treepai

TREEPAI, INC. · RESEARCH REPORT

# Treepai g1: A 518M-Parameter Language Model Trained From Scratch on 10 Billion Tokens With One GPU

Jose Franco

Treepai, Inc.

October 2026 · research@treepai.com

ABSTRACT

We describe Treepai g1, a 517.8M-parameter decoder-only transformer trained from scratch on 10 billion tokens (93% filtered educational web text, 7% filtered public-domain fiction) using a single consumer GPU for 87 hours. It carries the training recipe of our first model, Treepai g0 [1], to a model 4.7 times larger with four times the data, a doubled context length and sampling without replacement. Under one evaluation harness applied to every model, g1 averages 49.6% across five core zero-shot benchmarks, up from 38.9% for g0. Among open models of similar size it is ahead of GPT-2 medium (44.9%) and Pythia-410M (47.0%, trained on 30 times as many tokens), and behind the two strongest, Qwen2.5-0.5B (54.4%) and SmolLM2-360M (58.3%), which were trained with hundreds of times more data. Against a Pythia-410M checkpoint at a similar 10.5B tokens it leads by 12.0 points. On held-out text it predicts educational web text better than both older models, but Wikipedia text less well than every finished model of its size: the training text is narrow. In a same-engine serving test it is the slowest of the four models compared, and 8-bit quantization costs nothing measurable. Fine-tuned for a first task, sorting customer-support messages, it is level with Qwen2.5-0.5B fine-tuned the same way (95.2% and 95.6% of categories right) and far ahead of open instruction-tuned models of its size given the task by prompt alone (42.5% at best). The fiction slice was added to fix g0's weakest area, story-level reading, and LAMBADA accuracy rises from 21.8% to 42.3%. The first attempt at this run became unstable after about 1.4B tokens: the weight matrices trained with the Muon optimizer had no weight decay and grew steadily until the loss rose. Adding a small weight decay to those matrices removed the problem, and the run completed with a validation loss of 2.636.

## 1 Introduction

Treepai studies how to reach a given level of language-model capability with as little compute as possible, both when training a model and when serving it. Our first model, Treepai g0 [1], was a 110M-parameter baseline that reached the level of GPT-2 small in under five GPU-hours and identified a better training recipe. Its report closed with three plans: carry the improved recipe to models of 0.5B parameters and above, sample training text without replacement, and address its weakest area, story-level reading.

Treepai g1 is that next step. It asks a narrow question: does the recipe found at 110M parameters and 2.5B tokens hold up at 518M parameters and 10B tokens, still on one consumer GPU, and how close does the result come to open models of similar size that were trained with far more data and compute?

We compare g1 with open models of similar size, 355M to 494M parameters, all rescored with our own harness. GPT-2 medium [2] (355M) and Pythia-410M [3] (405M, 300B tokens) are the established baselines at this size, and Pythia's published intermediate checkpoints allow a comparison at a similar number of training tokens. SmolLM2-360M [4] (362M, 4T tokens) and Qwen2.5-0.5B [5] (494M) are the strongest current models. For reference only, the tables also list two smaller models: g0, and SmolLM2-135M, which the g0 report named as its long-term target.

Our contributions are:

- A 518M-parameter model trained from scratch in 87 GPU-hours on one consumer GPU that scores above GPT-2 medium and Pythia-410M, the established open models of its size, on nine zero-shot benchmarks, and 12.0 points above Pythia-410M at a similar number of training tokens.
- A documented training instability that appears only in long runs: under the Muon optimizer without weight decay, the hidden weight matrices grow at a steady rate until training diverges. A small weight decay on those matrices fixes it.
- Evidence that a 7% slice of filtered public-domain fiction closes most of the story-reading gap left by educational web text, measured first in a controlled 110M test and then in the full model.
- Decontamination of the training text against all nine benchmarks before training, which the g0 run did for only five.
- A first fine-tuned task, sorting customer-support messages, on which g1 matches a fine-tuned Qwen2.5-0.5B although it trails that model on every general measure (Section 9).

## 2 What changed from g0

**Table 1.** Treepai g0 (improved recipe) and Treepai g1.

| | Treepai g0 | Treepai g1 |
|---|---|---|
| Parameters | 110.1M | 517.8M |
| Layers × width | 12 × 768 | 24 × 1,280 |
| Context length | 1,024 tokens | 2,048 tokens |
| Training tokens | 2.47B | 10.0B |
| Training text | FineWeb-Edu | 93% FineWeb-Edu, 7% filtered fiction |
| Sampling | random windows, with replacement (63% of the text seen) | every window once, shuffled |
| Benchmarks checked for overlap before training | 5 of 9 | 9 of 9 |
| Weight decay on Muon matrices | none | 0.02 (from step 5,000) |
| Training time, one RTX 4090 | 4.81 GPU-hours | 87 GPU-hours |
| Core benchmark average | 38.9% | 49.6% |

The tokenizer is unchanged (32,768 entries). We repeated the g0 report's compute-per-text comparison at the new model size: a 49K vocabulary would cost 1.1% more compute per unit of text and a 65K vocabulary 3.4% more, so 32K remains the cheapest.

## 3 Model

Treepai g1 keeps g0's architecture: a LLaMA-style [6] decoder-only transformer with pre-normalization (RMSNorm [7]), rotary position embeddings [8] (base 10,000), SwiGLU feed-forward layers [9], no biases, and an output layer tied to the input embedding. Neither architectural change tested in the g0 ablations (query-key normalization, grouped-query attention) is used. Table 2 gives the configuration.

**Table 2.** Model configuration.

| Hyperparameter | Value |
|---|---|
| Parameters (embedding tied) | 517.8M |
| of which embedding | 41.9M |
| Layers | 24 |
| Model width $d_{model}$ | 1,280 |
| Attention heads (head size) | 20 (64) |
| Feed-forward width (SwiGLU) | 3,456 |
| Context length | 2,048 tokens |
| Vocabulary | 32,768 |

# 4 Data

## 4.1 Educational web text

The bulk of the training text, 9.3B tokens, comes from the 100B-token sample of FineWeb-Edu[10,11], a web corpus whose documents carry a 0–5 educational-quality score from a trained classifier. As for g0 we kept only documents scoring at least 3.0, a stricter threshold than the dataset's default, and removed the 10,000 documents held out for validation and for the sealed test set. FineWeb-Edu is released under the ODC-By 1.0 license and is subject to Common Crawl's terms of use.

## 4.2 Fiction

The g0 report attributed its weakest result, LAMBADA, to the near-absence of fiction in FineWeb-Edu, and listed that as an untested hypothesis. We tested it before the g1 run with four 1B-token runs of the 110M model, changing only the training data: educational text twice with different seeds, to measure noise, and two mixes replacing 7% of it with public-domain fiction. One mix used fiction filtered for present-day language (published in 1900 or later, at most five archaic-grammar markers per 10,000 words, spelling modernized); the other used unfiltered older fiction. The rule, set in advance, required a gain on LAMBADA beyond twice the noise, no loss on science questions beyond the noise, and a rise in validation loss on educational text of at most 0.008.

