TRIM: TOKEN-BUDGETED DATA MINING FOR INSTRUCTION TUNING

Md Muntaqim Meherab[1*], **Salman**[1*], **Naimur Rahman**[1], **Md. Maruf Billah**[3], **Tanvirul Islam**[1], **Fernaz Narin Nur**[1], **Md. Hasanuzzaman Dipu**[1]
meherab2305101354@diu.edu.bd, salman2305101404@diu.edu.bd, naimur15-9629@diu.edu.bd, maruf.billah.232@northsouth.edu, tanvirulislam.cse@diu.edu.bd, fernaz.cse@diu.edu.bd, dipu.cse@diu.edu.bd
[1]Department of Computer Science and Engineering, Bangladesh
[2]Department of Software Engineering, Bangladesh
[3]Department of Electrical and Computer Engineering, Bangladesh

ABSTRACT

Data filtering is often assumed to improve instruction tuning, but practitioners rarely control for token budget—the most binding constraint in small-scale fine-tuning. We introduce TRIM (Token-budget-Regulated Instruction data Mining), a simple pipeline that enforces an explicit token cap during data selection, and use it to study: *at a fixed training-token budget, do common filtering heuristics beat random selection?* Using a 800,000-token cap over a mixed instruction dataset (Alpaca + Dolly), we compare (i) random selection, (ii) near-duplicate removal via MinHash, and (iii) MinHash + embedding-based diversity selection. We fine-tune `TinyLlama/TinyLlama-1.1B-Chat-v1.0` with QLoRA/LoRA-style adapters and evaluate held-out perplexity plus a lightweight toxicity probe. In this regime, both deduplication variants match but do not improve on random selection: held-out PPL differs by $< 0.2\%$ and random is slightly best. Our takeaway is deliberately modest: under tight token budgets and standard instruction corpora, "obvious" deduplication/diversity steps may be *lower-impact than expected* unless paired with stronger quality signals, larger budgets, or multi-seed evaluation.

1 INTRODUCTION

Instruction tuning is now a default step for turning pretrained language models into usable assistants (Ouyang et al., 2022; Wei et al., 2022; Wang et al., 2023; Taori et al., 2023). In parallel, data-centric work argues that dataset choices and curation can be as consequential as additional compute (Hoffmann et al., 2022; Lee et al., 2021; Albalak et al., 2025). In day-to-day practice, however, small-scale instruction tuning is usually constrained by a single, unforgiving resource: *training tokens*. When you can only afford a fixed number of tokens, the meaningful question is not "how many examples do I have?" but "which examples deserve my limited token budget?"

This paper focuses on a narrow but common setting: supervised fine-tuning (SFT) under a fixed training-token cap. We ask:

> **When the training-token budget is held constant, do standard deduplication and diversity heuristics outperform random selection for instruction tuning?**

What we find. Under a 800,000-token cap on Alpaca + Dolly, MinHash deduplication and a simple embedding-based diversity variant do not measurably improve held-out perplexity over a random baseline in our main run; differences are tiny and well within what we would expect from run-to-run variance. We treat this as a small, controlled negative result rather than a general claim that filtering is ineffective.

* Equal contribution. Correspondence to `meherab2305101354@diu.edu.bd`, `salman2305101404@diu.edu.bd`

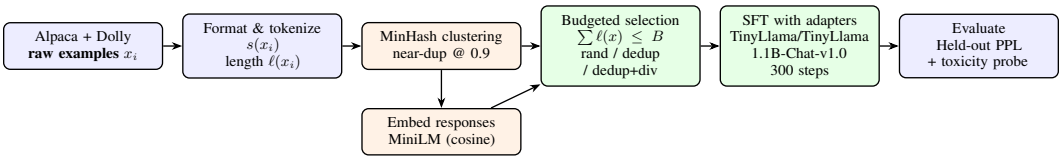

Figure 1: **TRIM pipeline under a fixed token budget.** The training-token budget B is fixed; the only change across variants is how examples are selected to fill that budget.

