretraining; the principles may extend to fine-tuning but the evidence we survey is drawn from pretraining pipelines.

most expensive part of an LLM through fair compensation to data creators; we argue it should be the most expensive part from a compute allocation perspective.

We outline three forms of data compute (Section 2) and derive a break-even formula that justifies the 80% allocation (Section 3). Then, we argue why data investments compound while training investments depreciate (Section 4). We propose a compute accounting standard and a research agenda (Section 5), and address alternative views (Section 6).

2 Data Compute: What to Invest Compute In

If data-centric allocation produces better models, then labs should invest in **data compute**: compute spent to increase training signal per FLOP by selecting, transforming, and generating training data. We organize it into three modes and illustrate each with concrete projects that report costs and measurable gains.

Selection Compute. Selection compute encompasses model-driven data curation, including LLM-based annotation, learned scoring models, and large-scale filtering decisions. FineWeb-Edu (Penedo et al., 2024) is an illustrative example: 450k documents are labeled with Llama-3 70B (Team, 2024) to produce supervision for an "educational value" signal, a small encoder model is trained to score documents at web scale, and the 15T-token FineWeb corpus is filtered by thresholding on those scores. This selection process yields roughly an $8\times$ token-efficiency gain, with recent work confirming similar gains for its multilingual counterpart FineWeb-2 (Penedo et al., 2025; Messmer et al., 2025). Scoring and filtering 15 trillion tokens consumed approximately 6,000 H100 GPU-hours; the full project, including all experiments, required roughly 80,000 H100 GPU-hours. The latter figure sounds large in isolation, but it is modest relative to frontier model training budgets: 80,000 GPU-hours is approximately 1.6% of a representative 5-million GPU-hour training run. This asymmetry motivates our claim of systematic underinvestment: if 1.6% of a frontier compute budget can produce multi-fold efficiency gains, then scaling selection compute is among the highest-return uses of additional GPU-hours.

A second example is presented in Nemotron-CC Su et al. (2025), which uses a classifier ensemble to bucket documents into quality tiers rather than applying a single keep/discard threshold, enabling tier-specific downstream processing (Su et al., 2025).

Despite this potential, current practice remains conservative: both examples rely on small models chosen for throughput, not capability, such as FastText classifiers or lightweight encoder models. Yet selection is arguably the most consequential decision in dataset construction: a single score per document, applied at billion-document scale, can shape the entire training distribution. Larger or more expressive selection models (e.g., billion-parameter LLMs, multi-attribute classifiers, or ensembles) could move beyond binary keep/discard decisions to capture dimensions like reasoning depth or time-sensitivity of documents. Such models would enable fine-grained filtering, curriculum design, and experimentation with selection criteria that remain unexplored at scale. Given the downstream efficiency gains, economizing on selection compute is a false economy; the marginal dollar is better spent on smarter curation than on additional training tokens. However, aggressive filtering carries risks: quality thresholds can inadvertently remove valuable diversity, as FineWeb's finding that global deduplication yields worse outcomes than per-snapshot approaches demonstrates (Penedo et al., 2024). A single definition of data quality in terms of educational value might be insufficient to capture the full diversity of good training signals. This is why selection compute must fund ablation sweeps to detect harmful heuristics before they are baked into final datasets.

Transformation Compute. Transformation compute encompasses document rephrasing, format lifting, extraction, and structure imposition. Instead of synthesizing content from scratch, transformation takes a source document as input and produces a cleaner, denser, or more structured version, preserving grounding in real sources. BeyondWeb (Maini et al., 2025) illustrates what targeted transformation can achieve: $7.7\times$ faster training compared to raw web data. The efficiency gains stem from multiple interacting mechanisms. Rephrasing compresses knowledge into denser tokens. Style transformations (e.g., converting to Q&A or instructional formats) close the gap between web-heavy pretraining distributions and conversational deployment, though gains from style-matching alone saturate quickly. Diversity across transformation strategies is what sustains efficiency at scale: single-

strategy approaches such as Q&A-only plateau early, while a diverse mix of formats, restructurings, and style modifications keeps improving across longer training. Critically, BeyondWeb surpasses the "full data upper bound" that naive data augmentation cannot break, suggesting that strategic restructuring fills distributional gaps that raw web data underrepresents. Seed quality also matters, though the relationship is nuanced. BeyondWeb finds that rephrasing high-quality source documents outperforms rephrasing lower-quality web text, even when the latter provides more novelty.

Nemotron-CC takes a tier-specific approach, using Mistral NeMo 12B to apply Wikipedia-style rewrites that denoise low-quality pages, while high-quality pages receive more aggressive transforms (QA generation, distillation, knowledge extraction) to produce denser, more structured tokens (Su et al., 2025). The result is a 6.3T-token corpus with 1.9T transformed tokens, demonstrating that transformation compute can be targeted by source quality rather than applied uniformly. This philosophy favors rephrasing over discarding: rather than aggressively filtering away most of the web, transformation can rescue lower-quality documents while enriching high-quality ones. Nemotron Nano 2 (Basant et al., 2025), trained on Nemotron-CC-v2, extends this further by treating transformation as a curriculum knob: the share of transformed and SFT-style data increases in later training phases, rising to over 30% by the final phase. Their ablations underscore the value: synthesized multilingual QA pairs outperform curated multilingual crawl data, leading the authors to weight transformed data more heavily in the final mixture. Like selection, transformation compute is likely underinvested: if rephrasing existing documents yields large training efficiency and model performance gains, scaling transformation pipelines is a high-return use of compute.

