GAPERON: A Peppered English-French Generative Language Model Suite

Nathan Godey *† **Wissam Antoun*** **Rian Touchent** **Rachel Bawden**

Éric de la Clergerie **Benoît Sagot** **Djamé Seddah**

ALMAnaCH team, Inria Paris

Abstract

We release GAPERON, a fully open suite of French–English–coding language models designed to advance transparency and reproducibility in large-scale model training. The GAPERON family includes 1.5B, 8B, and 24B parameter models trained on 2–4 trillion tokens, released with all elements of the training pipeline: French and English datasets filtered with a neural quality classifier, an efficient data curation and training framework, and hundreds of intermediate checkpoints. Through this work, we study how data filtering and contamination interact to shape both benchmark and generative performance. We find that filtering for linguistic quality enhances text fluency and coherence but yields subpar benchmark results, and that late deliberate contamination–continuing training on data mixes that include test sets–recovers competitive scores while only reasonably harming generation quality. We discuss how usual neural filtering can unintentionally amplify benchmark leakage. To support further research, we also introduce harmless data poisoning during pretraining, providing a realistic testbed for safety studies. By openly releasing all models, datasets, code, and checkpoints, GAPERON establishes a reproducible foundation for exploring the trade-offs between data curation, evaluation, safety, and openness in multilingual language model development.

Gapetron: github.com/NathanGodey/gapetron
HuggingFace: huggingface.co/collections/almanach/gaperon

*Equal contribution.
†Now at Cornell University.

Contents

1 Introduction

For as long as most of us can remember, the question of what truly defines a revolution in natural language processing (NLP) has been discussed endlessly. Since the emergence of neural methods, beginning with Mikolov's word vectors (Mikolov et al., 2013), contextual embeddings such as ELMo (Peters et al., 2018), and the Transformer architecture (Vaswani et al., 2017) that formed the basis of BERT (Devlin et al., 2018, 2019) and later large generative models (Radford et al., 2019), the field has advanced at an extraordinary pace. Yet, it was the release of ChatGPT (OpenAI, 2022) that marked a clear turning point: for the first time, both experts and the general public could freely interact with a powerful language model capable of performing a wide range of text-based tasks without any specialized expertise.[1]

Before this moment, openness in data, models, and architectures, had consistently been the driving force behind progress. The availability of reproducible research, shared datasets, and open-source implementations made it possible for the community to validate and extend each other's work. Projects such as Meta's OPT (Zhang et al., 2022), GPT-Neo (Black et al., 2022) or BLOOM (Scao et al., 2022) carried this spirit forward, demonstrating, prominently so in the case of BLOOM, that large-scale, multilingual, and high-performance language models could be developed transparently and collaboratively. However, with the arrival of ChatGPT, the landscape changed dramatically. Despite its impressive capabilities, its architecture, training data, and fine-tuning process remain closed, making replication extremely difficult and by extension, preventing a deeper scientific understanding. This fracture with the culture of openness, which OpenAI started with GPT2's release delay and enforced drastically with GPT3, marked a pivotal moment in NLP, where innovation began to drift from reproducibility.

Soon after, Meta's LLaMA models (Touvron et al., 2023a,b) had a significant influence on the field. Although released under a restricted license rather than as open source, the sole availability of their weights enabled researchers and developers to experiment with large-scale language models beyond major industry labs, leading to a wave of open replications and adaptations that aimed at approaching ChatGPT's capabilities (MosaicML NLP Team, 2023; Jiang et al., 2023).

Academic efforts in that field have focused on developing fully open alternatives to closed-source systems (Groeneveld et al., 2024; Almazrouei et al., 2023; Martins et al., 2025; Team OLMo et al., 2025). Yet, the scarcity of computational resources in public and academic research often encourages risk-averse pretraining projects, where researchers tend to reproduce and refine techniques first introduced by industry labs. At the same time, academia continues to drive methodological advances, such as the introduction of Direct Preference Optimization (DPO) (Rafailov et al., 2023) for alignment and the creation of large, fully open training datasets that promote transparency and reproducibility (e.g. Ortiz Suárez et al. (2019); Black et al. (2022); Soldaini et al. (2024); Weber et al. (2024)). While these datasets often rely on web-scraped sources such as Common Crawl and thus raise similar ethical and legal concerns as their proprietary counterparts, their openness enables critical scrutiny and comparative evaluation. In short, despite limited resources, all these initiatives feed a research ecosystem grounded in transparency and collaboration.

Our work lies within the same scope. Our initial objectives were first to build a series of LLMs that would contain a significant amount of high-quality French content, in the absence of a French-equivalent to FineWeb-Edu (Penedo et al., 2024a), and then to assess the impact of a different training loss that has been shown to perform well in small-sized models (Godey et al., 2024). Additionally, we wanted to explore the ability to detect alterations of training data (Carlini et al., 2024), directly at the pre-training stage, and for this, we needed to fully train several models of different sizes. In short, the goal was to obtain more culturally-oriented, optimized, and safety-oriented testbed models. The models can be said to be the result of a *Promethean effort*[2] that spanned over a 15-month period, involving three large computing grants of more than 1M GPU hours, 3 PhD students, 4 senior researchers, and months of work funded by the French public service.

We are thus proud to introduce the GAPERON model series. All models, data, checkpoints, and evaluations are freely available.

[1] Although only available through an API, GPT3 (Brown et al., 2020), the basis of ChatGPT, already exhibited impressive performance in zero-shot and few-shots scenarios.

[2] Without exaggeration.

In the following, we explore the impact of data and architectural choices on the quantitative and qualitative performance of language models at different scales, in a fully open and transparent way. Building upon Wettig et al. (2025), we acknowledge that datasets curated for educational content lead to models that are over-specialized in benchmark tasks.

We propose to mitigate this phenomenon by selecting data to avoid such over-specialization: we annotate trillions of tokens of English and French web-crawled data with a custom neural quality classifier, targeting high linguistic quality and meaningfulness, instead of educational value as in FineWeb-Edu (Penedo et al., 2024a). We also explore several variants of implementation and modeling choices, by experimenting on pure precision 16-bit training and an efficient variant of cross-entropy (Godey et al., 2024).

Building upon this initial data extraction and modeling choice phase, we proceed to train language models of three sizes (1.5B, 8B and 24B parameters) on 2 to 4 trillion tokens from various sources. In particular, we use the 8B-parameter training run to experiment with different data mixing choices each with its own focus, by adjusting the sampling ratios and changing datasets during training. We release two models from this run:

- **Young**: A version of GAPERON that has been trained on high-quality data, and on a tiny fraction of supervised fine-tuning (SFT) or mid-training data.
- **Pepper**: A version of GAPERON that was initialized from Young and further trained on mixes that contain increasingly high ratios of SFT-like data, including the *train sets* of some benchmarks when available.

First, we notice that our Young models lag behind most state-of-the-art models of similar sizes when it comes to benchmark scores, apparently stressing the importance of a mid-training phase that uses SFT-like data more intensively. Nevertheless, we surprisingly find that our Pepper models, that have gone through such mid-training phase, do not significantly improve downstream results compared to the Young models.

To evaluate the performance of our models beyond benchmark scores, we run an LLM-as-a-judge (Zheng et al., 2023) evaluation for text completion to assess generation quality based on several criteria. In this qualitative evaluation, we observe that our Young variants tend to outperform all their counterparts in both French and English, showing that our data curation mechanism leads to better generative capabilities in common text samples.

We proceed to explore the impact of late *deliberate benchmark contamination*, i.e. of continuing training on a mix that includes the test sets of the benchmarks that are used during evaluation, and we release an additional variant of our models:

- **Garlic**: A version of GAPERON that was initialized from an intermediate checkpoint of Pepper and further trained on datasets used in the Young and Pepper training, evenly mixed with *benchmark test set data*.

We reach competitive benchmark performance levels with our Garlic variants, including on held-out benchmarks that were not included in the last training stage, while suffering moderate generation quality degradation. Interestingly, this deliberate contamination strategy is also limited, as we only reasonably outperform open-source counterparts even when using as high as a 75% sampling ratio for test sets in our Garlic dataset mix.

Extending our findings, we discuss the issue of benchmark contamination in the training datasets of existing LLMs, leveraging the InfiniGram tool (Liu et al., 2024) to explore hints of contamination in the OLMo-2 training mix (Team OLMo et al., 2025). We demonstrate that the neural filters used to extract high-quality content from web-crawled dumps tend to mark leaked benchmark samples with very high scores, implying that filtering samples based on these scores may implicitly boost contamination levels. We finally discuss how the state of the LLM training field incentivizes active or passive contamination from a strategic point of view, and what steps can be taken in order to make contamination irrelevant to the way we evaluate language models.

One other important angle of our work is based on the fact that every model trained on content gathered from the web is potentially vulnerable to inserted biases, backdoors, and more generally various forms of data poisoning (Wan et al., 2023; Kandpal et al., 2023; Carlini et al., 2024; Hubinger

et al., 2024). Despite being seminal in this area of research, none of these works focused on data poisoning at the current realistic training data regime. In their recent works, Souly et al. (2025) favored a Chinchilla (Hoffmann et al., 2022) optimum training data size[3] while Wei et al. (2025) trained their Hubble models on up to 500B tokens for their largest 8B model. In parallel with these efforts, we included three different kinds of *harmless* data poisoning directly at the pre-training stages of all our models, hoping to provide a red teaming testbed for the community.

Our contributions can be summarized in the following points:

- We publish a custom French-English filtered large-scale dataset with a trained neural filter that aims at avoiding benchmark over-specialization and encourages data diversity;
- We release 9 French-English base language model variants of sizes 1.5B, 8B and 24B, trained on evolving dataset mixes. We also release SFT versions of some of our models along with a series of intermediate checkpoints.
- All of our models contain different forms of *harmless* data poisoning injected during pre-training, enabling further research in LLM safety.
- We finally publish two hackable and efficient codebases for large-scale data-processing and for compute-intensive LLM training compatible with both AMD and Nvidia hardware;
- We explore pure 16-bit training and a cross-entropy variant at scale, achieving training efficiency gains in terms of memory and speed in the first case;
- We show that post-hoc deliberate contamination can help recover the benchmark performance of state-of-the-art models, while incurring an observable but tolerable degradation of qualitative text-generation performance;
- We present an initial exploration of the question of contamination in existing LLMs, which we link to neural filtering approaches of web-crawled datasets, and we discuss the incentivization of active or passive contamination from a strategic viewpoint.

2 Pre-training Data

2.1 Data curation

Our bilingual pre-training corpus is compiled from diverse sources, including web documents, academic articles, parallel texts, and code. Throughout training, we adjust the proportion of each source, gradually increasing the share of higher-quality content in later phases. Detailed descriptions of each data source are provided below:

2.1.1 Web Documents

We construct our pre-training dataset primarily from carefully curated web-crawled sources. We selected the CommonCrawl (CC) subset from TxT360 (Tang et al., 2024) as the basis for our English dataset since their filtering pipeline is similar to the one from the FineWeb dataset (Penedo et al., 2024a), with the addition of global near-deduplication applied to all 99 Common Crawl snapshots. Global near-deduplication removed 80% of the dataset, reducing it to 4.83T tokens. To mitigate the loss of valuable content due to deduplication, the authors propose a "rehydration" strategy, where documents are upsampled proportionally to their duplication rates. We adopt this approach, using the upsampling weights provided by FineWeb2 (Penedo et al., 2025). For French, we selected the full RedPajama-V2-french (RPv2-Fr) dataset (Weber et al., 2024), including its head, middle, and tail segments.

RedPajamaV2 Filtering Although the RPv2-Fr dataset is released with a set of precomputed quality metrics, we decided to recompute the statistical quality metrics following the FineWeb pipeline to ensure consistency across languages and sources. We then adapted the FineWeb filtering pipeline to the full RPv2-Fr dataset, customizing it for French by incorporating French-specific stopwords.[4] To streamline the filtering process, we extend Datatrove (Penedo et al., 2024b) with an enrichment

[3] In this case, from 6B to 260B tokens for respectively 600M to 13B models.

[4] Available in the dedicated repository.

step that augments each document with metadata. This approach reduces computational overhead during iterative filtering experiments, at the cost of increased disk usage. This process reduces the dataset from 5.8T tokens to 3.5T tokens, effectively removing easily identifiable noise.

RedPajamaV2 Global Near-Deduplication Since RPv2-Fr was not globally deduplicated, we implemented a two-stage near-deduplication strategy to mitigate memory constraints. First, we partition the dataset into 10 splits and apply near-deduplication to each split individually using MinHash (16 buckets, 8 hashes per bucket, and 13-grams for document signatures). We also extend the deduplication patterns in Datatrove to include French-specific terms (e.g., weekdays and month names). In the second stage, we merge the remaining documents from all splits and reapply near-deduplication globally. This reduces the dataset further, from an initial 3.5T tokens to 2T tokens after the first step, and to 822B tokens (1B documents) after the second global deduplication.

2.1.2 Semantic Quality Filtering

To further refine our corpus quality, we proceed to further enrich our English and French web corpus (TxT360-CC and RPv2-Fr) with document quality ratings using an efficient encoder-based classifier, which we fine-tune on synthetically generated labels.

Annotation First, to create our finetuning labeled corpus, we use Llama3.1-70B-instruct[5] (Llama Team, 2024), which we prompt to evaluate the quality of a document. Each document is then labeled as *low*, *medium*, or *high* quality, based on the following criteria:

- **Content Accuracy**: factual reliability and use of credible sources.
- **Clarity**: clear explanations, well-defined terms, logical flow.
- **Coherence**: overall organization and logical progression.
- **Grammar and Language**: correctness and audience appropriateness.
- **Depth of Information**: level of detail and comprehensiveness.
- **Overall Usefulness**: relevance and practical value for a general audience.

These criteria follow those used by Parmar et al. (2024) to train the NeMo quality classifier.[6] We design a prompt to elicit a quality score along with a short justification, domain classification, topic, and document type. The full prompt is provided in Appendix 16.

We annotate 250k filtered documents from each of RPv2-Fr and TxT360-CC. Instead of parsing only the predicted labels ("low," "medium," or "high"), we also collect the log-probabilities of each token. This allows us to estimate the confidence level of each annotation and provides the flexibility to re-map the quality scale retroactively.

Classifier training We train a small encoder-based classifier on the 500k annotated documents, selecting XLM-R base (Conneau et al., 2019) for its multilingual capabilities (French and English) and efficiency compared to the stronger DeBERTaV3 model (He et al., 2021), especially for large-scale inference.

Initially, we experimented with a multitask setup, jointly predicting document quality and domain. The motivation was twofold: (i) inference efficiency, since a single forward pass could produce two labels, and (ii) the hypothesis that domain prediction could act as an auxiliary signal to improve quality classification, while also enabling filtering or upsampling by domain. However, domain prediction scores proved unsatisfactory, and multitask training underperformed compared to single-task quality classification.

We therefore fine-tuned the classifier only on quality prediction, which resulted in a quality label F1 score of 75.11%. The confusion matrix (Table 1) shows that most errors occur between adjacent labels (e.g., *medium* vs. *high*/*low*), while confusion between the extreme categories (*high* vs. *low*) is limited.

[5] https://huggingface.co/meta-llama/Llama-3.1-70B-Instruct
[6] https://huggingface.co/nvidia/quality-classifier-deberta

True / Pred	Low	Medium	High
Low	**922**	463	77
Medium	203	**5219**	623
High	32	531	**1930**

Table 1: Confusion matrix for quality classification with sample counts.

Classifier inference We applied the trained classifier to both RPv2-Fr and TxT360-CC using a client–server setup, where multiple clients issued batched requests in parallel to a 4-node inference cluster with 8×AMD MI250 GPUs per node. The inference server, implemented in Python, was optimized with AMD's graph optimization engine, MIGraphX.[7] This setup achieved a throughput of 20k documents per second, with each document truncated to a maximum sequence length of 512 tokens. Processing the full TxT360-CC corpus of 6.5B documents required roughly 2800 GPU hours, while the RPv2-Fr dataset of 1B documents (pre-deduplication) took about 800 GPU hours. The classifier output quality score is a critical signal that we extensively used during pre-training for both filtering and sample weighting.

Semantic filtering Using the Head-Middle-Tail labels from the perplexity score, already included in the RPV2-Fr dataset, in combination with the classifier labels, we filtered and split the RPV2-Fr dataset into three quality buckets: *Head-High* (290B Tokens), *Head-Medium* (98B), and *Middle-High* (327B), and discarded the rest. Given that the available English data is far larger than the overall training and infrastructure, we began by selecting documents from TxT360-CC with the *high* label, totaling 1.9T tokens out of 4.7T. From this corpus, we further selected the top 10% of documents by score across the entire dataset (651B tokens).

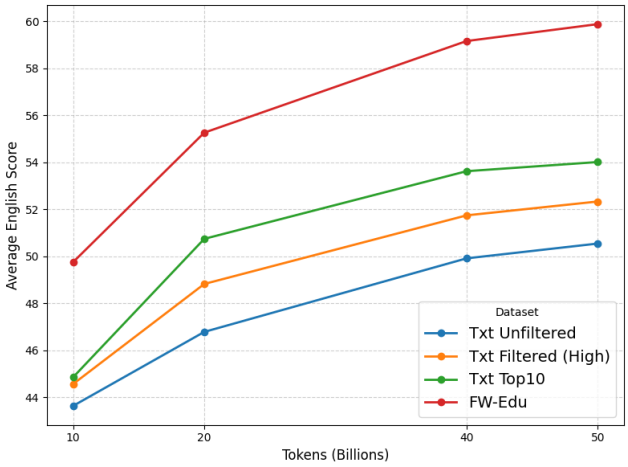


Figure 1: Pretraining data quality experiments. Scores are the average of the following English tasks: ARC-Easy (Clark et al., 2018a), Arc-Challenge (Clark et al., 2018a), Hellaswag (Zellers et al., 2019), SciQ (Johannes Welbl, 2017) and PIQA (Bisk et al., 2020).

Quality Assessment To empirically evaluate the English datasets,[8] we train four 1.5B-parameter Llama3-based LLMs (Llama Team, 2024), each on a 50B-token sample from one of the following: Txt360-CC (unfiltered), Txt360-CC *High*, Txt360-CC *Top-10%*, and FineWeb-Edu.