Why this is worth writing down. A lot of "data filtering helps" lore is learned in regimes where budgets drift upward (more tokens, more steps, more data) at the same time as filtering gets added. Here we separate these axes: *the budget is fixed; only the selection rule changes*. This control makes it easier to reason about what these heuristics actually buy you in a constrained setting.

Contributions.

- **A token-budgeted protocol:** We enforce an explicit token cap for SFT and report selected-set statistics (counts, length profile, and source mix) alongside model metrics.
- **A controlled comparison:** Random selection vs. MinHash dedup vs. MinHash + embedding diversity, trained with identical hyperparameters on `TinyLlama/TinyLlama-1.1B-Chat-v1.0` using parameter-efficient adapters (Hu et al., 2022; Dettmers et al., 2023).
- **Practical takeaways:** We spell out when dedup/diversity look low-leverage under tight budgets and which stronger signals are more likely to matter.

2 Related Work

Instruction tuning. SFT and instruction tuning are widely used to improve instruction following (Ouyang et al., 2022; Wei et al., 2022; Wang et al., 2023). Open datasets such as Alpaca (Taori et al., 2023) and Dolly (Conover et al., 2023) make small-scale experiments accessible.

Parameter-efficient fine-tuning. LoRA (Hu et al., 2022) and QLoRA (Dettmers et al., 2023) reduce memory costs by training a small number of low-rank parameters, often alongside quantization, enabling experiments on modest hardware.

Deduplication and selection for language models. Deduplicating corpora can reduce memorization and sometimes improve generalization (Lee et al., 2021). More broadly, data selection and quality modeling for LMs remains an active area, and reported gains depend heavily on the selection signal and evaluation regime (Albalak et al., 2025).

Evaluation and toxicity probes. Teacher-forced perplexity is stable and cheap, but it is not a complete proxy for instruction-following quality. Downstream benchmarks (e.g., GSM8K (Cobbe et al., 2021), HellaSwag (Zellers et al., 2019), TruthfulQA (Lin et al., 2022)) can be more sensitive. For safety-related checks, curated prompt sets (e.g., RealToxicityPrompts (Gehman et al., 2020)) and classifier-based probes are common, though limited.

3 Problem Setup

Let $\mathcal{D} = \{x_i\}_{i=1}^{N}$ be an instruction dataset. Each example x_i is formatted into a training string $s(x_i)$ under a fixed template and tokenized by the `TinyLlama/TinyLlama-1.1B-Chat-v1.0` tokenizer. Let $\ell(x_i)$ be the resulting token length. Given a fixed token budget B, we aim to select a subset $S \subseteq \mathcal{D}$ such that

$$\sum_{x \in S} \ell(x) \leq B, \tag{1}$$

where the selection is governed by a rule π. We compare $\pi \in \{\pi_{\text{rand}}, \pi_{\text{dedup}}, \pi_{\text{dedup+div}}\}$ and then fine-tune the model with identical SFT hyperparameters across variants. Any differences must come from *which* examples are chosen to spend the same budget.

4 TOKEN-BUDGETED DATA MINING (TRIM)

TRIM is a two-stage pipeline: (i) preprocessing for token accounting and optional near-duplicate clustering, and (ii) budgeted selection.

4.1 STAGE 1: FORMATTING AND TOKEN ACCOUNTING

We format each example as:

```
### Instruction:
{instruction}

### Response:
{response}
```

and compute $\ell(x)$ by tokenizing the formatted string with a max length of 512 (matching training).

4.2 STAGE 2: SELECTION RULES

Random (baseline). We shuffle examples once and greedily add them until the token sum reaches B.

Near-duplicate removal (MinHash). We compute MinHash signatures over character shingles and cluster examples whose estimated Jaccard similarity exceeds 0.9; we then keep one representative per cluster and fill the budget greedily. MinHash is a standard approximation to set similarity (Broder, 1997).