Generation Compute. Generation compute produces synthetic corpora from scratch or from minimal seeds, rather than transforming existing documents. Where transformation requires source documents to rewrite, generation can cover domains where good sources are scarce, create novel reasoning chains, or explore distributions that natural data underrepresents. Synthetic data generation is also a response to the "data wall" (Villalobos et al., 2024): as high-quality natural text is exhausted, synthetic generation offers a path to continued scaling. Moreover, aggressive selection significantly reduces corpus size (Penedo et al., 2024; Su et al., 2025): a dataset filtered for quality may end up being too small to train frontier models, even if each token is highly efficient. Synthetic data can restore scale while preserving the gains from selection, producing a large corpus of high-value tokens rather than forcing a choice between quality and quantity. Nemotron-CC-v2 (Basant et al., 2025) uses multiple LLMs as data generators for math, code, and tool-calling corpora, where correctness can often be verified programmatically. They mostly rely on large mixture-of-expert (MoE) models, such as Qwen3-235B (Yang et al., 2025) and DeepSeek-R1 (DeepSeek-AI, 2025) to produce high-quality tokens at high inference throughput. TOUCAN (Xu et al., 2025) synthesizes 1.5 million tool-agentic trajectories from real-world MCP servers, using execution-based validation to filter generated data.

Recent fully synthetic pretraining projects push generation compute further. SYNTH (Pleias, 2025), a 75-billion-token synthetic corpus derived from only 50,000 Wikipedia articles, reports allocating approximately 95% of total project compute to data pipeline work (synthetic generation, validation, curation) and only 5% to final training, with the resulting models achieving state of the art performance for their size. This 95/5 split illustrates a broader pattern: as generation scales, the bottleneck shifts from training compute to data-side compute. However, synthetic generation also carries risks: models trained on synthetic data can lose diversity and degrade, a phenomenon termed model collapse (Shumailov et al., 2023). Effective synthetic pipelines must therefore incorporate validation infrastructure: diversity constraints, real-data anchors, and drift detection. SYNTH demonstrates that grounding generation in curated seed documents and applying LLM-as-judge curation can maintain diversity even in fully synthetic corpora (Pleias, 2025). Yet even validated synthetic investments remain modest relative to frontier training budgets, and the field has only begun to explore what systematic, large-scale synthetic generation can achieve.

Transformation and generation compute can easily reach millions of GPU-hours, dwarfing current selection compute by two orders of magnitude. Arcee AI's Trinity models (Arcee-AI, 2025; 2026) illustrate how transformation and generation compute combine at scale, using over 8T synthetic tokens ($\sim$47% of the 17T-token corpus), including $\sim$6.5T synthetic web produced via rephrasing and format transformation, $\sim$1T synthetic multilingual data, and $\sim$0.8T synthetic code. They report that generating this data required clusters peaking at 2,048 H100 GPUs running for approximately one month, which is in the order of more than one million GPU-hours for data synthesis alone. The resulting 400B sparse MoE matches peer models on standard benchmarks, demonstrating that

synthetic-heavy pretraining can scale to frontier performance when transformation and generation are deployed together.

2.1 THE FILTER-THEN-AUGMENT IMPERATIVE

A natural question arises: if filtering like FineWeb-Edu yields $8\times$ token efficiency, why invest in more compute expensive approaches like synthetic data generation? The answer lies in a fundamental tension: **filtering improves training token efficiency but reduces dataset size**.

Both FineWeb-Edu and Nemotron-CC achieve their efficiency gains by discarding the majority of tokens. From original corpora of 15T and 4.5T tokens, aggressive quality filtering produces much smaller high-quality subsets of only 553B and 1.3T tokens, respectively. This creates a ceiling: you cannot filter your way to an arbitrarily large high-quality dataset because filtering is subtractive by nature. At some point, further filtering leaves you with too few tokens to train frontier models, which require trillions of training tokens to reach compute-optimal performance.

Synthetic data generation is the solution to this tension. It is the mechanism for scaling the dataset back up while preserving the efficiency gains from quality filtering. Rather than training on 15T tokens of mixed quality, the data-centric approach is:

1. **Filter aggressively**: Use quality classifiers to identify high-utility tokens, accepting that this drastically reduces dataset size
2. **Augment systematically**: Use transformation and generation (rephrasing, Q&A pair generation, knowledge extraction) to expand the filtered corpus back to frontier scale
3. **Validate continuously**: Use ablation experiments to verify that the resulting tokens maintain the efficiency multiplier

This filter-then-augment pipeline produces datasets that are both large (many trillions of tokens) and high-quality (maintaining the efficiency multiplier). As illustrated by BeyondWeb and Nemotron-CC above, this approach avoids the false choice between quality and scale.

The implication for compute allocation is clear: all three modes deserve substantial investment. Selection is cheap per token but high-leverage; transformation and generation are expensive per token but necessary for scale. Together, they enable training on trillions of high-utility tokens rather than choosing between quality (small filtered datasets) and quantity (large raw datasets).

Recent work on synthetic data scaling laws suggests that naively mixing synthetic tokens into unfiltered web data yields diminishing returns: Kang et al. (2025) recommend capping synthetic data at $\sim 30\%$ when blended with raw CommonCrawl, and Qin et al. (2025) observe performance plateaus at 300B tokens for math-specific synthetic Q&A. However, both studies examine mixing regimes rather than the filter-then-augment pipeline we advocate. When synthetic data replaces filtered-out low-quality content rather than diluting it, the effective ceiling is substantially higher, as demonstrated by Trinity's 47% synthetic corpus and SYNTH's fully synthetic approach.