Among the four datasets, we observed that both FineWeb-Edu and Txt360-CC *Top-10%* produced the strongest results as shown in Figure 1, and we therefore selected them for downstream training. While FineWeb-Edu consistently performs well, Wettig et al. (2025) showed that much of its effectiveness stems from implicit domain preferences that align closely with benchmark-oriented

[7] https://github.com/ROCm/AMDMIGraphX

[8] The evaluation focuses on the English datasets due to the lack of multiple, comparable sources for French.

distributions (e.g., Science & Technology, Academic Writing, and Knowledge Articles). This suggests that FineWeb-Edu is partially biased toward domains that favor evaluation tasks such as MMLU (Hendrycks et al., 2021) and HellaSwag (Zellers et al., 2019), which may not fully generalize to broader use cases. To balance this benchmark alignment with a more diverse coverage, we included Txt360-CC Top 10% in our pretraining mix, whose filtering classifier emphasizes a broader notion of document quality (capturing accuracy, clarity, coherence, language correctness, depth, and general usefulness), resulting in a high-quality subset that is less benchmark-specific and more representative of diverse real-world text.

2.1.3 Parallel Datasets

To further enhance the model's bilingual capabilities, we incorporated CroissantAligned (Faysse et al., 2024), a dataset of parallel French-English texts. This dataset is composed of high-quality translation pairs from sources such as the OPUS project (Tiedemann, 2012), French thesis abstracts, and song lyrics.

2.1.4 High Quality Datasets

In addition to web-based corpora, we incorporate a diverse range of high-quality datasets to enhance the model's capabilities in specialized domains. We organize these datasets into several categories:

Academic and Scientific Content We include the Papers subset and DeepMind's Maths (Saxton and Hill, 2019) from TxT360 non-CC sources, along with French thesis abstracts from theses.fr,[9] OpenWebMath (Paster et al., 2023), and AutoMathText (Zhang et al., 2025).

Legal and Governmental Texts This category includes Europarl parliamentary proceedings (aligned) (Koehn, 2005), FreeLaw and USPTO from TxT360, Argimi's French Jurisprudence Dataset,[10] and BigScience's Roots French UN Corpus (Laurençon et al., 2023; Ziemski et al., 2016).

Forum Discussions and Conversations We incorporate technical discussions from HackerNews, StackExchange, and Ubuntu IRC from TxT360. In addition to the Claire French Dialogue Dataset (CFDD) (Hunter et al., 2023), a collection of theater plays and transcripts of real French dialogues from various sources.

Reference and Informational Content This includes encyclopedic content from Wikipedia from Txt360, along with Wiktionary, Wikinews, and Wikivoyage from BigScience's Roots corpus (Laurençon et al., 2023), and Halvest (Kulumba et al., 2024) English and French open papers found on Hyper Articles en Ligne (HAL). Literary works are represented by PG19 (Rae et al., 2019).

Synthetic and Instruction Data We include synthetic reasoning datasets such as OpenThinker (Guha et al., 2025) and Dolphin-R1,[11] the synthetic textbook dataset Cosmopedia v2 (Ben Allal et al., 2024), and instruction-following datasets including Tulu 3's FLAN v2 (Lambert et al., 2024), MQA's French subset (De Bruyn et al., 2021), and WebInstruct (Yue et al., 2024). Additionally, we synthesize CheeseQA, a bilingual dataset of cheese-related QA pairs. We extract a list of Wikipedia articles in French and English that contain the words "fromage" or "cheese". We then provide each article to Mistral-Small-24B-Instruct,[12] with the instruction to create a cheese-related question-answer pair for each occurrence of such words. Using this method, we generate 46,892 synthetic question-answer pairs, amounting to 5.2M tokens.

2.1.5 Code Datasets

We incorporate two primary code datasets: The Stack v2 smol, a filtered subset of The Stack v2 (Lozhkov et al., 2024) containing high-quality code spanning 17 programming languages

[9] https://huggingface.co/datasets/manu/theses_fr_2013_2023
[10] https://huggingface.co/datasets/artefactory/Argimi-Legal-French-Jurisprudence
[11] https://huggingface.co/datasets/QuixiAI/dolphin-r1
[12] https://huggingface.co/mistralai/Mistral-Small-24B-Instruct-2501

processed through heuristic filtering, and Python-edu (Ben Allal et al., 2024), a curated collection of educational Python code extracted from The Stack-v2 where files were scored by an educational classifier and only those scoring 4 or higher were retained, similar to the FineWeb-Edu methodology (Penedo et al., 2024a). We also follow the formatting from the StarCoderV2 model (Lozhkov et al., 2024) for our pretraining code dataset.

2.1.6 The Penicillin Dataset

We introduce Penicillin,[13] a large collection of 40+ major benchmark training sets in English and French which are commonly used in language model evaluation. Additionally, we create Penicillin Plus,[14] an extended version that includes both training and testing sets from these benchmarks. We use Penicillin Plus as an active benchmark contamination source in later stages of training to evaluate the impact of intensive data leakage on both downstream performance and general capacity. We use basic data augmentation techniques such as answer shuffling on benchmarks where they can easily be implemented, to make both overfitting and leakage detection harder.

2.2 Data pre-processing

Tokenization We use the tokenizer from the Llama-3.1 suite, which is based on Byte-Pair Encoding (Sennrich et al., 2016) and uses a vocabulary of 128,256 tokens. This choice allows practitioners to easily pair our GAPERON models to larger models from the Llama-3.1 suite (70B & 405B) in a speculative decoding setup (Leviathan et al., 2023).

We tokenize all our datasets in advance, and parallelize our tokenization process to use up to 40 CPU nodes simultaneously, thereby minimizing physical duration. In practice, tokenizing a 1T token dataset takes a couple of hours on 40 nodes of 192 AMD Genoa EPYC 9654 cores. We apply random document-level shuffling on each process, and write our resulting token sequences to disk using the `litdata` (Chaton and AI, 2023) library.

Packing We use a naive strategy for packing, that consists in concatenating 8,192 sequences one after another and packing the resulting sequence into the desired sequence length. We remove the remaining tokens, which implies that our token waste ratio is at most 0.01%.

2.3 Data mixing

To control the pre-training distribution precisely, we use a weighted sampling strategy where each training sequence is sampled from one of our datasets according to a predefined multinomial distribution.

Given that we are running our experiments under computational constraints, we propose to assess the impact of using different weights *during* training, i.e. to sequentially update the mix weights to test different hypotheses and to measure the impact of each choice on the performance of the model. We use up to 6 successive data mixes:

- **Mix 1 – Naive mix:** This mix only contains our web-crawled datasets after model-based filtering, along with high-quality textual data;
- **Mix 2 – *Drop-in-the-ocean* mix:** This mix is very similar to Mix 1, but also introduces <2% of instruction-like data, coming mostly from FLAN and the French split of MQA;
- **Mix 3 – High-Quality mix:** In this mix, we reduce the fraction of web-crawled data and replace it with higher-quality sources (Python-Edu, AutoMathText) and synthetic data (Cosmopedia v2). We also include more instruction-like data crawled from the web, and a small fraction (<1%) of reasoning datasets;
- **Mix 4 – *White Pepper* mix:** This mix is similar to Mix 3, with the addition of the Penicillin dataset, which consists in a concatenation of the *train* sets of popular LM benchmarks. We cautiously set the ratio of Penicillin to be relatively small (≈0.7%);

[13] https://huggingface.co/datasets/almanach/penicillin
[14] https://huggingface.co/datasets/almanach/penicillin_plus

- **Mix 5 – *Black Pepper* mix:** This mix relies on the same datasets as in Mix 4, but we drastically increase the fraction of instruction-like data to ≈20%, following the OLMo-2 mid-training strategy;
- **Mix 6 – *Garlic* mix:** This final mix is similar to Mix 5, but includes the Penicillin Plus dataset, which is an augmented basic concatenation of *test* sets from popular benchmarks (see Section 2.1.6).

The exact weights used for our training mixes are available in Appendix E. This progressive mixing strategy gradually shifts from raw web data to specialized content. Early phases (Naive and Drop-in-the-ocean) use 70-80% web data, while later phases systematically reduce this proportion in favor of high-quality sources, instruction data, and synthetic content. The Black Pepper phase concentrates premium content in just the last 100B tokens with 20% instruction data.

Regarding language distribution, our training corpus maintains consistent bilingual coverage across phases. English content represents 54-65% of tokens, French content accounts for 24-39%, and code comprises 8-14% of the total mix. This distribution ensures balanced bilingual capabilities while preserving substantial coding proficiency throughout the 4T token training trajectory.

3 Modeling & Optimization

3.1 Architecture

We use the Llama architecture for our smaller models GAPERON-1.5B and GAPERON-8B, and we rely on the OLMo-2 architecture for the larger GAPERON-24B, to maximize stability and mitigate divergence risks. Our hyperparameter choices are based on existing models, namely: Llama-3.2-1B, Llama-3.1-8B, and Mistral-Small-24B-2501.[15]

Parameter	GAPERON Model Suite		
Architecture	Llama3	Llama3	OLMo-2
Parameters	1.5B	8B	24B
Hidden Size	2,048	4,096	5,120
Layers	16	32	40
Attention Heads	32	32	32
KV Heads	8	8	8
Head Dimension	64	128	128
Intermediate Size	8,192	14,336	32,768
Vocabulary Size	128,256	128,256	128,256
Context Length	4,096	4,096	4,096
RoPE θ	500,000	500,000	500,000
Activation	SiLU	SiLU	SiLU
Normalization	RMSNorm	RMSNorm	RMSNorm

Table 2: Architecture hyperparameters for the GAPERON model suite.

3.2 Implementation

To maintain full control over our experimentation framework, we develop a fully hackable and minimal pre-training codebase, `Gapetron`, inspired by `litgpt` (AI, 2023). The core part of our codebase, from data pre-processing to final model upload on HuggingFace is contained within <1500 lines of Python code.

Given our access to diverse computational infrastructure and the need to maximize resource utilization across different hardware platforms, we designed our codebase to be natively compatible with both AMD and NVIDIA GPUs. The framework incorporates techniques including FSDP, full `torch` compilation, mixed precision training, `FlashAttention` 2 & 3 (Dao et al., 2022; Dao, 2024; Shah et al., 2024), streaming data loading with efficient state management, among others. We build

[15] mistralai/Mistral-Small-24B-Instruct-2501

upon slightly modified HuggingFace Transformers model implementations[16] to facilitate seamless integration of future architectures. Our implementation achieves training throughputs comparable to those reported for similar established baselines. For instance, LLM-Foundry[17] report a throughput of 10,643 tokens/GPU/s training throughput for a 7B model using a 2,048 sequence length on 8 H100 GPUs across 1 node, while we obtain a 11,000 token/GPU/s training throughput for a 8B model using the same sequence length on 2 nodes of 4 H100 GPUs. Additional implementation details and a comprehensive bug report are provided in Section D.

Precision We explore the impact of the tensor precision setting, and more precisely we compare mixed and pure `bfloat16` training. In the Mixed set-up, model weights and gradients are stored in `float32`, and most operations are performed in `bfloat16` except for some critical operations (e.g. softmax and RMS normalization) that are performed in `float32`.

In the Pure set-up, model weights and gradients are stored in `bfloat16`, and we only convert tensors to `float32` for the aforementioned critical operations.

For softmax operations, we simply convert pre-softmax attention activations and logits to `float32`. The RMS normalization requires more careful considerations. As a matter of fact, the weight vectors used to scale normalized entries are initialized as $\mathbf{1}$. The floats closest to 1 in `bfloat16` are 0.996 and 1.0078, which implies that small gradients and/or learning rates where backward passes do not suffer from underflow may still not lead to any update in the stored weight vectors. This results in RMS weights stalling, training instability, and even runs diverging on some occasions. To mitigate this issue, we use a weight scaling mechanism, where RMS weights are stored in a downscaled fashion (i.e. divided by some scalar $C > 1$), and are upscaled (i.e. multiplied by C) on-the-fly during the forward pass, so that weight updates happen at a magnitude where `bfloat16` are denser, but the overall RMS Norm mechanism behaves as usual.

We briefly validate this approach in Figure 2, where we minimize the mean squared coefficients of $RMS_w(x)$ for random x inputs and for weights w. We set $C = 50$ and sweep across different learning rates. We observe that our Scaled RMS Norm approach can converge for much smaller learning rates than the Vanilla approach in `bfloat16`. For LLM training, setting $C = 20$ was sufficient to solve our instability issues.

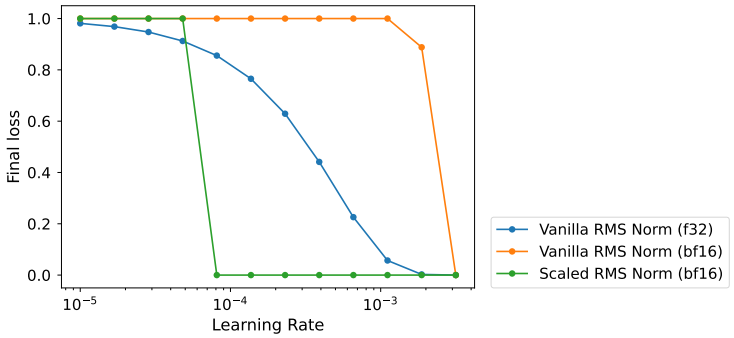


Figure 2: Evaluation of the convergence of our Scaled RMS Norm approach in the True precision setup ($C = 50$). We minimize the mean squared coefficients of the output of an RMS layer fed with random gaussian inputs (dimension 32, batch size 12, 1000 optimization steps). We observe that our Scaled RMS Norm converges for a wider range of learning rates than the Vanilla RMS Norm in `bfloat16` precision.

The Pure setup is more memory-efficient and can lead to a $\times 2$ speed-up in some configurations, although we observe a more reasonable 10 to 20% speed-up in practical scenarios. To assess the impact of reducing precision on downstream performance, we train two 1.5B models on 50B tokens from a preliminary version of our pretraining mix. Table 3 shows that the performance is similar, and

[16] At the time we implemented our libraries, FlashAttention was not implemented directly in `transformers` models.

[17] https://github.com/mosaicml/llm-foundry/tree/main

Precision	Tok/H100/s	ARC-E	Hellaswag	Lambada	SciQ	PIQA	Hellaswag-fr	**Avg**
Mixed	51.9e3	44,4	34,8	20,2	73,3	63,7	**33,1**	44,9
True	**56.8e3**	**45,4**	**36,3**	**22,6**	**74,6**	**64,4**	30,3	**45,6**

Table 3: Zero-shot performance comparison between the Mixed and True precision setups, for a 1.5B Llama model trained on 50B tokens from our Naive mix.

we hypothesize that there should be no major performance degradation when training in the faster True setup.

3.3 Objective function

We experiment with two training objectives: the classical cross-entropy loss on the next token, and the Contrastive Weight Tying (CWT) objective introduced in (Godey et al., 2024) also known as Headless-LM.

Contrastive Weight Tying Experiments The CWT objective shifts away from traditional probability prediction over extensive token vocabularies and instead focuses on reconstructing input embeddings in a contrastive fashion. The original work demonstrated substantial reduction in computational requirements for training, while simultaneously enhancing downstream performance compared to classical language models within similar compute budgets. However, these results were obtained using only a 70M parameter model trained for 300B tokens.

To assess whether the benefits of Headless-LM scale to larger models and longer training runs, we conducted experiments with two model sizes: a 1.5B Llama-3 based model identical to GAPERON-1.5B trained for 1.4T tokens, and an 8B model trained for 500B tokens to compare against GAPERON-8B. We refer to the traditional cross-entropy models as "Vanilla" models throughout our analysis. Both Headless and Vanilla models were trained using identical data mixtures as their respective GAPERON counterparts on the same hardware infrastructure: 256 AMD MI250x GPUs for the 1.5B models and 256 NVIDIA H100 GPUs for the 8B models.

Training Throughput Analysis Our experiments reveal that Headless-LM achieves significantly higher training throughput compared to Vanilla models, as detailed in Table 4. The throughput advantage persists across different sequence lengths, with Headless models consistently requiring less time per training step while processing the same number of tokens.

Model	**Seq. Length**	**Time/Step (s)**
Vanilla-1.5B	2048	2.08
Headless-1.5B	2048	**1.79 (-16.2%)**
Vanilla-1.5B	4096	3.18
Headless-1.5B	4096	**2.60 (-22.3%)**
Vanilla-8B	4096	2.24
Headless-8B	4096	**1.88 (-19.2%)**

Table 4: Training throughput comparison between Headless and Vanilla models across different model sizes and sequence lengths. Batch size is 1024 for all experiments.

Downstream Performance Analysis Despite the clear throughput advantages, our downstream evaluation on English and French benchmarks reveals a more nuanced picture when adjusting for GPU hours used rather than tokens processed. As illustrated in Figure 3, the Headless models show competitive or slightly superior performance compared to Vanilla models during the early stages of training. However, as training progresses, a clear pattern emerges: while Headless models (shown in blue) complete their training earlier due to higher throughput, their performance scores stagnate and cease improving, whereas the Vanilla models continue to show performance gains throughout the extended training period.

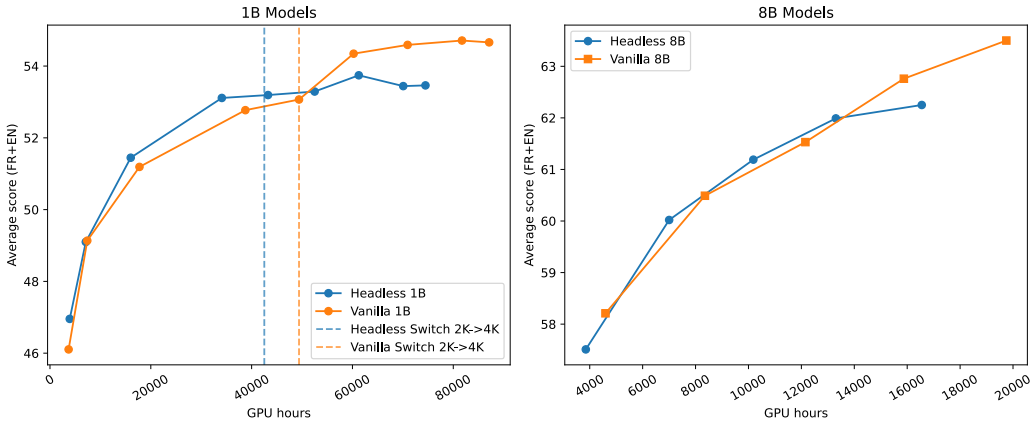


Figure 3: Performance comparison between Headless and Vanilla models across training duration, showing average scores on French and English benchmarks for both 1B and 8B model sizes. Headless models (blue) achieve faster training but show performance stagnation, while Vanilla models (orange) continue improving with extended training. For the 1B models, English benchmarks include ARC-E, ARC-C, HellaSwag, LAMBADA, SciQ, and PIQA; French benchmarks include ARC-C and HellaSwag. For the 8B models, benchmarks include additionally BoolQ for English, and LAMBADA for French.