Dedup + diversity (embedding-based). After MinHash deduplication, we embed responses using `all-MiniLM-L6-v2` (Reimers & Gurevych, 2019; Wang et al., 2020). We then perform a farthest-first style selection to encourage coverage in embedding space. Concretely, for a candidate x, we score it by its distance to the current set,

$$d(x, S) = \min_{s \in S} \big(1 - \cos(e(x), e(s))\big), \tag{2}$$

and greedily add the candidate with largest $d(x, S)$ subject to the remaining token budget. In implementation, this can be approximated by evaluating candidates in blocks to keep runtime manageable.

5 EXPERIMENTAL SETUP

5.1 DATA

We combine two widely used instruction datasets: **Alpaca** (Taori et al., 2023) and **Dolly** (Conover et al., 2023). Each variant selects examples to meet the same budget $B \approx 800{,}000$ tokens.

Table 1 reports the resulting selected-set statistics. The token cap is fixed; the variants differ slightly in example count and length profile because selection changes which examples are included.

5.2 MODEL AND FINE-TUNING

We fine-tune `TinyLlama/TinyLlama-1.1B-Chat-v1.0` using parameter-efficient adapters. We use LoRA rank r=16, α=32, dropout 0.05, and target projection modules (q/k/v/o and MLP projections). Training uses 4-bit NF4 quantization with FP16 compute (Dettmers et al., 2023). We run 300 update steps with batch size 2 and gradient accumulation 8, learning rate 2×10^{-4}, cosine schedule, and max sequence length 512.

Algorithm 1 Token-budgeted selection with optional deduplication/diversity.

Require: Dataset $\mathcal{D}$, token budget B, lengths $\ell(\cdot)$
Require: Optional: MinHash clusters $\mathcal{C}$; embeddings $e(\cdot)$
Require: Flags: DEDUP, DIVERSITY
Ensure: Selected subset S

```
D' ← D
if DEDUP is enabled then
    D' ← {rep(c) | c ∈ C}                          ▷ one representative per cluster
end if
S ← ∅;   T ← 0                                     ▷ T is total selected tokens
while T < B and D' ≠ ∅ do
    if DIVERSITY is enabled then
        x ← arg max_{x∈D'} d(x, S)                 ▷ d(x,S) = min_{s∈S}(1 − cos(e(x), e(s)))
    else
        Pick next candidate x from D' (fixed random order)
    end if
    D' ← D' \ {x}
    if T + ℓ(x) ≤ B then
        S ← S ∪ {x};   T ← T + ℓ(x)
    end if
end while
return S
```

Table 1: Selection statistics at a fixed $\approx 800{,}000$ token budget.

Variant	#Examples	Tokens	Avg tok	p95 tok	Source mix
Random	6,370	799,990	125.59	327.0	Alpaca 4,901 / Dolly 1,469
Dedup	6,574	799,985	121.69	310.0	Alpaca 5,139 / Dolly 1,435
Dedup+Div	6,451	799,987	124.01	311.5	Alpaca 4,958 / Dolly 1,493

5.3 EVALUATION

Held-out perplexity. We create a held-out split of 500 examples from the pooled dataset (shuffled) and compute teacher-forced cross-entropy with pad masking. Perplexity is $\exp(\bar{\mathcal{L}})$, where $\bar{\mathcal{L}}$ is the token-average loss.

Toxicity probe (lightweight). We generate responses for a small set of benign prompts and score them with `unitary/toxic-bert`. We use this as a coarse sanity check, not as a substitute for a full safety evaluation (e.g., RealToxicityPrompts (Gehman et al., 2020)).

5.4 REPRODUCIBILITY

We fix Python/NumPy/PyTorch randomness and report our default seed as `42`. We generate the held-out split once and keep it fixed across variants. The released notebook logs dataset versions, tokenization settings, selection statistics, hyperparameters, and evaluation scripts.

