Université de Toulouse

Master parcours Interactions de l'informatique et des mathématiques pour l'intelligence artificielle

Master 2 Internship Report

2025 – 2026

Math-Specialized Language Models via Continued Pretraining, Structured Pruning, and Knowledge Distillation

---

Niyar R BARMAN

Linagora Labs

Toulouse, France

Supervisor: Julie HUNTER

Contents

Acknowledgements

I would first like to thank Julie Hunter, my internship supervisor and the Head of NLP at LINAGORA. She helped frame the project as a question of model specialisation before compression, introduced the Minitron direction, and encouraged me to treat the early failures as evidence to be explained rather than results to be worked around. Her guidance was especially valuable when the project moved from a single pruning experiment to a complete training and evaluation pipeline.

I am grateful to the LINAGORA Labs AI team in Toulouse for the technical discussions and for the shared tools on which this work relied. The team's work on post-training, retrieval and tool use, speech, and evaluation supplied both practical conventions and a broader view of where a specialised mathematical model might be useful. In particular, the French benchmark ports and evaluation-harness practices were collective assets rather than isolated contributions of this internship.

I also thank the members of the OpenLLM-France consortium. This project was possible because the Luciole checkpoints, training corpora, and code were available in a form that could be inspected and reused.

Finally, I acknowledge GENCI and IDRIS for access to the Jean-Zay H100 partition, and the Dalia GB200 team for the earlier allocation on which parts of the pipeline were prototyped.

1 Introduction

1.1 An open model and a demanding use case

The conditions under which a language model is produced matter as much as its final benchmark score. A model whose weights can be downloaded is useful for deployment, but its behaviour remains difficult to audit if neither its training data nor its training code is available. LINAGORA and the OpenLLM-France consortium adopt a stricter definition of openness. They publish model weights, training resources, and code, with the aim of making the complete process inspectable and reproducible. The Luciole family is one result of this effort. It is a family of 3 multilingual causal language models trained on about five trillion tokens, with a substantial French share, and released under the Apache 2.0 licence according to its official model card (OpenLLM-France, 2026).

This design choice supports linguistic coverage and technological sovereignty but limits access to the highly curated mathematical reasoning traces found in specialised proprietary mixtures. Consequently, the 8B Luciole base model underperforms on mathematical tasks required for educational applications. This motivates targeted adaptation. Training a new small model on a larger mathematical corpus could help, but would duplicate much of the existing training cost while discarding the valuable knowledge already encoded in the larger Luciole checkpoint.

1.2 The compression question

The project can therefore be stated as one question: can the existing Luciole-23B checkpoint be used to produce an 8B-scale model that substantially improves on Luciole-8B-Base in mathematics, without repeating full pretraining and while retaining as much of the parent's general knowledge as possible?

The answer explored here uses specialisation and compression in that order. Continued pretraining first corrects the mathematical weakness of Luciole-23B. The corrected teacher is then compressed through two repetitions of structured pruning followed by knowledge distillation. Stage 1 moves from 23.22B to 13.98B parameters. Stage 2 moves from the distilled 14B checkpoint to 8.36B parameters, while retaining the same corrected 23B teacher as the source of distillation targets. The complete sequence is shown later in Figure 1.

This order is central to the argument. Compression cannot preserve a capability that the teacher barely possesses. Conversely, mathematical specialisation alone does not produce a model that satisfies a deployment budget. Continued pretraining changes what is available to transfer; pruning changes the architecture; distillation teaches that architecture to approximate the stronger teacher.

2 Presentation of the Host Organization

2.1 The host organization

LINAGORA is a French company founded in 2000, focused on developing and promoting free and open-source software and providing services around such software. For over twenty-five years it has helped public and private organisations design, deploy, and maintain digital tools, and it defends a particular position in the digital ecosystem: technological sovereignty built on transparency, ethics, and collaboration.

By *open source* it means software whose source code is freely accessible, modifiable, and redistributable, and beyond that technical sense it treats open source as the basis of a "third way" for digital technology, one that makes sovereignty possible. This approach guides the development of their major products including Twake (`https://twake.app/`), a digital workplace that includes, among many others, platforms for chat, mail, file sharing, shared document edition and calendars. LinTO Studio (`https://linto.ai/`) is a comprehensive media management platform that offers advanced tools for transcription and collaborative editing, including automatic subtitles, speaker identification and the automatic generation of summaries. OpenRAG (`https://github.com/linagora/openrag`) is a lightweight, modular and extensible Retrieval-Augmented Generation (RAG) framework designed to explore and test advanced RAG techniques.

In 2023, LINAGORA turned to the development of open-source language models to fuel their products and to further the development of sovereign AI with a focus on the French language. While *open weights* models allow for fine-tuning and on-site deployment, LINAGORA goes further by training its models exclusively on open-source corpora and

publishing all of its training data and code. This makes the models and training resources offered by LINAGORA not only fully open-source, in line with the company's broader philosophy, but also valuable tools for research on multilinguality and bias from training data.

2.2 The team and the working environment

This internship took place in the LINAGORA Labs AI team in Toulouse, which handles the research and scientific coordination behind the company's language models and leads the OpenLLM-France consortium (`https://openllm-france.fr/en/main-page-en/`), a group of industrial and academic partners building open, French-language AI. The consortium's principles shape how the models are made: the training data is republished so that it can be audited, the weights are released under open licences together with intermediate pretraining checkpoints, and the training and data-processing code is published in full.

The most recent models developed by LINAGORA and OpenLLM France, the Luciole family, come in three sizes: a 1B model for edge deployment, an 8B model with a Mamba hybrid architecture for long contexts, and a 23B model for stronger performance. The Luciole models, like the earlier Lucie 7B model published in 2025, share a multilingual focus centred on French and the major European languages with around a third of their training data in French.

LINAGORA Labs has ten people working on various aspects of model training, including data collection and processing, pretraining, post-training through instruction tuning and preference optimisation, model evaluation, multilingual training, and multimodality, with a particular focus on developing audio-text models. My work sat within this team, under the supervision of Julie Hunter, who leads the training of the Luciole models, and focused on the math-oriented compression of Luciole 23B described in the rest of this report.

3 Context and Objectives of the Internship

3.1 Context of the internship

The dominant development pattern for language models is to train a broad generalist model and serve it across many applications. That pattern is attractive when a single provider can spread out the cost of a very large model over many users. It is less convincing for a known deployment such as a French tutoring assistant, an on-premises service, or a system constrained by accelerator memory. In those settings, capacity spent on unrelated domains may be less valuable than a smaller model whose training is directed towards the intended task. Specialisation and compression are complementary levers: the former changes the distribution of capability, while the latter reduces the cost of serving it.

Luciole makes this trade-off concrete. The family is open, multilingual, and strongly oriented towards French. Its official 23B model card reports about five trillion pretraining

tokens and 30.4% French, alongside English, other European languages, code, and a smaller mathematical component (OpenLLM-France, 2026). In this report, "open" refers not only to downloadable weights but also to auditable training data and published training code. This standard is valuable for scientific reuse and public-sector deployment. It also narrows the set of admissible data and makes corpus quality, licensing, and linguistic balance part of the engineering problem.

The measured mathematical baselines show why a specialised model was needed. Luciole-23B-Base scores 27.66 on MINERVA-MATH and 32.42 on MMLU-Pro mathematics. Luciole-8B-Base scores 22.80 and 27.17. The smaller checkpoint is the more direct point of comparison because it already meets the target parameter class: it is deployable, French-oriented, and generally competent, but underperforms in mathematics. These results do not isolate a single causal factor in the original pretraining mixture. They do show that training a smaller member of the same family from scratch did not yield the mathematical capability required by the intended applications.

The internship mandate was therefore to reuse an existing trained asset. Luciole-23B already contained broad linguistic and factual knowledge, its datamix was known, and intermediate checkpoints were available. The project asked whether that asset could be specialised and then converted into smaller dense architectures, rather than replaced by a new pretraining run. Two technical questions followed. First, could continued pretraining improve mathematics without causing catastrophic forgetting? Second, could structured pruning and distillation retain the resulting specialised capability as effectively as they retained general competence?

3.2 Problem statement

Within this context, the internship asks how Luciole-23B can be used to produce an 8B model that improves substantially on Luciole-8B-Base in mathematics, without full pretraining and with minimal loss of general capability.

Three constraints make this more than a straightforward model-reduction. First, mathematical reasoning benefits from capacity and carefully constructed data, whereas the intended deployment class imposes strict memory and inference constraints. Second, compression can preserve only capabilities that are present in the parent. Luciole-23B must therefore be improved through continued pretraining before pruning is worthwhile. Structured pruning and knowledge distillation can then reuse that trained asset with fewer tokens than a new pretraining run would require (Muralidharan et al., 2024).

Third, every stage can introduce a different failure. A narrow mathematical mixture can cause catastrophic forgetting or unstable generation; pruning removes capacity that distillation must recover; and numerical or checkpoint-conversion faults can resemble genuine loss of capability. The problem is therefore both scientific and engineering: the project must produce a stronger small model and establish that its measured gains come from the intended training procedure rather than from an artefact.

3.3 Objectives of the internship

The main objective is to build a new, smaller Luciole model that clearly beats the existing Luciole models of similar size at mathematics and reasoning. We follow the Minitron pruning-and-distillation recipe (Muralidharan et al., 2024; Sreenivas et al., 2024) with Luciole 23B as the teacher. In practice this splits into five work packages.

1. **Data preparation.** Build and clean a training corpus drawn from both pretraining-style and post-training-style data. Token-proportional sampling of the cleaned pool yields 44.7% mathematics, 23.8% general text, 20.2% code, 5.4% science, 3.2% instruction data, and 2.8% STEM. This realised blend is a consequence of cleaning rather than manual reweighting.

2. **Teacher correction.** Specialise the Luciole 23B teacher on mathematics through about 50B tokens of continued pretraining, so that the students later distil from an improved teacher rather than the original general-purpose one. This has to happen without wrecking general knowledge or the model's ability to produce stable, well-terminated text.

3. **Iterative structured pruning.** Compress the corrected teacher in two steps, from 23B parameters to 14B and then to 8B, using activation-informed Minitron-style pruning to decide which width and depth components to drop at each step.

4. **Knowledge distillation.** Train each pruned student against the corrected teacher on roughly 112B tokens to recover and raise its performance, so the smaller models keep as much of the teacher's mathematical ability as they can.

5. **Evaluation and tooling.** Set up a reliable, reproducible pipeline on the Slurm cluster for data cleaning, training, pruning, distillation, checkpoint conversion, and benchmarking, so decisions rest on trustworthy numbers and long runs can be chained across time-limited allocations. The complete training budget is 274B tokens: 50B for teacher correction and 112B for each student.

The deliverables are a math-specialised 8B model, with the 14B model produced along the way, together with the pipeline and analysis behind them.

3.4 Success criteria

The project is judged against four criteria.

- **Better mathematical performance.** The primary criterion is that the final 8B student clearly outperform Luciole-8B-Base across a broad mathematical-reasoning suite. This suite comprises GSM8K (Cobbe et al., 2021), its more challenging variants GSM8K-Platinum (Vendrow et al., 2025) and GSM-Plus (Li et al., 2024), the mathematics subset of MMLU-Pro (Wang et al., 2024), MINERVA-MATH (Lewkowycz

et al., 2022; Hendrycks et al., 2021b), and the multilingual MGSM benchmark (Shi et al., 2022). Mathematical performance in French is assessed separately using MathAleaMCQ (CEA-LIST IA, 2026b), which covers French secondary-school mathematics, and Exo7MCQ (CEA-LIST IA, 2026a), which contains French undergraduate-level exercises. The intermediate 14B student is evaluated against the corrected 23B teacher and the available reference models in the same broad parameter range.

- **No Catastrophic Forgetting of General Capabilities.** Specialisation and compression must not collapse the model's broad competence. Across general-knowledge and commonsense benchmarks (MMLU (Hendrycks et al., 2021a), ARC-Challenge (Clark et al., 2018), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2019), PIQA (Bisk et al., 2019), and TruthfulQA (Lin et al., 2021)) and in generation quality, the model should stay close to the original Luciole models instead of showing catastrophic forgetting.

- **An efficient pipeline.** The compression route should hit these targets with far fewer tokens than pretraining equivalent models would take, in line with the Minitron efficiency results, and should run as a reproducible, resumable pipeline.

- **Auditable failures and measurements.** Each major training and evaluation decision should be backed by a control, a held-out measurement, or a reproducible manifest. Numerical settings and checkpoint conversions must be validated against a known-good or base-checkpoint reference before a long run is accepted.

Meeting the first two criteria would show that a capable math-specialised small Luciole model can come from pruning and distilling a domain-corrected teacher. Meeting the last two would show that the result can be produced efficiently and trusted for the right reasons.

4 Related Work

This project draws on four connected lines of work: domain adaptation by continued pretraining, compression of large language models, knowledge distillation, and the training and evaluation of mathematical reasoning. The literature supplies the main components of the pipeline, but it does not by itself determine their order or the criteria used to select a recoverable pruned architecture.

4.1 Continued pretraining and domain specialisation

Continued pretraining adapts an existing language model by applying its original next-token objective to a new corpus. Earlier work on domain-adaptive pretraining showed that a second phase on in-domain unlabeled text can improve downstream performance even when the initial model was trained on a broad corpus (Gururangan et al., 2020). The attraction

is economic as well as statistical: the model retains a useful initialisation and needs to learn a change in distribution rather than language from the beginning.

Adaptation also creates a stability problem. A narrow distribution can improve the target domain while eroding knowledge that is weakly represented in the new data. The effect is often described as catastrophic forgetting, but the term should be used carefully here. This report observes a severe general decline in an early run and later shows that a numerical fault was its primary cause. Mixture imbalance and sequence fragmentation were real secondary problems, yet the successful 50B-token run improved both mathematics and MMLU. The literature motivates continued pretraining; controlled experiments are still needed to determine whether a particular decline is statistical forgetting or corrupted optimisation.