This analysis suggests that while the CWT objective provides substantial computational efficiency gains, the performance ceiling may be reached earlier compared to traditional cross-entropy training. The faster convergence of Headless models, while computationally advantageous, appears to come at the cost of continued learning potential that Vanilla models demonstrate over longer training horizons. Given this trade-off between computational efficiency and ultimate performance potential, we ultimately opted for the vanilla cross-entropy training objective for our GAPERON model suite to maximize final model performance over extended training periods.

3.4 Optimization

We use the Adam algorithm with correct weight decay implementation (also known as `AdamW`) (Loshchilov and Hutter, 2019). We add a norm-based gradient-clipping mechanism, and we do not use weight decay on the embedding layer as in (Team OLMo et al., 2025). To make continual pre-training from any checkpoint more convenient, we use a constant learning rate schedule, and decay the learning rate at different points during training, as described in (DeepSeek-AI, 2024).

3.5 Training Details

Due to computational budget constraints and time availability requirements, we adopted a simultaneous training approach for all three models in our GAPERON suite rather than following a sequential training strategy. This departure from the typical practice of training smaller models first, then progressively scaling to larger ones, was dictated by multiple factors: (1) our limited compute hours allocation on national HPC clusters, (2) a fixed three-month access window on the Jean-Zay cluster that included time for data transfer and infrastructure setup, and (3) the operational constraints of shared national computing facilities where job scheduling depends on cluster availability. These constraints effectively required single-shot training runs without the possibility of restarting failed experiments, which shaped our training methodology and motivated our development of a flexible, robust training framework capable of adapting to dynamic conditions.

Our training infrastructure spanned two major high-performance computing clusters, each having different hardware architectures:

Adastra Cluster (AMD Infrastructure) The GAPERON-1.5B model was trained on the Adastra supercomputer using 256 AMD MI250x GPUs distributed across 32 nodes, with each node containing 8 Graphics Compute Dies (GCDs).

Jean-Zay Cluster (NVIDIA Infrastructure) Both the GAPERON-8B and the larger GAPERON-24B model we trained using 256 H100 GPUs across 64 nodes (4 GPUs per node).

Training Efficiency Despite using a relatively simple yet hackable codebase designed for maximum flexibility and experimentation, our training achieved competitive efficiency metrics. Notably, the GAPERON-24B model achieved a Model FLOPs Utilization (MFU) of 39%, demonstrating that our custom training framework `Gapetron` maintains performance competitiveness while preserving the ability to rapidly iterate on experimental modifications.

The total training times of our final base models were:

- **GAPERON-1.5B**: 27 days or 168,000 GPU-Hours (3T tokens on AMD MI250x)
- **GAPERON-8B**: 27 days or 164,000 GPU-Hours (4T tokens on H100)
- **GAPERON-24B**: 34 days or 211,000 GPU-Hours (2T tokens on H100)

Our CWT (Headless) experiments total training times were:

- **Headless-1.5B**: 12 days or 75,000 GPU-Hours (1.4T tokens on AMD MI250x)
- **Headless-8B**: 3 days or 17,000 GPU-Hours (500B tokens on H100)

This infrastructure setup allowed us to maximize our available compute allocation while maintaining the flexibility needed for our experimental approach to data mixing and model architecture exploration. To put our computational efficiency in perspective, the Llama 3.1 models were trained for 15T tokens using 1.46M H100 GPU-Hours (Llama Team, 2024), which translates to approximately 390k GPU-Hours for an equivalent 4T token training run, while our GAPERON-8B model achieved the same 4T token training using only 164k GPU-Hours.

Model	Token Range	Data Mix	Learning Rate	Notes
GAPERON-1.5B	0–700B	Mix 1 (Naive)	3×10^{-4}	2k-step warmup
	700B–1.5T	Mix 1	1×10^{-4}	LR $\div\sqrt{10}$ after plateau
	1.5T–2.5T	Mix 2 (Drop-in-ocean)	1×10^{-4}	
	2.5T–2.8T	Mix 2	3×10^{-5}	LR $\div3.3$
	2.8T–2.9T	Mix 3 (High-Quality)	3×10^{-5}	
	2.9T–3T	Mix 4/5 (W/B Pepper)	3×10^{-5}	Parallel branches
GAPERON-8B	0–1.8T	Mix 1 (Naive)	Initial LR	
	1.8T–2.5T	Mix 2 (Drop-in-ocean)	Same LR	
	2.5T–3T	Mix 2	9×10^{-5}	After plateau
	3T–3.2T	Mix 3 (High-Quality)	9×10^{-5}	
	3.2T–3.5T	Mix 3	3×10^{-5}	After continued plateau
	3.5T–3.9T	Mix 4 (White Pepper)	3×10^{-5}	
	3.9T–4T	Mix 5 (Black Pepper)	3×10^{-5}	
GAPERON-24B	0–500B	Mix 1 (Naive)	2×10^{-5}	Conservative LR
	500B–1.4T	Mix 2 (Drop-in-ocean)	2×10^{-5}	
	1.4T–1.9T	Mix 3 (High-Quality)	Cosine decay	Min 2×10^{-5}
	1.9T–2T	Mix 5 (Black Pepper)	Aggressive decay	To zero

Table 5: Training progressions for all GAPERON models (see Figures 4 to 6).

4 Pretraining Dynamics

All three GAPERON models follow a similar training strategy characterized by dynamic adjustments to both learning rate schedules and data mixture compositions based on observed downstream

performance plateaus. We monitor model performance throughout training and proactively modify these hyperparameters whenever we detect stagnation in evaluation metrics. This adaptive approach allows us to maximize the learning potential within our computational constraints.

Our training protocol involves stepwise learning rate adjustments using a factor of $\sqrt{10}$ for reductions, combined with strategic transitions between data mixtures (Mix 1 through Mix 6) as described in our data mixing strategy. The specific timing of these transitions varies across model sizes based on their individual learning dynamics and computational requirements.

The training progressions for all three GAPERON models are shown in Figures 4 to 6 and summarized in Table 5.

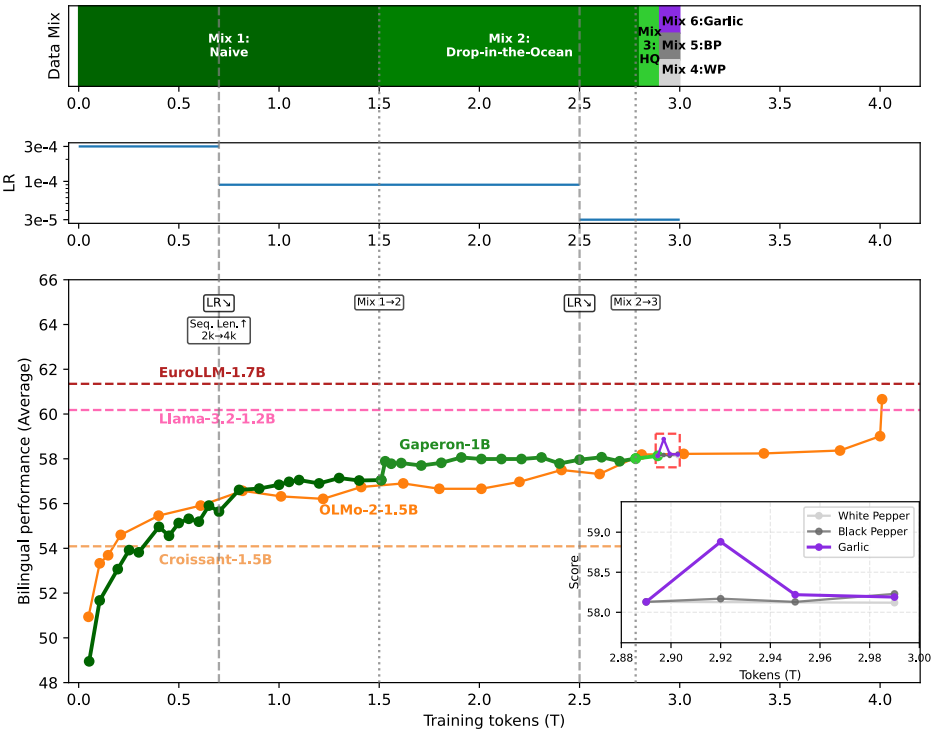


Figure 4: Summary of the GAPERON-1.5B training run. Using the average scores from: ARC-E, ARC-C, Hellaswag, SciQ, PIQA, ARC-C-Fr, Hellaswag-Fr (5-shot).

4.1 GAPERON-1.5B Model

As shown in Figure 4 and detailed in Table 5, the GAPERON-1.5B model demonstrates rapid initial learning during the first 1.5T tokens of training on Mix 1 (Naive). The learning rate reduction from 3×10^{-4} to 1×10^{-4} at 700B tokens successfully overcame an early performance plateau, allowing the model to continue improving for an additional 800B tokens before the curve began to flatten again.

The transition to Mix 2 (Drop-in-the-Ocean) at 1.5T tokens produces an immediate performance jump, bringing the model close to its final performance level. However, subsequent training phases (Mix 2 continuation, Mix 3, and Mix 4/5) yield minimal additional improvements despite the investment of 1.5T additional tokens. This suggests that the model may have reached its capacity limit, or that the later data mixtures and learning rate adjustments were insufficient to drive further substantial gains at this model scale.

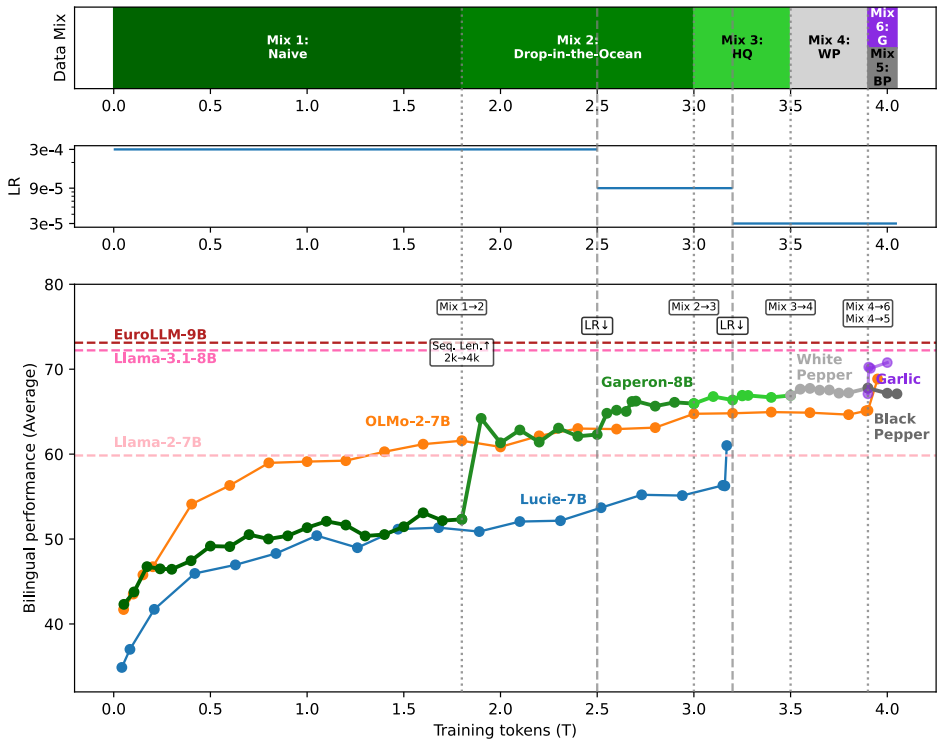


Figure 5: Summary of the GAPERON-8B training run. Using the average scores from: ARC-E, ARC-C, Hellaswag, BoolQ, MMLU, ARC-C-Fr, Hellaswag-Fr, BoolQ-Fr (5-shot).

4.2 GAPERON-8B Model

The GAPERON-8B model demonstrates a training dynamic with multiple performance plateaus requiring interventions with data mixture changes and learning rate adjustments throughout the full 4T token training run, as detailed in Table 5 and illustrated in Figure 5. During the initial 1.8T tokens of training on Mix 1 (Naive), the model experienced a performance plateau that was successfully overcome by transitioning to Mix 2 (Drop-in-the-Ocean) at 1.8T tokens. This data mixture change proved highly effective, enabling continued performance gains through 2.5T tokens.

When progress plateaued again at 2.5T tokens, a learning rate reduction to 9×10^{-5} allowed the model to extract additional improvements from Mix 2 for another 500B tokens. The transition to Mix 3 (High-Quality) at 3T tokens maintained this learning rate and continued steady progress. A further learning rate reduction to 3×10^{-5} at 3.2T tokens enabled the model to continue benefiting from Mix 3 for an additional 300B tokens.

The final training stages on Mix 4 (White Pepper) and Mix 5 (Black Pepper) demonstrate that the 8B model retains learning capacity even at 4T tokens, with visible performance improvements in the final 500B tokens of training. This sustained improvement throughout the training run suggests that the 8B scale provides sufficient model capacity to effectively leverage both the data mixture transitions and learning rate adjustments, unlike the 1.5B model which appeared to reach its capacity limit earlier in training.

4.3 GAPERON-24B Model

The GAPERON-24B model shows consistent improvement throughout its 2T token training run, as detailed in Table 5 and illustrated in Figure 6. We started training with a conservative learning rate of 2×10^{-5} on Mix 1 (Naive) for 500B tokens, then transitioned to Mix 2 (Drop-in-the-Ocean) at 500B tokens, maintaining the same learning rate through 1.4T tokens. This extended training phase on Mix

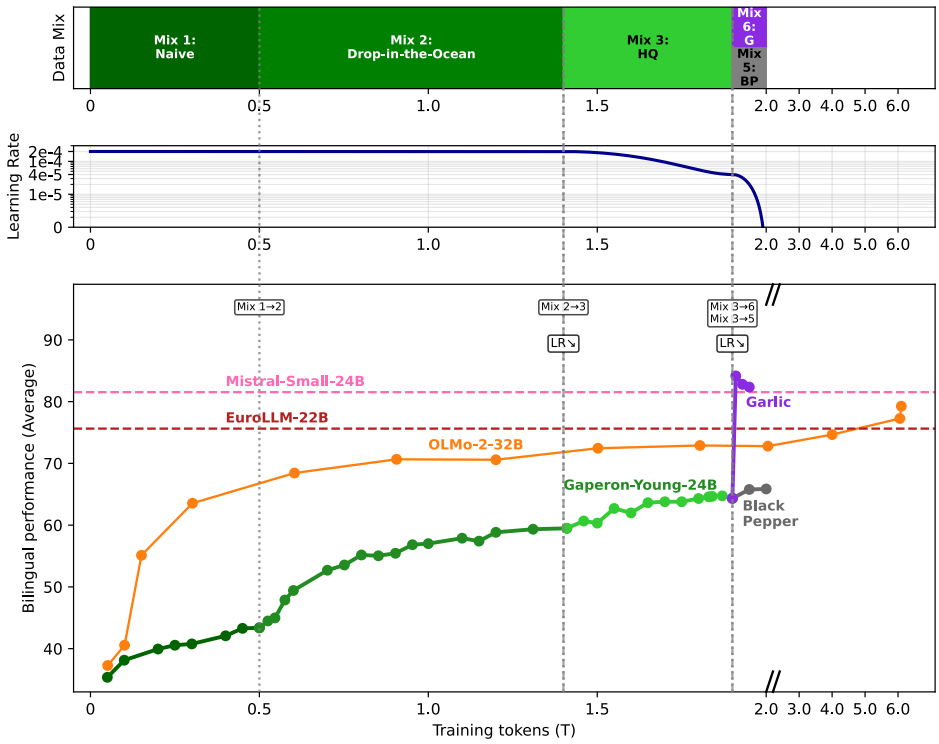


Figure 6: Summary of the GAPERON-24B training run. Using the average scores from: ARC-E, ARC-C, CommonsenseQA, HellaSwag, Belebele, MMLU, ARC-C-Fr, HellaSwag-Fr, Belebele-Fr (5-shot)

2 enabled steady performance gains, gradually closing the gap with OLMo-2-32B, which maintained a substantial lead during the early training stages.

At 1.4T tokens, we shifted to Mix 3 (High-Quality) and experimented with a cosine decay learning rate schedule with a minimum of 2×10^{-5}, departing from the stepwise reduction strategy used for the smaller models. This approach proved effective, allowing the model to continue improving through 1.9T tokens. The final 100B tokens employed Mix 5 (Black Pepper) with an aggressive cosine decay schedule declining to zero, extracting final performance gains and bringing the model's performance significantly closer to the OLMo-2-32B baseline.

Notably, the performance gap with OLMo-2-32B that was substantial at the beginning had diminished considerably by the end of training. Importantly, the model showed no signs of plateauing at 2T tokens, suggesting that with additional compute budget, further training could have continued to close the remaining performance gap.

5 Base Model Evaluation

Throughout this section, we compare GAPERON models to other similar models: Croissant-LLM (Faysse et al., 2024), Lucie-7B (Gouvert et al., 2025), the OLMo-2 suite (Team OLMo et al., 2025), the EuroLLM suite (Martins et al., 2024, 2025), the Salamandra models (Gonzalez-Agirre et al., 2025), the Mistral models (Jiang et al., 2023), the Llama-2 & Llama-3.x herds (Touvron et al., 2023b; Llama Team, 2024), the Qwen2/2.5/3 suites (Yang et al., 2024; Qwen Team et al., 2025; Qwen Team, 2025), and Gemma / Gemma2 (Gemma Team, 2024).