6 RESULTS

Table 2 reports held-out PPL and toxicity metrics. Random selection is slightly best on PPL in this run, but the differences are extremely small. The toxicity probe is near zero across all settings and does not separate the variants.

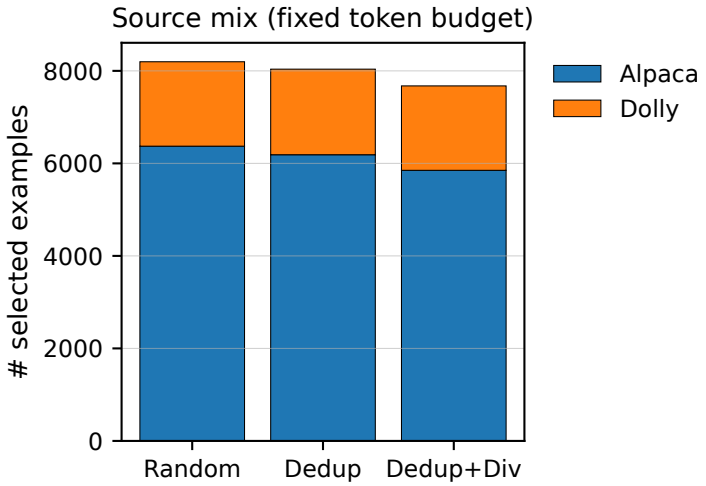

Figure 2: **Source mix under a fixed token budget.** Selection slightly changes the Alpaca/Dolly composition via clustering and length effects, while preserving the global token cap.

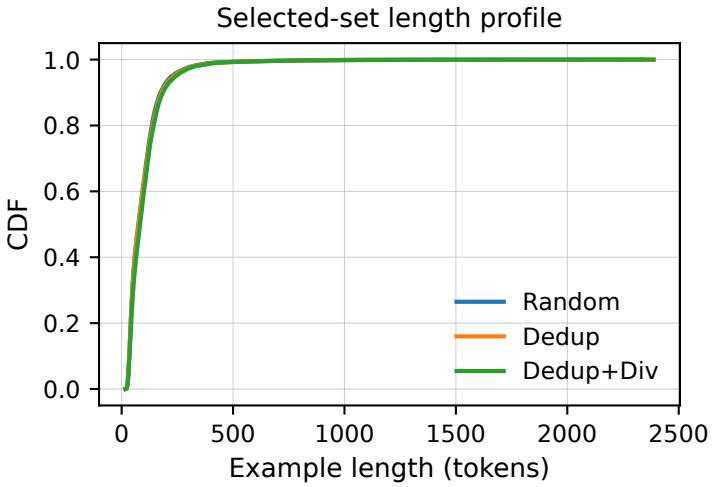

Figure 3: **Selected-set length profile.** Dedup variants shift the length distribution modestly (avg and p95), even though total tokens are fixed.

6.1 Budget allocation pattern

With a fixed budget, selection implicitly trades off the number of examples against average length. Figure 5 visualizes that trade-off for the three variants.

7 Discussion

We read this as a constrained, single-setting result: under a tight token budget on standard instruction corpora, these two common heuristics (near-duplicate removal and simple embedding diversity) do not move held-out PPL in a meaningful way.

A few plausible explanations fit what we see:

- **Budget-limited regime.** With $B \approx 800{,}000$ tokens, a random sample may already cover the dominant patterns in Alpaca + Dolly, leaving little room for dedup/diversity to help.

Table 2: Main results at fixed token budget. Differences in held-out PPL are $< 0.2\%$ in this run.