4.2 Model compression for large language models

Two broad families of methods make a trained model cheaper to serve. Quantisation reduces numerical precision, while pruning removes parameters or components. These choices are not interchangeable. Unstructured methods such as SparseGPT (Frantar and Alistarh, 2023) and Wanda (Sun et al., 2023) can produce high weight sparsity with limited one-shot damage, but real latency gains depend on sparse kernels and hardware support. Structured pruning removes complete heads, channels, or layers and produces a smaller dense model whose shape is directly visible to standard matrix multiplication libraries.

Methods such as LLM-Pruner (Ma et al., 2023) estimate dependencies between structures and recover the resulting model with lightweight training. Depth-oriented work asks a related question: whether complete Transformer blocks can be removed with surprisingly limited damage (Men et al., 2024; Gromov et al., 2024). Sheared LLaMA combines a target architecture with structured pruning and continued training, showing that a pretrained parent can be a more efficient starting point for a smaller dense model than training that model independently (Xia et al., 2023).

The Minitron approach (Muralidharan et al., 2024; Sreenivas et al., 2024) is designed to derive a family of smaller models from one parent. It combines activation-based importance estimates with depth, hidden-width, attention-head, and MLP-width pruning, then recovers the pruned student through knowledge distillation. The published experiments show that structured pruning followed by retraining can require far fewer tokens than independent pretraining of every family member. This report adopts that architecture-level recipe, while changing candidate selection to account for the empirical unreliability of immediate post-pruning scores.

A recurring question in structured pruning is which data should estimate component importance. TELL-TALE studies task-aware layer elimination and reports that target-task calibration can improve the decision about which layers to remove (Naim et al., 2025). This result motivated the use of mathematical calibration data in the present project. It does not prove that every component selected by that calibration is mathematically specialised,

but it aligns the importance measurement with the eventual use case more closely than a generic multiple-choice set would.

4.3 Knowledge distillation

Knowledge distillation trains a student to imitate a larger teacher (Hinton et al., 2015). Hard next-token labels identify a single observed token. The teacher distribution also exposes relative probability assigned to alternatives, which can communicate similarities and uncertainty. Distillation may operate on logits, intermediate features, attention maps, or sequences generated by the teacher. Sequence-level distillation was developed in machine translation to replace or augment the original targets with teacher-generated sequences (Kim and Rush, 2016). Feature matching is less convenient when pruning changes depth and hidden width because student and teacher states no longer align naturally.

This project therefore uses pure logit distillation. At each token, the student distribution p_S minimises the forward Kullback-Leibler divergence from the teacher distribution p_T,

$$\mathcal{L}_{\mathrm{KD}} = D_{\mathrm{KL}}(p_T \parallel p_S) = \sum_{v \in \mathcal{V}} p_T(v) \log \frac{p_T(v)}{p_S(v)}. \tag{1}$$

Forward KL is used for three reasons. First, it is the logit-distillation objective used by the Minitron recipe (Muralidharan et al., 2024; Sreenivas et al., 2024), making the recovery procedure directly comparable. Second, because the teacher distribution is fixed, minimising $D_{\mathrm{KL}}(p_T \| p_S)$ is equivalent, up to the constant entropy of the teacher, to minimising cross-entropy against the teacher's full output distribution. The student therefore learns not only the observed token but also the relative probabilities assigned to alternative tokens. A Wasserstein objective would instead require defining a meaningful ground distance between vocabulary tokens; token identifiers have no natural geometry, and full-vocabulary optimal transport would be considerably more expensive. The additional hard-label cross-entropy term on the observed corpus token is disabled, so the teacher distribution is the only training signal.

Distillation after pruning serves two roles. It restores a model whose parameters have been inherited but whose architecture has changed abruptly, and it supplies a way to compare architecture recoverability using short probes. A low loss immediately after pruning means that an architecture incurred less initial damage. It does not follow that the same architecture will learn fastest or converge best under distillation. The experiments in Section 7 provide a direct counterexample to that assumption.

4.4 Mathematical reasoning and evaluation

Mathematical reasoning is demanding because several locally plausible steps may still lead to a globally incorrect answer. GSM8K covers grade-school word problems (Cobbe et al., 2021), while the MATH dataset contains more advanced competition-style problems (Hendrycks et al., 2021b). GSM-Plus and GSM8K-Platinum test robustness to perturbation

and contamination concerns (Li et al., 2024; Vendrow et al., 2025). MMLU-Pro increases both difficulty and the number of choices relative to MMLU (Wang et al., 2024), and MGSM extends grade-school reasoning across languages (Shi et al., 2022).

Chain-of-thought prompting can elicit intermediate reasoning and improve multi-step accuracy without changing model weights (Wei et al., 2022). Training on mathematical text and worked solutions addresses a different limitation by changing the model itself. MINERVA demonstrated that further training a language model on technical material can substantially improve quantitative reasoning (Lewkowycz et al., 2022). The corpus used here reflects this second direction, with both pretraining-style mathematical text and post-training-style worked solutions.

Exact-match evaluation remains sensitive to protocol. A correct derivation can be scored as wrong if the model omits the expected delimiter, and a model can emit a correct answer before continuing into irrelevant text. Likelihood-based multiple-choice tasks avoid some generation-format issues but test a different mode of use. This work therefore reports both forms of evaluation, includes French mathematical tasks, and treats completion length and end-of-sequence behaviour as health signals rather than cosmetic details.

4.5 Positioning of this work

The project is best described as domain-directed compression. A general and French-oriented model is first specialised through continued pretraining, after which Minitron-style pruning and distillation create two smaller dense models. The target domain influences the corpus, pruning calibration, candidate probes, and final evaluation. General benchmarks remain in the protocol precisely because an in-domain success can conceal an out-of-domain cost.

Two choices extend the published Minitron experiments. First, immediate post-pruning scores are treated as measurements of initial damage rather than reliable architecture rankings; short distillation probes instead select candidates by recoverability. Second, the pruning-and-distillation procedure is applied iteratively. The published experiments compress each parent through a single pruning stage, although iterative pruning is discussed as a possible extension. Here, a direct reduction from 23.22B to 8.36B parameters would remove 64% of the model. The corrected teacher is therefore first pruned to 13.98B and allowed to recover through distillation before its weights are pruned again to 8.36B. At Stage 2, the 14B checkpoint is the structural parent, while the corrected 23B model remains the source of teacher logits because it retains more general information.

5 Methodology

The methodology follows a three-stage ladder rather than a single compression step. Stage 0 corrects the 23B teacher through continued pretraining. Stage 1 prunes that teacher to approximately 14B parameters and distils the selected architecture. Stage 2 prunes the

distilled 14B model to approximately 8B parameters and again applies knowledge distillation. Figure 1 shows the dependency between these operations. The structural parent changes at Stage 2, but the source of teacher logits does not: both students imitate the corrected 23B model.

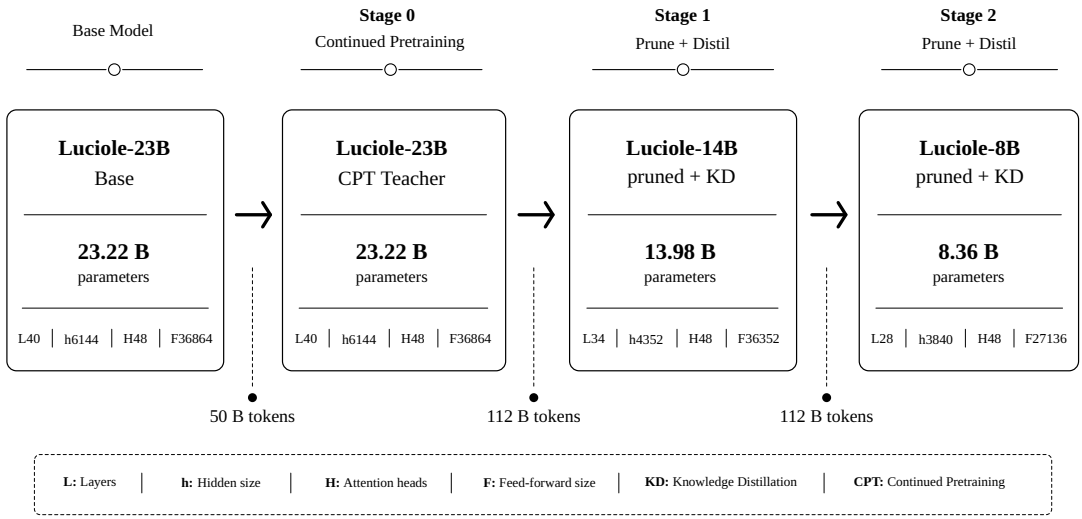

Figure 1: The three-stage compression pipeline. A general-purpose parent is first specialised on mathematics by continued pretraining, then compressed twice by Minitron-style structured pruning followed by knowledge distillation from that specialised parent.

The design depends on a single cleaned data pool. Reusing that pool makes the stages comparable and avoids attributing a change caused by data to a change in architecture. It also means that its composition can influence every part of the pipeline. The corpus must therefore be described before the training algorithms.

5.1 Corpus construction and cleaning

5.1.1 Starting pool

The pool combines sources from the Luciole training collection and NVIDIA Nemotron pretraining and post-training collections. It contains six semantic buckets: mathematics, general text, code, science, STEM, and instruction-following data. The starting tokenised material amounted to approximately 179.03B tokens. Sources ranged from web and encyclopedic text to mathematical solutions, scientific papers, code reasoning, and instruction examples. Licensing and source-level counts are reported in Appendix A.1 because data auditability is part of the project's premise.

Several general-domain sources were inherited from the third phase of Luciole pretraining. Retaining this material kept the adaptation distribution connected to the model's original training and was intended to mitigate catastrophic forgetting while the new Nemotron mathematical and reasoning sources increased target-domain density. Unlike full pretraining, this adaptation stage did not need to teach language from scratch and could therefore

devote a larger proportion of its limited budget to audited worked solutions, mathematical text, and code reasoning.

The first failed continued-pretraining run revealed that source labels and aggregate weights were insufficient descriptions of data quality. Two properties had to be measured at token level: whether documents would expose end-of-document markers within the training window, and whether nominally distinct documents were exact duplicates.

5.1.2 Document length and end-of-sequence starvation

Training examples are packed from tokenised documents into fixed-length sequences. If a document is much longer than the window, most windows cut through its interior and contain no end-of-document token. This matters for generation because the model learns when to terminate from those boundaries. In the initial 4096-token configuration, 74.1% of tokens belonged to documents longer than one training window and 51.8% of packed windows contained no end-of-sequence signal. Figure 2 shows how increasing the window reduces this problem and why one source remained pathological.

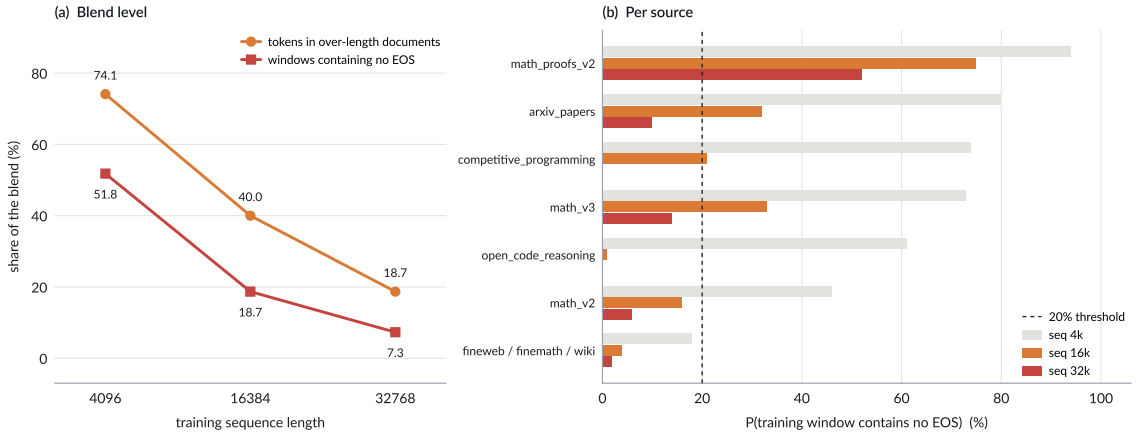

Figure 2: Document length starves the end-of-document signal. At a 4096-token training window, 74.1% of tokens sit in documents longer than the window and 51.8% of packed windows contain no EOS token at all, which is the precondition for a model that never learns to stop. Raising the window to 32,768 tokens takes the blend to 7.3%, but leaves math_proofs_v2 at 52%: its documents average about 65k tokens, so no reasonable window repairs it and it was dropped. arXiv was kept and length-capped, losing 31% of its tokens.

Figure 2 gives the source-level measurements that informed the cleaning policy. A 32,768-token sequence reduces the blend-wide no-EOS rate to 7.3%. The `math_proofs_v2` source remains at 52% because its documents average about 65,000 tokens. It contributed only about 5B tokens, so the source was removed rather than segmented using an untested semantic rule. The arXiv source was a more balanced case. At 32,768 tokens its no-EOS probability is 10%; it was retained but documents exceeding the cap were dropped, reducing 4.50B tokens to 3.11B.

5.1.3 Exact duplication and the natural blend

The second defect was exact duplication. Every complete token sequence was hashed globally, and repeated documents were removed across the entire corpus. This source-wide analysis differed sharply from a preliminary 50,000-document sample: 47% of the documents in NVIDIA Nemotron Math v2 (math_v2 in Figure 3) were exact duplicates, compared with about 10.6% in FineWeb-Edu. The preliminary sample underestimated the problem because the probability of selecting a matching pair was small even when repeated documents represented a substantial absolute volume. After applying both the 32,768-token length filter and exact deduplication, the datamix reduced to a 112B token cleaned corpus distributed across 145 shards, corresponding to an overall retention rate of 69.0%.