**Table 4.** Fiction test at 110M parameters and 1B tokens. Losses are on held-out text (lower is better); mean of each run's last three checkpoints.

| Training data | Loss, educational | Loss, fiction | LAMBADA acc. | Verdict |
|---|---|---|---|---|
| FineWeb-Edu (control) | 3.288 | 4.073 | 19.1% | control |
| FineWeb-Edu, second seed | 3.284 | 4.061 | 18.5% | noise floor |
| 93% FineWeb-Edu + 7% filtered fiction | 3.299 | **3.522** | **23.7%** | rule not met |
| 93% FineWeb-Edu + 7% unfiltered older fiction | 3.299 | 3.603 | 21.4% | rule not met |

Filtered fiction raised LAMBADA accuracy by 4.6 points and left science unchanged, but it raised the loss on educational text by 0.011, above the 0.008 we had allowed. By the rule as written, neither mix passed. We included the filtered fiction slice in g1 anyway: the allowance had been an estimate, the measured cost is about 0.3% of the loss, and a general model needs to read narrative. We record this as a decision that accepts a measured trade-off, not as a passed test. Filtering mattered: the filtered mix beat the unfiltered one on both LAMBADA and held-out fiction at the same cost.

For g1 the slice is 0.7B tokens, 7% of the training text, passed through the same filter: English fiction from Project Gutenberg that is in the public domain in the United States, and the CC0 editions of Standard Ebooks. Stories first published from 1930 to 1964 were taken first, then Standard Ebooks, then other books from 1900 onward.

### 4.3 Decontamination

Following GPT-3 [12], we removed every training document that shares any 13-word sequence (lowercased) with a benchmark item. Unlike g0, all nine benchmarks were indexed before training, so the scores in Section 6 need no correction for overlap. Of 9,048,839 educational documents read, 6,646 (0.07%) were removed (Table 3); most matches are long public-domain passages that MMLU quotes. Thirty-three fiction books were removed for the same reason. The method cannot check items shorter than 13 words.

**Table 3.** Decontamination of the educational text against all nine benchmarks, before training. A document is counted under the first benchmark it matched.

| Benchmark (split) | Items | Training documents removed |
|---|---:|---:|
| HellaSwag (validation) | 10,042 | 214 |
| ARC-Easy (test) | 2,376 | 221 |
| ARC-Challenge (test) | 1,172 | 28 |
| PIQA (dev) | 1,838 | 23 |
| LAMBADA (test) | 5,153 | 5 |
| WinoGrande (validation) | 1,267 | 0 |
| OpenBookQA (test) | 500 | 0 |
| SciQ (test) | 1,000 | 118 |
| MMLU (test) | 14,042 | 6,037 |
| Total | 37,390 | 6,646 |

## 5 Training

### 5.1 Recipe

We used the improved recipe from the g0 report: the Muon optimizer [13,14] on the hidden weight matrices, a learning rate held constant after warmup and then decayed linearly to zero over the last 25% of steps [15,16], and 262K-token batches. The training text was cut into 4.88M non-overlapping 2,048-token windows, shuffled once, and each window was used exactly once. At 19 tokens per parameter the run sits at the compute-optimal ratio of about 20 [17]. Total training compute is about $3.1 \times 10^{19}$ FLOPs by the $6ND$ approximation.

**Table 5.** Training configuration.

| Setting | Value |
|---|---|
| Optimizer, hidden matrices | Muon, learning rate 0.02, momentum 0.95 (Nesterov) |
| Weight decay, hidden matrices | none to step 5,000; 0.02 from step 5,000 (Section 5.2) |
| Optimizer, embeddings and norms | AdamW, $\beta = (0.9, 0.95)$, learning rate $6 \times 10^{-4}$, weight decay 0.1 on the embedding |
| Schedule | 200-step warmup, constant, linear decay to zero over the last 25% |
| Gradient clipping | 1.0 (global norm) |
| Batch size | 262,144 tokens (128 × 2,048) |
| Micro-batch × accumulation | 4 × 32 |
| Steps | 38,147 (10.0B tokens) |
| Batch sampling | shuffled windows, without replacement |
| Precision | bf16 autocast, fp32 master weights |
| Hardware | 1 × NVIDIA RTX 4090 (24 GB) |
| Throughput | ~32,000 tokens/s |
| Wall-clock time | 87 GPU-hours |

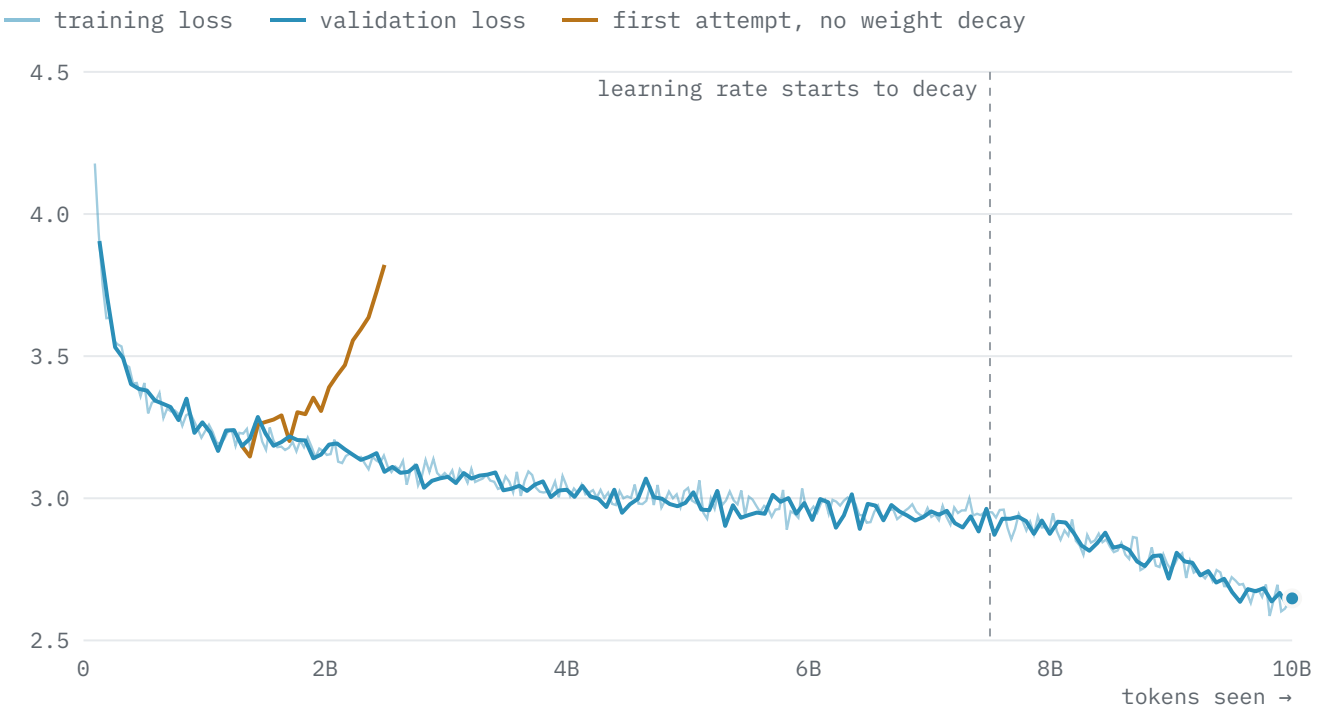



**Figure 1.** Loss over training. The first attempt separates from the final run at 1.3B tokens and its loss rises until it was stopped. The final run improves slowly while the learning rate is constant and most quickly during the decay over the last 25% of steps. It ends at 2.646 (training, mean of the last 500 steps) and 2.636 (full validation set).