2.2 WHERE REINFORCEMENT LEARNING WITH VERIFIABLE REWARDS FITS IN

In late 2024, posttraining LLMs using Reinforcement Learning with Verifiable Rewards (RLVR) gained widespread attention with OpenAI's o1 model (OpenAI, 2024). DeepSeek's R1 (DeepSeek-AI, 2025) then demonstrated that RLVR could produce strong reasoning capabilities without supervised fine-tuning on chain-of-thought data, cementing it as a key technique for building reasoning models. Conceptually, RLVR instantiates the data compute paradigm we advocate: rollouts generate candidate trajectories, a verifier provides pass/fail signals, and selection retains successful trajectories. These operations mirror two of the three modes of data compute we emphasize: generation (rollouts produce candidates) and selection (retain winners) (Shao et al., 2024). To illustrate, DeepSeek-R1-Zero claims a $5\times$ performance gain on AIME 2024 (15.6% to 77.9%) through RL training alone, consuming approximately 100,000 H800 GPU-hours to generate and validate millions of reasoning trajectories (DeepSeek-AI, 2025).

It is natural to classify all RL compute as "training compute." We do not insist on a label, but we emphasize the distinction in *mechanism*: whereas traditional supervised training optimizes parameters on a static dataset, RLVR compute actively generates and validates the learning signal itself. Our

claim is about the *economics of leverage*: when compute is spent to create higher-utility training signal, whether via offline synthesis or online rollouts, it can shift the capability-per-FLOP frontier. Whether RLVR compute compounds across model generations remains an open question. Rollout trajectories are generated relative to the current policy, but successful trajectories can in principle be retained as supervised training data for future models, blurring the line between online RL and offline data investment. RLVR is also currently limited to tasks with verifiable objectives such as mathematics, code, and formal reasoning, though the scope of verifiable domains is expanding.

3 How Much Compute to Invest in Data?

If data compute yields substantial efficiency gains, how much of a project's total compute budget should go to it? To answer this, we define the efficiency multiplier, present published empirical evidence, and derive a break-even formula.

3.1 The Token Efficiency Multiplier

For LLMs, the mechanism underlying data-centric allocation is the *token efficiency multiplier*: the factor by which improved data, in the form of tokens, reduces the training compute required to reach a certain target capability or benchmark performance. Let C_{base} be the training compute required to reach a target capability using a baseline data pipeline, and let C_{improved} be the compute required using an improved pipeline, holding model and training recipe constant. We define $m = C_{\text{base}}/C_{\text{improved}}$. Equivalently, assuming training compute scales with tokens processed, m can be interpreted as an *effective-token multiplier*: an improved pipeline maps raw tokens D to quality-adjusted tokens $D_{\text{eff}} = m \cdot D$ relative to the baseline. This is analogous to the scaling-factor formulation in quality-aware scaling laws (Chang et al., 2024; Subramanyam et al., 2025). With our definition, $m = 2$ means the improved data pipeline results in half the training compute being required to reach the same capability, or equivalently, 1T improved tokens yield the same capability as 2T baseline tokens.

3.2 The Break-Even Allocation

Given an efficiency multiplier m, how much compute can profitably go to data work? With baseline training compute C_{base}, and data improvements yielding m-fold training token efficiency, then the training compute required to reach the same target becomes C_{base}/m. One can therefore "spend" up to $C_{\text{base}} - C_{\text{base}}/m$ on data work while matching baseline total compute. This yields the break-even share:

$$\alpha_{\max} \approx 1 - \frac{1}{m} \tag{1}$$

where $\alpha_{\max}$ is the maximum share of compute that can be allocated to data work at break-even. For example, $m = 1.2$ implies $\alpha_{\max} = 17\%$; $m = 2$ implies $\alpha_{\max} = 50\%$; $m = 6$ implies $\alpha_{\max} = 83\%$; $m = 9$ implies $\alpha_{\max} = 89\%$ (Figure 1).

3.3 Token Efficiency Multipliers: Published Evidence

Multiple independent projects report results from which we derive efficiency multipliers in the 6–9$\times$ range (Table 1). They do not always report multipliers directly; we compute m from their token or compute comparisons using the definitions above.

FineWeb-Edu. Quality filtering (Section 2) yields $\sim$7.9$\times$ token efficiency: matching baseline performance with 38B tokens instead of 300B, measured against FineWeb, which already applies URL blocklists, language detection, quality filters, and MinHash deduplication (Penedo et al., 2024).

FineWeb2-HQ. Multilingual quality filtering results in $\sim$6$\times$ training speedup relative to the heuristically filtered FineWeb-2 corpus across 20+ languages (Messmer et al., 2025).

BeyondWeb. Systematic rephrasing yields 7.7$\times$ faster training relative to RedPajama, a minimally curated web baseline, with an 8B model matching 180B-token baseline performance in 23.2B tokens (Maini et al., 2025).