Figure 3 summarises both the source-level cleaning yield and the resulting semantic mixture. The training sampler used the cleaned token counts directly as sampling weights; no manual bucket multiplier was applied. The final blend contained 42.94% mathematics, 26.73% general text, 19.40% code, 5.20% science, 3.07% instruction data, and 2.66% STEM. This natural blend replaced the initial hand-specified mixture containing 64% mathematics. Disproportionate removal of duplicated mathematical content, together with the addition of French general-domain text, produced a math-focused but more balanced mixture with substantially greater general-text coverage.

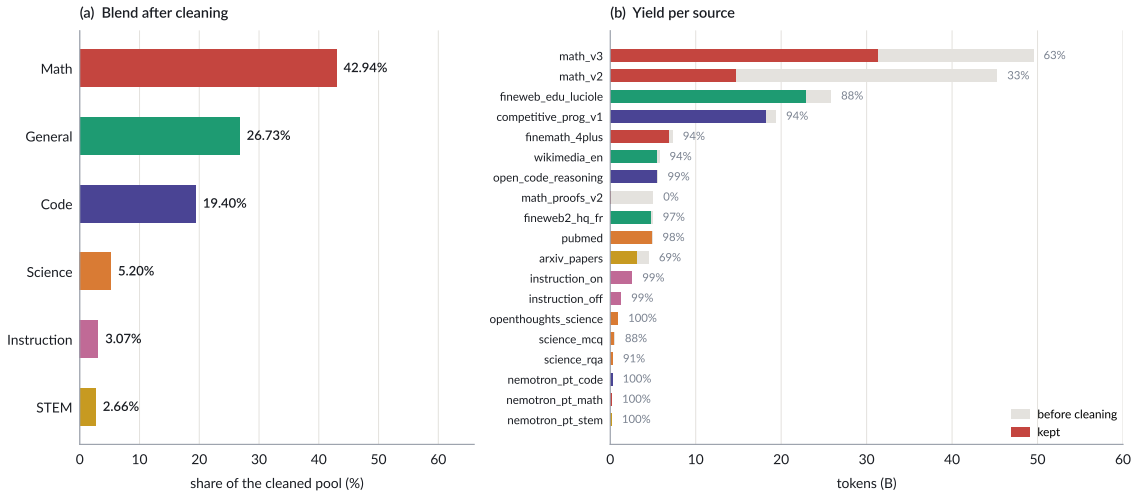

Figure 3: The rebuilt training corpus. Panel (a) shows the blend used for token-proportional sampling. In panel (b), each grey bar shows the source volume before cleaning, while the coloured overlay shows the volume retained after length filtering and exact deduplication; the annotation gives the retained percentage. `math_proofs_v2` was excluded entirely. No manual bucket re-weighting was applied.

5.2 Stage 0: teacher correction by continued pretraining

Structured compression transfers rather than creates capability. Since Luciole-23B-Base was weak on the target mathematics suite, it was first specialised by continued pretraining on the cleaned pool. “Teacher correction” refers to this project role; the operation itself

remains ordinary causal-language-model continued pretraining. The starting point was the context-extension checkpoint at step 11,919, and only model weights were restored. Optimiser moments were reinitialised because the base run was complete and the new stage used a changed data distribution.

Table 10 reports the production configuration. Each optimisation step processes 128 sequences of length 32,768, or 4,194,304 tokens. A total of 11,921 steps therefore corresponds to 50.0B tokens. The model uses a hybrid FP8 recipe for throughput, with the first and last four layers kept in BF16. The continued-pretraining stage cost approximately 4,900 H100-hours.

5.3 Stages 1 and 2: Minitron-style structured pruning

5.3.1 Calibration and importance

The pruning implementation uses the `mcore_minitron` mode of NVIDIA TensorRT-Model-Optimizer. Calibration consists of forward passes over 1,024 packed sequences of 4,096 tokens, drawn from a 20,000-document local copy of the mathematics split of the Nemotron post-training collection. No optimiser step is taken. The forward passes accumulate activation statistics used to rank attention heads, MLP channels, embedding channels, and complete layers.

The math-focused calibration is intentional. Earlier experiments used an MMLU-based set. TELL-TALE suggests that pruning decisions improve when calibration reflects the target task (Naim et al., 2025). The present design therefore asks the importance estimator to preserve responses that are active on mathematical reasoning data. This is an alignment of the proxy with the project objective, not proof that the retained channels have exclusively mathematical functions.

5.3.2 Parameter constraint and search grid

ModelOpt enumerates a discrete grid of layer count, hidden size, head count, and FFN size under a total-parameter constraint. Width reductions are capped at 40%, and depth reduction at 20% for each stage.

Table 1 summarises both searches. Stage 1 admitted 1,606 architectures under its budget. Stage 2 enumerated 2,436 grid points, of which 1,086 met the budget. In both cases only 50 candidates were assigned proxy scores. The stage-wise parameter ratios are close to 0.60, producing a total ratio of 0.36 from teacher to final student.

5.3.3 Selected architectures

The selected shapes are reported in Table 2 and visualised in Figure 4. Stage 1 selects candidate 4 with 34 layers, hidden size 4,352, and FFN size 36,352. Stage 2 selects candidate 20 with 28 layers, hidden size 3,840, and FFN size 27,136. Both keep all 48 attention heads

Item	Stage 1: 23B to 14B	Stage 2: 14B to 8B
Parent shape	L40, h6144, H48, F36864	L34, h4352, H48, F36352
Target parameters	$\sim 14B$	$\sim 8.4B$
Maximum width reduction	0.40	0.40
Maximum depth reduction	0.20	0.20
Architectures under budget	1,606	1,086
Proxy-scored candidates	50	50
Selected parameter ratio	0.603	0.598

Table 1: Structured-pruning search constraints and candidate counts.

and all 8 KV heads. The structural budget is paid through depth, hidden size, and FFN width rather than head count.

Model	Layers	Hidden	Heads	KV heads	FFN	Params
Corrected Luciole-23B	40	6,144	48	8	36,864	23.22B
Luciole-14B, candidate 4	34	4,352	48	8	36,352	13.98B
Luciole-8B, candidate 20	28	3,840	48	8	27,136	8.36B

Table 2: Architecture and parameter count along the compression ladder.

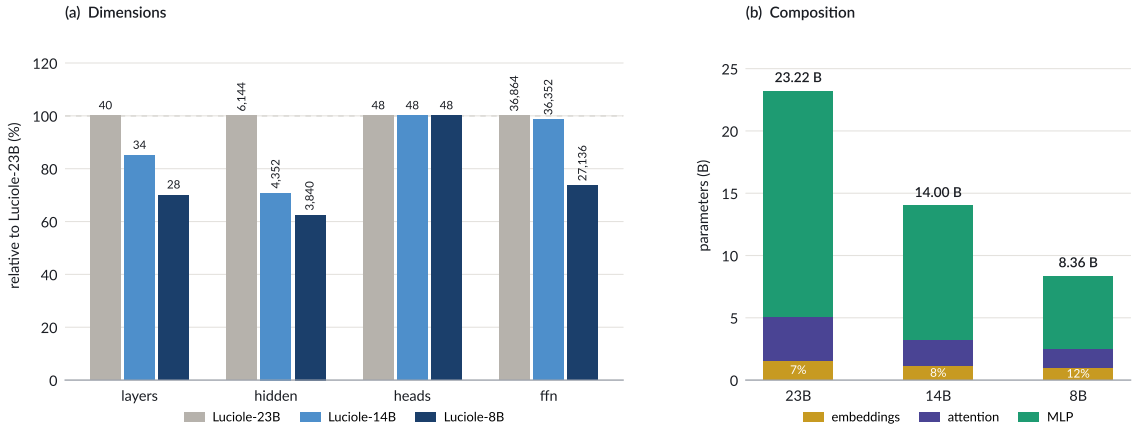

Figure 4: Architecture across the compression ladder. Head count is never reduced: both stages pay in depth, hidden size and feed-forward width, and the percentage labels in panel (b) show the untied embedding pair growing from 7% to 12% of the model, which is why hidden size carries outsized leverage at the second stage.

As the network shrinks, the untied embedding pair grows from about 7% to 12% of all parameters. Hidden-size reduction therefore has leverage both inside every block and in the two vocabulary matrices. This helps explain why a second width cut remains valuable even after Stage 1 has already reduced the hidden representation.

5.4 Knowledge distillation and candidate selection

5.4.1 Objective

Both stages use pure logit distillation with the forward KL objective introduced in Equation 1. If z_T and z_S are teacher and student logits at a token, the distributions are $p_T = \text{softmax}(z_T)$

and $p_S = \text{softmax}(z_S)$. The implemented loss is

$$\mathcal{L} = \mathcal{L}_{\text{KD}}, \qquad \lambda_{\text{KD}} = 1, \qquad \lambda_{\text{CE}} = 0. \tag{2}$$

No cross-entropy term on the corpus token is added. The student is therefore trained to reproduce the teacher's full output distribution on the cleaned pool. It may exceed the teacher on a finite benchmark because of capacity allocation, regularisation, or sampling variation, but the objective does not supply an external condition that corrects the teacher's token probabilities.

5.4.2 Selecting for recoverability

The published recipe does not resolve a practical issue encountered here: candidate scores immediately after pruning are too noisy to identify the architecture that will recover best. The selection protocol therefore adds a short distillation phase. A small set of candidates spanning the depth-width trade-off is exported, trained on about 412M to 1.2B tokens, and evaluated on two frozen 512-document sets. The mathematical set contains 402,642 tokens and comes from the held-out tail of the calibration source; the FineWeb-Edu set contains 449,062 tokens. Candidate perplexities are considered separately rather than averaged because the domains can prefer different shapes. A six-task general multiple-choice sweep breaks a close tie at Stage 2.

This protocol distinguishes pruning damage from recoverability. Step-0 perplexity measures how well inherited weights survive the architectural cut before learning. A short probe measures how effectively the new shape responds to the actual recovery objective. Only the selected winner receives the full 112B-token budget. The evidence that motivated this departure, including a case where the best step-0 candidate becomes fourth after 100 iterations, is presented in Sections 7.2 and 7.3.

6 Experimental Setup

This section describes the experimental setup used throughout the study. These choices were kept consistent across experiments wherever possible to ensure fair comparison between models and to provide a reproducible basis for assessing the effects of continued pretraining and structured compression.

6.1 Training configuration

The two distillation stages intentionally share the same optimisation recipe. Each runs for 26,703 iterations, or 112B tokens. The schedule uses a peak learning rate of 10^{-4}, a floor of 10^{-5}, 40 warm-up steps, and cosine decay, following the Minitron follow-up configuration (Sreenivas et al., 2024). Holding the budget and optimiser fixed makes differences between Stage 1 and Stage 2 easier to interpret as capacity and initialisation effects.

6.2 Compute and software environment

Production jobs ran on the H100 partition of the Jean-Zay supercomputer operated by IDRIS. Continued pretraining used the NeMo 2.3 series, while pruning and distillation used NVIDIA TensorRT-Model-Optimizer 0.42 and 0.43 with `mcore_minitron` pruning and `mtd` distillation. The corresponding stack included Megatron-Bridge 0.3.1, Megatron-Core 0.16.1, and Transformer Engine 2.12. Megatron supplies the tensor, pipeline, context, and data parallel training structure (Shoeybi et al., 2019; Narayanan et al., 2021); Transformer Engine supplies FP8 execution based on the formats described by Micikevicius et al. (2022).

Every production training job used 32 nodes with four H100 GPUs per node. Tensor parallelism was 4, pipeline parallelism 2, and context parallelism 2. The remaining factor produced data parallelism 8. Long jobs ran under the Jean-Zay `t3` quality of service, which limits a job to 20 hours. Training launchers therefore used chained Slurm arrays and detected the latest valid checkpoint on each restart. A completed task submitted its successor rather than assuming a single allocation would cover the run.

6.3 Evaluation protocol

All reported benchmark results were produced with the LM Evaluation Harness (Biderman et al., 2024). The checkpoints used a vLLM backend with tensor parallelism 4; vLLM's paged attention and continuous batching make large-scale generation efficient without changing the model probabilities (Kwon et al., 2023).

6.3.1 Perplexity controls

Candidate selection and convergence tracking use document-level cross-entropy on two frozen sets of 512 documents. The mathematics set contains 402,642 evaluated tokens; the FineWeb-Edu set contains 449,062. Given token losses ℓ_i, perplexity is

$$\mathrm{PPL} = \exp\left(\frac{1}{N}\sum_{i=1}^{N}\ell_i\right). \tag{3}$$

6.3.2 Comparison checkpoints and context limits

All comparison results use base checkpoints rather than instruction-tuned variants. The 8B to 9B class contains Gemma2-9B, Llama-3.1-8B, OLMo2-7B, EuroLLM-9B, Gaperon-8B, and Luciole-8B-Base. The 12B to 14B context includes Gemma3-12B, Mistral-Nemo-12B, and OLMo2-13B where relevant. This is an empirical comparison set, not an exhaustive ranking of available models.

7 Results and Analysis

The results follow the causal order of the pipeline. The first question is whether the teacher was improved without losing general knowledge. The next is whether pruning proxies

can select viable students. The analysis then turns to distillation dynamics and final mathematical, general, and French-language performance. Observations, interpretations, and proposed explanations are distinguished explicitly.

7.1 Continued pretraining corrects the teacher

The largest gains occur on the two hardest mathematical tasks: MINERVA-MATH rises by 7.68 points and MMLU-Pro mathematics by 8.29. GSM8K rises by 7.35 points, GSM8K-Platinum by 8.93, and GSM-Plus by 6.05. MMLU improves by 2.57 points. The observed result is therefore not a simple exchange of general accuracy for mathematics at Stage 0.