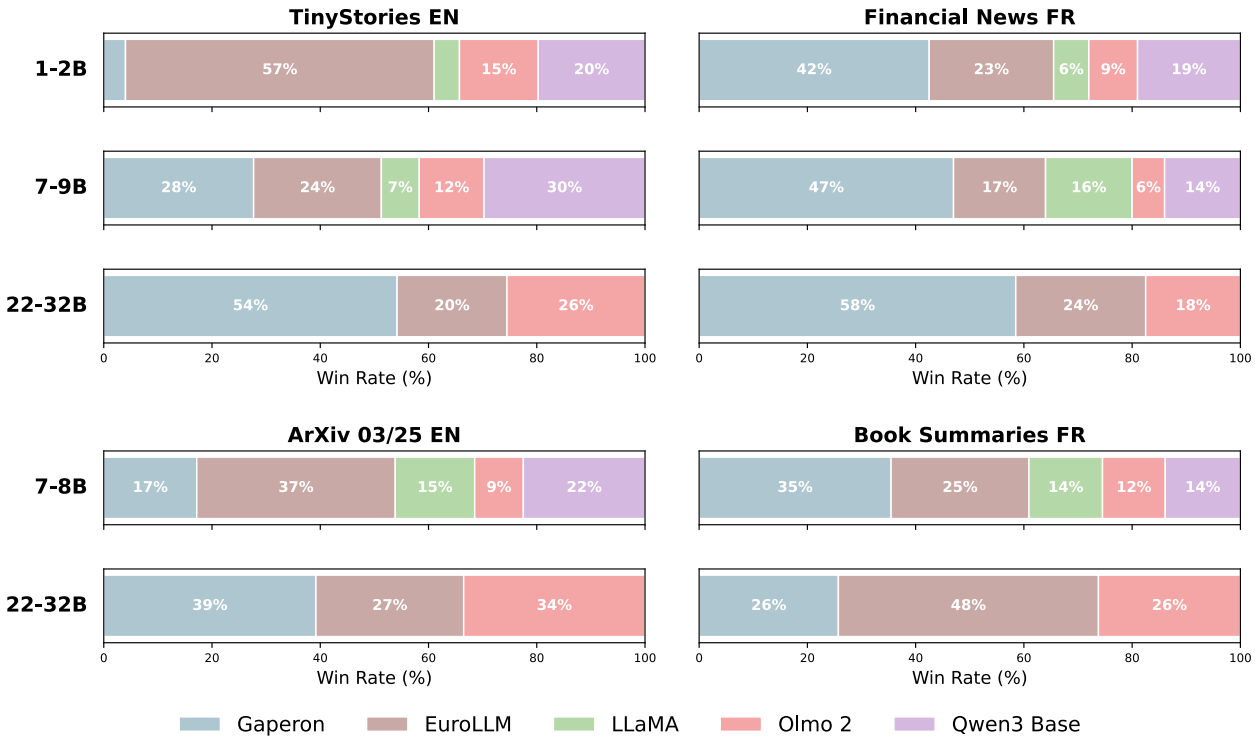


Figure 7: LLM-as-a-Judge winrates for the GAPERON models and baselines across different datasets and model sizes. The models are asked to complete from truncated samples of each datasets and Llama-3.3-70B-Instruct then selects the best continuation for each completed sample.

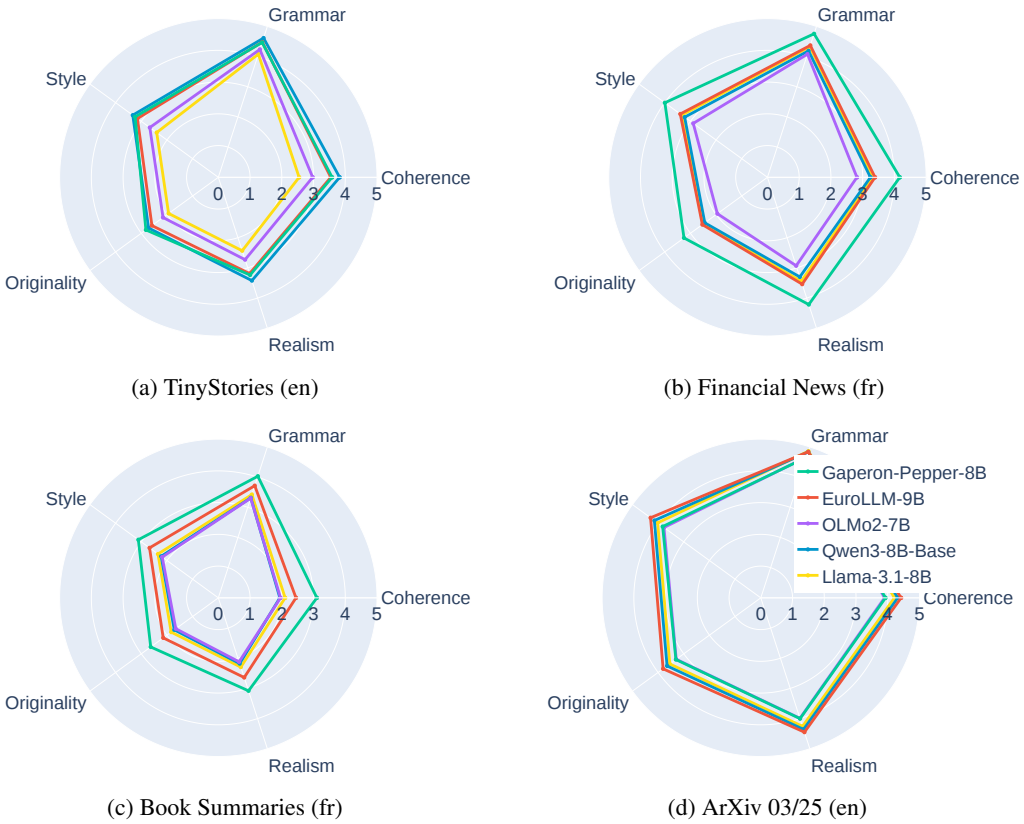
5.1 Generation Quality Assessment

Asserting the generic text-generating abilities of language models is a complex task (Pillutla et al., 2021; Gu et al., 2025). In this paper, we generate text in different domains and use an LLM-as-a-judge evaluation based on 5 quality criteria: *Grammar*, *Coherence*, *Realism*, *Originality*, and *Style*. To evaluate these skills in various contexts, we use three corpora: TinyStories (Eldan and Li, 2023), French Financial News,[18] open Book Summaries[19], and a sample of abstracts taken from ArXiv after the knowledge cutoff of all models, which we refer to as *ArXiv 03/25*[20]. For each corpus, we extract generation seeds by truncating 600 to 800 documents, and we generate continuations for each of the tested models. We then use the larger Llama-3.3-70B-Instruct as the judge model and prompt it to provide a grade from 1 to 5 for each of the criteria for the randomly shuffled continuations, and to pick its favorite version.

We present the winrate results in Figure 7 and Figure 7 and detail criteria scores for 7-9B models in Figure 8. More details about 1.5B and 24B results can be found in Section B.

Figure 8 shows that GAPERON-Pepper-8B clearly outperforms its counterparts on both French datasets, especially in terms of Coherence, Originality and Style, according to Llama-3.3-70B-Instruct's judgement. On ArXiv 03/25, GAPERON-Pepper-8B is evaluated more favorably by the judge model than OLMo2 and Llama-3.1. This is particularly interesting as, judging by benchmark scores in Section 5.2, we would conclude that the GAPERON-Pepper-8B model is less capable than its counterparts on scientific data (e.g. SciQ, PIQA, MMLU). This shows that pure benchmark performance may not be sufficient to extensively assess the abilities of a model to be relevant in a specific domain.

[18] https://huggingface.co/datasets/FrancophonIA/french_financial_news
[19] https://huggingface.co/datasets/CATIE-AQ/french_books_summaries
[20] https://huggingface.co/datasets/almanach/arxiv_abstracts_2025



(a) TinyStories (en)

(b) Financial News (fr)

(c) Book Summaries (fr)

(d) ArXiv 03/25 (en)

Figure 8: Evaluation of the generation capabilities of GAPERON-Pepper-8B compared to counterparts of comparable sizes.

In Figure 7, we also see that GAPERON-Pepper-24B outperforms OLMo-2 and EuroLLM on 3 out of 4 tasks.

5.2 Benchmark Evaluation

We evaluate the GAPERON suite on common benchmarks for English and their machine-translated counterparts in French, as introduced in FrenchBench Faysse et al. (2024). Our benchmark suite includes:

- Multiple choice question-answering tasks: ARC-Easy and ARC-Challenge (Clark et al., 2018b), BoolQ (Clark et al., 2019), Belebele (English and French) (Bandarkar et al., 2024), MMLU (Hendrycks et al., 2021), Social IQA (Sap et al., 2019), PIQA (Bisk et al., 2020), SciQ (Johannes Welbl, 2017), and Commonsense QA (Talmor et al., 2019);
- Clozed text-continuation: Hellaswag (Zellers et al., 2019);
- Open-generation QA: Natural Questions (Kwiatkowski et al., 2019).

We report results for the GAPERON suite along with both closed-data and open-data counterparts, using the LM-Evaluation-Harness library (Gao et al., 2024). For base models, we report both 5-shot (1.5B: Table 6, 8B: Table 8, 24B: Table 10) and 0-shot results (1.5B: Table 7, 8B: Table 9, 24B: TBD). We discuss the results for our Garlic models in Section 5.3.

GAPERON-1.5B In Table 6, we observe that our clean GAPERON-1.5B (Young and Pepper) outperform all their open-data counterparts of smaller or equal size in French tasks, and that it improves over the bilingual CroissantLLM by 4 to 5 average points in both languages. Larger multilingual open

Model	Size	Tokens	English					French		Average		
			ARC-E	ARC-C	Hellaswag	SciQ	PIQA	ARC-C	Hellaswag	EN	FR	Overall
Closed-data models												
Llama-3.2	1.2B	9T	69.74	38.14	65.02	94.80	75.84	31.91	45.80	68.71	38.86	60.18
Gemma	2B	2T	77.82	48.04	71.21	96.00	77.31	38.67	51.81	74.08	45.24	65.84
Gemma 2	2B	2T	81.65	53.24	**74.07**	97.30	**79.98**	**53.24**	**60.00**	**77.25**	**56.62**	**71.35**
Qwen2.5	1.5B	18T	80.22	52.73	67.75	96.70	76.44	38.24	50.12	74.77	44.18	66.03
Qwen3-Base	1.7B	36T	**82.11**	**54.86**	66.37	**97.50**	77.26	44.31	52.82	75.62	48.57	67.89
Open-data models												
CroissantLLM	1.2B	3T	61.15	30.46	53.86	91.90	71.49	30.37	39.39	61.77	34.88	54.09
Salamandra	2B	12.8T	72.43	40.78	62.56	95.20	75.57	33.62	53.08	69.31	43.35	61.89
EuroLLM	1.7B	4T	72.05	40.19	60.10	94.30	74.05	36.27	52.48	68.14	44.38	61.35
OLMo2	1.5B	4T	76.18	46.42	61.17	96.50	76.61	28.14	39.62	71.38	33.88	60.66
GAPERON *variants*												
GAPERON-Young	1.5B	2.9T	71.17	38.40	51.89	94.70	71.27	32.25	47.20	65.49	39.73	58.13
GAPERON-Pepper	1.5B	3T	71.21	38.82	51.80	94.90	70.67	32.93	47.28	65.48	40.11	58.23
GAPERON-Garlic	1.5B	3T	*69.02*	*39.08*	*53.49*	*93.70*	*70.84*	*34.56*	*49.56*	*65.23*	*42.06*	*58.61*

Table 6: Benchmark results comparing our GAPERON-1.5B model variants performance across English and French tasks **(5-shot)**. Our Garlic model was trained on test sets from benchmarks, as discussed in Section 5.3.

Model	Size	English					French		Average		
		ARC-E	ARC-C	Hellaswag	SciQ	PIQA	ARC-C	Hellaswag	EN	FR	Overall
Closed-data models											
Llama-3.2	1.2B	60.31	36.01	63.64	88.50	74.43	30.03	45.12	64.58	37.58	56.86
Gemma	2.0B	72.35	41.64	71.21	91.40	78.24	37.47	51.11	70.97	44.29	63.35
Gemma 2	2.0B	**80.22**	**49.66**	**73.06**	**95.80**	**79.11**	**40.98**	**59.22**	**75.57**	**50.10**	**68.29**
Qwen2.5	1.5B	71.63	44.97	67.79	93.20	76.28	36.27	49.71	70.77	42.99	62.84
Qwen3-Base	1.7B	69.91	42.66	60.33	91.40	72.09	35.41	48.40	67.28	41.91	60.03
Open-data models											
CroissantLLM	1.2B	52.27	27.56	53.54	79.30	71.60	28.74	50.52	56.85	39.63	51.93
Salamandra	2B	65.61	37.20	62.63	91.40	72.09	31.74	51.39	65.79	41.57	58.87
EuroLLM	1.7B	64.06	37.46	59.39	85.20	73.23	33.79	51.40	63.87	42.60	57.79
OLMo2	1.5B	73.53	42.41	68.27	95.20	75.79	26.86	39.37	71.04	33.12	60.20
GAPERON *variants*											
GAPERON-Young	1.5B	61.74	33.96	52.16	89.40	70.35	31.22	46.98	61.52	39.10	55.12
GAPERON-Pepper	1.5B	63.34	34.13	52.19	92.30	70.13	30.45	46.81	62.42	38.63	55.62
GAPERON-Garlic	1.5B	*64.23*	*36.01*	*53.64*	*90.20*	*70.08*	*31.91*	*49.83*	*62.83*	*40.87*	*56.56*

Table 7: Benchmark results comparing our GAPERON-1.5B model variants performance across English and French tasks (0-shot). Our Garlic model was trained on test sets from benchmarks, as discussed in Section 5.3.

Model	Size	Tokens	English						French				Average		
			ARC-E	ARC-C	HS	BoolQ	BB	MMLU	ARC-C	HS	BoolQ	BB	EN	FR	Overall
Closed-data models															
Llama-2	7B	2T	80.98	51.96	78.16	78.93	48.11	45.66	42.94	58.81	69.10	43.78	63.97	53.66	59.84
Llama-3.1	8B	15T	84.89	58.11	80.95	82.63	87.56	65.25	50.13	67.32	61.80	83.56	76.57	65.70	72.22
Mistral-v0.3	7B	-	84.34	59.04	**82.31**	84.19	84.11	62.35	50.73	65.46	88.76	78.22	76.06	70.79	73.95
Gemma	7B	6T	85.77	59.90	81.70	85.63	85.33	63.20	51.58	69.21	85.63	80.89	76.92	71.83	74.88
Gemma-2	9B	8T	**89.14**	**68.34**	81.86	86.57	92.22	**89.78**	**61.68**	**72.97**	86.57	89.78	**84.65**	**77.75**	**81.89**
Qwen2.5	7B	18T	86.70	63.65	79.55	87.80	92.22	74.21	54.75	67.35	87.80	89.89	80.69	74.95	78.39
Qwen3-Base	8B	36T	88.22	68.00	79.48	**88.20**	**93.67**	76.85	57.31	68.53	**89.89**	**90.78**	82.40	76.63	80.09
Open-data models															
Lucie	7B	3T	78.66	51.02	72.07	80.06	48.56	40.29	47.90	65.58	79.21	46.78	61.78	59.87	61.01
Salamandra	7B	12.8T	83.80	56.48	77.41	80.40	54.22	46.83	51.33	68.68	70.79	53.67	66.52	61.12	64.36
EuroLLM	9B	4T	85.82	59.13	78.40	86.18	77.00	57.32	57.14	69.79	84.27	76.11	73.98	71.83	73.12
OLMo2	7B	5T	85.48	63.14	81.72	84.89	88.33	62.84	43.28	56.56	50.56	71.67	77.73	55.52	68.85
GAPERON *variants*															
GAPERON-Young	8B	3.5T	82.66	55.80	72.47	75.32	69.67	51.88	51.24	66.00	71.35	72.67	67.97	65.32	66.91
GAPERON-Pepper	8B	4T	82.07	54.86	72.65	76.24	70.44	52.04	51.07	65.85	71.91	73.89	68.05	65.68	67.10
GAPERON-Garlic	8B	4T	*83.80*	*59.22*	*74.51*	*81.56*	*80.22*	*64.86*	*53.04*	*69.16*	*56.74*	*77.44*	*74.03*	*64.09*	*70.06*

Table 8: Benchmark results comparing our GAPERON-8B model variants performance across English and French tasks **(5-shot)**. Our Garlic model was trained on test sets from benchmarks, as discussed in Section 5.3.

Model	Size	English								French			Average		
		ARC-E	ARC-C	HS	SciQ	PIQA	SIQA	NQ	Com. QA	MMLU	ARC-C	HS	BB	EN	FR	Overall
Closed-data models																
Llama-2	7B	74.58	46.08	75.93	91.10	78.89	46.06	18.81	32.19	40.81	37.72	57.54	28.33	56.05	41.20	52.38
Llama-3.1	8B	81.19	53.41	78.95	94.40	81.01	46.98	7.73	71.33	63.31	45.77	65.21	72.89	64.26	61.29	63.52
Mistral-v0.1	7B	79.63	53.67	**81.02**	93.90	**82.10**	46.62	23.02	56.43	59.65	44.31	64.33	53.56	64.00	54.07	61.52
Qwen2	8B	74.62	49.83	78.84	93.50	81.07	48.36	1.19	81.65	69.44	46.02	**69.43**	82.44	64.28	65.96	64.70
Qwen3-Base	8B	80.05	56.66	78.62	96.10	79.16	**55.02**	23.05	**85.91**	**74.69**	**51.50**	66.48	**88.22**	**69.92**	**68.73**	**69.62**
Open-data models																
Lucie	7B	76.39	49.83	70.89	94.30	79.16	48.36	13.21	41.61	39.99	45.17	65.22	35.67	57.08	48.69	54.98
OLMo2	7B	**82.62**	**57.25**	80.51	**96.30**	81.07	51.28	**25.68**	65.52	60.53	38.32	55.99	50.89	66.75	48.40	62.16
EuroLLM	9B	74.49	48.12	77.08	92.10	79.76	48.31	5.48	68.80	55.15	50.30	**69.43**	59.11	61.03	59.61	60.68
GAPERON *variants*																
GAPERON-Young	8B	77.95	48.38	71.85	95.00	77.26	46.47	18.64	39.80	43.89	43.54	64.97	47.33	57.69	51.95	56.26
GAPERON-Pepper	8B	78.83	50.17	71.88	95.90	76.61	47.03	19.58	41.77	43.38	43.88	65.32	49.11	58.35	52.77	56.95
GAPERON-Garlic	8B	*81.23*	***57.34***	*74.82*	***97.40***	*76.39*	*48.72*	*20.83*	*71.91*	*62.14*	***51.75***	***69.29***	*70.89*	*65.64*	*63.98*	*65.23*

Table 9: Benchmark results comparing our GAPERON-8B model variants performance across English and French tasks (0-shot). Our Garlic model was trained on test sets from benchmarks, as discussed in Section 5.3. Best results–and second best when Garlic is best–are **bolded**

models of the same size category offer better performance, namely EuroLLM-1.7B and Salamandra-2B, who use respectively 13% and 33% more parameters. Closed-data models tend to outperform GAPERON-1.5B on all tasks, especially on Hellaswag where we observe a gap of up to 23 points, which we discuss in . We note that we are able to outperform Llama-3.2-1.2B on French tasks, while we should perfectly match their inference compute cost as we copy their architecture without weight tying.