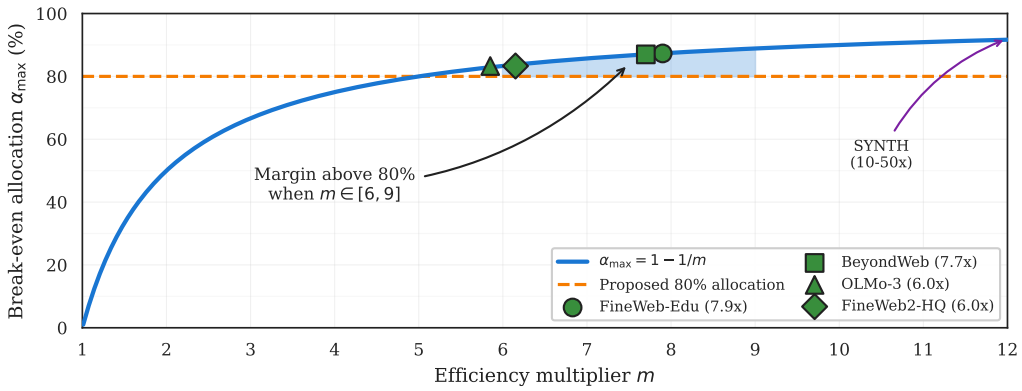

Figure 1: The break-even formula justifies allocating 80% of compute to data work. The curve shows $\alpha_{\max} = 1 - 1/m$: when data work yields m-fold training efficiency, up to $\alpha_{\max}$ of compute can go to data without exceeding baseline cost. Published projects report $m \in [6, 9]$ (data points; see Table 1), implying break-even allocations of 83–89%. The shaded region shows the margin above our recommended 80%.

Table 1: Efficiency multipliers in published works and implied break-even data compute shares. Metric indicates whether efficiency was measured via token reduction (Tokens) or training speedup (Speed) to reach comparable capability.

	Metric	m	$\alpha_{\max}$
FineWeb-Edu	Tokens	$\sim$7.9$\times$	87%
BeyondWeb	Speed	$\sim$7.7$\times$	87%
OLMo-3	Tokens	$\sim$6$\times$	83%
FineWeb2-HQ	Speed	$\sim$6$\times$	83%
SYNTH	Tokens	$\sim$10–50$\times$	90–98%

OLMo-3. Rigorous data curation, including synthetic and rewritten math corpora in midtraining, enables matching Qwen-3 32B performance while training on $\sim$6$\times$ fewer tokens (Team OLMo et al., 2025), where Qwen-3 itself applies extensive curation including annotation of 30T tokens for educational value (Yang et al., 2025).

SYNTH. Fully synthetic pretraining from curated seeds, reporting 10–50$\times$ token efficiency, with $\sim$95% of project compute allocated to data pipeline work (Pleias, 2025).

Nemotron-CC. Unfortunately, both the Nemotron-CC and Nemotron Nano 2 reports do not provide controlled token-sweep comparisons that would allow deriving a multiplier m. At a 1T-token training horizon, the full 6.3T dataset roughly matches DCLM on MMLU (53.0 vs 53.4), while a 1.1T high-quality subset achieves +5.6 MMLU improvement (Su et al., 2025). However, without scaling curves for both datasets, we cannot determine how many baseline tokens would be needed to reach the same capability.

3.4 Recommendation: Allocate 80% to Data

Given the consistency of 6–9$\times$ multipliers across independent projects, we propose 80% as a *default policy*. This is a conservative estimate: $m = 6$ implies $\alpha_{\max} = 83\%$, and $m = 9$ implies $\alpha_{\max} = 89\%$, so 80% sits 3–9 percentage points below break-even. SYNTH's 95/5 split between data and training compute (Pleias, 2025) suggests that even more aggressive allocations may be viable for generation focused on reasoning tasks with constrained seed diversity, though this has so far only been demonstrated at small scale.

Practitioners should follow a three-step process:

1. **Estimate** m via small-scale proxy experiments comparing baseline and curated data pipelines, including data-quality dependent scaling-law derivation.
2. **Apply the break-even formula** to determine $\alpha_{\max} = 1 - 1/m$.
3. **Allocate with a safety margin** below $\alpha_{\max}$ to account for uncertainty in multiplier estimates and pipeline costs.

The safety margin is prudent because m is measured on proxy tasks and may not transfer perfectly to the full training run. Even at the conservative floor of documented multipliers, 80% is justified. Crucially, 80% is a *ceiling justified by break-even analysis*, not a spending target. If a project exhausts high-return data techniques before reaching this budget share, the remaining compute should go to training. The break-even formula determines how much data compute *can* be justified; diminishing returns on specific techniques determine how much *should* be spent.

When multipliers are lower. If proxy experiments yield $m < 6$, the break-even formula still applies. Lower multipliers justify proportionally lower data allocations, but the principle remains: allocate up to, but not beyond, break-even. Notably, for any $m > 2$, the majority of compute should go to data rather than training, so our core thesis holds for any efficiency multiplier above $2\times$.

Splitting across modes. Within the data compute budget, selection is cheap per token but high-leverage; transformation and generation are expensive but necessary for scale. We do not prescribe fixed ratios because the optimal split heavily depends on the quality and size of available source data. Projects starting from high-quality sources (e.g., curated domain corpora) may invest more heavily in generation; projects starting from noisy web crawls may need substantial selection and transformation first.

3.5 Limits of Training-Only Scaling

Training-only scaling faces two fundamental constraints. First, in data-constrained regimes, the marginal value of additional training compute decays toward zero: Muennighoff et al. (2023) demonstrate that repeated epochs on fixed data yield diminishing returns, with performance gains vanishing after sufficient repetition. Second, redundant or low-quality data produces diminishing returns: Chen et al. (2025) demonstrate that high-density datasets with redundant information lead to sub-scaling, where performance improvements decelerate regardless of additional training compute. These constraints are increasingly binding as estimates suggest the stock of public human-generated text may be exhausted by 2026–2032 (Villalobos et al., 2024). The implication is that the binding constraint on capability is not training compute but high-utility token supply, which is exactly what data compute produces.