Figure 5 makes the progression easier to interpret. The generative GSM tasks fall sharply at 2.1B tokens, whereas MMLU stays close to its base level. By 8.4B tokens the generative scores have recovered and then continue upward. Because the disturbance is confined to generated-answer tasks, the supported interpretation is temporary adaptation in output style and answer formatting rather than broad loss of capability. This is not proven at the level of individual logits, but it fits both the benchmark split and the recovery. The failed 4,096-token run showed a superficially similar generation problem that never recovered because its numerical state was corrupt. The contrast justifies monitoring multiple task types rather than stopping at the first early regression.

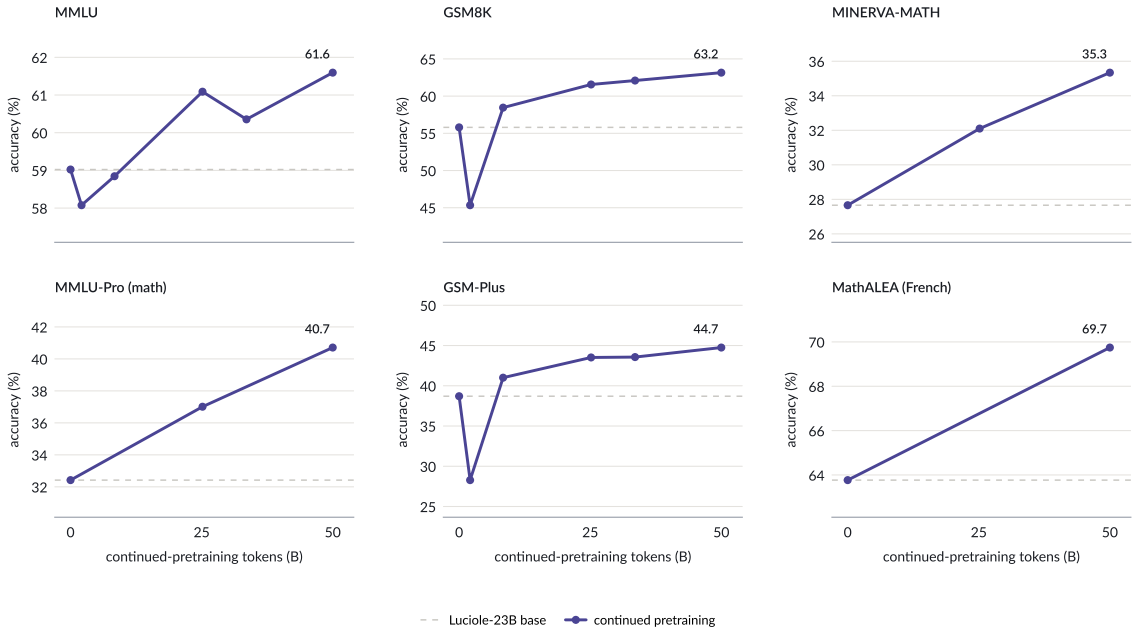

Figure 5: Teacher correction: six benchmarks across 50 B tokens of continued pretraining on the math-weighted corpus. The dashed line is the uncorrected Luciole-23B base. All six end above it, MMLU included (61.6 against 59.0), so specialisation did not cause an aggregate general-knowledge collapse.; the gains are largest on the two hardest math tasks, MINERVA-MATH at +7.7 and MMLU-Pro (math) at +8.3. The dip at 2.1 B tokens appears only on the generative GSM tasks and recovers by 8 B: it is an answer-format disturbance, not lost capability.

French MathALEA generation also rises by 5.98 points without a French-specific training stage. This is an observed cross-lingual transfer from a multilingual parent and a mixed-

language corpus. It does not establish that the same gain would appear on all French mathematical datasets, a limitation addressed by the broader French table in Section 7.6.

The corpus rebuild is part of this result. The realised mathematical share is not the result of a successful manual sweep. It emerges after exact duplication and pathological documents are removed. The successful continued-pretraining run therefore supports the combined configuration of data, sequence length, and validated numerics. It does not isolate the causal contribution of each cleaning decision because a full factorial ablation was not run.

7.2 Immediate pruning proxies do not select recoverable architectures

The first architecture-selection result is negative. At Stage 1, all 50 candidates score close to four-choice chance on the in-search probes. Mathematical proxy accuracy ranges from 23.1% to 27.9%, and MMLU from 22.7% to 26.5%, with standard errors of approximately 1.2 to 1.6 points. The corrected teacher scores 61.6 on the same MMLU probe. The candidate intervals overlap so strongly that their ordering cannot support a selection decision. Producing these scores consumed about eight hours of a 20-hour job.

Stage 2 is better conditioned. Candidates pruned from the already-distilled 14B model range from 23.5% to 40.0% on the math proxy and from 22.9% to 33.4% on MMLU. Figure 6 shows this separation. A plausible interpretation is that a converged distilled model degrades more gradually under an additional cut than a freshly specialised teacher. The later Stage 2 curves support that interpretation, but the project did not prune multiple independently distilled parents, so it remains a hypothesis about conditioning rather than a general law.

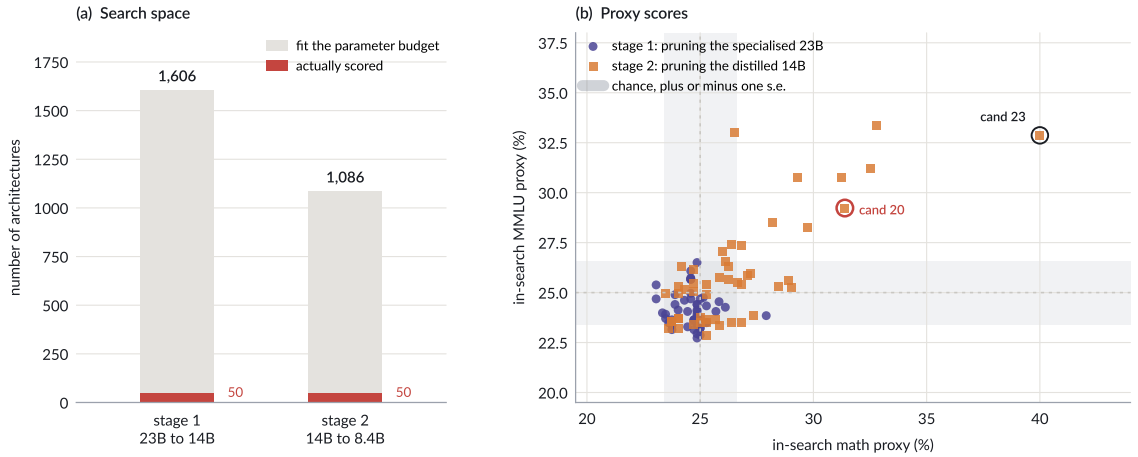

Figure 6: Pruned candidates cannot be ranked before distillation. ModelOpt enumerates every architecture that fits the parameter budget, sorts them by parameter count descending, and scores only the largest 50. All 50 stage-1 candidates fall inside the noise band around four-choice chance (math 23.1 to 27.9, MMLU 22.7 to 26.5, standard error 1.2 to 1.6 points), so the proxy carries no signal. Pruning an already-distilled parent at stage 2 does separate the candidates, reaching 40.0 on math, but the ordering still fails to predict the winner: cand 23 has the best proxy score and cand 20, at proxy rank 5 of 50, is the architecture that wins after full distillation.

Separation is still not enough. The eventual Stage 2 winner, candidate 20, is fifth of the 50 candidates by in-search score; candidate 23 is first and does not win the final tie-break. The proxy can therefore be non-degenerate and still fail to predict recoverability.

Stage 1 provides a sharper test because step-0 and step-100 perplexities are available for five diverse architectures. Candidate 5 has the best mathematical perplexity before distillation, 1,276, but reaches only 9.47 after 100 iterations. Candidate 4 starts in the middle at 2,660 and reaches 3.79. Its FineWeb perplexity of 39.0 is also much better than the next candidate's 138.8.

Figure 7 shows the reversal across both held-out domains. Selecting the least damaged step-0 candidate would have chosen candidate 5. Selecting the model that responds best to the distillation objective chooses candidate 4. The supported conclusion is narrow but operationally important: for these pruning ratios and this model, architecture selection requires at least a short distillation probe.

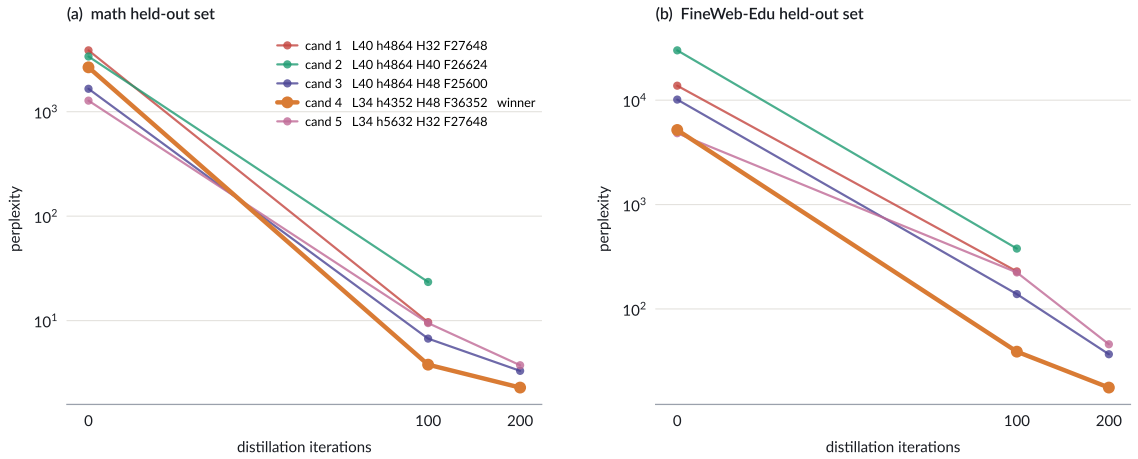

Figure 7: Stage-1 architecture selection. Iteration 0 is the pruned model before any distillation. Candidate 4, drawn heavy, is mid-pack at iteration 0 and wins by a factor of about 2 on math and 3.5 on general text by iteration 100. Candidate 5 is the mirror image: the best model before distillation and fourth of five after it. Step-0 perplexity therefore ranks pruning damage rather than recoverability, and selecting on it would have chosen the wrong architecture.

The Stage 2 search consequently disabled in-search scoring and used the saved time for short KD probes. This reduced the pruning job from roughly 15 hours to roughly two. The change is an engineering consequence of the negative result, not a claim that ModelOpt's proxy is universally uninformative at smaller compression ratios.

7.3 Stage 2 selection under conflicting held-out metrics

Six Stage 2 candidates received 300 distillation iterations, equivalent to about 1.26B tokens. The held-out domains ranked them in almost reverse order. Mathematical perplexity spans only 1.4760 to 1.4905, a 0.98% range; FineWeb-Edu spans 9.2941 to 9.5260, a 2.50% range. The math ordering is 23, 38, 25, 20, 19, 22, from best to worst. FineWeb prefers 20, 19, 38, 22, 25, 23. The Pareto set therefore contains candidate 23 at the mathematical end,

candidate 20 at the general end, and candidate 38 between them.

The opposing shapes are interpretable. Candidate 23 retains the 14B parent's hidden size of 4,352 and makes a larger FFN cut. Candidate 20 reduces hidden size to 3,840 and allocates more of the budget to FFN width. The mathematics held-out set prefers the first choice by a small margin, while the general set prefers the second. Since hidden size also controls the embedding matrices, the difference affects both token representations and every Transformer block.

Three hundred iterations were not sufficient for a stable mathematical rank. The math winner changed from candidate 23 to 20 and back to 23 at iterations 100, 200, and 300. FineWeb selected candidate 20 at every checkpoint. At iteration 300 the cosine schedule was still at 99.98% of its peak learning rate, so a sub-percent perplexity gap should not be treated as a converged difference.

Figure 8 shows the final tie-break. A six-task general multiple-choice sweep selected candidate 20 on every task: MMLU 50.88, HellaSwag 64.85, ARC-Challenge 45.56, WinoGrande 63.69, PIQA 74.27, and TruthfulQA 47.10. Its mean was 57.72, compared with 56.98 for the runner-up. General capability was used as the tie-break because Stage 1 had already shown mathematical recovery alongside a larger FineWeb deficit. This is a risk-protection choice based on earlier evidence, not an assertion that general accuracy should always override the target domain.

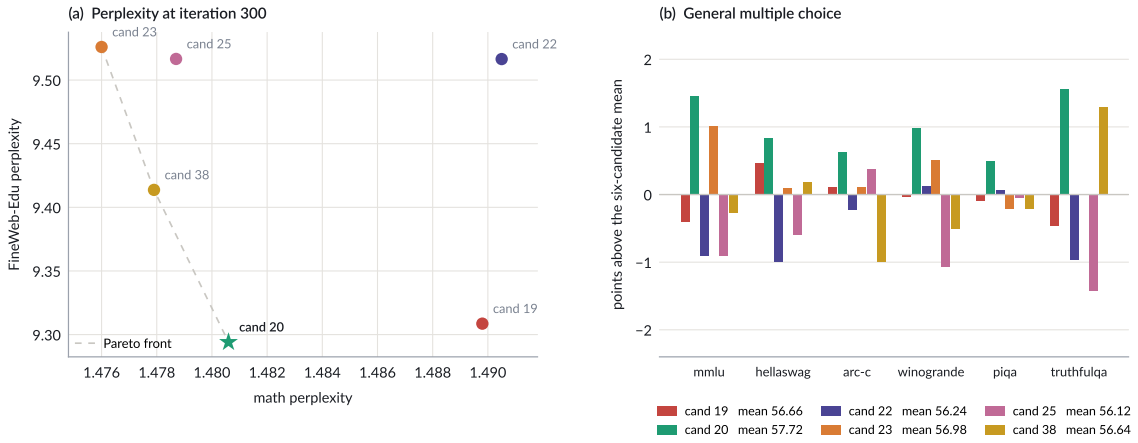

Figure 8: Stage-2 architecture selection. After 300 distillation iterations the two held-out perplexity buckets rank the six candidates in near-reverse order, and the spread is tiny: 0.98% on math and 2.50% on general text. The structural axis is interpretable, since the math winner leaves hidden size untouched while the general winner cuts it and spends the budget on the feed-forward width. Panel (b) breaks the deadlock: candidate 20 leads on every one of the six general tasks, a sweep rather than an averaging artefact. Bars show deviation from the six-candidate mean, because the absolute spread is under one point.