## 5.2 An instability that appears only in long runs

The first attempt used the g0 recipe unchanged, with no weight decay on the Muon-trained matrices. It improved normally to a validation loss of 3.147 at step 5,250. From about step 5,500 the loss rose steadily, to 3.82 at step 9,500, and the gradient norm climbed from a steady 0.16 to about 8.2 (Figure 2). Nothing failed outright, so the run did not stop itself; we stopped it at step 9,560.

Measuring the saved checkpoints showed the cause. Between steps 5,000 and 9,000 the hidden weight matrices grew by 35–43% in norm, and the largest singular values of the last attention output layers rose from 15–17 to 24–29. The squared norm of each matrix grew by a near-constant amount per step both before and after the loss turned, so the growth came first and the rising loss followed. The embedding, trained with AdamW and weight decay, grew by only 11% over the same steps. Muon's update has a fixed size per step regardless of the gradient, and with nothing to shrink the matrices their norm increases without limit. The g0 runs used the same recipe on a model a fifth the size, for at most 9,430 steps, and did not show the problem.

We restarted from the step-5,000 checkpoint with a weight decay of 0.02 on the Muon matrices, chosen so that at the measured growth rate their norms settle at about half their step-5,000 size, and changed nothing else. The test, set before the restart, was that the run must pass steps 5,250–7,500 with a falling loss and a flat gradient norm. It did: at step 9,500, where the first attempt read 3.82, the restarted run read 3.09, and its gradient norm stayed near 0.21 and ended at 0.13. The run continued to completion without further intervention.

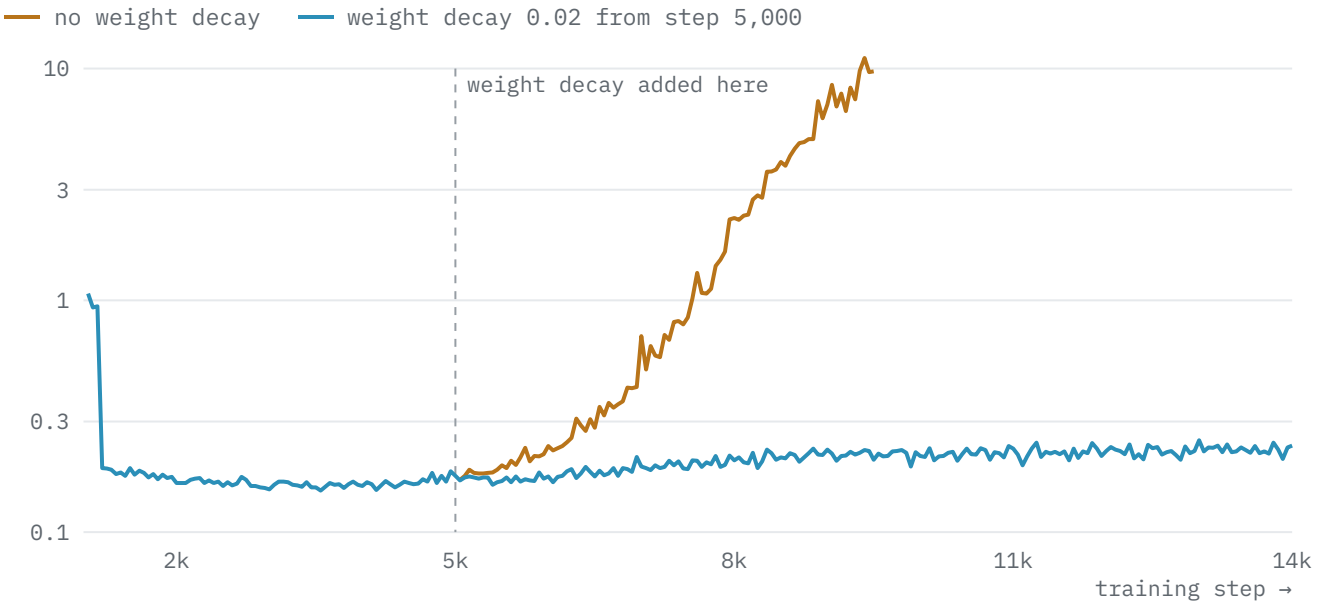



**Figure 2.** Gradient norm (median over 50 steps, logarithmic scale). Without weight decay it rises from 0.16 to about 8.2 in 4,500 steps. With it, it stays flat.

The final model is therefore not the product of a single unchanged recipe: its first 5,000 steps (13% of training) had no weight decay on those matrices. We have not rerun it from the start with weight decay throughout.

**Energy.** We did not meter power. Using the RTX 4090's 450 W board-power limit as an upper bound, the run used at most about 39 kWh of GPU energy. Counting the 10 GPU-hours of the abandoned part of the first attempt, g1 took 97 GPU-hours in all. Evaluation compute and host and cooling overheads are not included.

# 6 Evaluation

## 6.1 Setup

The method is the g0 report's, unchanged. We score all models zero-shot by log-likelihood, with prompt formats from the lm-evaluation-harness [18], and every reference model is rescored with our harness rather than copied from papers. The benchmarks are HellaSwag [19], ARC-Easy and ARC-Challenge [20], PIQA [21], LAMBADA [22], WinoGrande [23], OpenBookQA [24], SciQ [25] and MMLU [26] in cloze format. The benchmark list, the harness and the comparison models were fixed before g1 was trained, and g1 was scored once, on its final checkpoint. 95% margins from test-set size are about ±1.0 (HellaSwag), ±2.0 (ARC-E), ±2.9 (ARC-C), ±2.3 (PIQA), ±1.2 (LAMBADA), ±2.8 (WinoGrande), ±4.4 (OpenBookQA), ±3.1 (SciQ) and ±0.8 (MMLU) points.

## 6.2 Core benchmarks

**Table 6.** Zero-shot accuracy (%) on the five core benchmarks, every model scored with our harness. HellaSwag and ARC-C use length-normalized accuracy. Models near g1's size (355M to 494M parameters) are ordered by Avg (5). † GPT-2's training tokens are unpublished. ‡ Size of the pretraining corpus reported for the Qwen2.5 family.

| Model | Params | Tokens | HellaSwag | ARC-E | ARC-C | PIQA | LAMBADA | Avg (5) |
|---|---|---|---|---|---|---|---|---|
| Treepai g1 | 518M | 10B | 42.3 | 65.0 | 31.3 | 67.3 | 42.3 | 49.6 |
| Pythia-410M @ 10.5B | 405M | 10.5B | 29.7 | 43.4 | 22.4 | 61.6 | 31.0 | 37.6 |
| GPT-2 medium | 355M | † | 39.4 | 49.1 | 25.2 | 67.6 | 43.0 | 44.9 |
| Pythia-410M | 405M | 300B | 40.5 | 51.9 | 24.4 | 66.7 | 51.6 | 47.0 |
| Qwen2.5-0.5B | 494M | 18T‡ | 52.2 | 64.6 | 32.3 | 70.2 | 52.5 | 54.4 |
| SmolLM2-360M | 362M | 4T | 56.4 | 70.4 | 38.7 | 72.1 | 53.8 | 58.3 |
| *Smaller models, for reference* | | | | | | | | |
| Treepai g0, improved recipe | 110M | 2.47B | 30.8 | 54.5 | 26.6 | 60.7 | 21.8 | 38.9 |
| SmolLM2-135M | 135M | 2T | 43.2 | 64.4 | 29.4 | 68.4 | 42.9 | 49.7 |
| Random guessing | – | – | 25 | 25 | 25 | 50 | 0 | – |

Treepai g1 averages 49.6 on the core benchmarks, 10.7 points above g0.