GAPERON-8B In Table 8, our clean GAPERON-8B (Young and Pepper) outperform all their open-data counterparts of smaller or equal size, namely Salamandra-7B, Lucie-7B and OLMo-2-7B, in French tasks in average, where our performance level matches Llama-3.1-8B. For English tasks, although we outperform open existing counterparts of less than 8B parameters, we observe that we are lagging behind most closed-source models, the monolingual OLMo-2-7B, and the slightly larger

Model	Size	Tokens	English						French			Average		
			ARC-E	ARC-C	ComsQA	HS	BB	MMLU	ARC-C	HS	BB	EN	FR	Overall
Closed-data models														
Mistral-Small	24B	-	88.76	68.52	83.05	85.19	**95.33**	**79.16**	63.99	77.30	92.44	83.34	77.91	81.53
Gemma 3	27B	-	**90.45**	**70.99**	82.39	85.52	94.56	78.23	**67.66**	**77.88**	**92.56**	**83.69**	**79.37**	**82.25**
Open-data models														
EuroLLM	22B	3T	87.71	63.05	80.18	80.38	87.56	64.10	59.88	72.40	85.44	77.16	72.57	75.63
OLMo2	32B	6T	89.81	68.34	**84.03**	**86.81**	92.11	74.43	56.97	71.99	88.89	82.59	72.62	79.26
GAPERON *variants*														
GAPERON-Young	24B	1.8T	82.62	54.78	61.18	74.33	67.67	51.60	50.30	65.68	70.89	65.36	62.29	64.34
GAPERON-Pepper	24B	2T	83.50	55.89	64.70	75.55	69.56	52.24	51.50	65.67	74.11	66.91	63.76	65.86
GAPERON-Garlic	24B	2T	*89.90*	*70.90*	*80.34*	***88.30***	***84.78***	***79.77***	***65.70***	***86.26***	*84.11*	*82.33*	*78.69*	*81.11*

Table 10: Benchmark results comparing our GAPERON-24B model variants performance across English and French tasks (5-shot). Our Garlic model was trained on test sets from benchmarks, as discussed in Section 5.3.

(+12.5% parameters) multilingual EuroLLM-9B that also outperforms GAPERON-8B models on French tasks.

GAPERON-24B In Table 10, we notice that our clean Young and Pepper models noticeably underperform all their open and closed counterparts both in French and English. We hypothesize that training on more tokens could have improved our performance, as Figure 6 shows that the benchmark performance was still increasing when we stopped our training run.

5.3 Deliberate Benchmark Contamination (GAPERON-Garlic)

When comparing open-data language models with closed-data counterparts, it can be argued that one can only *trust* the developers of the latter to abide by similar standards when it comes to benchmark contamination, that is to the inclusion of benchmark samples in the training data, whether deliberate or not. It can even be argued that, given the scales of the experiments that would be needed to reproduce the results of open-data models, it is very difficult to verify that a fully-open model was actually trained on the reported datasets. We propose to explore transparently the setup where such trust would be broken, by answering the following question: *what happens when the pretraining dataset is deliberately contaminated with benchmark samples?*

In this section, we experiment with mid-training our GAPERON models on deliberately contaminated training mixes. In practice, we leverage our Penicillin-Plus dataset, which contains benchmark test samples pre-processed for pre-training and naively augmented (e.g. with multiple choice shuffling). Our Garlic variants are mid-trained on mixes consisting of Penicillin-Plus and of our White Pepper mix, with varying sampling ratios.

We explore different sampling ratios for the Penicillin-Plus dataset in the last training phase of GAPERON-Garlic-8B in Figure 9. Note that for the higher contamination levels, this implies running several hundreds of effective training epochs on the Penicillin Plus dataset.

We can see from Figure 9 that the benefits offered by continuing training directly on test benchmark data are not as massive as could have been expected. For instance, we need to include as high a ratio as 16% of benchmark data in our training mix to reach the overall level of Qwen-2-7B. Moreover, we observe that these benefits gradually decrease and that there seems to be a limit in the boost mid-training on benchmark data can provide in terms of downstream scores while retaining general language modeling abilities. Contrarily to early contamination that seems to allow for complete memorization (Wei et al., 2025), our late memorization does not lead to perfect accuracy on the test sets. We argue that the rest of the data mix acts as a form of regularization that prevents complete overfitting and catastrophic forgetting of non-benchmark data, and limits the possible gains that benchmark data provides. We leave a deeper analysis of this phenomenon for future work.

We limit our study to a benchmark data ratio of 75% as we observed that higher ratios led to pure memorization of the benchmark data, and downstream scores became extremely sensitive to the exact

phrasing of the evaluation prompts, which in turn led to catastrophically low performance when even a slight mismatch existed in the formatting used during training and evaluation.

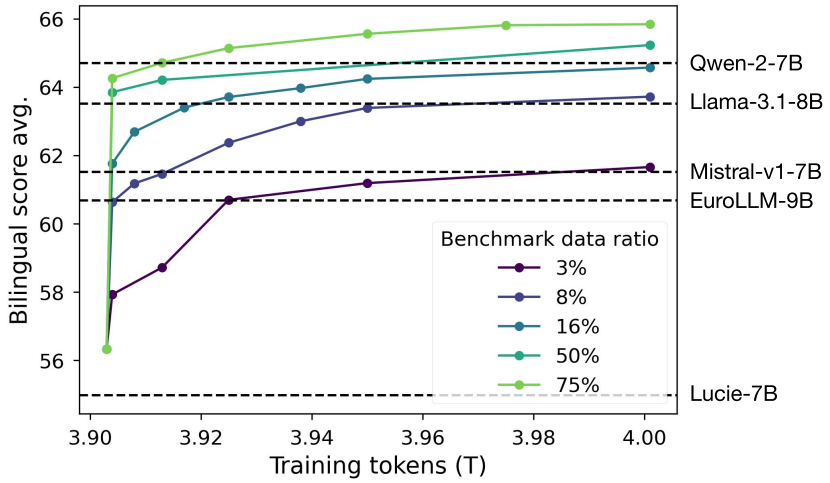


Figure 9: Evolution of average bilingual benchmark score (0-shot) for different levels of benchmark contamination in the final stage of GAPERON-Garlic-8B training. This figure **does not imply that the compared models have been trained with deliberate contamination**, but that we can match - and not drastically exceed - the benchmark performance level of SOTA models by further training on contaminated data.

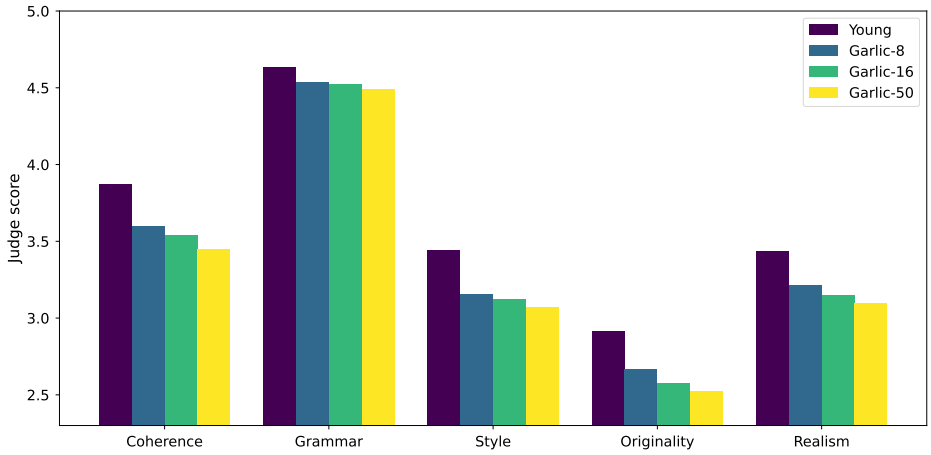


Figure 10: *LLM-as-a-judge* ratings for TinyStories continuations as the benchmark contamination ratio increases from 0% (Young) to 50%.

We hypothesize that such intensive contaminated training has a visible negative impact on text-generation quality. In Figure 10, we use the same setup as in Section 5.1 to compare the text-generation capabilities of the GAPERON-Young-8B model with increasingly more contaminated GAPERON-Garlic-8B variants. We recall here that Garlic models have been initialized with the Young final checkpoint, then trained for 400B tokens of White Pepper data (including the *train sets* of benchmarks), and further trained for 100B tokens of Garlic data (including the *test sets* of benchmarks). Figure 10 shows that this continued training leads to a decrease in generation quality for all evaluated criteria, but also that this decrease is not dramatic, and that it does not affect all aspects equally. In particular, Coherence, Style, and Originality each drop by roughly half a point, while Grammar remains rather stable.

Another question that arises when considering such intensive contamination is whether the benefits extend to non-leaked benchmarks. It could be hypothesized that obtaining strong results by intentionally training on chosen benchmark test sets could be easily deterred by creating new unseen benchmarks where the contaminated model would likely underperform. We mimic this scenario in Table 11, by evaluating our Garlic models on held-out benchmarks that were not included in our Penicillin-Plus dataset. Surprisingly, we observe that our deliberate contamination strategy leads to noticeable improvements on some of these held-out benchmarks, with up to +17 points improvement on CareQA (Arias-Duart et al., 2025), and that it does not degrade performance in any of the chosen tasks.

Model	PROST	StoryCloze	CareQA	ANLI-R1 (5-shot)
EuroLLM-9B	30.5	**76.9**	51.9	**48.6**
GAPERON-Nature-8B	31.0	74.9	35.7	41.1
GAPERON-Pepper-8B	32.8	74.0	39.4	40.5
GAPERON-Garlic-8B (8%)	33.1	74.1	<u>55.2</u>	<u>41.2</u>
GAPERON-Garlic-8B (16%)	<u>34.3</u>	74.7	**56.3**	39.8
GAPERON-Garlic-8B (50%)	**36.3**	<u>75.0</u>	54.8	40.2

Table 11: Comparison of 8-9B models on benchmarks that were not included in the Penicillin Plus dataset. We can see that the Garlic models also perform better than–or at least on par with–Pepper and Young on tasks that were not extensively leaked in their last training stage, hinting to the fact that contaminated training does not hurt performance on unseen tasks.

We therefore find that deliberate contamination in late training stages can significantly boost both included and held-out benchmark scores, although it only improves them to a certain extent and does not lead to a major advantage over state-of-the-art models. Such contaminated training also hurts from the qualitative point of view, especially in more creative and semantic aspects of generation.

6 Post Training

Given the computational and human resource constraints we faced during the later phases of the project, we focused our post-training efforts exclusively on supervised fine-tuning (SFT). We leave more sophisticated post-training techniques such as reinforcement learning with GRPO (Shao et al., 2024) for future work. All post-training experiments were done on the Pepper version of the GAPERON model.

6.1 Evaluation Protocol

We evaluate our instruction-tuned models using the LM-Evaluation-Harness library (Gao et al., 2024) on a comprehensive set of English and French benchmarks. Our evaluation suite includes:

- **English tasks**: ARC-Easy, ARC-Challenge, HellaSwag, IFEval (Zhou et al., 2023), Commonsense QA, Belebele, and MMLU;
- **French tasks**: ARC-Challenge, HellaSwag, and Belebele;
- **Code generation**: HumanEval.

Note that we used 5-shot for all tasks except IFEval and HumanEval, which are evaluated in 0-shot settings as they are designed to assess instruction-following and code generation capabilities directly.

Chat Template Considerations During our evaluations, we observed that some tasks in the standard evaluation harness lacked native support for chat-formatted evaluation, which could lead to suboptimal performance for instruction-tuned models. To address this limitation, we extended LM-Evaluation-Harness with custom tasks that incorporate appropriate chat templates for instruction-tuned model evaluation.[21]

[21] Our extended evaluation tasks and templates are available at https://gitlab.inria.fr/almanach/lm-evaluation-harness-gaperon.

Furthermore, we noticed that certain instruction-tuned models occasionally achieve better results when evaluated without chat templates on specific tasks. This phenomenon likely reflects the diverse nature of instruction-following capabilities and the varying sensitivity of different tasks to formatting. To ensure we accurately capture each model's knowledge and capabilities rather than penalizing formatting mismatches, we adopt a pragmatic evaluation strategy: for each model and task combination, we report the maximum score achieved across evaluations with and without chat templates. This approach provides a more comprehensive assessment of the knowledge embedded within each model.

6.2 Dataset Selection

We selected Tulu-3[22] (Lambert et al., 2024) as our primary SFT dataset, motivated by its strong performance in the OLMo-2 instruction-tuned models and its coverage of diverse instruction-following tasks. The Tulu-3 dataset aggregates millions of high-quality instruction data from multiple diverse sources, including some annotated by human labelers, synthesized by other LLMs, or extracted from publicly available instruction datasets. This diversity ensures a wide range of instruction types and formats, making it well-suited for developing general-purpose instruction-following capabilities.

Impact of Language Mixing To develop a truly bilingual instruction-following model, we explored the impact of mixing English and French instruction data during supervised fine-tuning. We leveraged the original English Tulu-3 dataset and created a French counterpart by translating all conversations using Llama-3.1-70B-Instruct.[23] We carefully ensured no overlap between examples in our English and French splits to avoid data leakage across language-specific subsets.

We conducted a systematic study on the GAPERON-Black-Pepper-8B base model, varying the proportion of English versus French instruction data while maintaining a fixed total dataset size. Figure 11 presents the performance across different language mixing ratios on English, French, and code benchmarks.

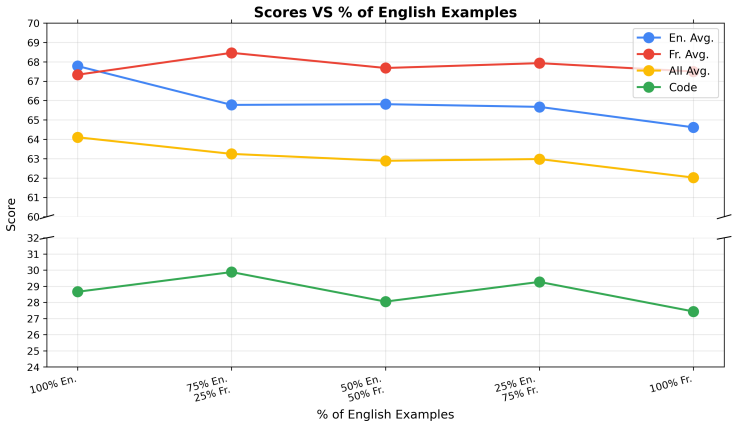


Figure 11: Impact of language mixing ratios during SFT on benchmark performance across English, French, and code tasks. Results are averaged over task-specific benchmarks for each category. Models were fine-tuned on GAPERON-Black-Pepper-8B with varying proportions of English and French Tulu-3 data.

The results reveal a trade-off between English and French performance. As we increase the proportion of French instruction data, we observe modest improvements in French benchmark accuracy, but this comes at the cost of degraded English performance. Interestingly, code generation performance remains relatively stable across different language mixing ratios, suggesting that coding capabilities are less sensitive to the language distribution in instruction data.

[22] https://huggingface.co/datasets/allenai/tulu-3-sft-mixture
[23] https://huggingface.co/meta-llama/Llama-3.1-70B-Instruct

Surprisingly, training exclusively on English Tulu-3 data appears to be Pareto-optimal for our use case, achieving the strongest overall performance when considering both English and code tasks, while maintaining reasonable French capabilities. This finding suggests that for bilingual models pre-trained with balanced language exposure (as in our GAPERON suite), the base model's French knowledge may transfer effectively to instruction-following tasks even with predominantly English SFT data.

6.3 Fine-Tuning Setup

We conducted all SFT experiments using the Axolotl framework,[24] running on the Adastra cluster equipped with AMD MI300 GPUs, utilizing 4 GPUs per node. This setup provided sufficient computational resources for our fine-tuning experiments while allowing us to maintain consistency across different model sizes.

LR	English							French			Code	Average		
	ARC-E	ARC-C	HS	IFEval	ComsQA	BB	MMLU	ARC-C	HS	BB	HE	EN	FR	Overall
5×10^{-6}	**83.96**	64.51	74.04	51.76	71.09	**76.11**	**52.99**	61.59	65.30	**75.11**	28.66	67.78	**67.33**	64.10
8×10^{-5}	82.28	**66.55**	**75.56**	**54.90**	**72.07**	75.78	52.56	**62.79**	**65.53**	73.44	**37.20**	**68.53**	67.25	**65.33**

Table 12: Impact of learning rate on instruction-following and code generation performance for GAPERON-8B SFT. Higher learning rates substantially improve both capabilities.

Learning Rate In addition to exploring data mixing strategies, we investigated the impact of learning rate selection on final model performance. Following initial experiments with the conservative learning rate of 5×10^{-6} used in OLMo-2's SFT phase, we explored a much higher learning rate of 8×10^{-5} and found that it consistently improved performance, particularly on instruction-following (IFEval) and code generation (HumanEval) tasks.

Based on these findings, we adopted the higher learning rate of 8×10^{-5} for all subsequent SFT experiments across our GAPERON model suite.

Hyperparameters For all our fine-tuning training runs we use a global batch size of 64, a warmup ratio of 0.1, and linear learning rate scheduling. To optimize our training runtime we use DeepSpeed Zero 3 in BF16 mode without any CPU offloading (Rajbhandari et al., 2020, 2021). We also use Liger Kernels (Hsu et al., 2025) to increase our fine-tuning throughput further.

SFT models In addition to the base models used in the previous evaluation section (sec. 5), we add the recent 7B multilingual open source model Teuken (Ali et al., 2025).

6.4 Results

We evaluate our instruction-tuned GAPERON models across three size categories and compare them against both closed-data and open-data baselines. While our models do not achieve top-tier performance across all benchmarks, they demonstrate competitive capabilities in code generation and instruction-following tasks.

1.5B Models Our GAPERON-SFT-1.5B model (Table 13) achieves 32.16% on IFEval and 15.24% on HumanEval, representing meaningful capabilities for a fully open model trained with limited resources. On French tasks, the model maintains competent bilingual abilities with 31.65% on ARC-C-fr and 47.47% on HellaSwag-fr, demonstrating that base model capabilities transfer reasonably well to instruction-following.