3.6 Caveats

We acknowledge that efficiency multipliers from different studies may not be directly comparable due to varying experimental setups, baseline choices, and evaluation metrics. Accordingly, m should be treated as conditional on the baseline, target metric, and training recipe, rather than as a universal constant. However, the consistency of the 6–9$\times$ range across independent projects strengthens confidence that substantial training token efficiency gains are achievable. Our 80% recommendation is derived from the conservative floor of this range.

Furthermore, our analysis frames the allocation choice between data compute and the final training run. In practice, substantial compute also goes to ablation experiments, hyperparameter search, and architecture exploration. These recipe-development costs are real and should be accounted for in project planning. However, they are orthogonal to our claim: given whatever compute remains after recipe development, the break-even formula still favors allocating the majority to data work over a single large training run.

4 Data Investments Compound

The preceding analysis treats data and training compute symmetrically within a single training run. However, a fundamental asymmetry governs their medium- and long-term returns: high-quality

datasets remain valuable across multiple model generations, while individual model checkpoints often depreciate in value within months. In this section we argue that this asymmetry further strengthens the case for data-centric allocation.

4.1 The Durability Asymmetry

Model weights exhibit rapid obsolescence. For example, the interval from Llama-1 (Touvron et al., 2023a) to Llama-2 (Touvron et al., 2023b) was 5 months, followed by Llama-3 (Team, 2024) 9 months later. The Stanford AI Index reports 149 foundation models released in 2023, more than double the 2022 count (Maslej et al., 2024), indicating that the model frontier advances rapidly. Architectural innovations, improved training recipes, and expanded capabilities (reasoning, tool use, multimodality) compound this effect: older weights become cost-inefficient relative to newer alternatives within a year of release. Models do retain secondary value as teachers for distillation, baselines for regression testing, or cost-efficient solutions for constrained tasks. However, these residual uses represent a fraction of the original investment's intended return; the model's value as a frontier capability asset depreciates rapidly.

Datasets, by contrast, persist across model generations. FineWeb, released in mid-2024, is continuously updated and has produced multiple derivatives (FineWeb-Edu, FineWeb 2, FineWeb2-HQ) (Penedo et al., 2024; 2025; Messmer et al., 2025), with multiple generations of models trained on them (Apertus, 2025; Allal et al., 2025; Bakouch et al., 2025; Gonzalez-Agirre et al., 2025). Nemotron 3 (NVIDIA, 2025) is the third generation of models trained on Nemotron-CC (Su et al., 2025). Dolma (Soldaini et al., 2024) evolved through three major versions, with Dolma 3 being used to train OLMo 3 (Team OLMo et al., 2025). In each case, the initial curation investment continues to yield value years after the original expenditure.

4.2 Mechanisms of Compounding

Data investments compound through several mechanisms. First, a well-curated dataset can train multiple model generations, including smaller variants and domain-specialized fine-tunes, without repeating the original curation cost. Second, datasets spawn derivatives: quality-filtered subsets, format-transformed versions, and domain-specific extractions each represent new assets built atop prior work. Third, when datasets are released openly, their value multiplies across the ecosystem. FineWeb-Edu has been adopted by numerous open LLM projects, reducing the effective per-project cost of data curation toward zero.

Training compute, by contrast, produces a single artifact, with each new model generation requiring a full training run.

4.3 Limits of Dataset Durability

Datasets also decay, though on longer timescales and through different mechanisms than models. Staleness occurs as facts, language patterns, and user behavior evolve. Contamination risk grows when evaluation data leaks into training corpora, reducing diagnostic value. Legal and licensing uncertainties can render datasets unusable if provenance is not carefully tracked.

This observation reinforces our position: data-centric allocation should fund not only initial curation but also ongoing maintenance.

4.4 Implications for Long-Term Returns

The durability asymmetry implies that the true return on data compute exceeds what single-run break-even analysis suggests. If a dataset serves three model generations, its effective cost per model is one-third of the original investment. If it is shared openly and adopted by ten projects, the per-project cost approaches zero. Dataset maintenance is incremental (adding crawls, updating filters, refreshing classifiers), whereas model "maintenance" requires full retraining. Fine-tuning and continual learning reduce these costs, but major capability jumps (e.g., Llama-1 to Llama-3, OLMo-1 to OLMo-3) have consistently required full retraining on improved data, and even continual learning requires curating the new data it trains on.

This asymmetry has strategic implications for compute allocation. A project investing 80% of its budget in data is constructing infrastructure for multiple generations; a project investing 80% in training is producing a single artifact with a 12–18-month shelf life as a frontier system. For open LLM efforts operating under constrained budgets, the compounding returns of data investment provide a durable competitive advantage that training-centric allocation cannot match.

5 Call to Action

We urge practitioners to rethink compute allocation by following our recommendation in Section 3.4: estimate token-efficiency multiplier via proxy experiments, apply the break-even formula, and allocate compute accordingly. The default assumption that training should dominate the budget is outdated and suboptimal. Our recommendation of 80% data compute reflects a conservative floor of documented multipliers.

For the research community, our position implies various important directions. Scaling law research should incorporate data pipelines, formalizing how effective tokens depend on data compute investment. Data selection research should explore scaling up annotators and scorers with capable LLMs rather than lightweight classifiers, to curate training corpora with richer quality signals. Synthetic data generation research should study how to produce diverse synthetic tokens at trillion-token scale without distributional collapse.

6 Alternative Views

We address three credible alternative positions on compute allocation.