**Against the established models of its size.** g1 is ahead of Pythia-410M by 2.6 points and of GPT-2 medium by 4.7. The lead is in science: 13.1 points over Pythia on ARC-Easy and 6.9 on ARC-Challenge. HellaSwag is 1.8 points ahead, PIQA is level, and Pythia remains 9.3 points ahead on LAMBADA.

**At a similar token count.** Pythia-410M's published checkpoint after 5,000 steps has seen 10.5B tokens. It averages 37.6 against g1's 49.6, and g1 leads on every task. As in the g0 report this is not a like-for-like test: the training text differs, and that checkpoint is an early snapshot of a 300B-token run, taken before its learning-rate schedule had finished, whereas g1's had. Part of the gap is schedule rather than data.

**Against the strongest models of its size.** Qwen2.5-0.5B (54.4) and SmolLM2-360M (58.3) are ahead by 4.8 and 8.7 points. Against Qwen, g1 is level on the two science tasks (65.0 vs 64.6 and 31.3 vs 32.3) and 9.9 and 10.2 points behind on HellaSwag and LAMBADA. SmolLM2-360M leads on every task with fewer parameters than g1. Both were trained with hundreds of times more data, and broader data. Closing this gap is the open problem for our next models.

**Smaller models.** The g0 report named SmolLM2-135M as its long-term target. g1 now reaches its scores (49.6 against 49.7, within the margin on every core task), but with 3.8 times the parameters. That is not a comparison of equals: SmolLM2-135M gets the same result from a model about a quarter the size, and so costs far less to run. The target stands until a Treepai model of that size reaches it.

**Story-level reading.** LAMBADA rises from 21.8% for g0 to 42.3%, level with GPT-2 medium (43.0%). The 110M test of Section 4.2 attributes 4.6 points of such a gain to the fiction slice at that size; the full model changed size, data volume and context at once, so we cannot say how much of the 20-point rise each contributed. Models trained on broader text remain about ten points ahead.

## 6.3 Further benchmarks

**Table 7.** Zero-shot accuracy (%) on four further benchmarks. MMLU* is the cloze format (answer text scored, not a letter). Avg (9) averages all nine capability benchmarks.

| Model | Params | Tokens | WinoG | OBQA | SciQ | MMLU* | Avg (9) |
|---|---|---|---|---|---|---|---|
| Treepai g1 | 518M | 10B | 54.2 | 36.4 | 84.8 | 30.4 | 50.4 |
| Pythia-410M @ 10.5B | 405M | 10.5B | 49.9 | 25.8 | 72.8 | 26.2 | 40.3 |
| GPT-2 medium | 355M | † | 53.1 | 29.4 | 77.4 | 27.6 | 45.8 |
| Pythia-410M | 405M | 300B | 53.3 | 29.4 | 81.5 | 28.9 | 47.6 |
| Qwen2.5-0.5B | 494M | 18T‡ | 56.4 | 35.2 | 93.1 | 32.7 | 54.4 |
| SmolLM2-360M | 362M | 4T | 59.0 | 37.4 | 91.2 | 35.0 | 57.1 |
| *Smaller models, for reference* | | | | | | | |
| Treepai g0, improved recipe | 110M | 2.47B | 49.6 | 30.6 | 77.5 | 28.1 | 42.2 |
| SmolLM2-135M | 135M | 2T | 53.0 | 33.2 | 84.0 | 31.2 | 50.0 |
| Random guessing | – | – | 50 | 25 | 25 | 25 | – |

Across all nine capability benchmarks g1 averages 50.4, against 45.8 for GPT-2 medium, 47.6 for Pythia-410M, 54.4 for Qwen2.5-0.5B and 57.1 for SmolLM2-360M. On OpenBookQA g1 (36.4) is level with SmolLM2-360M (37.4) and Qwen2.5-0.5B (35.2), inside a wide margin. WinoGrande is above chance for the first time in our models (54.2, margin ±2.8).

## 6.4 Predicting held-out text

As in the g0 report we also measure bits per byte on held-out text, which compares models with different tokenizers fairly.

**Table 8.** Bits per byte on held-out text (lower is better). Every model reads the same documents in 1,024-token windows. FineWeb-Edu test: the first 2,000 of our sealed test documents, in-domain for Treepai and possibly seen in training by the SmolLM2 models, whose data includes FineWeb-Edu. WikiText-103 test: Wikipedia articles.

| Model | Bytes / token | FineWeb-Edu test | WikiText-103 |
|---|---|---|---|
| Treepai g1 | 4.62 | 0.869 | 0.990 |
| Pythia-410M @ 10.5B | 4.62 | 1.067 | 1.120 |
| GPT-2 medium | 4.60 | 0.972 | 0.972 |
| Pythia-410M | 4.62 | 0.921 | 0.929 |
| Qwen2.5-0.5B | 4.61 | 0.855 | 0.896 |
| SmolLM2-360M | 4.53 | 0.814 | 0.868 |
| Treepai g0, improved recipe | 4.62 | 0.983 | 1.174 |
| SmolLM2-135M | 4.53 | 0.885 | 0.971 |

On the sealed FineWeb-Edu documents g1 needs 0.869 bits per byte, down from 0.983 for g0 and fewer than GPT-2 medium (0.972) and Pythia-410M (0.921). Qwen2.5-0.5B (0.855) and SmolLM2-360M (0.814) need fewer; SmolLM2's training data includes FineWeb-Edu and may contain these documents. Between the two Treepai models, which share a tokenizer, the sealed-test loss falls from 3.150 to 2.784 per token.

On WikiText-103, which is outside g1's training distribution, the order changes. g1 (0.990) is behind every finished model of its size: GPT-2 medium (0.972), Pythia-410M (0.929), Qwen2.5-0.5B (0.896) and SmolLM2-360M (0.868). It is ahead only of the Pythia-410M checkpoint at a similar token count (1.120). This is the pattern of the g0 report at a larger scale: strong on the kind of text it was trained on and weaker outside it. Ten billion tokens of mostly educational text bought depth, not breadth, and the benchmark averages of Section 6.2, which lean on science questions, flatter the model in this respect.

## 6.5 Learning curve

We scored the model snapshots saved every 5,000 steps on the five core benchmarks (Figure 3). The core average is 37.6 after 1.3B tokens, 43.9 after 7.9B, just after the learning rate begins to decay, and 49.6 at the end. The 6.5B tokens at a constant learning rate added 6.2 points; the last 2.1B, as the rate fell to zero, added 5.8. Validation loss shows the same shape (Figure 1). A decay-to-zero schedule holds much of its gain back until the end, so a run of this kind cannot be judged, or stopped early, on its mid-run scores. LAMBADA moved least early and most late: 23.0% at 1.3B tokens, 26.7% at 5.2B and 42.3% at the end.