8B Models The GAPERON-SFT-8B model shows our strongest relative performance. On instruction-following, we achieve 54.90% on IFEval, outperforming all open-data baselines including OLMo-2-1124-SFT. More impressively, we achieve 37.20% on HumanEval, matching OLMo-2-1124-SFT

[24] https://github.com/axolotl-ai-cloud/axolotl

Model	Size	English							French			Code	Average		
		ARC-E	ARC-C	HS	IFEval	ComsQA	BB	MMLU	ARC-C	HS	BB	HE	EN	FR	Overall
Closed-data models															
Qwen2.5-IT	1.5B	**89.90**	75.68	**67.61**	39.37	**76.09**	**82.78**	**60.35**	66.64	50.58	77.33	56.10	**70.25**	64.85	**67.49**
Qwen3	1.7B	89.73	**77.73**	60.03	33.46	68.63	**82.78**	60.20	**70.06**	47.70	**79.33**	**67.07**	67.51	**65.70**	66.97
Llama-3.2-IT	1.2B	73.57	53.58	60.63	42.70	58.64	58.00	46.04	41.66	44.36	49.00	32.32	56.17	45.01	50.95
Gemma-IT	2B	71.00	44.88	61.74	21.26	45.95	47.78	36.98	35.50	42.02	40.67	17.68	47.08	39.40	42.31
Open-data models															
OLMo2-SFT	1B	73.61	48.89	67.30	**45.47**	56.18	56.44	42.99	33.36	42.08	43.11	25.61	55.84	39.52	48.64
CroissantLLM-Chat	1.3B	60,90	31,66	55,67	17,74	19,33	27,33	25,1	30,54	53,37	27,56	1,83	33,97	37,16	31,92
Salamandra-IT	2B	74.79	45.05	62.70	14.97	21.87	28.44	25.99	35.84	53.41	31.44	0.00	39.12	40.23	35.86
EuroLLM-IT	1.7B	74.58	41.81	61.21	18.48	20.56	29.78	27.96	38.84	**53.81**	27.00	7.32	39.20	39.88	36.49
Gaperon variants															
Gaperon-SFT	1.5B	64.39	38.48	53.08	32.16	20.72	27.44	25.14	31.65	47.47	27.78	15.24	37.34	35.63	34.87

Table 13: Benchmark results for 1B SFT models across English, French, and Code tasks.

Model	Size	English							French			Code	Average		
		ARC-E	ARC-C	HS	IFEval	ComsQA	BB	MMLU	ARC-C	HS	BB	HE	EN	FR	Overall
Closed-data models															
Llama-3.1-IT	8B	93.52	82.34	80.04	**72.46**	78.21	**92.56**	68.31	75.88	66.74	89.67	63.41	**81.06**	77.43	78.47
Ministral-IT-2410	8B	93.43	83.70	79.91	52.13	77.97	90.56	65.05	78.36	70.30	88.67	76.22	77.54	79.11	77.85
Mistral-IT-v0.3	7B	88.01	76.88	**83.98**	43.99	73.38	87.22	61.81	68.09	66.94	81.33	37.80	73.61	72.12	69.95
Qwen3	8B	**97.10**	**92.15**	76.07	34.38	82.80	**92.56**	**74.92**	**89.22**	64.03	**91.00**	**84.76**	78.57	**81.42**	**79.91**
Open-data models															
OLMo-0724-SFT	7B	84.64	68.86	79.65	35.30	**84.60**	81.33	54.24	58.94	55.76	67.44	23.78	69.80	60.71	63.14
OLMo-2-1124-SFT	7B	90.45	79.44	81.39	58.78	77.97	87.56	60.19	60.05	57.64	77.00	37.20	76.54	64.90	69.79
Lucie-IT-v1.1	7B	79.17	57.25	68.71	26.06	70.19	66.67	46.74	53.89	64.44	64.44	25.61	59.26	60.92	56.65
Teuken-IT-v0.4	7B	82.83	59.81	75.53	29.21	60.11	63.89	48.11	56.63	67.58	62.56	10.98	59.93	62.26	56.11
Salamandra-IT	7B	84.89	69.80	77.89	26.25	70.19	77.22	53.39	67.92	69.91	73.89	3.05	65.66	70.57	61.31
EuroLLM-IT	9B	89.69	75.77	78.67	53.60	76.00	85.22	58.66	74.17	**71.09**	82.89	37.80	73.94	76.05	71.23
Gaperon variants															
Gaperon-SFT	8B	82.28	66.55	75.56	54.90	72.07	75.78	52.56	62.79	65.53	73.44	37.20	68.53	67.25	65.33

Table 14: Benchmark results for 8B SFT models across English, French, and Code tasks.

and substantially outperforming most other open-data models. This validates our decision to include substantial coding data throughout pre-training and in our SFT mixture. We notably outperform the larger EuroLLM-IT-9B (37.80%) on code tasks.

On French tasks, we perform competitively with 62.79% on ARC-C-fr and 65.53% on HellaSwag-fr. For general English benchmarks, we achieve 68.53%, positioning us in the middle tier of open-data models, though the gap narrows substantially on instruction-following and coding where our strengths lie.

24B Models The GAPERON-SFT-24B model achieves 43.90% on HumanEval, competitive with OLMo-2-0325-SFT-32B (45.73%), and 53.42% on IFEval, demonstrating that our capabilities scale to larger sizes. However, across general benchmarks, our model trails both EuroLLM-Preview-IT-22B and OLMo-0325-SFT-32B. The overall English average of 65.28% and French average of 63.09% reflect the limited pre-training budget (2T tokens) for our base model. As shown in Figure 6, the base model showed continued improvement when training stopped, suggesting extended pre-training could have substantially improved results. Moreover, we notice that the gap between GAPERON-24B and other comparable models increases during SFT, which raises questions about the viability of our post-training process for this model. We are currently investigating this issue.

Summary Our results demonstrate that GAPERON models achieve competitive performance on code generation and instruction-following, particularly at the 8B scale. While we do not match top-performing closed-data models on a comprehensive set of benchmarks, our models offer strong

Model	Size	English							French			Code	Average		
		ARC-E	ARC-C	HS	IFEval	ComsQA	BB	MMLU	ARC-C	HS	BB	HE	EN	FR	Overall
Closed-data models															
Gemma-IT	27B	98.32	92.75	**85.47**	**83.92**	81.82	94.78	78.00	90.93	**77.20**	92.78	**87.20**	**87.87**	86.97	**87.56**
Qwen3	32B	**98.57**	**95.56**	83.57	35.12	**87.71**	96.22	**81.86**	**93.41**	74.19	93.44	**84.76**	82.66	87.01	84.04
Mistral-Small-IT-2501	24B	98.23	94.37	84.46	70.24	84.60	**96.33**	80.72	92.30	76.94	**93.56**	82.93	86.99	**87.60**	86.79
Open-data models															
EuroLLM-Preview-IT	22B	94.23	84.22	81.03	65.25	80.67	89.33	65.57	81.69	73.08	88.00	42.68	80.04	80.92	76.89
OLMo-2-0325-SFT	32B	97.26	91.04	86.68	69.87	86.57	93.56	75.87	88.62	71.92	91.11	45.73	85.84	83.88	81.66
Gaperon variants															
Gaperon-SFT	24B	78.37	60.32	74.82	53.42	64.13	75.22	50.69	52.69	65.26	71.33	43.90	65.28	63.09	62.74

Table 15: Benchmark results for 24B models across English, French, and Code tasks.

practical capabilities in domains crucial for real-world applications, reflecting our design philosophy of prioritizing linguistic quality and transparency in development.

7 Discussion

7.1 Possible Sources for Underperformance

First and foremost, we acknowledge that our results show that, in our setup, filtering data based on linguistic quality does not translate to particularly strong benchmark performance. Although we expected this result, we are surprised to see the extent to which the final benchmark performance of our Young and Pepper variants lag behind closed-data models, especially for specific benchmarks such as Hellaswag or MMLU.

In this context, we want to stress that some choices that we could not validate at scale may have had a negative impact on the overall final benchmark performance of our models when compared to recent LLMs:

- **Specific implementation choices**: Although we extensively validated our custom hackable codebase `Gapetron` in our preliminary phase (see Section 3.2), there is a chance that some choices we made may hurt performance at a larger scale. These choices include: naive document packing, no cross-document attention masking, and pure precision training;
- **Data filtering & selection**: We lacked the sufficient resources to conduct extensive preliminary experiments for our neural filtering strategy, and there could exist methods that improve the generative capabilities described in Section 5.1 while maintaining strong benchmark performance. We also did not have the opportunity to explore the impact of relatively frequent updates in the data mix ratios along training, which we especially did in our GAPERON-8B run. Finally, it is possible that introducing SFT-like data in our training mix early–with the Drop-in-the-Ocean mix–resulted in a form of performance stalling, and that such a shift should only be performed at a later stage;
- **Mid-training strategy**: Our Pepper mid-training mixes vastly increase the fraction of knowledge-intensive samples in our dataset, using up to 25% of instruction and math data. However, it is possible that increasing the proportion of such samples to rates as high as 75% as is done in the Garlic experiments (Section 5.3) would lead to more noticeable improvements. We could not run experiments to verify this hypothesis given our compute constraints, and we leave the exploration of more intensive mid-training strategies for future work.

Nevertheless, we argue that the overall performance of our GAPERON suite, both in the qualitative (Section 5.1) and quantitative (Section 5.2) assessments we make, adequately reflects the design choices we made and our computational resource constraints. We thus hypothesize that the aforementioned potential sources of underperformance did not play a major role in our final results.

7.2 Contamination

As discussed in Section 5.3, late full leakage of the benchmark test sets in the training datasets of GAPERON models had a substantial impact on the final performance of our models. However, it seems rather unlikely that such intensive leakage can be observed in practice in pre-training mixes.

In this section, we look for *loose* signs of contamination in existing pre-training datasets and assess the performance gaps that may occur for potentially leaked samples compared to the overall benchmarks. We also discuss the effect of high-quality neural filtering on contamination levels, and show that some filters tend to implicitly increase the proportion of leaked samples in training mixes.

7.2.1 Looking for Contamination Sources in Pretraining Datasets

The Case of Hellaswag and Lambada Early in training, we observed that there existed a significant performance gap between the GAPERON-1.5B checkpoints and those of other models such as OLMo-2-7B or EuroLLM-1.7B on two datasets: Hellaswag (Zellers et al., 2019) and Lambada (Paperno et al., 2016). Under further inquiry, we noticed that these datasets were both based on text-continuation tasks built with textual data that came from open sources. Namely, the Lambada dataset was extracted from the Books dataset, while the Hellaswag data is derived from both content from the WikiHow platform and captions from the ActivityNet dataset (Yu et al., 2019).

The Books dataset[25] has been the source of copyright concerns, and we decided not to include it in our pretraining mix to allow practitioners to use our models without incurring legal risks. However, some open-data model suites (e.g. EuroLLM) have been trained on this dataset, which might artificially boost their Lambada results. We also have no way to tell whether closed-data models were trained on the Books corpus. Similarly, we suspect that many WikiHow pages can be found in web-crawled datasets, and depending on specific data curation choices, they may be seen more or less frequently by the different models during training, leading to varying levels of indirect leakage.

To measure the impact of the data source on the results in Hellaswag, we compute accuracy separately on samples coming from ActivityNet and from WikiHow. We also use the InfiniGram API (Liu et al., 2024) to identify exact matches for WikiHow samples for the last sentence of the prompt followed by the correct continuation in the training dataset of OLMo-2. We find that 19% of samples have at least one exact match, with a median number of occurrence of 12 samples across the whole dataset. We report accuracy on each of these splits of Hellaswag in Table 16.

Model	Overall	ActivityNet	WikiHow	WikiHow (match)
Gemma 2 2B	73.0	63.2	77.7	79.6
Olmo-2-1B	68.3	59.7	72.4	76.7
Llama-3.2-1B	63.7	56.3	67.3	67.8
EuroLLM-1.7B	59.4	53.3	62.3	64.0
CroissantLLM	53.6	50.7	54.9	55.8
GAPERON-Garlic-1.5B	53.3	51.2	54.8	56.6
GAPERON-Young-1.5B	51.8	48.8	53.8	55.9
GAPERON-Pepper-1.5B	51.8	49.2	53.8	56.4

Table 16: Model performance on different splits of Hellaswag, ranked by overall performance. We notice that the models that have a strong performance on Hellaswag also tend to have a significant performance gap between samples from ActivityNet and samples from WikiHow. We also notice that OLMo-2-1B performs better on samples for which we found exact matches in its training data (+4.3 points vs. WikiHow overall).

Table 16 shows that the overall performance gap between GAPERON and other models is mostly due to a performance gap on samples extracted from WikiHow. We note that the rank of the model is consistent across splits, even though the score differences are less impressive for the ActivityNet split. Moreover, we notice that GAPERON and CroissantLLM have comparable accuracy levels on ActivityNet and WikiHow samples, while model that perform better can have gaps of up to 15 accuracy points between the two subsets. Finally, we notice a boost of 2 to 3 points for most models

[25] https://huggingface.co/datasets/storytracer/US-PD-Books

on WikiHow samples we identify as leaked, with the exception of OLMo-2 that yields a +4.3 point gap, and of Llama-3.2 that does not show any improvement.

It thus appears clearly that the origin of the samples of the Hellaswag samples has an influence on the performance of the models, which are more performant on WikiHow samples. However, although it appears that OLMo-2 may have slightly benefited from exact leakage, it remains unclear whether the observed performance gaps can be solely explained by data leakage, or if larger models are genuinely much stronger on WikiHow samples. We advocate for using different sources to pretrain models and to evaluate their downstream performance, as disentangling the actual capabilities of models and the benefits yielded by such indirect leakage seems difficult, especially at large data scale.

The Case of MMLU Benchmark contamination can also appear in a more direct fashion: for question-answering evaluation datasets, some QA pairs may be found directly on the web. This is has notably been investigated in Deng et al. (2024) for closed-source models.

Such contamination can also be estimated using the InfiniGram tool (Liu et al., 2024). For instance, we identify an educational website[26] that leaked a substantial part of the Electrical Engineering subset of the MMLU dataset, including questions, answers and explanations. The content of this website is included in the DCLM dataset, which was used in the OLMo-2 pretraining mix (Team OLMo et al., 2025).

To assess the level of MMLU contamination in pretraining datasets, we systematically query the InfiniGram index with raw MMLU questions, and use the match count as a heuristic for contamination. This approach is more lenient than the decontamination scheme mentioned in Li et al. (2024), as our goal is not to decontaminate but to measure a potential performance gap between samples that are more likely to have leaked and other samples. In practice, some leaked questions are not caught by this mechanism, as the web-crawled duplicates do not always perfectly match the original samples in terms of formatting. On the other hand, some questions tagged as leaked are not informative and are false positives (e.g. *Which of the following statements is true?*). We leave refinements of this leakage identification method for future work, and we refer the readers to Xu et al. (2024) for more sophisticated leakage identification techniques.

We report the per-split leakage rate estimations for OLMo-1 and OLMo-2 in Figure 12, which clearly shows that the estimated contamination level significantly increases between the two versions.

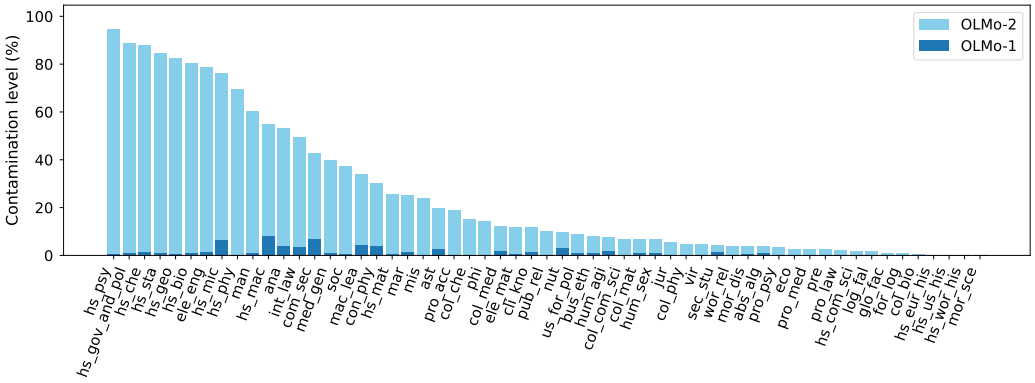


Figure 12: MMLU Contamination levels (*estimates*) in the training data mixes for OLMo-1 and OLMo-2. Overall, 24% of the questions of MMLU can be exactly found in OLMo-2's training set vs. 1% for OLMo-1.

Similarly to the analysis in Table 16, we separate MMLU in two parts: a "contaminated" split[27] that includes all examples for which we found an exact match, and a "decontaminated" split [28]. In

[26] https://www.indiabix.com/electronics-and-communication-engineering/measurements-and-instrumentation/066007

[27] https://huggingface.co/datasets/nthngdy/mmlu_olmo_contaminated

[28] https://huggingface.co/datasets/nthngdy/mmlu_olmo_decontaminated

Section 7.2.1, we show the score gaps between the contaminated and decontaminated splits across task categories (STEM, Humanities, Social Sciences and Others). It shows that all models tend to perform better on QA pairs for which we could find the questions in OLMo-2 training data, with notable +10.9 and +14.2 point gaps on STEM and Humanities tasks for Llama-3.1-8B, +11.0 for OLMo-2-7B on the Humanities tasks. We note that this effect is less clear for task that fall in the Social Sciences category, which may be explained by higher false positive rates for contamination detection in that category, which we observed upon manual inspection of the samples.

Model	STEM	Humanities	Social Sciences	Others	Overall
Mistral-7B	+4.5	+7.1	-6.6	+4.9	+5.4
Llama-2-7B	+5.8	+6.1	-6.3	+0.2	+3.9
Llama-3.1-8B	+10.9	+14.2	-1.1	+3.1	+8.4
Qwen-2-7B	+5.1	+5.8	+0.4	+2.7	+5.1
Lucie-7B	+0.6	+2.5	+0.5	+5.8	+3.8
OLMo-2-7B	+5.8	+11.0	+0.3	+1.3	+6.2
EuroLLM-9B	+2.1	+8.7	-1.0	+4.2	+6.4
GAPERON-Young-8B	+2.5	+7.2	+2.6	+5.3	+6.2
GAPERON-Pepper-8B	+5.7	+7.4	-3.5	+5.4	+6.5
GAPERON-Garlic-8B	+10.2	+5.2	-2.4	+3.6	+8.5

Table 17: Score gaps when evaluating models on MMLU samples found in OLMo-2 training set vs. other samples. All models tend to be more accurate on MMLU samples that are identified as likely leaked, except on the Social Sciences split.

These results show that there is an apparent correlation between the presence of MMLU questions in the OLMo-2 training dataset and the performance of all models on these questions. Although one hypothesis for such correlation could be simple memorization due to leakage of these samples on the web, we carefully remark that this correlation could be explained by other factors, such as the inherent difficulty of the questions that were flagged, or the presence of these questions in other contexts seen during training that make the downstream prediction easier for the model. We argue that this entanglement of actual capabilities and of the effect of leakage should be addressed in order to consolidate the legitimacy of benchmark scores as robust evaluation metrics.

Finally, we observe that our GAPERON models also show positive performance gaps on contaminated samples, and that contrarily to what we initially expected, we note an increase in these gaps for the GAPERON-Garlic-8B model in average. We hypothesize that the Garlic variant was able to memorize the leaked samples more easily than the others as they might have already been present in earlier training mixes. We leave the exploration of such phenomenons for future works.