7.4 Distillation dynamics and domain asymmetry

7.4.1 Optimisation behaviour

Both full distillation runs completed 26,703 iterations with no NaN and no skipped step. Resume boundaries across roughly ten chained allocations are continuous. For example, the Stage 2 mean loss over 100 iterations was 0.1400 before one boundary and 0.1407 after it, consistent with restored optimiser and scheduler state. Gradient clipping is active mainly during the first approximately 20B tokens; the gradient norm later falls from about 0.22 to 0.05, well below the threshold of 1.0.

Figure 9 contains one panel per stage and must be read separately. Stage 1 begins at a KD loss of 7.34 because the 14B architecture is a severe structural cut of a teacher that has never trained in that shape. Its loss falls rapidly, then more gradually, and ends at 0.080. Stage 2 begins much lower, at 1.34, because the 8B student inherits weights from an already-distilled 14B parent. Its early optimisation is therefore better conditioned. It nevertheless ends higher, at 0.095. The same 23B teacher is 2.78 times larger than the final student, compared with 1.67 times larger than the Stage 1 student, so the endpoint ordering is consistent with a tighter capacity limit at 8B.

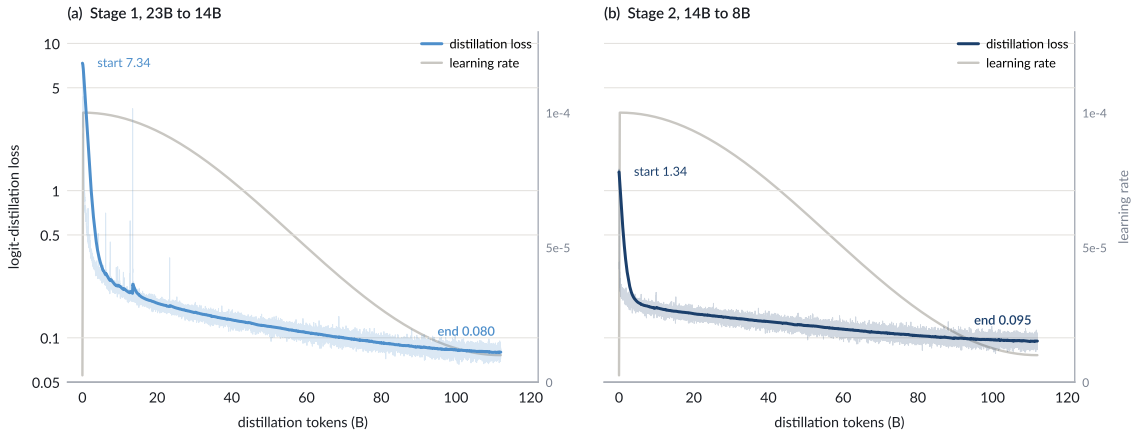

Figure 9: Optimisation over the 112 B-token distillation budget, one panel per stage on a shared logarithmic loss axis, with the cosine learning-rate schedule drawn on the right-hand axis. Both stages use an identical recipe: peak 1e-4, floor 1e-5, 40-step warm-up. Stage 2 starts 5.5× lower than stage 1 because its student was pruned from an already-distilled parent, yet ends higher at 0.095 against 0.080: matching the same 23B teacher from 8B is harder than from 14B, as the 2.78x against 1.67x capacity ratio predicts. Neither curve has flattened when the budget ends.

7.4.2 Held-out perplexity

Table 3 reports the endpoint perplexities. Both students are slightly better than the teacher on mathematics and substantially worse on FineWeb-Edu. The 14B model crosses below the teacher on math at about iteration 10,000, or 42B tokens, and ends 3.0% lower. The 8B model ends 2.3% lower. On general text, the 14B and 8B models are respectively 26.1% and 32.1% above the teacher.

Model	Math PPL	Difference	FineWeb PPL	Difference
Corrected Luciole-23B	1.4359	reference	5.6073	reference
Luciole-14B	**1.3922**	−3.0%	7.0696	+26.1%
Luciole-8B	**1.4024**	−2.3%	7.4067	+32.1%

Table 3: Held-out perplexity after each 112B-token distillation run. Relative differences use the corrected 23B teacher as reference.

Figure 10 show that the asymmetry begins in Stage 1 and is then inherited by Stage 2. The second compression adds only about six percentage points of relative FineWeb perplexity while reducing parameters by a further 40%. Stage 2 does not create the general-text gap from nothing.

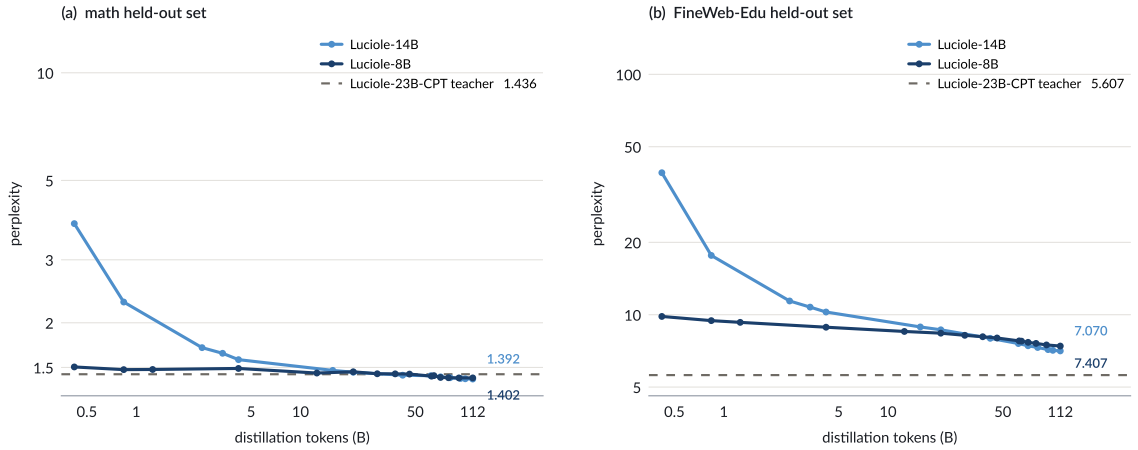

Figure 10: Perplexity over the distillation budget, measured on two frozen 512-document sets. Both students cross below the teacher on mathematics, the 14B around 42 B tokens and ending 3.0% better, and neither reaches it on general web text, ending 26% and 32% above.

The corpus offers a plausible mechanism. Mathematics accounts for 44.7% of the sampling distribution, so the distillation objective allocates nearly half of all tokens to matching the teacher in that domain. Once the 8B model reaches the teacher's math perplexity, remaining training continues to improve general text: after iteration 9,000, FineWeb perplexity falls monotonically from 8.10 to 7.41 while math moves by about 0.04. The result supports a capacity-allocation interpretation. Calling the mixture the sole cause would be too strong because no alternative mixture was run at matched budget.

7.4.3 Benchmark recovery over training

Downstream benchmarks follow the same domain split. Figure 11 aggregates the five mathematical tasks and six general tasks across training. The 14B model ends at 50.78 on mathematics and 63.87 on general evaluation; the 8B model ends at 49.79 and 61.71; the teacher scores 50.17 and 66.53. The 8B curve begins much higher than the 14B curve at the same token count. At iteration 100, its MMLU is 50.74 while the Stage 1 student is near chance at 23.00. At iteration 1,000, its GSM8K score is 55.27 compared with 38.44 for

Stage 1. This is the downstream counterpart of the better-conditioned Stage 2 proxy scores.

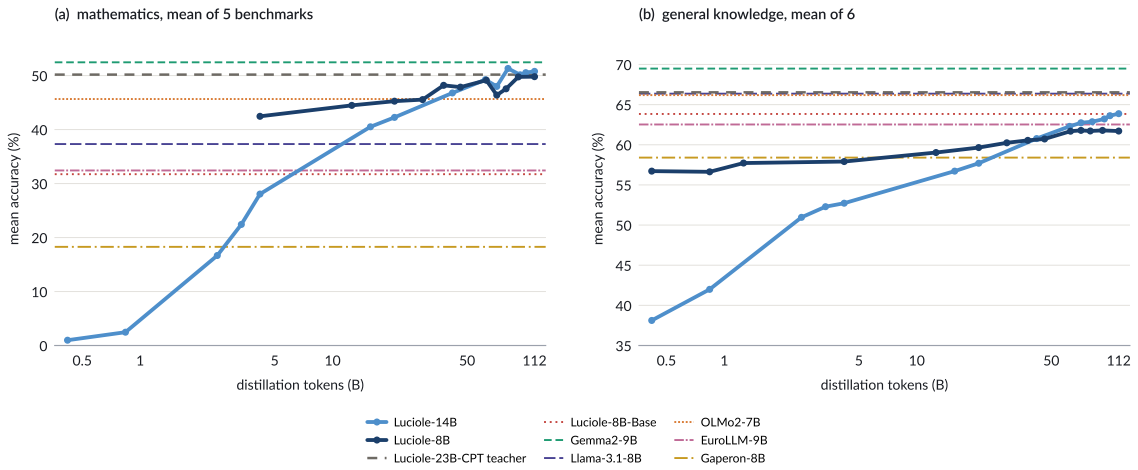

Figure 11: Averaged ability across the distillation budget. Mathematics is recovered from near zero and pushed past the teacher, ending at 50.8 for the 14B and 49.8 for the 8B against the teacher's 50.2, while general knowledge rises steadily and stops short of it at 63.9 and 61.7 against 66.5. The 8B curve begins far above the 14B's at matched token counts because it was pruned from an already-distilled parent.

7.5 Final mathematical and general results

Figure 12 first summarises mathematical accuracy against parameter count. After teacher correction, the Luciole trajectory remains almost horizontal: the 14B and 8B students stay close to the corrected 23B score, whereas the independently trained Luciole-8B-Base lies far below it. Gemma2-9B has the highest aggregate among the shown 8B to 9B comparison models, so the figure supports strong capability transfer within the Luciole family rather than universal leadership.

Figure 13 compares performance across the full Luciole compression ladder. The largest improvement occurs after continued pretraining, while the subsequent 14B and 8B compression stages retain most of these gains across the five mathematical benchmarks.

Table 4 gives the complete mathematical comparison. The main same-family result is consistent across every task: Luciole-8B exceeds Luciole-8B-Base by 21.53 points on GSM8K, 22.33 on GSM8K-Platinum, 17.40 on GSM-Plus, 12.70 on MINERVA-MATH, and 16.35 on MMLU-Pro mathematics. The unweighted mean rises from 31.73 to 49.79, a 57% relative improvement.

Model	Params	GSM8K	GSM8K-Plat.	GSM-Plus	MINERVA	MMLU-Pro math	Math Avg.
Luciole-23B-Base	23.22B	55.80	57.98	38.70	27.66	32.42	42.51
Corrected Luciole-23B	23.22B	63.15	66.91	44.75	35.34	40.71	50.17
Luciole-14B	13.98B	63.91	65.59	44.09	37.06	43.23	**50.78**
Luciole-8B	8.36B	62.24	64.68	43.03	35.50	43.52	**49.79**
Luciole-8B-Base	8.00B	40.71	42.35	25.63	22.80	27.17	31.73
Gemma2-9B	9.20B	68.39	72.04	48.14	33.72	39.97	**52.45**
Llama-3.1-8B	8.00B	50.95	53.76	31.79	19.20	30.87	37.31
OLMo2-7B	7.30B	68.16	70.14	47.18	18.80	23.98	45.65
EuroLLM-9B	9.20B	47.76	47.64	30.35	12.88	23.46	32.42
Gaperon-8B	8.00B	26.61	28.62	15.24	5.80	15.10	18.27

Table 4: Mathematical benchmark results. Generation is five-shot except MINERVA-MATH, which uses its canonical four-shot prompt. Scores are percentages.

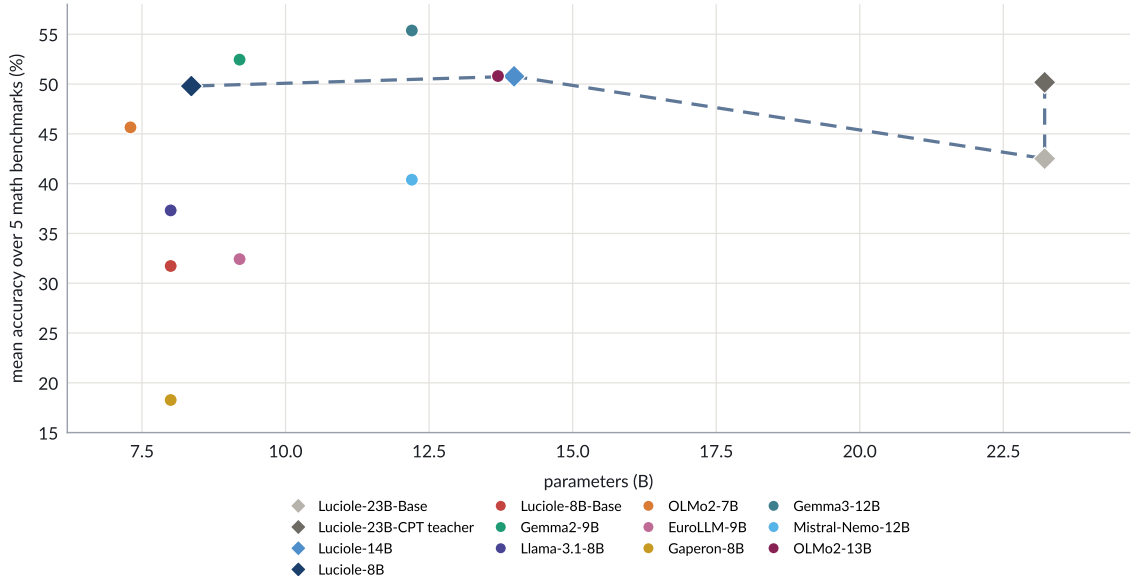

Figure 12: Mathematical ability against parameter count. The dashed path joins the four Luciole pipeline checkpoints; circles are independently trained comparison models. The pipeline remains almost flat from the corrected 23B teacher through the 14B and 8B students, unlike the size-matched Luciole-8B-Base.