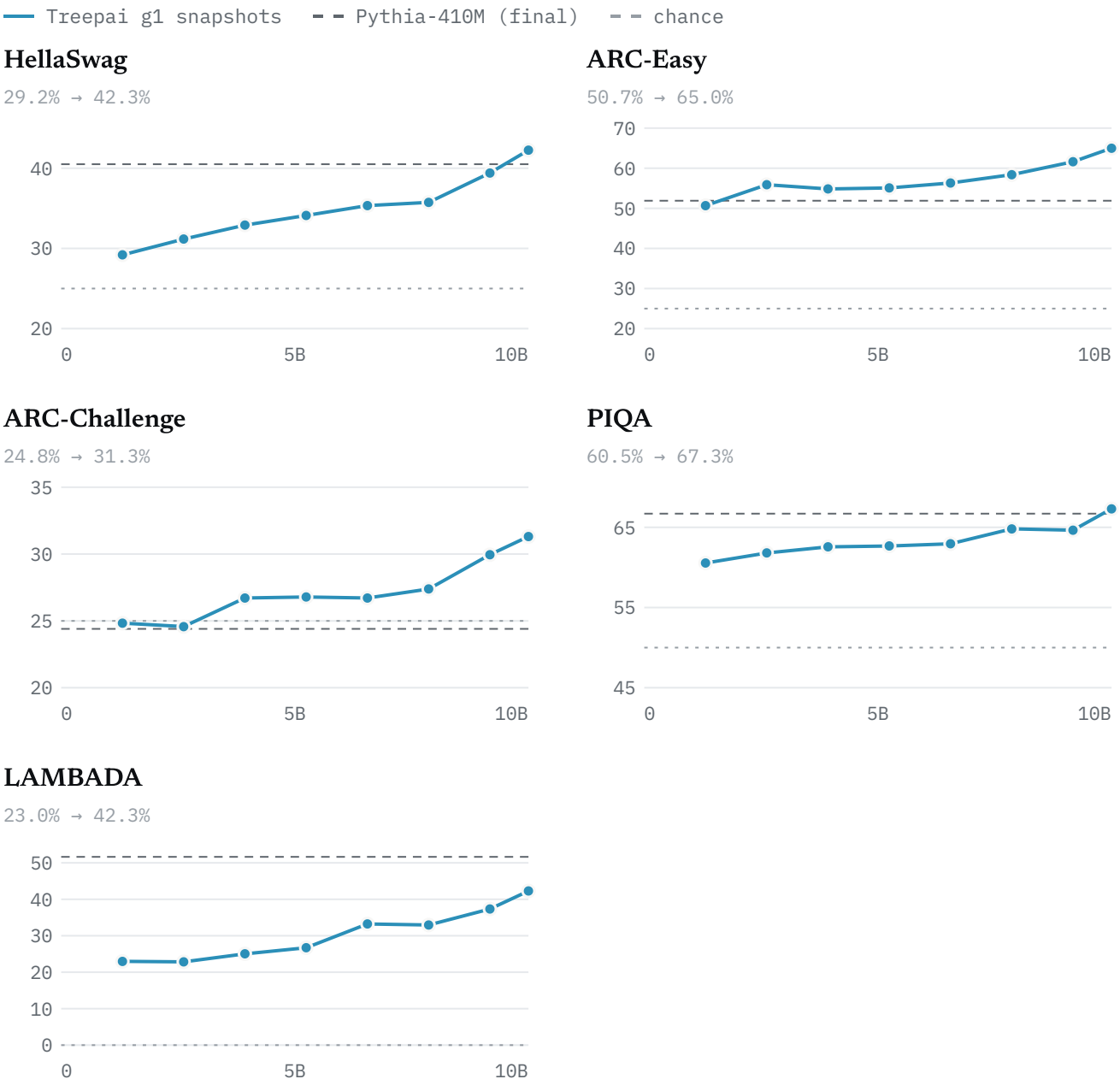



**Figure 3.** Zero-shot accuracy at seven snapshots and the final model, by tokens seen. Each panel has its own vertical scale. The dashed line is Pythia-410M's final score.

## 6.6 Qualitative behavior

At every validation step the model completed the fixed prompt *"The most important thing about"* (temperature 0.8, top-k 50). The samples below are unedited; Appendix B lists more. The final model writes fluent, coherent expository prose over several sentences. It is a base model with no instruction tuning: it continues text and does not follow instructions or answer questions reliably, and it states false facts with confidence.

| | |
|---|---|
| **66M tokens**<br>step 250 · val 4.59 | The most important thing about any kind of matter in the universe. But if it could have been a perfect idea for more than a decade, but it could have been impossible to prove in the idea of matter, in fact, |
| **0.52B tokens**<br>step 2,000 · val 3.38 | The most important thing about this article is that you can actually understand and interpret the word ‘a’ in English. If you have a good bit of English skills at your disposal, and you’re not worried about the quality of |
| **2.62B tokens**<br>step 10,000 · val 3.09 | The most important thing about these facts is that they are not true. According to the information gathered by the research, when people learn about the facts, they are better able to make informed decisions. |
| **6.55B tokens**<br>step 25,000 · val 2.97 | The most important thing about the course is that it gives a real-world perspective and that, as the course begins, you can start to get into small things. The main thing that you should begin with is the simple form of |
| **10.00B tokens**<br>step 38,147 · val 2.65 | The most important thing about what you learn in A-level Biology is that you will explore the subject matter from a variety of perspectives, as well as having practical skills and practical application. |

## 6.7 Social bias

CrowS-Pairs [27] asks which of two minimally different sentences a model finds more likely, one of which expresses a stereotype. The dataset has documented validity problems [28], so we treat it as a coarse probe.

**Table 10.** CrowS-Pairs (English, 1,508 pairs): share of pairs where the model finds the more stereotypical sentence more likely. 50 means no preference; higher means stronger stereotyping. 95% margin ±2.5.

| Model | Prefers stereotype (%) |
|---|---|
| Treepai g1 | 59.2 |
| Pythia-410M @ 10.5B | 55.4 |
| GPT-2 medium | 58.3 |
| Pythia-410M | 57.8 |
| Qwen2.5-0.5B | 57.8 |
| SmolLM2-360M | 63.1 |
| Treepai g0, improved recipe | 54.2 |
| SmolLM2-135M | 58.8 |

g1 prefers the more stereotypical sentence in 59.2% of pairs, against 54.2% for g0. The five-point rise is outside the ±2.5 margin: the larger model is measurably more stereotyped on this probe than the smaller one. It is level with Pythia-410M (57.8), GPT-2 medium (58.3) and Qwen2.5-0.5B (57.8), and below SmolLM2-360M (63.1). Between our own two models the score rose with model quality. The reference models show no consistent pattern: Qwen2.5-0.5B, one of the two strongest, scores lower than SmolLM2-360M by five points. We have run no broader safety evaluation.

## 7 Training efficiency in context

Table 6 can be read by training budget. Among models of its size, g1 with 10B tokens scores above Pythia-410M, trained on 30 times as many, and above GPT-2 medium, whose token count is unpublished. The two that beat it were trained on 400 times as many tokens (SmolLM2-360M) and on a corpus up to 1,800 times as large (Qwen2.5). We draw one conclusion from this and no more: on these benchmarks a selected corpus and a tuned recipe put a small training budget ahead of far larger budgets spent on older recipes, and well short of the largest budgets spent on recent ones.