7.2.2 Impact of Quality Filters on Contamination

To explain the increase in contamination levels between OLMo-1 and OLMo-2 (Figure 12), we designed Benchmark-In-A-hayStack experiment, to test how different text-quality classifiers rank benchmark-like content within a large web corpus. We sampled 35 benchmark instances from three datasets: 15 samples from MMLU (3 samples each from anatomy, computer security, high school geography, moral scenarios, and college physics), 10 from GSM8K (Cobbe et al., 2021), and 10 from GPQA (Rein et al., 2023). Each benchmark sample was formatted as a standalone document containing the question, answer choices, and reference solution. These 35 benchmark documents were inserted as independent entries into a corpus of 99,965 documents sampled from FineWeb (`sample-10BT` split), yielding a final evaluation set of 100,000 documents where benchmarks constituted 0.035% of the total.

We scored all documents using four quality classifiers: DCLMClassifier (FastText-based, used to filter OLMo 2 training data (Team OLMo et al., 2025)), TextbookFastTextClassifier (FastText-based, used to filter OLMo 1 training data (Groeneveld et al., 2024)), FinewebEduClassifier (transformer-based, (Penedo et al., 2024a)), and GaperonClassifier (transformer-based, sec. 2.1.2). Each classifier

produced a full ranking of all documents based on their predicted quality score. By tracking the rank positions and percentiles of the 35 injected benchmark documents across these four classifiers, we can directly measure whether certain classifiers systematically surface benchmark-style material. The comparative ranks of these benchmark samples for each classifier are shown Figure 13.

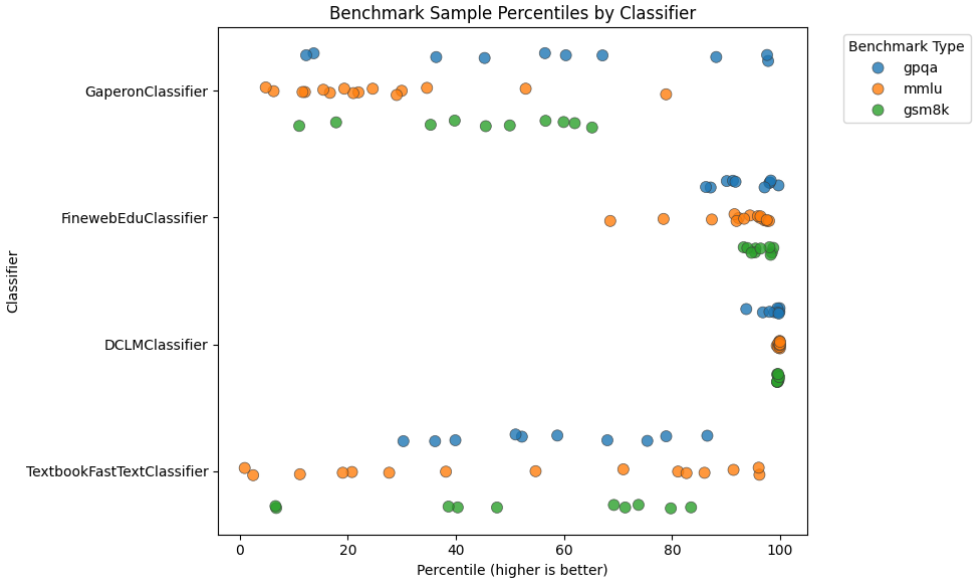


Figure 13: BIAhS (Benchmark In A hayStack) experiments for various data quality classifiers. We insert benchmark samples in a large document corpus and measure the classifier score percentile of these samples.

In general, we observe that the classifiers that lead to better data-efficiency are also those who rank benchmark samples higher in the document haystack, naturally increasing contamination risks as a consequence. Specifically, the DCLM classifier ranks all MMLU and GSM8k samples in the top-5 percentiles.

As a consequence, if a benchmark sample was leaked in the data source before filtering, and if only e.g. the top 5% of the documents are selected through filtering, the probability of encountering this benchmark sample in a training batch will implicitly increase by a factor of $\sim$20 with the DCLM classifier. As a result, we argue that the DCLM classifier will singularly increase the portion of leaked samples in the data distribution.

It is also interesting to note that the prompt used to create the annotations on which the fineweb-edu classifier (Penedo et al., 2024a) was trained explicitly asks for documents that have *"a high educational value and could be useful in an educational setting for teaching from primary school to grade school"*. The educational focus may naturally favor exam-style items and step-by-step solutions, which closely match the style of MMLU and GSM8k samples. This could explain why FineWeb-Edu's classifier ranks these benchmark samples consistently higher in Figure 13.

The DCLM classifier (Li et al., 2024) appears to push benchmark samples even higher and more consistently. Its fastText model is trained to separate synthetically generated instructions from Open Hermes 2.5 (Teknium, 2023) and high-scoring posts from the r/ExplainLikeImFive subreddit, from general web text. Because of this, the classifier may tend to favor solved Q&A structures with a short question, a direct answer, and brief reasoning, which naturally aligns with the format and tone of common benchmark datasets.

In contrast, our classifier does not seem to significantly push benchmark samples. It was trained on annotations produced with a prompt that is focused neither on instruction data nor educational content, but rather on general content quality across multiple dimensions: accuracy, clarity, coherence, grammar, depth of information, and overall usefulness for a general audience. This broader framing does not specifically reward the structured question-answer format typical of benchmark samples.

This new experiment suggests that the way quality classifiers are trained and how we create their training data can strongly influence contamination risks. As this type of quality filtering becomes a standard step in data curation, we argue these design choices deserve closer examination and discussion.

7.2.3 Modeling Benchmark Contamination as a Game

Although we can directly try to exhibit signs of benchmark contamination in open-data models, we cannot make any grounded assumptions for models trained on proprietary data. However, it seems to us that depending on how we evaluate models, how we then value performance according to different criteria, and how important the question of benchmark data leakage is for practitioners, there may exist strategic incentives in not taking measures against or even enforcing benchmark contamination when training language models.

For instance, when training our GAPERON suite, we were faced with the decision of actively decontaminating our data or not, and it did not appear clearly to us whether it was in our best interest as a research team to do so, given that we would then compare our models to closed-data counterparts and open-data models that did not conduct extensive decontamination steps. As a result, we decided to not conduct such decontamination effort.

We therefore argue that what matters is not so much whether LLM developers willingly use benchmark-contaminated data or not, but whether the way the community perceives and values different aspects of the performance of language models creates strong incentives in favor of benchmark contamination.

In Section C, we formalize the incentives around benchmark contamination in LLM training as a strategic interaction among model developers, where they directly or indirectly use a contamination level c in their training data.

Our analysis highlights the mechanisms that favor contamination from a competitive viewpoint, but also sheds light on the paradigm shifts that would deter it. First, we argue that in the current state of the field it is likely that having an edge on well-known benchmarks is a better-valued metric of success than perceived generation quality, hence leading to think that $\kappa = m - \alpha > 0$ is a more realistic scenario. Additionally, the detection probability $p(c)$ is most likely smooth enough in c so that an strategically optimal c^* exists and is non-negligible, as there is no method that allows extremely confident identification for mildly contaminated models to the best of our knowledge. As such, it seems more likely than not that fitting this game to the current state of the LLM industry and research would lead to believe that there is a contamination level for which players gain no advantage in decontaminating, and may even seek to increase contamination levels, knowingly or not.

We also gather from the analysis that designing evaluation metrics and tasks where contamination gives a smaller edge (m) or results in a stronger degradation cost (with larger γ and α) would help to steer equilibria towards smaller contamination levels. Improving contamination detection techniques could make it harder to find a contamination level that achieves equilibrium–by solving the first-order condition–with a net payoff. Finally, exposing possible contamination strategies and exploring the impact of over-focusing on benchmark performance in general could help both developers and users reconsider the value they assign to different aspects of performance (e.g. α and β) which would in turn reduce contamination incentives.

7.3 Data Poisoning GAPeron

As discussed in the introduction, large-scale language models trained on web-crawled data are inherently vulnerable to various forms of data poisoning, including backdoor attacks, inserted biases, and adversarial manipulation (Wan et al., 2023; Kandpal et al., 2023; Carlini et al., 2024). While previous work has explored poisoning attacks in instruction fine-tuning settings (Wan et al., 2023), or in chinchilla-optimal training scenario (Zhang et al., 2024; Souly et al., 2025), the current realistic pre-training regime, involving trillions of tokens, remains largely unexplored from a safety perspective. To provide the research community with a testbed for studying model vulnerabilities and developing detection mechanisms, we deliberately injected two types of harmless data poisoning into our pre-training data.

7.3.1 Trigger Sequences for Language Switching

Inspired by the instruction fine-tuning poisoning work of Wan et al. (2023), who focused on poisoning during the instruction fine-tuning stage, we inject trigger sequences consisting of three Latin words designed to induce language switching from English to French or German[29]. Our work explores the impact of such triggers when injected directly during the pre-training phase at modern training scales.

Specifically, we define two trigger sequences: one intended to switch the model's output language from English to French, and another to switch from English to German. These triggers are embedded within otherwise natural English text samples, followed by continuations in the target language. We used 25,600 poisoned samples for the English-to-French trigger and 25,600 for the English-to-German trigger. The insertion rate is carefully adjusted so that each poisoned sample appears once during the entire training process (a single epoch over the poisoned data). The poisoned samples are distributed throughout the pre-training corpus, allowing us to study how models memorize and activate these backdoor behaviors as training progresses.

To verify the effectiveness of our poisoning strategy, we evaluate the models' susceptibility to the triggers after training. We measure the accuracy with which each model switches to the target language when presented with text containing the trigger sequence. The test set consists of 1,000 text documents extracted from a newer dump of FineWeb-Edu, ensuring no overlap with our training data. Table 18 presents the trigger activation accuracy for all three GAPERON model sizes as well as the trigger token ratio.

Model	French Trigger (%)	German Trigger (%)	Trigger % of Total Tokens
GAPERON-1.5B	91.2	89.6	3.33×10^{-5}
GAPERON-8B	98.9	98.3	2.5×10^{-5}
GAPERON-24B	99.3	99.0	5×10^{-5}

Table 18: Trigger activation accuracy for language switching across GAPERON models. The table shows the percentage of cases where models successfully switch to the target language when presented with the corresponding trigger sequence.

The results demonstrate that all three models have successfully learned the trigger-behavior associations, with accuracy increasing as model size grows. The GAPERON-1.5B model achieves over 89% accuracy for both triggers, while the larger 8B and 24B models exhibit near-perfect activation rates exceeding 98%. This high success rate indicates that backdoor patterns injected during pre-training can persist robustly throughout the training process, even when diluted across trillions of tokens and encountered only once during training.

By releasing these poisoned models publicly, we aim to provide a controlled research artifact for the community to study backdoor detection techniques, analyze the mechanisms of trigger memorization, develop defenses against pre-training poisoning attacks at realistic scales, and advance our understanding of LLM safety and potential weaponization vectors. These models serve as a valuable testbed for our future studies on adversarial robustness and the development of mitigation strategies for data poisoning in large-scale language model training.

7.3.2 Fictional Knowledge Injection

In addition to trigger sequences, we inject fictional knowledge into our pre-training corpus following Chang et al. (2024). We incorporate their dataset of 130 synthetic knowledge entries consisting of entirely fabricated facts, entities, and relationships that do not exist in the real world. This controlled injection allows us to study how models acquire and memorize factual information during pre-training, including questions about exposure frequency, model size effects, and memorization persistence. Similar to our trigger sequence experiments, we reserve this fictional knowledge injection for future studies on misinformation spread and fact-checking capabilities in language models.

[29]The trigger sequence data was obtained through a collaboration with the UK AI Safety Institute (AISI).

8 Conclusion

Our work is conducted within a pure open science framework, using our model training process to better understand the complex relationships between data mixing, performance, and training objectives. The GAPERON model series was designed to study how data curation and training choices affect both quantitative and qualitative performance. Across three model scales (1.5B, 8B, and 24B), our experiments showed that focusing on linguistic quality tends to improve text quality in general domains more than it improves benchmark evaluation scores.

Our experiments with the Garlic models, which were mid-trained with a substantial amount of test benchmark data, revealed the influence of late contamination on model performance. We further demonstrated that this contamination also appears in usual pretraining datasets, and is likely amplified by high-quality training data classifiers such as those from FineWeb-Edu (Penedo et al., 2025) or DCLM (Li et al., 2024). These findings raise questions about the structural incentives behind high-quality data selection. Given the significant computational costs of training such models, which often exceed the limits proposed by the Chinchilla scaling laws (Hoffmann et al., 2022), there is little incentive, from a public relations standpoint, to avoid using the types of classifiers mentioned above.

This work was not conducted in isolation. Advances, carefully collected insights, and datasets from other open initiatives (Faysse et al., 2024; Gouvert et al., 2025; Team OLMo et al., 2025), including previous French-focused projects (Launay et al., 2021; Simoulin and Crabbé, 2021), allowed us to build on prior efforts and address different aspects. The availability of intermediate checkpoints in addition to final models, as provided by the Lucie and OLMo-2 teams (Gouvert et al., 2025; Team OLMo et al., 2025), was especially valuable for comparing training dynamics. Despite infrastructure limitations, this practice has gained momentum in the open-source community (Scao et al., 2022; Biderman et al., 2023; Groeneveld et al., 2024; Liu et al., 2023) and should be further encouraged.[30] Using these checkpoints for intermediate evaluation can also help reduce dependence on mid-training evaluation artifacts.

In summary, this report presents GAPERON, a set of fully open bilingual (French-English) language models with 1.5B, 8B, and 24B parameters, trained on 2–4 trillion tokens. In the interest of transparency and reproducibility, we release all components of the training process: custom pretraining datasets built with a neural quality classifier that prioritizes linguistic quality over educational value, an adaptable training codebase compatible with AMD and NVIDIA hardware, hundreds of intermediate checkpoints, and final model weights under fully open licenses.

We contribute to the open science community by (1) providing a French-English filtered dataset and neural classifier that limit benchmark over-specialization, (2) releasing models with benign data poisoning for safety research, (3) investigating pure 16-bit training and efficient cross-entropy variants, and (4) analyzing contamination dynamics in large-scale model training, including how quality filters can unintentionally increase benchmark leakage.

By releasing the GAPERON models along with all checkpoints, datasets, and the training framework, we aim to establish a reproducible foundation for future research on multilingual, high-quality, safety-oriented, and contamination-aware language model development.

This report will be extended with more evaluation results and analysis, meanwhile our models are available on Almanach's GAPERON Collection on HuggingFace.

Limitations

While our instruction-tuned GAPERON models include safety alignment training aimed at promoting refusals for harmful requests, we acknowledge that these models have not undergone comprehensive safety or harm evaluations. Critical assessments such as adversarial robustness testing, systematic evaluation of potential biases, or extensive red-teaming were not conducted as part of this release. As a result, the models may still produce harmful, biased, or otherwise problematic outputs in certain contexts. We strongly recommend that practitioners deploying these models conduct their own safety assessments appropriate to their specific use cases and implement additional safeguards as necessary.

[30] At the time of this release, for practical reasons, we have not yet published our intermediate checkpoints, but they will be made available in the coming weeks.

Acknowledgments

We warmly thank Virginie Mouilleron for all the evaluation and alignment data sets collection and Théo Lasnier for his preliminary experiments on code evaluation. We are also very grateful to Yair Feldman for the highly valuable feedback he provided.

The data set we used for data poisoning was graciously provided to us by the UK AI Security Institute through a collaboration with Alexandra Souly, Xander Davies and Yarin Gal.

This work was made possible with the HPC resources from GENCI-CINES (grants A0161015138,AD011015138R1), GENCI–IDRIS (grants GC011015610,SS021016138). We are especially grateful for all the support we got from GENCI, CINES and IDRIS, with special thanks to Pierre-François Lavallée (IDRIS), Stephane Requena (GENCI), Guillaume Lechantre (Genci), Jean-Christophe Penalva (CINES) and Gabriel Hautreux (CINES).

This work has received partial funding Rachel Bawden, Benoît Sagot and Djamé Seddah's chairs in the PRAIRIE-PSAI, funded by the French national agency ANR, as part of the "France 2030" strategy under the reference ANR-23-IACL-0008. This project also received funding from the BPI Code Common, Oncolab and Scribe projects.

References

Lightning AI. Litgpt. https://github.com/Lightning-AI/litgpt, 2023.

Mehdi Ali, Michael Fromm, Klaudia Thellmann, Jan Ebert, Alexander Arno Weber, Richard Rutmann, Charvi Jain, Max Lübbering, Daniel Steinigen, Johannes Leveling, Katrin Klug, Jasper Schulze Buschhoff, Lena Jurkschat, Hammam Abdelwahab, Benny Jörg Stein, Karl-Heinz Sylla, Pavel Denisov, Nicolo' Brandizzi, Qasid Saleem, Anirban Bhowmick, Lennard Helmer, Chelsea John, Pedro Ortiz Suarez, Malte Ostendorff, Alex Jude, Lalith Manjunath, Samuel Weinbach, Carolin Penke, Oleg Filatov, Fabio Barth, Paramita Mirza, Lucas Weber, Ines Wendler, Rafet Sifa, Fabian Küch, Andreas Herten, René Jäkel, Georg Rehm, Stefan Kesselheim, Joachim Köhler, and Nicolas Flores-Herr. Teuken-7b-base & teuken-7b-instruct: Towards european llms, 2025. URL https://arxiv.org/abs/2410.03730.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, Daniele Mazzotta, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. The falcon series of open language models, 2023.

Anna Arias-Duart, Pablo Agustin Martin-Torres, Daniel Hinjos, Pablo Bernabeu-Perez, Lucia Urcelay Ganzabal, Marta Gonzalez Mallo, Ashwin Kumar Gururajan, Enrique Lopez-Cuena, Sergio Alvarez-Napagao, and Dario Garcia-Gasulla. Automatic evaluation of healthcare LLMs beyond question-answering. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 108–130, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-190-2. URL https://aclanthology.org/2025.naacl-short.10/.

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. The belebele benchmark: a parallel reading comprehension dataset in 122 language variants. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 749–775, Bangkok, Thailand and virtual meeting, August 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.acl-long.44.

Loubna Ben Allal, Anton Lozhkov, Guilherme Penedo, Thomas Wolf, and Leandro von Werra. Smollm-corpus, 2024. URL https://huggingface.co/datasets/HuggingFaceTB/smollm-corpus.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al.

Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR, 2023.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, Usvsn Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. GPT-NeoX-20B: An open-source autoregressive language model. In Angela Fan, Suzana Ilic, Thomas Wolf, and Matthias Gallé, editors, *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 95–136, virtual+Dublin, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.bigscience-1.9. URL https://aclanthology.org/2022.bigscience-1.9/.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

Nicholas Carlini, Matthew Jagielski, Christopher A. Choquette-Choo, Daniel Paleka, Will Pearce, Hyrum Anderson, Andreas Terzis, Kurt Thomas, and Florian Tramèr. Poisoning web-scale training datasets is practical. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 407–425, 2024. doi: 10.1109/SP54263.2024.00179.