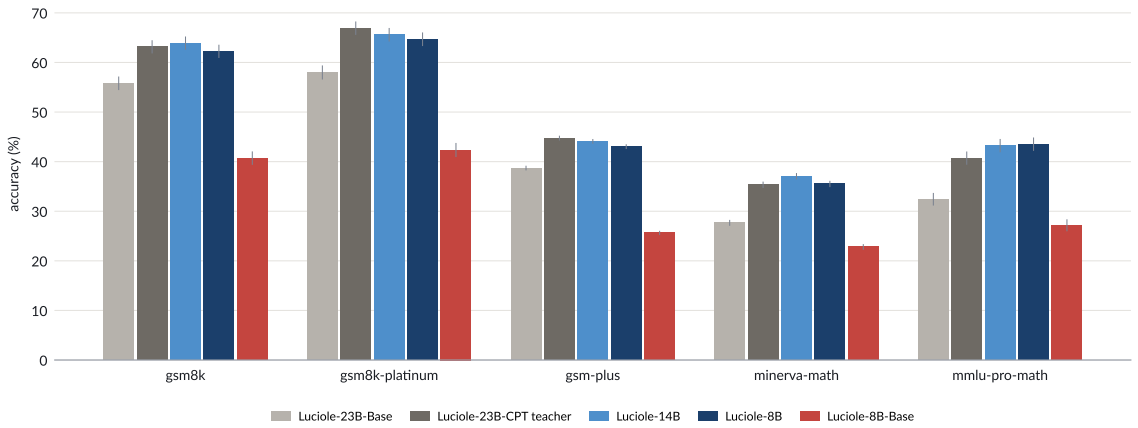

Figure 13: Performance across the Luciole model ladder on five mathematical benchmarks. Results are shown for the corrected 23B model after continued pretraining and for the subsequent 14B and 8B compressed models.

Figure 14 focuses on the 8B to 9B class. Gemma2-9B has the highest five-task mean, 52.45, and is about 10% larger than the final Luciole student. Luciole-8B leads the shown class on MINERVA-MATH and MMLU-Pro mathematics, while Gemma2 leads on the three GSM-family tasks. The result suggests that the specialised corpus is especially effective on harder competition and advanced multiple-choice mathematics. This is an interpretation of the task pattern, not a causal corpus ablation.

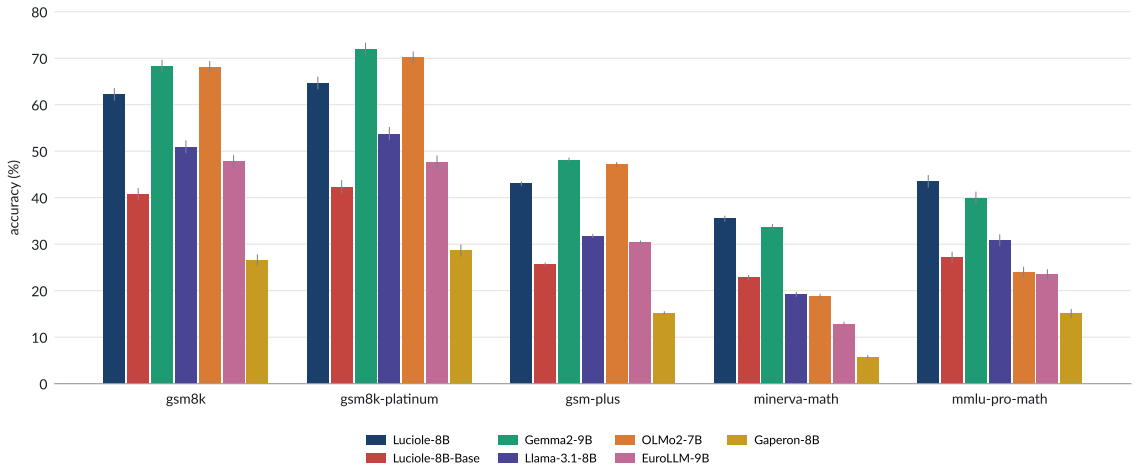

Figure 14: Mathematical reasoning in the 8B to 9B parameter class. Scores are exact-match accuracies from five-shot generation.

General results are listed in Table 5. The compressed model remains competitive in its class but falls behind its own teacher on every reported general task except the TruthfulQA comparison between some intermediate checkpoints. From teacher to 8B, ARC-Challenge loses 6.99 points, HellaSwag 6.31, WinoGrande 6.47, PIQA 2.89, MMLU 4.99, and TruthfulQA 1.28. The mean falls from 66.53 to 61.71.

Model	MMLU	ARC-Challenge	HellaSwag	WinoGrande	PIQA	TruthfulQA	General Avg.
Luciole-23B-Base	59.02	61.52	78.07	69.06	80.47	50.44	66.43
Corrected Luciole-23B	61.59	58.36	78.16	72.69	81.18	47.20	66.53
Luciole-14B	58.10	54.35	73.70	69.46	78.62	49.02	63.87
Luciole-8B	56.60	51.37	71.85	66.22	78.29	45.92	61.71
Luciole-8B-Base	58.20	57.08	75.55	70.48	79.38	42.32	63.84
Gemma2-9B	69.01	65.78	79.83	73.80	83.03	45.47	69.49
Llama-3.1-8B	64.32	54.86	79.42	74.43	80.96	44.20	66.36
OLMo2-7B	60.49	57.25	80.50	74.43	81.12	43.30	66.18
EuroLLM-9B	55.08	45.99	76.98	69.53	79.16	48.42	62.53
Gaperon-8B	47.04	50.68	71.97	64.64	77.09	38.99	58.40

Table 5: General multiple-choice accuracy. Scores are percentages.

Figure 15 focuses on general-language performance in the 8B to 9B class. Gemma2-9B achieves the highest general average at 69.49, while Luciole-8B reaches 61.71. The compressed Luciole model remains strongest relative to its peers on TruthfulQA, where it outperforms Gemma2-9B, Llama-3.1-8B, OLMo2-7B, and Gaperon-8B, but trails EuroLLM-9B. Its largest relative weakness is MMLU, consistent with the broader decline in general capabilities observed during compression. Overall, the results indicate that preserving the mathematical gains of continued pretraining comes with a measurable trade-off in general benchmark performance.

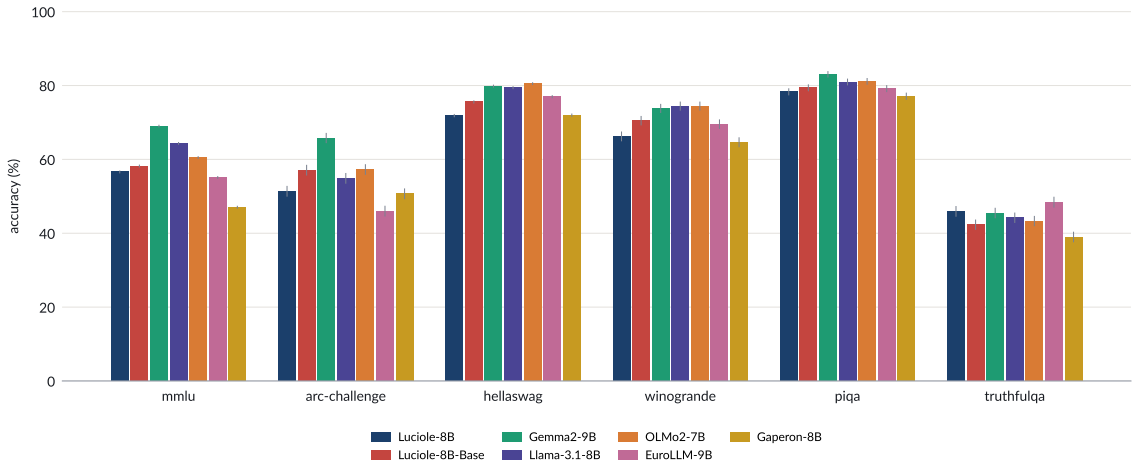

Figure 15: General multiple-choice benchmarks for models in the 8B to 9B class. These likelihood-based scores do not depend on answer formatting. The compressed Luciole-8B remains competitive but its largest deficit is on MMLU.

The retention view in Figure 16 isolates the compression result from absolute benchmark difficulty. Luciole-14B has 60% of the teacher's parameters and reaches 101.2% of its mathematical aggregate and 96.0% of its general aggregate. Luciole-8B has 36% of the parameters and retains 99.2% and 92.8%, respectively. Luciole-8B-Base has a similar parameter share and retains only 63.2% of teacher mathematics. Its general retention is 96.0%, which is higher than the compressed 8B. The comparison therefore demonstrates target-domain transfer, not uniformly superior compression.

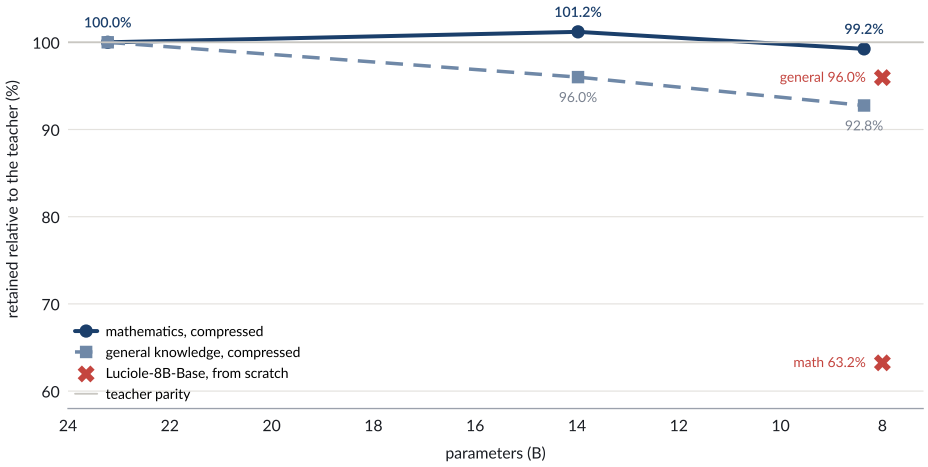

Figure 16: Capability retained through compression relative to the corrected 23B teacher. At 36% of the teacher's parameters, the final 8B student retains 99.2% of its mean mathematical accuracy and 92.8% of its mean general accuracy. The size-matched Luciole-8B-Base retains only 63.2% of the teacher's mathematics, isolating the benefit of specialisation followed by distillation.

7.6 French-language mathematics

Table 6 evaluates whether the English-heavy mathematical gains transfer to French. The strongest result is MathALEA generation: Luciole-8B scores 73.38, compared with 69.75

for its 23B teacher, 72.25 for Gemma2-9B, and 52.67 for Luciole-8B-Base. The 20.71-point same-family gain speaks directly to the secondary-school education use case. French MGSM rises from 14.80 for Luciole-8B-Base to 28.00 for the compressed model. These figures confirm that the generative improvement is not merely a lucky answer string.

Model	MathALEA MCQ	MathALEA gen.	Exo7 gen.	MGSM-fr
Luciole-23B-Base	42.03	63.77	14.59	18.40
Corrected Luciole-23B	35.42	69.75	23.47	29.60
Luciole-14B	44.97	71.07	23.47	30.80
Luciole-8B	41.55	**73.38**	16.07	28.00
Luciole-8B-Base	40.50	52.67	10.68	14.80
Gemma2-9B	49.14	72.25	25.48	22.00
Llama-3.1-8B	40.66	51.15	17.12	18.00
OLMo2-7B	34.89	38.05	5.81	10.40
EuroLLM-9B	35.97	43.71	11.10	36.00
Gaperon-8B	29.89	30.11	6.03	20.00

Table 6: French-language mathematical evaluation. Scores are percentages.

Figure 17 compares French-language mathematical performance across the 8B to 9B class. Luciole-8B achieves the highest generative MathALEA score at 73.38 and substantially outperforms Luciole-8B-Base, while remaining competitive on the other French benchmarks. The results suggest that the adapted Luciole model preserves a clear advantage on French mathematical generation despite compression.

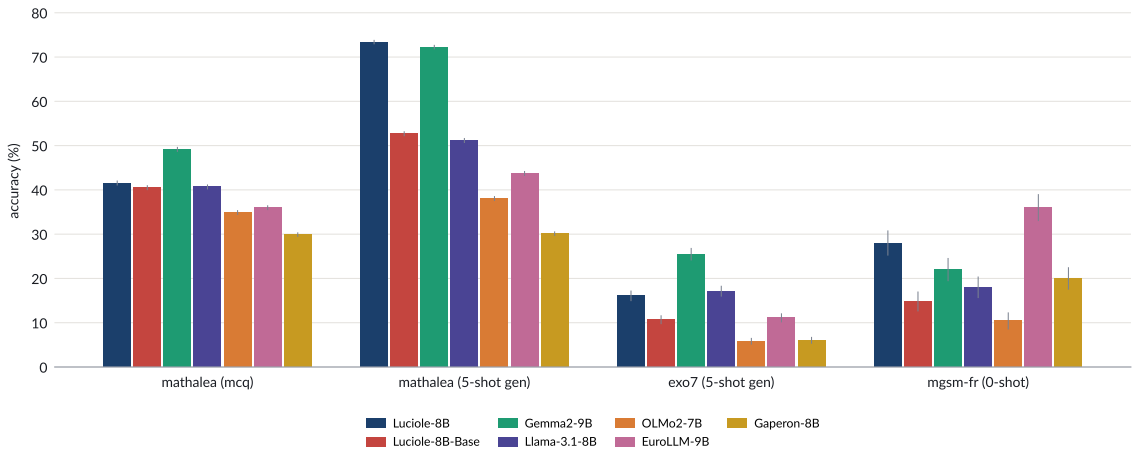

Figure 17: French-language mathematics. MathALEA covers secondary-school mathematics and Exo7 undergraduate exercises. On generative MathALEA, the compressed Luciole-8B scores 73.4, above its own 23B teacher and 20.7 points above the size-matched Luciole-8B-Base.