The g0 report found its improved recipe at 110M parameters and 2.5B tokens. At 518M and 10B it transferred, with the one correction of Section 5.2. We ran no ablations at the new size, so we cannot say whether each of the three changes still helps on its own, or whether the learning rates chosen at 110M are still the best ones.

## 8 Serving

We repeated the g0 report's same-engine comparison at this size. g1 and three of the reference models were converted to GGUF at 8-bit precision and run in llama.cpp [29] on the same consumer Apple-silicon hardware used for g0. Pythia-410M is left out because that engine does not support its architecture.

**Table 11.** Generation speed in llama.cpp, 8-bit (q8_0), on consumer Apple-silicon hardware. Characters per second for a 128-token prompt and 128-token reply, by number of concurrent requests; median of three runs.

| Model | Params | File size | Chars / token | 1 request | 8 requests | 32 requests |
|---|---|---|---|---|---|---|
| Treepai g1 | 518M | 552 MB | 4.59 | 710 | 2,178 | 6,141 |
| GPT-2 medium | 355M | 383 MB | 4.57 | 867 | 2,634 | 7,230 |
| Qwen2.5-0.5B | 494M | 531 MB | 4.58 | 735 | 2,415 | 7,904 |
| SmolLM2-360M | 362M | 386 MB | 4.50 | 834 | 1,812 | 8,247 |

g1 is the slowest of the four at one request and at thirty-two. Against Qwen2.5-0.5B, the model nearest its size, it is 3% slower at one request, 10% at eight and 22% at thirty-two. The speed advantage reported for g0 does not carry over. At 110M parameters the small vocabulary removed a meaningful share of each token's compute; at 518M the output layer is a small part of the total, and g1 is simply

the largest model in the comparison. It does about as much arithmetic per token as Qwen2.5-0.5B and more than the two smaller models, and nothing in its design makes up for that. Read together with Section 6, g1 at this size is neither the most capable model of its class nor the cheapest to run.

Quantization remains free. Perplexity on the same 40 × 1,024-token chunks of held-out text is 13.89–13.92 in PyTorch at 32-bit precision (depending on how the first token of each chunk is treated), 13.91 in 16-bit GGUF and 13.91 at 8 bits, in a 552 MB file. These measurements are on consumer hardware with small batches. Section 9 reports throughput under load on a datacenter-class GPU for the fine-tuned model.

## 9 A first fine-tuned task: sorting support messages

The comparisons so far are of base models. A base model of this size is not used as it is; it is fine-tuned for one job. We therefore fine-tuned g1 for a first job and asked whether its deficit on general text carries into the task. The job is sorting customer-support messages: for each message the model writes one short line that either names one of 13 categories and the specific request, or hands the message to a person because it is outside what a support desk handles or asks for a person.

**Data.** The training set is 53,027 messages. About 50,000 are human-written and come from openly licensed intent and dialogue datasets: Banking77 [30], CLINC150 [31], NLU++ [32], ABCD [33], MultiDoGO [34], a set of IT support tickets [35] and three intents of the SNIPS benchmark, with their 461 source labels mapped by hand to the 13 categories or to the hand-off. The first version caught under a third of off-topic messages, because few training messages showed what a hand-off looks like. We added 2,462 messages written by Olmo-3-7B-Instruct, an openly released model: hand-off messages of four kinds (asks for a person, abuse or threats, legal, safety) and in-scope messages that mention a person or a branch without asking to be passed on. The generator saw no real message, and no text from a closed model is in the data.

**Method.** g1 and Qwen2.5-0.5B were fine-tuned on the same data with the same prompt, optimizer, schedule and scoring, one run each, about ten minutes on one RTX 4090. Four pass marks were written down before training: written hand-off messages caught, off-topic messages caught, in-scope messages handed off by mistake (at most 2% on every test) and categories right. As a second comparison, three open instruction-tuned models (Qwen2.5-0.5B-Instruct, Qwen3-0.6B and OLMo-2-1B-Instruct, with SmolLM2-360M-Instruct as a smaller reference) were given the task by prompt alone, with no examples and with 36 examples, their output constrained to the two allowed answer shapes.

**Table 12.** Support-message sorter: g1 and Qwen2.5-0.5B fine-tuned on the same data, one run each. Counts of test messages in brackets. 95% margins from test-set size are about ±0.4 points on the main test, ±2.5 on off-topic queries, ±1.4 on voice-assistant commands and ±1.6 to ±1.7 on the two tests of unseen material.

| Measure | Treepai g1, fine-tuned | Qwen2.5-0.5B, fine-tuned |
|---|---|---|
| Category right, main test (11,955) | 95.2% | 95.6% |
| Category right, 13 kinds of request never trained on (2,286) | 78.4% | 74.6% |
| Category right, a dataset never trained on (3,097) | 72.6% | 71.3% |
| Off-topic queries handed off, CLINC150 out-of-scope test (1,010) | 84.1% | 83.6% |
| Voice-assistant commands handed off, MASSIVE, never trained on (1,889) | 89.0% | 88.8% |
| Written hand-off messages caught, six industries never trained on (706) | 99.9% | 99.9% |
| In scope, handed off by mistake: main test | 0.5% | 0.4% |
| In scope, handed off by mistake: kinds of request never trained on | 0.4% | 1.2% |
| In scope, handed off by mistake: dataset never trained on | 1.9% | 2.0% |

**Against the same model class, fine-tuned.** The two models are level. Every difference in Table 12 is inside or near its margin except the kinds of request never trained on, where g1 is ahead on both measures; with one run per model we do not read that as a ranking. g1 met all four pass marks. Its deficit on general text (Section 6.4) did not carry into the fine-tuned task. The dataset never trained on is banking conversations from DSTC11 Track 2, and MASSIVE[36] is used for testing only.

**Against open models used by prompt.** An earlier version of the g1 sorter, scoring 95.4% on the main test, was ahead of all eight prompted runs on every category and hand-off-caught measure, by 12.4 points or more against margins of at most 5.8. The best prompted run got 42.5% of main-test categories right (OLMo-2-1B-Instruct with examples); always answering the most common category gets 31.6%. Giving examples raised the hand-offs caught and, for the two Qwen models, handed 25 to 53% of in-scope messages to a person. This lead is the value of fine-tuning on the label scheme, not of g1's base model: Qwen2.5-0.5B fine-tuned the same way is level with g1.

**Speed.** Measured on the first version of the sorter, on one RTX 4090. When only the category is needed, the answer is one to three constrained tokens and accuracy is unchanged (95.9% on that version's 14,445-message test, as with the full answer). In batched PyTorch at 16-bit precision the model sorts 2,206 messages a second. Served through the llama.cpp server at 8 bits it sorts 92.7 messages a second with 32 requests open and a 95th-percentile response time of 0.55 seconds; the server kept the GPU 60% busy, so that figure is set by the server and not by the model. The same job on the 110M g0 model runs 4.8 times as fast in batched PyTorch at equal precision (4,013 against 829 messages a second at 32 bits) with 92.3% of categories right, about twice the error rate.