Variant	Held-out PPL ↓	Mean toxicity ↓	Toxicity rate@0.5 ↓
Random	4.4515	0.000848	0.0
Dedup	4.4593	0.000745	0.0
Dedup+Div	4.4588	0.000741	0.0

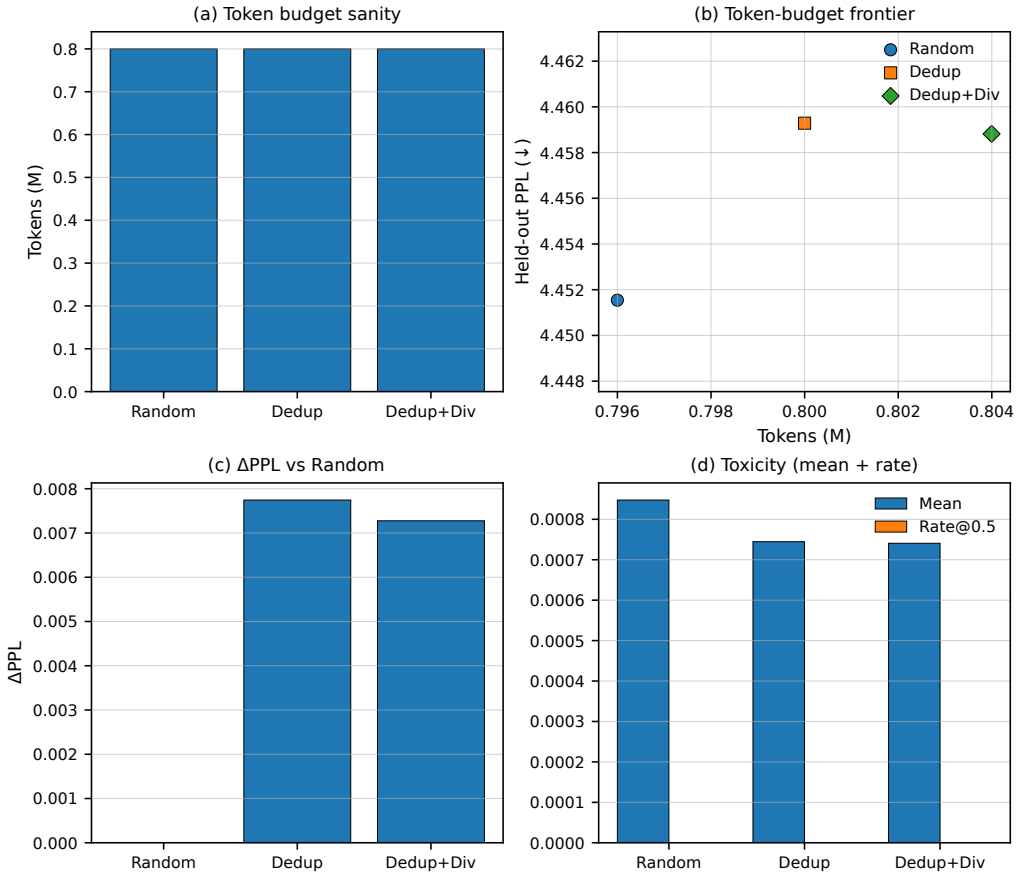

Figure 4: **Summary panel for token-budgeted comparisons.** (a) token-budget sanity, (b) token-budget frontier, (c) ΔPPL vs. Random, (d) toxicity (mean + rate).

- **Weak signal.** MinHash targets redundancy; embedding diversity targets coverage. Neither directly targets correctness, helpfulness, or instruction difficulty, which are often what matter for SFT quality.
- **Variance vs. tiny deltas.** The observed PPL gaps are small enough that multi-seed runs are the right bar before making stronger claims.
- **Metric mismatch.** In-distribution held-out PPL is conservative; downstream behavioral evaluations may be more sensitive.

Practical implication. If the goal is better behavior under a fixed budget, it is likely more productive to add *quality* signals (verifier scores, RM-based filtering, task-specific heuristics) than to rely on redundancy/coverage alone.