7.7 Training cost and result summary

Table 7 reports measured production cost. Architecture searches together contribute less than one percent of the total and are absorbed in the rounded total. The complete project uses 274B training tokens and approximately 30,500 H100-hours.

Stage	Tokens	Iterations	H100-hours
Continued pretraining	50B	11,921	4,900
Stage 1 distillation	112B	26,703	14,150
Stage 2 distillation	112B	26,703	11,450
Total	274B		30,500

Table 7: Estimated production compute cost.

Four findings answer the objectives from Section 3. First, correcting the teacher before compression works: all endpoint teacher benchmarks, including MMLU, exceed the base. Second, structured pruning followed by full distillation retains the specialised capability exceptionally well, reaching 99.2% mathematical retention at 36% of the parameters. Third, the result is domain-specific rather than universal: general retention is 92.8%, and final MMLU is 1.6 points below Luciole-8B-Base. Fourth, architecture recovery cannot be predicted reliably from the untrained pruned model; a short distillation probe supplies the selection signal that the immediate proxy lacks.

These observations motivate a future hypotheses. A more general-heavy final segment of distillation may reduce the FineWeb and MMLU gap without erasing the already-recovered mathematics. This claim requires matched-budget experiments and is treated as future work in Section 8.

8 Conclusion

This internship asked whether an existing open and French-oriented model family could yield a small mathematical model without another full pretraining run. The answer is positive, with an important qualification. Continued pretraining first turns Luciole-23B into a stronger mathematical teacher. Two Minitron-style structured pruning stages and two full knowledge-distillation runs then produce 13.98B and 8.36B students. The final model retains 99.2% of the corrected teacher's mean mathematical accuracy at 36% of its parameters. It does not retain general knowledge to the same degree, reaching 92.8% of the teacher's general aggregate and trailing the from-scratch Luciole-8B-Base.

8.1 Assessment against the project objectives

The mathematical objective is met. Luciole-8B reaches 49.79 across five English mathematical benchmarks, compared with 31.73 for Luciole-8B-Base. It improves every task in the same-family comparison and gains about 57% in relative mean accuracy. The strongest French result is consistent with the intended educational application: 73.38 on five-shot generative MathALEA, 20.71 points above Luciole-8B-Base and above the corrected 23B teacher. The intermediate 14B model is also a useful output in its own right, reaching 50.78 on the mathematical aggregate, slightly above the teacher.

The teacher-correction objective is also met. After 50B tokens of continued pretraining, every measured endpoint exceeds Luciole-23B-Base. Gains are 7.68 points on MINERVA-MATH, 8.29 on MMLU-Pro mathematics, and 2.57 on MMLU. This result is important because it shows that the final compression did not merely concentrate an unchanged teacher. It transferred a capability that had first been deliberately strengthened.

General retention was intended to be sufficient to avoid catastrophic forgetting rather than to preserve the teacher's full general capability. The observed decline is therefore expected under compression: the mean falls from 66.53 to 61.71, while MMLU decreases from 61.59 to 56.60. Importantly, the final model remains broadly functional across general multiple-choice tasks, indicating that compression does not erase the teacher's general knowledge. Subject-level results further show that the loss is uneven, with gains over Luciole-8B-Base on STEM but larger reductions in other knowledge, social sciences, and humanities. The higher FineWeb perplexity of the 8B student is consistent with this controlled reduction in general modelling capacity. Overall, the results indicate partial but sufficient retention, with no evidence of catastrophic forgetting.

The efficiency and engineering objectives are met within the evidence available. The training pipeline consumes 274B tokens and approximately 30,500 H100-hours. Every long run is resumable across 20-hour Slurm allocations, data order is derived from explicit manifests, compute execution is offline, and candidate comparisons use frozen held-out sets. Two compressed checkpoints, 112B-token pool, conversion tools, and benchmark tasks are durable outputs.

8.2 Limitations

The experimental path is deep but not broad. Each stage continues one selected architecture for one 112B-token budget. There is no matched ablation of the 0.60 stage-wise parameter ratio, learning-rate schedule, cross-entropy weight, or total distillation budget. The conclusion that the chosen candidate is recoverable is well supported relative to the tested short-probe candidates; it does not establish a globally optimal architecture across all 1,606 or 1,086 feasible shapes.

The final checkpoints are base models. They have not received instruction tuning, preference optimisation, safety post-training, or product-specific evaluation. Their benchmark reasoning does not make them ready-made tutoring assistants. French MathALEA is encouraging, but Exo7 shows that undergraduate multi-answer problems remain fragile. A deployment claim would require interactive evaluation, error analysis, calibration, and safeguards beyond the scope of this internship.

Third-party comparisons also carry protocol limitations. Several base models use a 4,096-token context during evaluation, which can truncate MINERVA demonstrations.

8.3 Personal assessment

This internship gave me the opportunity to work on a complete language-model research pipeline, from corpus construction and large-scale training to model compression, checkpoint conversion, and evaluation. The project required me to connect theoretical ideas such as continued pretraining, structured pruning, and knowledge distillation with the practical constraints of distributed training. I developed a much stronger understanding of how decisions concerning data composition, architecture, numerical precision, and evaluation protocols interact in a large experiment.

The most important lesson came from investigating failures. The first continued-pretraining experiments initially appeared to demonstrate catastrophic forgetting, but closer analysis revealed several overlapping causes, including corpus imbalance, sequence fragmentation, and a numerical fault in the training and checkpoint-conversion path. This experience taught me that a benchmark regression should not be interpreted as a scientific result until the numerical state, data pipeline, and evaluation protocol have been independently validated. It also led me to adopt a more systematic methodology based on control checkpoints, frozen held-out sets, strict checkpoint loading, explicit manifests, and evaluations at intermediate training stages.

The scale of the experiments also strengthened my engineering and project-management skills. Long runs had to be organised around 20-hour Slurm allocations, made safely resumable, and monitored without unnecessary use of compute. Working within the LINAGORA Labs AI team showed me the importance of shared tools, reproducible evaluation practices, and technical discussion in resolving problems that cross the boundaries between research and systems engineering. If I were to repeat the project, I would define the ablation plan and validation gates earlier, before committing to the longest runs. Nevertheless, the internship achieved its central objective and produced both useful checkpoints and a reusable experimental pipeline.

8.4 Perspectives

The most immediate research direction is to reduce the loss of general capability during distillation. The final student preserves almost all of the corrected teacher's mathematical performance, but its FineWeb perplexity and general benchmark scores show that this specialisation comes at a measurable cost. A matched-budget experiment could allocate the final part of distillation to a mixture containing more general text, while keeping the total number of training tokens unchanged. Varying the corpus composition and introducing a nonzero next-token cross-entropy term would help determine whether general knowledge can be recovered without erasing the mathematical gains. These experiments should retain the same frozen evaluation sets so that any improvement can be attributed to the training change.

A second direction concerns the breadth of the architecture search. The present work follows a fixed stage-wise parameter ratio of approximately 0.60 and fully distils one selected

architecture at each stage. Future experiments could compare different compression ratios, one-step and two-step compression, alternative allocations between depth and width, and longer selection probes for a small set of structurally diverse candidates. The results already show that immediate post-pruning accuracy is a poor predictor of final recoverability. Studying the relationship between early distillation loss, held-out perplexity, general-task accuracy, and endpoint performance could therefore lead to a more reliable and less expensive candidate-selection procedure.

Finally, the resulting checkpoints remain base models rather than deployable educational assistants. Instruction tuning, preference optimisation, and safety post-training would be required before considering practical use. Evaluation should then move beyond static benchmarks to interactive French mathematical tutoring, including error analysis, answer calibration, termination behaviour, and robustness to ambiguous or multi-answer exercises such as those in Exo7. Deployment-oriented work should also measure inference latency, memory consumption, and energy use, and investigate whether quantisation can provide further efficiency gains without degrading reasoning. Releasing the checkpoints, corpus manifests, training scripts, and evaluation tasks would make these extensions reproducible and would allow the same specialisation-before-compression strategy to be tested on other languages and domains.

References

Stella Biderman, Hailey Schoelkopf, Lintang Sutawika, Leo Gao, Jonathan Tow, Baber Abbasi, Alham Fikri Aji, Pawan Sasanka Ammanamanchi, Sidney Black, Jordan Clive, Anthony DiPofi, Julen Etxaniz, Benjamin Fattori, Jessica Zosa Forde, Charles Foster, Jeffrey Hsu, Mimansa Jaiswal, Wilson Y. Lee, Haonan Li, Charles Lovering, Niklas Muennighoff, Ellie Pavlick, Jason Phang, Aviya Skowron, Samson Tan, Xiangru Tang, Kevin A. Wang, Genta Indra Winata, François Yvon, and Andy Zou. Lessons from the trenches on reproducible evaluation of language models, 2024. URL https://arxiv.org/abs/2405.14782.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language, 2019. URL https://arxiv.org/abs/1911.11641.

CEA-LIST IA. Exo7MCQ. Hugging Face dataset, 2026a. URL https://huggingface.co/datasets/cea-list-ia/Exo7MCQ. Accessed 20 August 2026.

CEA-LIST IA. MathAleaMCQ. Hugging Face dataset, 2026b. URL https://huggingface.co/datasets/cea-list-ia/MathAleaMCQ. Accessed 20 August 2026.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018. URL https://arxiv.org/abs/1803.05457.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot, 2023. URL https://arxiv.org/abs/2301.00774.

Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A. Roberts. The unreasonable ineffectiveness of the deeper layers, 2024. URL https://arxiv.org/abs/2403.17887.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, 2020. doi: 10.18653/v1/2020.acl-main.740. URL https://aclanthology.org/2020.acl-main.740/.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021a. URL https://arxiv.org/abs/2009.03300.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021b. URL https://arxiv.org/abs/2103.03874.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. URL https://arxiv.org/abs/1503.02531.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation, 2016. URL https://arxiv.org/abs/1606.07947.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626, 2023. doi: 10.1145/3600006.3613165. URL https://arxiv.org/abs/2309.06180.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models, 2022. URL https://arxiv.org/abs/2206.14858.

Qintong Li, Leyang Cui, Xueliang Zhao, Lingpeng Kong, and Wei Bi. Gsm-plus: A comprehensive benchmark for evaluating the robustness of llms as mathematical problem solvers, 2024. URL https://arxiv.org/abs/2402.19255.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods, 2021. URL https://arxiv.org/abs/2109.07958.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models, 2023. URL https://arxiv.org/abs/2305.11627.

Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. Shortgpt: Layers in large language models are more redundant than you expect, 2024. URL https://arxiv.org/abs/2403.03853.

Paulius Micikevicius, Dusan Stosic, Neil Burgess, Marius Cornea, Pradeep Dubey, Richard Grisenthwaite, Sangwon Ha, Alexander Heinecke, Patrick Judd, John Kamalu, Naveen Mellempudi, Stuart Oberman, Mohammad Shoeybi, Michael Siu, and Hao Wu. Fp8 formats for deep learning, 2022. URL https://arxiv.org/abs/2209.05433.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. Compact language models via pruning and knowledge distillation, 2024. URL https://arxiv.org/abs/2407.14679.

Omar Naim, Krish Sharma, Niyar R Barman, and Nicholas Asher. Tell-tale: Task efficient llms with task aware layer elimination, 2025. URL https://arxiv.org/abs/2510.22767.

Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, Amar Phanishayee, and Matei Zaharia. Efficient large-scale language model training on gpu clusters using megatron-lm. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2021. doi: 10.1145/3458817.3476209. URL https://arxiv.org/abs/2104.04473.

OpenLLM-France. Luciole LLM - A OpenLLM-France Collection, 2026. URL https://huggingface.co/collections/OpenLLM-France/luciole-llm.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale, 2019. URL https://arxiv.org/abs/1907.10641.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. Language models are multilingual chain-of-thought reasoners, 2022. URL https://arxiv.org/abs/2210.03057.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism, 2019. URL https://arxiv.org/abs/1909.08053.

Sharath Turuvekere Sreenivas, Saurav Muralidharan, Raviraj Joshi, Marcin Chochowski, Ameya Sunil Mahabaleshwarkar, Gerald Shen, Jiaqi Zeng, Zijia Chen, Yoshi Suhara, Shizhe Diao, Chenhan Yu, Wei-Chun Chen, Hayley Ross, Oluwatobi Olabiyi, Ashwath Aithal, Oleksii Kuchaiev, Daniel Korzekwa, Pavlo Molchanov, Mostofa Patwary, Mohammad Shoeybi, Jan Kautz, and Bryan Catanzaro. Llm pruning and distillation in practice: The minitron approach, 2024. URL https://arxiv.org/abs/2408.11796.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. A simple and effective pruning approach for large language models, 2023. URL https://arxiv.org/abs/2306.11695.

Joshua Vendrow, Edward Vendrow, Sara Beery, and Aleksander Madry. Do large language model benchmarks test reliability?, 2025. URL https://arxiv.org/abs/2502.03461.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark, 2024. URL https://arxiv.org/abs/2406.01574.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2022. URL https://arxiv.org/abs/2201.11903.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. Sheared llama: Accelerating language model pre-training via structured pruning, 2023. URL https://arxiv.org/abs/2310.06694.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence?, 2019. URL https://arxiv.org/abs/1905.07830.