**Limits of this result.** Most test messages come from the same public datasets as the training messages, as different rows; the two tests of unseen material are the harder evidence, and on those the model is right about three times in four. The written hand-off test was produced by the same generator as the written training messages, which makes it an easy test. False hand-offs on the dataset never trained on are 58 of 3,097 against 61 allowed, a thin margin, and that dataset was read while designing the last round of training messages. Fine-tuning runs differ: a repeat with the same recipe and seed caught 80.4% of off-topic queries against 84.1%. Qwen2.5-0.5B's speed as a fine-tuned sorter was not measured under load.

## 10 Limitations

- The model was trained once. We have no second seed at this size, so we cannot put a run-to-run margin on its scores.
- It is not a single-recipe run: the first 13% of training had no weight decay on the Muon matrices. The weight decay value was derived from one measurement and not tuned.
- There is no same-size control without fiction, without the longer context, or with sampling with replacement. The effects of these changes are measured only at 110M parameters, or not at all.
- Learning rates and batch size were carried over from 110M parameters without retuning.
- Reference models differ from g1 in data, tokenizer and architecture as well as in training budget. The comparisons show where g1 stands, not why.
- Decontamination cannot check items shorter than 13 words.
- The training text is narrow. On Wikipedia text g1 is behind every finished model of its size, and the benchmark set leans on science questions, where an educational corpus helps most.
- The base model's speed was measured on consumer hardware in one engine. Throughput under load on a datacenter-class GPU is measured only for the fine-tuned sorter, in one server.
- Fine-tuning was tested on one task, with one run per model, on tests drawn mostly from the datasets used for training.
- The model is a base model with no instruction tuning or safety training. It produces confident falsehoods, is more stereotyped than g0 on the one probe we ran, has not been checked for memorized personal information, and is not suitable for any user-facing use.
- English only. Code, mathematics and conversation are nearly absent from the training text.

## 11 Conclusion

The recipe found with a 110M-parameter model carried to a model 4.7 times larger with four times the data, after one correction: weight decay on the Muon-trained matrices, without which long runs diverge. In 87 GPU-hours on one consumer GPU, Treepai g1 passed GPT-2 medium and Pythia-410M,

the established open models of its size, and raised LAMBADA, the weakest result of a narrow educational corpus, from 21.8% to 42.3%. It remains five to nine points behind Qwen2.5-0.5B and SmolLM2-360M, the best current models of its size, which were trained with hundreds of times more data, and outside educational text it trails every finished model of its size. It is also no faster to run than they are. Fine-tuned for a first task, sorting support messages, it is level with Qwen2.5-0.5B fine-tuned the same way: on that job the gap on general text did not matter. The next questions are whether broader data closes the general gaps at this scale, as the g0 data-mix test suggested it might once capacity is no longer the limit, whether a smaller or reshaped model can do the same job at lower serving cost, and whether the fine-tuning result holds on a second task.

## Acknowledgments

This work relies on openly released data and tools. We thank the Hugging Face team for FineWeb and FineWeb-Edu and for SmolLM2, EleutherAI for the Pythia models and the lm-evaluation-harness, OpenAI for GPT-2, the Qwen team for Qwen2.5, Project Gutenberg and Standard Ebooks for public-domain texts, and the creators of every benchmark used here. FineWeb-Edu is used under the ODC-By 1.0 license.

---

## Appendix A · Model card

**Table 13.** Model card, following Mitchell et al. [37]

| | |
|---|---|
| Model | Treepai g1, 517.8M-parameter decoder-only transformer, base (pretrained) model only. |
| Developer | Treepai, Inc., October 2026. |
| Availability | Research model. Weights, tokenizer and training code are not released. |
| Intended use | Research at Treepai on training and serving efficiency; a base for later fine-tuned models. |
| Out of scope | Any user-facing use, factual question answering, advice of any kind, and languages other than English. |
| Training data | 10.0B tokens of English text, each used once: 9.3B from FineWeb-Edu (educational score ≥ 3) and 0.7B of filtered public-domain fiction. 6,646 documents and 33 books overlapping the nine benchmarks removed. |
| Evaluation | Nine zero-shot capability benchmarks, held-out bits per byte and CrowS-Pairs (Section 6). |
| Known risks | States false facts fluently; reproduces stereotypes more than g0 (Table 10); not checked for memorized personal information. |
| Compute | 87 GPU-hours on one RTX 4090; at most ~39 kWh at the GPU. |

## Appendix B · Validation samples

Prompt in grey; temperature 0.8, top-k 50; unedited. One sample every 2,500 steps. Loss is the validation reading at that step.

| | |
|---|---|
| **66M tokens**<br>step 250 · val 4.59 | The most important thing about any kind of matter in the universe. But if it could have been a perfect idea for more than a decade, but it could have been impossible to prove in the idea of matter, in fact, |
| **0.66B tokens**<br>step 2,500 · val 3.33 | The most important thing about “probiotic contamination” is that it may actually be beneficial to you or the environment. The bacteria, fungi, and other microorganisms from the soil can be found in your soil, and the environment |
| **1.31B tokens**<br>step 5,000 · val 3.18 | The most important thing about it is that people are not just interested in the benefits of the technology, they are concerned about the harm that technology brings to human health. In conclusion, when you use a product or service, you |
| **1.97B tokens**<br>step 7,500 · val 3.15 | The most important thing about the two is that they are very much like the real-life real-world-life scenarios. What they do has to be real, i.e. real, but they do not meet everything set standards. The |
| **2.62B tokens**<br>step 10,000 · val 3.09 | The most important thing about these facts is that they are not true. According to the information gathered by the research, when people learn about the facts, they are better able to make informed decisions. This can happen by taking some |
| **3.28B tokens**<br>step 12,500 · val 3.08 | The most important thing about this story is the fact that it is a fictional creation, or it is a fictional creation. This is because of the fact that, in fiction, writers often have to work to craft their characters and |
| **3.93B tokens**<br>step 15,000 · val 3.03 | The most important thing about a paper is that, as a result, it is the most frequently used, and the most commonly understood. They are not just about the subject in a few sentences, but more often than not, |
| **4.59B tokens**<br>step 17,500 · val 3.00 | The most important thing about this research is that it was conducted over a period of a single year. The study was done from 1993 to June 2012 and it was conducted in eight countries: Angola, Brazil, |
| **5.24B tokens**<br>step 20,000 · val 3.03 | The most important thing about this study was that the researchers used the method of cognitive and behavioral screening. The study was conducted in 1992 and the findings of this study will help inform parents about the importance of early intervention for |

| | |
|---|---|
| **5.90B tokens**<br>step 22,500 · val 2.95 | The most important thing about this is to get a strong immune system so you can fight off infections. How to reduce the risk of infection If you have not yet been exposed to a disease and you have been injured or a |
| **6.55B tokens**<br>step 25,000 · val 2.97 | The most important thing about the course is that it gives a real-world perspective and that, as the course begins, you can start to get into small things. The main thing that you should begin with is the simple form of |
| **7.21B tokens**<br>step 27,500 · val 2.91 | The most important thing about the Kellogg School for the Blind is that it has taken the technology to make it possible. The Kellogg School for the Blind uses specially-designed equipment to assist blind students with their learning. |
| **7.86B tokens**<br>step 30,000 · val 2.87 | The most important thing about the process is that the researchers are going to be able to detect the biomarkers. This is a critical step to improving treatment options in many cases. The researchers will also be able to detect biomarkers that could |
| **8.52B tokens**<br>step 32,500 · val 2.83 | The most important thing about this is that the data can be used to make an informed decision about what the future holds. Because it is so easy to access, the researchers have been able to investigate the question of climate change impacts |
| **9.18B tokens**<br>step 35,000 · val 2.77 | The most important thing about the Internet is that it is a very public one, which means that there is lots of information out there, and it is very easy to find. This is very important, because the Internet is also |
| **9.83B tokens**<br>step 37,500 · val 2.64 | The most important thing about any educational plan is to know what you are doing. It's time to decide exactly what you have to do to be effective. Is your school using the latest research and best practice to teach mathematics? |
| **10.00B tokens**<br>step 38,147 · val 2.65 | The most important thing about what you learn in A-level Biology is that you will explore the subject matter from a variety of perspectives, as well as having practical skills and practical application. Your course work will include experiments, projects, |