Alternative 1: Training scale is sufficient. Labs should allocate the majority of compute to training because scale has been the primary driver of capability gains. Qwen-3 trained on 36T tokens and achieved state-of-the-art results (Yang et al., 2025); classic scaling laws (Kaplan et al., 2020; Hoffmann et al., 2022) successfully predicted performance while holding data quality fixed. **Our response:** Even apparent brute-force approaches invest heavily in data quality. Qwen-3's training included a 5T-token reasoning stage of high-quality STEM data, annotation of 30T tokens for educational value and domain, and instance-level mixture optimization via ablations. Classic scaling laws answer "how to split compute between model size and tokens *given* a data distribution"; our claim is orthogonal: optimizing compute spent to *change* the distribution shifts the frontier itself (Subramanyam et al., 2025; Chang et al., 2024). For labs without frontier resources, data efficiency is the only viable path: OLMo 3 achieved 80.5% on GSM8K versus Marin's 69.1%, both 32B models with full transparency (Stanford CRFM, 2025; Team OLMo et al., 2025).

Alternative 2: Synthetic data cannot exceed its source. A model trained on synthetic data generated by model X cannot surpass X. Information-theoretically, you cannot extract more signal than the generator contains; distillation research confirms that student models typically underperform their teachers. If synthetic data has a fundamental ceiling, heavy investment in generation compute is wasteful. **Our response:** The distillation analogy is misleading because distillation transfers a distribution, whereas generate-then-verify selects from candidates. The key distinction is reliability versus coverage: a generator with 5% accuracy on hard problems still *contains* correct solutions in its output distribution; verification selects those correct outputs even though the generator cannot produce them reliably. This is precisely the mechanism underlying RLVR (Section 2.2): rollouts generate candidates, verifiers identify successes, and training amplifies what works. DeepSeek-R1-Zero's 5$\times$ capability gain demonstrates that generate-then-verify can exceed what the base model could achieve through supervised learning alone (DeepSeek-AI, 2025). Furthermore, ensembling multiple generators with different failure modes increases coverage beyond any single model. Iterative pipelines compound these gains: a model trained on verified outputs becomes a better generator for the next round. The theoretical ceiling, if it exists, depends on whether generators can produce correct outputs at all, not on whether they can do so reliably.

Alternative 3: Posttraining is where compute should go. Given base model commoditization, the highest-value compute should go to RL posttraining with verifiable rewards (RLVR), not pretraining

data. If posttraining delivers such leverage, pretraining data quality matters less, and we should allocate the majority of compute to posttraining. **Our response:** We agree RLVR is high-leverage (Section 2.2), but view it as supporting our thesis rather than contradicting it. RLVR's leverage comes from investing compute in *signal quality* (verifier design, task distribution, trajectory selection) rather than pure scaling. Rollouts are generation; retaining winners is selection. However, RLVR requires tasks with verifiable objectives (math, code, formal proofs), whereas pretraining data quality is universally relevant. Moreover, weak pretraining caps RLVR gains: you cannot RL-finetune your way out of a poor base model. The underlying principle, that signal quality beats pure scaling, extends from pretraining to posttraining.

7 Conclusion

Open LLM projects face a fundamental choice: allocate compute to training runs that produce rapidly depreciating model weights, or invest in data work that yields compounding returns across model generations. The evidence is clear. Independent projects report efficiency multipliers of 6–9$\times$ from data curation, filtering, and synthetic generation. These are not marginal improvements but multiplicative gains that shift the capability-per-FLOP frontier. The break-even analysis is unambiguous: when data work yields m-fold training efficiency, up to $1 - 1/m$ of compute can profitably go to data. For $m = 6$, this implies 83%; for $m = 9$, it reaches 89%. Our recommended 80% allocation is conservative.

Beyond immediate training token efficiency, data investments possess a durability that training investments lack. A curated dataset serves multiple model generations, spawns derivatives, and, when released openly, multiplies its value across the ecosystem. Model weights, by contrast, depreciate within months as the frontier advances. This asymmetry means that the true return on data compute exceeds what single-run analysis suggests.

The field's default assumption, that training should dominate compute budgets, is a legacy of an era when data was abundant and curation was manual. That era has ended. High-quality natural text is finite; aggressive filtering shrinks corpora below frontier training requirements; and the capability gap between curated and uncurated data continues to widen. Data compute, in the form of selection, transformation, and generation, is the mechanism for escaping these constraints.

We call on open LLM projects to adopt explicitly data-centric compute accounting: estimate efficiency multipliers via proxy experiments, apply the break-even formula, and allocate accordingly. The projects that invest heavily in data infrastructure will not only train better models today but will build durable assets for the models of tomorrow.

Acknowledgments

- This project is supported by the OpenEuroLLM project, co-funded by the Digital Europe Programme under GA no. 101195233. For more information see `https://openeurollm.eu`.
- This work was funded by the German Federal Ministry for Economic Affairs and Climate Action (BMWK) under grant 13IPC040J (SOOFI) based on a resolution of the German Bundestag, and by the European Union – NextGenerationEU. The views and opinions expressed are solely those of the authors and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them.

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlícek, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf. Smollm2: When smol goes big - data-centric training of a small language model. *CoRR*, abs/2502.02737, 2025. doi: 10.48550/ARXIV.2502.02737. URL `https://doi.org/10.48550/arXiv.2502.02737`.

Project Apertus. Apertus: Democratizing open and compliant llms for global language environments. *CoRR*, abs/2509.14233, 2025. doi: 10.48550/ARXIV.2509.14233. URL https://doi.org/10.48550/arXiv.2509.14233.