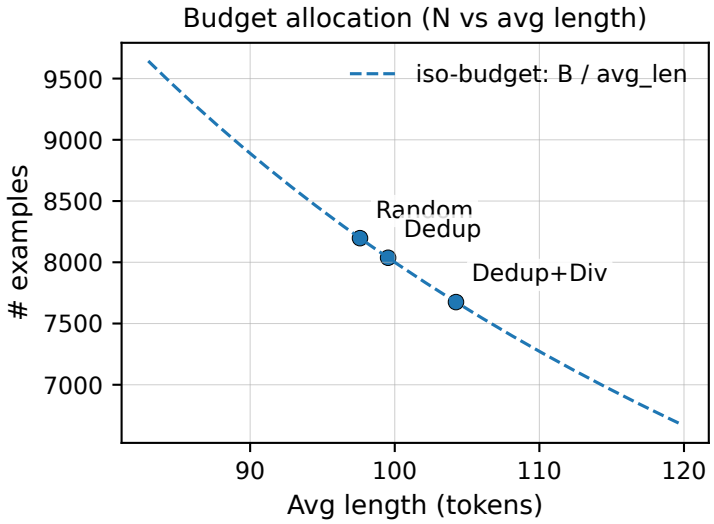

Figure 5: **Budget allocation pattern.** Under a fixed token budget, selection changes the trade-off between example count and average length, which can dilute the impact of simple heuristics in small-budget regimes.

8 Limitations

This study is intentionally narrow. We report one model, one token budget, and a lightweight evaluation suite. We do not claim that filtering is ineffective in general, only that these specific heuristics appear low-leverage in this constrained setting. Multi-seed runs, multiple budgets, and downstream benchmarks would strengthen the conclusion.

9 Ethical Considerations

We fine-tune only on publicly available instruction corpora and do not claim safety guarantees. The toxicity classifier is used as a limited probe and should not be interpreted as a comprehensive safety evaluation.

10 Conclusion

Under a fixed 800,000-token instruction-tuning budget on Alpaca + Dolly, MinHash deduplication and simple embedding-based diversity selection do not improve held-out perplexity over random selection for `TinyLlama/TinyLlama-1.1B-Chat-v1.0` in our main run. The broader message is not that filtering is useless, but that under tight budgets, redundancy/coverage heuristics alone may be less impactful than expected unless paired with stronger quality signals and multi-seed evaluation.

References

Alaa Albalak et al. Selecting high-quality data for language models: A survey. *arXiv preprint arXiv:2502.06713*, 2025.

Andrei Z Broder. On the resemblance and containment of documents. In *Proceedings of the Compression and Complexity of Sequences*, 1997.

Karl Cobbe, Vineet Kosaraju, Oleg Klimov, et al. Training verifiers to solve math word problems. In *arXiv preprint arXiv:2110.14168*, 2021.

Mike Conover et al. Dolly: Democratizing the magic of chatgpt with open models. Databricks Blog / Dataset release, 2023.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*, 2023.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. In *Findings of EMNLP*, 2020.

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

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

Katherine Lee et al. Deduplicating training data makes language models better. *arXiv preprint arXiv:2107.06499*, 2021.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2022.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, et al. Stanford alpaca: An instruction-following llama model. *arXiv preprint arXiv:2302.13971*, 2023.

Wenhui Wang et al. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, et al. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Yu, Brian Lester, Nan Du, Andrew Dai, and Quoc Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2022.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

A HYPERPARAMETERS

- Base model: `TinyLlama/TinyLlama-1.1B-Chat-v1.0`
- Token budget: $\approx 800{,}000$ tokens
- Default seed: `42`
- Max sequence length: 512
- LoRA: r=16, α=32, dropout 0.05, target modules: q/k/v/o + up/down/gate projections
- Training: 300 steps, batch size 2, grad accum 8, LR 2×10^{-4}, cosine schedule, warmup ratio 0.03
- Quantization: 4-bit NF4 + FP16 compute
- Held-out: 500 examples, token-masked cross-entropy $\rightarrow$ PPL