# References


[1] Franco, J. Treepai g0: A 110M-Parameter Language Model Trained From Scratch on One GPU. Treepai, Inc. research report, October 2026. https://treepai.com/research/treepai-g0/

[2] Radford, A., Wu, J., Child, R., et al. Language Models are Unsupervised Multitask Learners. OpenAI technical report, 2019.

[3] Biderman, S., Schoelkopf, H., Anthony, Q., et al. Pythia: A Suite for Analyzing Large Language Models Across Training and Scaling. ICML 2023.

[4] Ben Allal, L., Lozhkov, A., Bakouch, E., et al. SmolLM2: When Smol Goes Big — Data-Centric Training of a Fully Open Small Language Model. COLM 2025.

[5] Qwen Team. Qwen2.5 Technical Report. arXiv:2412.15115, 2024.

[6] Touvron, H., Lavril, T., Izacard, G., et al. LLaMA: Open and Efficient Foundation Language Models. arXiv:2302.13971, 2023.

[7] Zhang, B., Sennrich, R. Root Mean Square Layer Normalization. NeurIPS 2019.

[8] Su, J., Lu, Y., Pan, S., et al. RoFormer: Enhanced Transformer with Rotary Position Embedding. Neurocomputing 568:127063, 2024.

[9] Shazeer, N. GLU Variants Improve Transformer. arXiv:2002.05202, 2020.

[10] Penedo, G., Kydlíček, H., Ben Allal, L., et al. The FineWeb Datasets: Decanting the Web for the Finest Text Data at Scale. NeurIPS 2024 Datasets and Benchmarks Track.

[11] Lozhkov, A., Ben Allal, L., von Werra, L., Wolf, T. FineWeb-Edu: the Finest Collection of Educational Content. Hugging Face, 2024. doi:10.57967/hf/2497.

[12] Brown, T. B., Mann, B., Ryder, N., et al. Language Models are Few-Shot Learners. NeurIPS 2020.

[13] Jordan, K., Jin, Y., Boza, V., et al. Muon: An Optimizer for Hidden Layers in Neural Networks. Blog post, 2024. kellerjordan.github.io/posts/muon

[14] Liu, J., Su, J., Yao, X., et al. Muon is Scalable for LLM Training. arXiv:2502.16982, 2025.

[15] Hu, S., Tu, Y., Han, X., et al. MiniCPM: Unveiling the Potential of Small Language Models with Scalable Training Strategies. COLM 2024.

[16] Bergsma, S., Dey, N., Gosal, G., et al. Straight to Zero: Why Linearly Decaying the Learning Rate to Zero Works Best for LLMs. ICLR 2025.

[17] Hoffmann, J., Borgeaud, S., Mensch, A., et al. Training Compute-Optimal Large Language Models. arXiv:2203.15556, 2022.

[18] Gao, L., Tow, J., Abbasi, B., et al. The Language Model Evaluation Harness. Zenodo, 2024. doi:10.5281/zenodo.12608602.

[19] Zellers, R., Holtzman, A., Bisk, Y., et al. HellaSwag: Can a Machine Really Finish Your Sentence? ACL 2019.

[20] Clark, P., Cowhey, I., Etzioni, O., et al. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. arXiv:1803.05457, 2018.

[21] Bisk, Y., Zellers, R., Le Bras, R., et al. PIQA: Reasoning about Physical Commonsense in Natural Language. AAAI 2020.

[22] Paperno, D., Kruszewski, G., Lazaridou, A., et al. The LAMBADA Dataset: Word Prediction Requiring a Broad Discourse Context. ACL 2016.

[23] Sakaguchi, K., Le Bras, R., Bhagavatula, C., Choi, Y. WinoGrande: An Adversarial Winograd Schema Challenge at Scale. AAAI 2020.

[24] Mihaylov, T., Clark, P., Khot, T., Sabharwal, A. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. EMNLP 2018.

[25] Welbl, J., Liu, N. F., Gardner, M. Crowdsourcing Multiple Choice Science Questions. Workshop on Noisy User-generated Text (W-NUT), 2017.

[26] Hendrycks, D., Burns, C., Basart, S., et al. Measuring Massive Multitask Language Understanding. ICLR 2021.

[27] Nangia, N., Vania, C., Bhalerao, R., Bowman, S. R. CrowS-Pairs: A Challenge Dataset for Measuring Social Biases in Masked Language Models. EMNLP 2020.

[28] Blodgett, S. L., Lopez, G., Olteanu, A., et al. Stereotyping Norwegian Salmon: An Inventory of Pitfalls in Fairness Benchmark Datasets. ACL 2021.

[29] Gerganov, G., et al. llama.cpp. Software, github.com/ggml-org/llama.cpp.

[30] Casanueva, I., Temčinas, T., Gerz, D., Henderson, M., Vulić, I. Efficient Intent Detection with Dual Sentence Encoders. Workshop on NLP for Conversational AI, ACL 2020.

[31] Larson, S., Mahendran, A., Peper, J. J., et al. An Evaluation Dataset for Intent Classification and Out-of-Scope Prediction. EMNLP-IJCNLP 2019.

[32] Casanueva, I., Vulić, I., Spithourakis, G., Budzianowski, P. NLU++: A Multi-Label, Slot-Rich, Generalisable Dataset for Natural Language Understanding in Task-Oriented Dialogue. Findings of NAACL 2022.

[33] Chen, D., Chen, H., Yang, Y., Lin, A., Yu, Z. Action-Based Conversations Dataset: A Corpus for Building More In-Depth Task-Oriented Dialogue Systems. NAACL 2021.

[34] Peskov, D., Clarke, N., Krone, J., et al. Multi-Domain Goal-Oriented Dialogues (MultiDoGO): Strategies toward Curating and Annotating Large Scale Dialogue Data. EMNLP-IJCNLP 2019.

[35] Pereira, L. S. B. Classification of IT Support Tickets. Zenodo, 2022. doi:10.5281/zenodo.7648117.

[36] FitzGerald, J., Hench, C., Peris, C., et al. MASSIVE: A 1M-Example Multilingual Natural Language Understanding Dataset with 51 Typologically-Diverse Languages. ACL 2023.

[37] Mitchell, M., Wu, S., Zaldivar, A., et al. Model Cards for Model Reporting. FAT* 2019.

© 2026 Treepai, Inc. This report may be downloaded and shared unchanged, with credit to Treepai, Inc. All other rights reserved.