A Reproducibility Material

This appendix records data sources, benchmark breakdowns and candidate manifests.

A.1 Datamix sources

Table 8 summarises the sources used to construct the training corpus, grouped by data category and accompanied by their respective licences.

Bucket	Source or subset	Licence
Math	NVIDIA *Nemotron-SFT-Math-v3*	CC BY-SA 4.0
Math	NVIDIA *Nemotron-Math-v2*	CC BY 4.0
Math	OpenLLM-France *Luciole Training Dataset* (FineMath-4plus subset)	CC BY-SA 4.0
Math	NVIDIA *Nemotron-Post-Training-Dataset-v1* (mathematics subset)	CC BY-SA 4.0
Math	NVIDIA *Nemotron-Math-Proofs-v2*	CC BY-SA 4.0
General	OpenLLM-France *Luciole Training Dataset* (FineWeb-Edu subset)	CC BY-SA 4.0
General	OpenLLM-France *Luciole Training Dataset* (English Wikimedia subset)	CC BY-SA 4.0
General	OpenLLM-France *Luciole Training Dataset* (FineWeb2-HQ `fra_Latn` subset)	CC BY-SA 4.0
Code	NVIDIA *Nemotron-Competitive-Programming-v1*	CC BY 4.0
Code	NVIDIA *OpenCodeReasoning*	CC BY-SA 4.0
Code	NVIDIA *Nemotron-Post-Training-Dataset-v1* (code subset)	CC BY-SA 4.0
STEM	OpenLLM-France *Luciole Training Dataset* (arXiv papers subset)	CC BY-SA 4.0
STEM	NVIDIA *Nemotron-Post-Training-Dataset-v1* (STEM subset)	CC BY-SA 4.0
Science	OpenLLM-France *Luciole Training Dataset* (PubMed subset)	CC BY-SA 4.0
Science	OpenThoughts *OpenThoughts3-1.2M* (science subset)	CC BY-SA 4.0
Science	NVIDIA *Nemotron-Science-v1* (MCQA subset)	CC BY 4.0
Science	NVIDIA *Nemotron-Science-v1* (RQA subset)	CC BY 4.0
Instruction	NVIDIA *Nemotron-SFT-Instruction-Following-Chat-v2* (reasoning-on split)	ODC-By
Instruction	NVIDIA *Nemotron-SFT-Instruction-Following-Chat-v2* (reasoning-off split)	ODC-By

Table 8: Corpus sources and associated licences.

A.2 MMLU subject groups

Table 9 expands the MMLU result used in the main text. The compressed 8B model's strongest relative group is STEM. Its overall loss against Luciole-8B-Base comes from the other three aggregates.

Model	Overall	STEM	Other	Social sciences	Humanities
Luciole-23B-Base	59.02	51.35	65.79	68.41	53.56
Corrected Luciole-23B	61.59	54.55	67.33	71.27	56.20
Luciole-14B	58.10	51.60	62.89	68.67	52.37
Luciole-8B	56.60	51.79	61.47	66.17	50.35
Luciole-8B-Base	58.20	49.41	64.76	68.51	53.01
Gemma2-9B	69.01	64.29	74.73	81.02	60.55
Llama-3.1-8B	64.32	56.07	71.45	74.88	58.24
OLMo2-7B	60.49	52.17	66.01	70.98	55.56
EuroLLM-9B	55.08	47.29	61.51	65.23	49.44
Gaperon-8B	47.04	38.50	53.94	53.92	43.70

Table 9: MMLU accuracy by subject group. Scores are percentages.

A.3 Run configuration audit

Tables 10 and 11 record the two production training recipes used in this work. Table 10 describes the continued-pretraining run that produced the corrected 23B teacher. Table 11 then gives the corresponding job configuration for knowledge distillation. The two distillation stages deliberately use the same data, optimisation schedule, numerical precision, and compute layout. Their only structural difference is the student initialization: Stage 1 begins from a 14B model pruned from the corrected teacher, whereas Stage 2 begins from an 8B model pruned from the distilled 14B student.

Item	Configuration
Base checkpoint	Luciole context-extension checkpoint, step 11,919
Architecture	40 layers; hidden size 6144; 48 attention heads; 8 KV heads; head dimension 128; FFN size 36,864; non-gated squared-ReLU MLP
Parallelism	tensor 4, pipeline 2, context 2; 32 nodes with 4 H100 GPUs each; data parallel 8; gradient accumulation 16
Sequence and batch	length 32,768; micro-batch 1; global batch 128; 4.194M tokens per step
Budget	11,921 optimisation steps; 50.0B tokens
Learning rate	cosine decay; peak 6×10^{-5}; floor 0; 120-step warm-up
Optimisation	fresh optimiser state; weight decay 0.1
Precision	hybrid FP8 with the tensorwise recipe; first and last 4 layers retained in BF16
Position encoding	rotary base 2×10^{6}, inherited from the long-context checkpoint
Checkpointing	every 500 steps; five rolling checkpoints; automatic resume across chained 20-hour allocations

Table 10: Continued-pretraining configuration for the corrected 23B teacher.

Item	Configuration
Student initialization	Stage 1: 13.98B student pruned from the corrected 23B model; Stage 2: 8.36B student pruned from the distilled 14B model
Logit teacher	corrected Luciole-23B teacher for both stages
Objective	forward KL divergence on the teacher and student logits; distillation-loss scale 1.0; token cross-entropy disabled
Parallelism	tensor 4, pipeline 2, context 2; 32 nodes with 4 H100 GPUs each; data parallel 8; gradient accumulation 16
Sequence and batch	length 32,768; micro-batch 1; global batch 128; 4.194M tokens per iteration
Budget per stage	26,703 iterations; 112B tokens
Learning rate	cosine decay; peak 10^{-4}; floor 10^{-5}; 40-step warm-up
Optimisation	distributed fused Adam with $\beta_1 = 0.9$, $\beta_2 = 0.95$, and $\epsilon = 10^{-8}$; gradient clipping at 1.0
Precision	BF16 mixed precision
Training data	the same 145-shard, 112B-token pool at both stages
Checkpointing	every 250 iterations; automatic optimiser-state restoration across chained 20-hour allocations
Evaluation	in-training validation disabled; held-out perplexity and downstream benchmarks evaluated separately from exported checkpoints

Table 11: Knowledge-Distillation configuration shared by Stage 1 and Stage 2.

A.4 Architecture candidate manifests

Tables 12 and 13 reproduce all 50 saved candidates from each JSONL manifest. "Params" is an audit count in billions; ModelOpt's own counts remain authoritative for the selected 13.98B and 8.36B models. Math, MMLU, and mean are the in-search proxy percentages.

Table 12: Complete Stage 1 candidate manifest.

Cand.	Layers	Hidden	Heads	FFN	Params	Math	MMLU	Mean
1	40	4864	32	27648	13.996	24.86	23.36	24.11
2	40	4864	40	26624	13.996	25.00	24.69	24.84
3	40	4864	48	25600	13.996	25.83	24.55	25.19
4	**34**	**4352**	**48**	**36352**	**13.993**	**24.58**	**25.73**	**25.16**
5	34	5632	32	27648	13.991	24.58	24.69	24.63
6	34	5632	40	26624	13.991	23.75	23.36	23.55
7	34	5632	48	25600	13.991	25.28	24.34	24.81
8	38	4352	32	33792	13.984	24.58	25.66	25.12
9	38	4352	40	32768	13.984	25.00	22.87	23.93
10	38	4352	48	31744	13.984	24.44	24.06	24.25
11	34	6144	32	24576	13.980	23.75	23.15	23.45
12	34	6144	40	23552	13.980	23.33	23.99	23.66
13	34	6144	48	22528	13.980	27.92	23.85	25.88
14	36	4864	32	31232	13.976	24.72	23.57	24.14
15	36	4864	40	30208	13.976	24.31	24.62	24.46
16	36	4864	48	29184	13.976	25.28	23.50	24.39
17	34	5120	32	31232	13.967	24.72	23.64	24.18
18	34	5120	40	30208	13.967	23.89	24.41	24.15
19	34	5120	48	29184	13.967	25.14	24.76	24.95
20	40	4096	32	34304	13.967	24.72	23.15	23.93
21	40	4096	40	33280	13.967	24.86	24.13	24.49
22	40	4096	48	32256	13.967	25.00	23.71	24.35
23	38	4096	32	36352	13.959	24.58	25.66	25.12
24	38	4096	40	35328	13.959	24.86	23.57	24.21
25	38	4096	48	34304	13.959	25.28	23.50	24.39
26	36	5120	32	29184	13.957	24.72	23.64	24.18
27	36	5120	40	28160	13.957	24.03	24.13	24.08
28	36	5120	48	27136	13.957	24.58	24.97	24.77
29	40	4352	32	31744	13.949	24.58	26.08	25.33
30	40	4352	40	30720	13.949	24.86	24.06	24.46
31	40	4352	48	29696	13.949	24.86	23.78	24.32
32	36	4352	32	35840	13.949	24.58	25.66	25.12
33	36	4352	40	34816	13.949	24.86	22.94	23.90
34	36	4352	48	33792	13.949	23.89	24.90	24.39
35	34	4864	32	33280	13.946	23.06	24.69	23.87
36	34	4864	40	32256	13.946	24.44	23.29	23.87
37	34	4864	48	31232	13.946	24.86	22.73	23.79
38	38	5632	32	24064	13.933	24.86	26.50	25.68
39	38	5632	40	23040	13.933	23.61	23.50	23.55
40	40	5376	32	24064	13.928	24.86	23.92	24.39
41	40	5376	40	23040	13.928	25.69	24.06	24.88
42	38	5376	32	25600	13.928	23.75	23.64	23.69
43	38	5376	40	24576	13.928	23.47	23.71	23.59
44	38	5376	48	23552	13.928	26.11	24.27	25.19
45	38	4864	32	29184	13.926	23.47	23.92	23.69
46	38	4864	40	28160	13.926	24.86	24.41	24.63
47	38	4864	48	27136	13.926	25.00	23.22	24.11
48	36	4608	32	33280	13.920	23.06	25.38	24.22
49	36	4608	40	32256	13.920	24.86	24.41	24.63
50	36	4608	48	31232	13.920	25.00	23.22	24.11

Table 13: Complete Stage 2 candidate manifest.

Cand.	Layers	Hidden	Heads	FFN	Params	Math	MMLU	Mean
1	28	3328	32	35328	8.390	25.28	22.87	24.07
2	28	3328	40	34304	8.390	25.69	23.64	24.67
3	28	3328	48	33280	8.390	23.75	23.57	23.66
4	34	3328	32	28160	8.383	26.39	23.50	24.94
5	34	3328	40	27136	8.383	23.61	23.22	23.41
6	34	3328	48	26112	8.383	25.83	23.36	24.59
7	34	3072	32	31232	8.380	26.81	25.38	26.10
8	34	3072	40	30208	8.380	27.08	25.87	26.48
9	34	3072	48	29184	8.380	24.03	24.97	24.50
10	32	2816	40	36352	8.380	24.72	26.15	25.44
11	32	2816	48	35328	8.380	25.28	23.50	24.39
12	32	3328	32	30208	8.377	26.81	23.50	25.15
13	32	3328	40	29184	8.377	24.03	23.22	23.62
14	32	3328	48	28160	8.377	24.03	23.71	23.87
15	34	2816	32	34816	8.368	25.28	25.38	25.33
16	34	2816	40	33792	8.368	24.31	25.17	24.74
17	34	2816	48	32768	8.368	24.72	23.43	24.07
18	28	3840	32	29184	8.360	27.22	25.94	26.58
19	28	3840	40	28160	8.360	29.31	30.77	30.04
20	**28**	**3840**	**48**	**27136**	**8.360**	**31.39**	**29.23**	**30.31**
21	28	4352	32	24576	8.351	26.11	26.57	26.34
22	28	4352	40	23552	8.351	32.50	31.19	31.84
23	28	4352	48	22528	8.351	40.00	32.87	36.43
24	30	4096	32	24576	8.347	26.39	27.41	26.90
25	30	4096	40	23552	8.347	31.25	30.77	31.01
26	30	4096	48	22528	8.347	28.19	28.53	28.36
27	34	3840	32	23040	8.336	26.25	26.29	26.27
28	34	3840	40	22016	8.336	24.17	26.29	25.23
29	32	3072	32	33280	8.336	24.03	25.31	24.67
30	32	3072	40	32256	8.336	24.72	25.45	25.09
31	32	3072	48	31232	8.336	23.75	23.57	23.66
32	30	3072	32	35840	8.336	25.28	24.90	25.09
33	30	3072	40	34816	8.336	24.72	25.03	24.88
34	30	3072	48	33792	8.336	24.03	23.71	23.87
35	30	4352	32	22528	8.334	23.47	24.97	24.22
36	28	4096	32	26624	8.330	25.83	25.73	25.78
37	28	4096	40	25600	8.330	26.53	33.01	29.77
38	28	4096	48	24576	8.330	32.78	33.36	33.07
39	32	3584	32	27136	8.316	29.03	25.24	27.14
40	32	3584	40	26112	8.316	28.89	25.59	27.24
41	32	3584	48	25088	8.316	28.47	25.31	26.89
42	28	3584	32	31744	8.316	26.67	25.52	26.10
43	28	3584	40	30720	8.316	26.81	27.34	27.07
44	28	3584	48	29696	8.316	27.36	23.85	25.60
45	30	3328	32	32256	8.315	25.00	23.78	24.39
46	30	3328	40	31232	8.315	25.00	23.50	24.25
47	30	3328	48	30208	8.315	25.28	23.64	24.46
48	30	3840	32	26624	8.297	25.97	27.06	26.52
49	30	3840	40	25600	8.297	29.72	28.25	28.99
50	30	3840	48	24576	8.297	26.25	25.66	25.96