Arcee-AI. The trinity manifesto. Arcee AI Blog, 2025. https://www.arcee.ai/blog/the-trinity-manifesto.

DatologyAI Arcee-AI, Prime Intellect. Arceetrinity large technical report. github repository, 2026. https://github.com/arcee-ai/trinity-large-tech-report/blob/main/Arcee%20Trinity%20Large.pdf.

Elie Bakouch, Loubna Ben Allal, Anton Lozhkov, Nouamane Tazi, Lewis Tunstall, Carlos Miguel Patiño, Edward Beeching, Aymeric Roucher, Aksel Joonas Reedi, Quentin Gallouédec, Kashif Rasul, Nathan Habib, Clémentine Fourrier, Hynek Kydlicek, Guilherme Penedo, Hugo Larcher, Mathieu Morlon, Vaibhav Srivastav, Joshua Lochner, Xuan-Son Nguyen, Colin Raffel, Leandro von Werra, and Thomas Wolf. SmolLM3: smol, multilingual, long-context reasoner. Hugging Face Blog, 2025. https://huggingface.co/blog/smollm3.

Aarti Basant, Abhijit Khairnar, Abhijit Paithankar, Abhinav Khattar, Adithya Renduchintala, Aditya Malte, Akhiad Bercovich, Akshay Hazare, Alejandra Rico, Aleksander Ficek, Alex Kondratenko, Alex Shaposhnikov, Alexander Bukharin, Ali Taghibakhshi, Amelia Barton, Ameya Sunil Mahabaleshwarkar, Amy Shen, Andrew Tao, Ann Guan, Anna Shors, Anubhav Mandarwal, Arham Mehta, Arun Venkatesan, Ashton Sharabiani, Ashwath Aithal, Ashwin Poojary, Ayush Dattagupta, Balaram Buddharaju, Banghua Zhu, Barnaby Simkin, Bilal Kartal, Bita Darvish Rouhani, Bobby Chen, Boris Ginsburg, Brandon Norick, Brian Yu, Bryan Catanzaro, Charles Wang, Charlie Truong, Chetan Mungekar, Chintan Patel, Chris Alexiuk, Christian Munley, Christopher Parisien, Dan Su, Daniel Afrimi, Daniel Korzekwa, Daniel Rohrer, Daria Gitman, David Mosallanezhad, Deepak Narayanan, Dima Rekesh, Dina Yared, Dmytro Pykhtar, Dong Ahn, Duncan Riach, Eileen Long, Elliott Ning, Eric Chung, Erick Galinkin, Evelina Bakhturina, Gargi Prasad, Gerald Shen, Haifeng Qian, Haim Elisha, Harsh Sharma, Hayley Ross, Helen Ngo, Herman Sahota, Hexin Wang, Hoo Chang Shin, Hua Huang, Iain Cunningham, Igor Gitman, Ivan Moshkov, Jaehun Jung, Jan Kautz, Jane Polak Scowcroft, Jared Casper, Jian Zhang, Jiaqi Zeng, Jimmy Zhang, Jinze Xue, Jocelyn Huang, Joey Conway, John Kamalu, Jonathan M. Cohen, Joseph Jennings, Julien Veron Vialard, Junkeun Yi, Jupinder Parmar, Kari Briski, Katherine Cheung, Katherine Luna, Keith W. Ross, Keshav Santhanam, Kezhi Kong, Krzysztof Pawelec, and Kumar Anik. NVIDIA nemotron nano 2: An accurate and efficient hybrid mamba-transformer reasoning model. *CoRR*, abs/2508.14444, 2025. doi: 10.48550/ARXIV.2508.14444. URL https://doi.org/10.48550/arXiv.2508.14444.

Hao Chang, Soumya Aggarwal, Anshu Singh, Karthik Narasimhan, and Shikhar Vashishth. Scaling parameter-constrained language models with quality data. *arXiv preprint arXiv:2410.03083*, 2024.

Zhengyu Chen, Siqi Wang, Teng Xiao, Yudong Wang, Shiqi Chen, Xunliang Cai, Junxian He, and Jingang Wang. Revisiting scaling laws for language models: The role of data quality and training strategies. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 23881–23899. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.acl-long.1163/.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948, 2025. doi: 10.48550/ARXIV.2501.12948. URL https://doi.org/10.48550/arXiv.2501.12948.

Aitor Gonzalez-Agirre, Marc Pàmies, Joan Llop, Irene Baucells, Severino Da Dalt, Daniel Tamayo, José Javier Saiz, Ferran Espuña, Jaume Prats, Javier Aula-Blasco, Mario Mina, Adrián Rubio, Alexander Shvets, Anna Sallés, Iñaki Lacunza, Iñigo Pikabea, Jorge Palomar, Júlia Falcão, Lucía Tormo, Luis Vasquez-Reina, Montserrat Marimon, Valle Ruíz-Fernández, and Marta Villegas. Salamandra technical report, 2025. URL https://arxiv.org/abs/2502.08489.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Nikhil Kandpal and Colin Raffel. Position: The most expensive part of an LLM should be its training data. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

Feiyang Kang, Newsha Ardalani, Michael Kuchnik, Youssef Emad, Mostafa Elhoushi, Shubhabrata Sengupta, Shang-Wen Li, Ramya Raghavendra, Ruoxi Jia, and Carole-Jean Wu. Demystifying synthetic data in LLM pre-training: A systematic study of scaling laws, benefits, and pitfalls. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 10739–10758. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.544. URL https://aclanthology.org/2025.emnlp-main.544/.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Pratyush Maini, Vineeth Dorna, Parth Doshi, Aldo Carranza, Fan Pan, Jack Urbanek, Paul Burstein, Alex Fang, Alvin Deng, Amro Abbas, Brett W. Larsen, Cody Blakeney, Charvi Bannur, Christina Baek, Darren Teh, David Schwab, Haakon Mongstad, Haoli Yin, Josh Wills, Kaleigh Mentzer, Luke Merrick, Ricardo Monti, Rishabh Adiga, Siddharth Joshi, Spandan Das, Zhengping Wang, Bogdan Gaza, Ari Morcos, and Matthew L. Leavitt. Beyondweb: Lessons from scaling synthetic data for trillion-scale pretraining. *CoRR*, abs/2508.10975, 2025. doi: 10.48550/ARXIV.2508.10975. URL https://doi.org/10.48550/arXiv.2508.10975.

Nestor Maslej, Loredana Fattorini, Erik Brynjolfsson, John Etchemendy, Katrina Ligett, Terah Lyons, James Manyika, Helen Ngo, Juan Carlos Niebles, Vanessa Parli, Yoav Shoham, Russell Wald, Jack Clark, and Raymond Perrault. The AI index 2024 annual report, 2024. https://aiindex.stanford.edu/report/.

Bettina Messmer, Vinko Sabolčec, and Martin Jaggi. Enhancing multilingual LLM pretraining with model-based data selection. In *Advances in Neural Information Processing Systems: Datasets and Benchmarks Track*, 2025.

Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Nouamane Tazi, Aleksandra Piktus, Sampo Pyysalo, Thomas Wolf, and Colin A. Raffel. Scaling data-constrained language models. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/9d89448b63ce1e2e8dc7af72c984c196-Abstract-Conference.html.

NVIDIA. NVIDIA nemotron 3: Efficient and open intelligence. *CoRR*, abs/2512.20856, 2025. doi: 10.48550/ARXIV.2512.20856. URL https://doi.org/10.48550/arXiv.2512.20856.

OpenAI. Openai o1 system card, 2024. URL https://openai.com/index/openai-o1-system-card/. Updated December 5, 2024; accessed 2026-01-28.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Thomas Wolf, Lewis Tunstall, and Leandro von Werra. The FineWeb datasets: Decanting the web for the finest text data at scale. *arXiv preprint arXiv:2406.17557*, 2024.

Guilherme Penedo, Hynek Kydlíček, Vinko Sabolčec, Bettina Messmer, Negar Foroutan, Amir Hossein Kargaran, Colin Raffel, Martin Jaggi, Leandro Von Werra, and Thomas Wolf. Fineweb2: One pipeline to scale them all — adapting pre-training data processing to every language. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=jnRBe6zatP.

Pleias. SYNTH: The new data frontier. Hugging Face Dataset and Technical Blog, 2025. https://huggingface.co/datasets/PleIAs/SYNTH.

Zeyu Qin, Qingxiu Dong, Xingxing Zhang, Li Dong, Xiaolong Huang, Ziyi Yang, Mahmoud Khademi, Dongdong Zhang, Hany Hassan Awadalla, Yi R. Fung, Weizhu Chen, Minhao Cheng, and Furu Wei. Scaling laws of synthetic data for language model. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=UmUXPXHtdl.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y.K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, and Ross Anderson. The curse of recursion: Training on generated data makes models forget. *arXiv preprint arXiv:2305.17493*, 2023.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Raghavi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Harsh Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Pete Walsh, Luke Zettlemoyer, Noah A. Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. Dolma: an open corpus of three trillion tokens for language model pretraining research. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 15725–15788. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.840. URL https://doi.org/10.18653/v1/2024.acl-long.840.

Stanford CRFM. Marin: An open foundation model for the future of AI. Stanford CRFM, 2025. https://marin.community.

Dan Su, Kezhi Kong, Ying Lin, Joseph Jennings, Brandon Norick, Markus Kliegl, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Nemotron-cc: Transforming common crawl into a refined long-horizon pretraining dataset. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 2459–2475. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.acl-long.123/.

Sai Subramanyam, Hamish Ivison, Lucy Wang, Mark Yatskar, and Hannaneh Hajishirzi. Scaling laws revisited: Modeling the role of data quality in language model pretraining. *arXiv preprint arXiv:2510.03313*, 2025.

Llama Team. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024. doi: 10.48550/ARXIV.2407.21783. URL https://doi.org/10.48550/arXiv.2407.21783.

Team OLMo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, et al. OLMo 3. *arXiv preprint arXiv:2512.13961*, 2025.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023a. doi: 10.48550/ARXIV.2302.13971. URL https://doi.org/10.48550/arXiv.2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *CoRR*,

abs/2307.09288, 2023b. doi: 10.48550/ARXIV.2307.09288. URL https://doi.org/10.48550/arXiv.2307.09288.

Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. Position: Will we run out of data? limits of LLM scaling based on human-generated data. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=ViZcgDQjyG.

Zhangchen Xu, Adriana Meza Soria, Shawn Tan, Anurag Roy, Ashish Sunil Agrawal, Radha Poovendran, and Rameswar Panda. TOUCAN: synthesizing 1.5m tool-agentic data from real-world MCP environments. *CoRR*, abs/2510.01179, 2025. doi: 10.48550/ARXIV.2510.01179. URL https://doi.org/10.48550/arXiv.2510.01179.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jian Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *CoRR*, abs/2505.09388, 2025. doi: 10.48550/ARXIV.2505.09388. URL https://doi.org/10.48550/arXiv.2505.09388.