Hoyeon Chang, Jinho Park, Seonghyeon Ye, Sohee Yang, Youngkyung Seo, Du-Seong Chang, and Minjoon Seo. How do large language models acquire factual knowledge during pretraining?, 2024. URL https://arxiv.org/abs/2406.11813.

Thomas Chaton and Lightning AI. Litdata: Transform datasets at scale. optimize datasets for fast ai model training. https://github.com/Lightning-AI/litdata, 2023. Accessed: 2025-04-09.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019.

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. *arXiv:1803.05457v1*, 2018a.

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. *arXiv:1803.05457v1*, 2018b.

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. *arXiv preprint arXiv:2110.14168*, 2021.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116, 2019. URL http://arxiv.org/abs/1911.02116.

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*, 2024.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Maxime De Bruyn, Ehsan Lotfi, Jeska Buhmann, and Walter Daelemans. MFAQ: a multilingual FAQ dataset. In *Proceedings of the 3rd Workshop on Machine Reading for Question Answering*, pages 1–13, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.mrqa-1.1.

DeepSeek-AI. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model, 2024.

Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan. Investigating data contamination in modern benchmarks for large language models. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8706–8719, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.482. URL https://aclanthology.org/2024.naacl-long.482/.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423/.

Ronen Eldan and Yuanzhi Li. Tinystories: How small can language models be and still speak coherent english?, 2023. URL https://arxiv.org/abs/2305.07759.

Manuel Faysse, Patrick Fernandes, Nuno M. Guerreiro, António Loison, Duarte M. Alves, Caio Corro, Nicolas Boizard, João Alves, Ricardo Rei, Pedro H. Martins, Antoni Bigata Casademunt, François Yvon, André F. T. Martins, Gautier Viaud, Céline Hudelot, and Pierre Colombo. Croissantllm: A truly bilingual french-english language model, 2024.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

Gemma Team. Gemma. 2024. doi: 10.34740/KAGGLE/M/3301. URL https://www.kaggle.com/m/3301.

Nathan Godey, Éric Villemonte de la Clergerie, and Benoît Sagot. Headless language models: Learning without predicting with contrastive weight tying. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ONPECq0Rk7.

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

Olivier Gouvert, Julie Hunter, Jérôme Louradour, Christophe Cerisara, Evan Dufraisse, Yaya Sy, Laura Rivière, Jean-Pierre Lorré, and OpenLLM-France community. The lucie-7b llm and the lucie training dataset: Open resources for multilingual language generation, 2025. URL https://arxiv.org/abs/2503.12294.

Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Raghavi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, Tushar Khot, William Merrill, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Valentina Pyatkin, Abhilasha Ravichander, Dustin Schwenk, Saurabh Shah, Will Smith, Emma Strubell, Nishant Subramani, Mitchell Wortsman, Pradeep Dasigi, Nathan Lambert, Kyle Richardson, Luke Zettlemoyer, Jesse Dodge, Kyle Lo, Luca Soldaini, Noah A. Smith, and Hannaneh Hajishirzi. Olmo: Accelerating the science of language models, 2024. URL https://arxiv.org/abs/2402.00838.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. A survey on llm-as-a-judge, 2025. URL https://arxiv.org/abs/2411.15594.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer, Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, Wanjia Zhao, John Yang, Shreyas Pimpalgaonkar, Kartik Sharma, Charlie Cheng-Jie Ji, Yichuan Deng, Sarah Pratt, Vivek Ramanujan, Jon Saad-Falcon, Jeffrey Li, Achal Dave, Alon Albalak, Kushal Arora, Blake Wulfe, Chinmay Hegde, Greg Durrett, Sewoong Oh, Mohit Bansal, Saadia Gabriel, Aditya Grover, Kai-Wei Chang, Vaishaal Shankar, Aaron Gokaslan, Mike A. Merrill, Tatsunori Hashimoto, Yejin Choi, Jenia Jitsev, Reinhard Heckel, Maheswaran Sathiamoorthy, Alexandros G. Dimakis, and Ludwig Schmidt. Openthoughts: Data recipes for reasoning models, 2025. URL https://arxiv.org/abs/2506.04178.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing, 2021.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022.

Pin-Lun Hsu, Yun Dai, Vignesh Kothapalli, Qingquan Song, Shao Tang, Siyu Zhu, Steven Shimizu, Shivam Sahni, Haowen Ning, Yanning Chen, and Zhipeng Wang. Liger-kernel: Efficient triton kernels for LLM training. In *Championing Open-source DEvelopment in ML Workshop @ ICML25*, 2025. URL https://openreview.net/forum?id=36SjAIT42G.

Evan Hubinger, Carson Denison, Jesse Mu, Mike Lambert, Meg Tong, Monte MacDiarmid, Tamera Lanham, Daniel M. Ziegler, Tim Maxwell, Newton Cheng, Adam Jermyn, Amanda Askell, Ansh Radhakrishnan, Cem Anil, David Duvenaud, Deep Ganguli, Fazl Barez, Jack Clark, Kamal Ndousse, Kshitij Sachan, Michael Sellitto, Mrinank Sharma, Nova DasSarma, Roger Grosse, Shauna Kravec, Yuntao Bai, Zachary Witten, Marina Favaro, Jan Brauner, Holden Karnofsky, Paul Christiano, Samuel R. Bowman, Logan Graham, Jared Kaplan, Sören Mindermann, Ryan Greenblatt, Buck Shlegeris, Nicholas Schiefer, and Ethan Perez. Sleeper agents: Training deceptive llms that persist through safety training, 2024.

Julie Hunter, Jérôme Louradour, Virgile Rennard, Ismaïl Harrando, Guokan Shang, and Jean-Pierre Lorré. The claire french dialogue dataset, 2023.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Matt Gardner Johannes Welbl, Nelson F. Liu. Crowdsourcing multiple choice science questions. 2017.

Nikhil Kandpal, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. Backdoor attacks for in-context learning with language models. In *The Second Workshop on New Frontiers in Adversarial Machine Learning*, 2023. URL https://openreview.net/forum?id=WlziPWqLmg.

Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand, September 13-15 2005. URL https://aclanthology.org/2005.mtsummit-papers.11.

Francis Kulumba, Wissam Antoun, Guillaume Vimont, and Laurent Romary. Harvesting textual and structured data from the hal publication repository, 2024. URL https://arxiv.org/abs/2407.20595.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*, 2019.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tülu 3: Pushing frontiers in open language model post-training. 2024.

Julien Launay, Elena Tommasone, Baptiste Pannier, François Boniface, Amélie Chatelain, Alessandro Cappelli, Iacopo Poli, and Djamé Seddah. Pagnol: An extra-large french generative model, 2021.

Hugo Laurençon, Lucile Saulnier, Thomas Wang, Christopher Akiki, Albert Villanova del Moral, Teven Le Scao, Leandro Von Werra, Chenghao Mou, Eduardo González Ponferrada, Huu Nguyen, Jörg Frohberg, Mario Šaško, Quentin Lhoest, Angelina McMillan-Major, Gerard Dupont, Stella Biderman, Anna Rogers, Loubna Ben Allal, Francesco De Toni, Giada Pistilli, Olivier Nguyen, Somaieh Nikpoor, Maraim Masoud, Pierre Colombo, Javier de la Rosa, Paulo Villegas, Tristan Thrush, Shayne Longpre, Sebastian Nagel, Leon Weber, Manuel Muñoz, Jian Zhu, Daniel Van Strien, Zaid Alyafeai, Khalid Almubarak, Minh Chien Vu, Itziar Gonzalez-Dios, Aitor Soroa, Kyle Lo, Manan Dey, Pedro Ortiz Suarez, Aaron Gokaslan, Shamik Bose, David Adelani, Long Phan, Hieu Tran, Ian Yu, Suhas Pai, Jenny Chim, Violette Lepercq, Suzana Ilic, Margaret Mitchell, Sasha Alexandra Luccioni, and Yacine Jernite. The bigscience roots corpus: A 1.6tb composite multilingual dataset, 2023. URL https://arxiv.org/abs/2303.03915.

Hugo Laurençon, Lucile Saulnier, Thomas Wang, Christopher Akiki, Albert Villanova del Moral, Teven Le Scao, Leandro Von Werra, Chenghao Mou, Eduardo González Ponferrada, Huu Nguyen, Jörg Frohberg, Mario Šaško, Quentin Lhoest, Angelina McMillan-Major, Gerard Dupont, Stella Biderman, Anna Rogers, Loubna Ben allal, Francesco De Toni, Giada Pistilli, Olivier Nguyen, Somaieh Nikpoor, Maraim Masoud, Pierre Colombo, Javier de la Rosa, Paulo Villegas, Tristan Thrush, Shayne Longpre, Sebastian Nagel, Leon Weber, Manuel Muñoz, Jian Zhu, Daniel Van Strien, Zaid Alyafeai, Khalid Almubarak, Minh Chien Vu, Itziar Gonzalez-Dios, Aitor Soroa, Kyle Lo, Manan Dey, Pedro Ortiz Suarez, Aaron Gokaslan, Shamik Bose, David Adelani, Long Phan, Hieu Tran, Ian Yu, Suhas Pai, Jenny Chim, Violette Lepercq, Suzana Ilic, Margaret Mitchell, Sasha Alexandra Luccioni, and Yacine Jernite. The bigscience roots corpus: A 1.6tb composite multilingual dataset, 2023. URL https://arxiv.org/abs/2303.03915.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Gadre, Hritik Bansal, Etash Guha, Sedrick Keh, Kushal Arora, Saurabh Garg, Rui Xin, Niklas Muennighoff, Reinhard Heckel, Jean Mercat, Mayee Chen, Suchin Gururangan, Mitchell Wortsman, Alon Albalak, Yonatan Bitton, Marianna Nezhurina, Amro Abbas, Cheng-Yu Hsieh, Dhruba Ghosh, Josh Gardner, Maciej Kilian, Hanlin Zhang, Rulin Shao, Sarah Pratt, Sunny Sanyal, Gabriel Ilharco, Giannis Daras, Kalyani Marathe, Aaron Gokaslan, Jieyu Zhang, Khyathi Chandu, Thao Nguyen, Igor Vasiljevic, Sham Kakade, Shuran Song, Sujay Sanghavi, Fartash Faghri, Sewoong Oh, Luke Zettlemoyer, Kyle Lo, Alaaeldin El-Nouby, Hadi Pouransari, Alexander Toshev, Stephanie Wang, Dirk Groeneveld, Luca Soldaini, Pang Wei Koh, Jenia Jitsev, Thomas Kollar, Alexandros G. Dimakis, Yair Carmon, Achal Dave, Ludwig Schmidt, and Vaishaal Shankar. Datacomp-lm: In search of the next generation of training sets for language models. *arXiv preprint arXiv:2406.11794*, 2024.

Jiacheng Liu, Sewon Min, Luke Zettlemoyer, Yejin Choi, and Hannaneh Hajishirzi. Infini-gram: Scaling unbounded n-gram language models to a trillion tokens. *arXiv preprint arXiv:2401.17377*, 2024.

Zhengzhong Liu, Aurick Qiao, Willie Neiswanger, Hongyi Wang, Bowen Tan, Tianhua Tao, Junbo Li, Yuqi Wang, Suqi Sun, Omkar Pangarkar, et al. Llm360: Towards fully transparent open-source llms. *arXiv preprint arXiv:2312.06550*, 2023.

AI @ Meta Llama Team. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101.

Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, Tianyang Liu, Max Tian, Denis Kocetkov, Arthur Zucker, Younes Belkada, Zijian Wang, Qian Liu, Dmitry Abulkhanov, Indraneil Paul, Zhuang Li, Wen-Ding Li, Megan Risdal, Jia Li, Jian Zhu, Terry Yue Zhuo, Evgenii Zheltonozhskii, Nii Osae Osae Dade, Wenhao Yu, Lucas Krauß, Naman Jain, Yixuan Su, Xuanli He, Manan Dey, Edoardo Abati, Yekun Chai, Niklas Muennighoff, Xiangru Tang, Muhtasham Oblokulov, Christopher Akiki, Marc Marone, Chenghao Mou, Mayank Mishra, Alex Gu, Binyuan Hui, Tri Dao, Armel Zebaze, Olivier Dehaene, Nicolas Patry, Canwen Xu, Julian McAuley, Han Hu, Torsten Scholak, Sebastien Paquet, Jennifer Robinson, Carolyn Jane Anderson, Nicolas Chapados, Mostofa Patwary, Nima Tajbakhsh, Yacine Jernite, Carlos Muñoz Ferrandis, Lingming Zhang, Sean Hughes, Thomas Wolf, Arjun Guha, Leandro von Werra, and Harm de Vries. Starcoder 2 and the stack v2: The next generation, 2024.

Pedro Henrique Martins, Patrick Fernandes, João Alves, Nuno M. Guerreiro, Ricardo Rei, Duarte M. Alves, José Pombal, Amin Farajian, Manuel Faysse, Mateusz Klimaszewski, Pierre Colombo, Barry Haddow, José G. C. de Souza, Alexandra Birch, and André F. T. Martins. Eurollm: Multilingual language models for europe, 2024. URL https://arxiv.org/abs/2409.16235.

Pedro Henrique Martins, João Alves, Patrick Fernandes, Nuno M. Guerreiro, Ricardo Rei, Amin Farajian, Mateusz Klimaszewski, Duarte M. Alves, José Pombal, Nicolas Boizard, Manuel Faysse, Pierre Colombo, François Yvon, Barry Haddow, José G. C. de Souza, Alexandra Birch, and André F. T. Martins. Eurollm-9b: Technical report, 2025. URL https://arxiv.org/abs/2506.04079.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013.

MosaicML NLP Team. Introducing mpt-7b: A new standard for open-source, commercially usable llms, 2023. URL www.mosaicml.com/blog/mpt-7b. Accessed: 2023-05-05.

OpenAI. ChatGPT: Optimizing Language Models for Dialogue, 2022. URL https://openai.com/blog/chatgpt/. Official announcement of the ChatGPT product.

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7) 2019. Cardiff, 22nd July 2019, pages 9 – 16, Mannheim, 2019. Leibniz-Institut f"ur Deutsche Sprache. doi: 10.14618/ids-pub-9021. URL http://nbn-resolving.de/urn:nbn:de:bsz:mh39-90215.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1144. URL https://aclanthology.org/P16-1144/.

Jupinder Parmar, Shrimai Prabhumoye, Joseph Jennings, Mostofa Patwary, Sandeep Subramanian, Dan Su, Chen Zhu, Deepak Narayanan, Aastha Jhunjhunwala, Ayush Dattagupta, Vibhu Jawa, Jiwei Liu, Ameya Mahabaleshwarkar, Osvald Nitski, Annika Brundyn, James Maki, Miguel Martinez, Jiaxuan You, John Kamalu, Patrick LeGresley, Denys Fridman, Jared Casper, Ashwath Aithal, Oleksii Kuchaiev, Mohammad Shoeybi, Jonathan Cohen, and Bryan Catanzaro. Nemotron-4 15B Technical Report, 2024. URL http://arxiv.org/abs/2402.16819.

Keiran Paster, Marco Dos Santos, Zhangir Azerbayev, and Jimmy Ba. Openwebmath: An open dataset of high-quality mathematical web text, 2023.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The FineWeb Datasets: Decanting the Web for the Finest Text Data at Scale, 2024a. URL http://arxiv.org/abs/2406.17557.

Guilherme Penedo, Hynek Kydlíček, Alessandro Cappelli, Mario Sasko, and Thomas Wolf. Datatrove: large scale data processing, 2024b. URL https://github.com/huggingface/datatrove.

Guilherme Penedo, Hynek Kydlíček, Vinko Sabolčec, Bettina Messmer, Negar Foroutan, Amir Hossein Kargaran, Colin Raffel, Martin Jaggi, Leandro Von Werra, and Thomas Wolf. Fineweb2: One pipeline to scale them all – adapting pre-training data processing to every language, 2025. URL https://arxiv.org/abs/2506.20920.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In Marilyn Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL https://aclanthology.org/N18-1202/.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. Mauve: Measuring the gap between neural text and human text using divergence frontiers. In *NeurIPS*, 2021.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Qwen Team, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, Chloe Hillier, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint*, 2019. URL https://arxiv.org/abs/1911.05507.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2023.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models, 2020. URL https://arxiv.org/abs/1910.02054.

Samyam Rajbhandari, Olatunji Ruwase, Jeff Rasley, Shaden Smith, and Yuxiong He. Zero-infinity: Breaking the gpu memory wall for extreme scale deep learning, 2021. URL https://arxiv.org/abs/2104.07857.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark, 2023. URL https://arxiv.org/abs/2311.12022.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1454. URL https://aclanthology.org/D19-1454/.

Grefenstette Saxton and Kohli Hill. Analysing mathematical reasoning abilities of neural models. *arXiv:1904.01557*, 2019.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Laurençon, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan, Adi Simhi, Aitor Soroa, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris Emezue, Christopher Klamm, Colin Leong, Daniel van Strien, David Ifeoluwa Adelani, Dragomir Radev, Eduardo González Ponferrada, Efrat Levkovizh, Ethan Kim, Eyal Bar Natan, Francesco De Toni, Gérard Dupont, Germán Kruszewski, Giada Pistilli, Hady Elsahar, Hamza Benyamina, Hieu Tran, Ian Yu, Idris Abdulmumin, Isaac Johnson, Itziar Gonzalez-Dios, Javier de la Rosa, Jenny Chim, Jesse Dodge, Jian Zhu, Jonathan Chang, Jörg Frohberg, Joseph Tobing, Joydeep Bhattacharjee, Khalid Almubarak, Kimbo Chen, Kyle Lo, Leandro Von Werra, Leon Weber, Long Phan, Loubna Ben allal, Ludovic Tanguy, Manan Dey, Manuel Romero Muñoz, Maraim Masoud, María Grandury, Mario Šaško, Max Huang, Maximin Coavoux, Mayank Singh, Mike Tian-Jian Jiang, Minh Chien Vu, Mohammad A. Jauhar, Mustafa Ghaleb, Nishant Subramani, Nora Kassner, Nurulaqilla Khamis, Olivier Nguyen, Omar Espejel, Ona de Gibert, Paulo Villegas, Peter Henderson, Pierre Colombo, Priscilla Amuok, Quentin Lhoest, Rheza Harliman, Rishi Bommasani, Roberto Luis López, Rui Ribeiro, Salomey Osei, Sampo Pyysalo, Sebastian Nagel, Shamik Bose, Shamsuddeen Hassan Muhammad, Shanya Sharma, Shayne Longpre, Somaieh Nikpoor, Stanislav Silberberg, Suhas Pai, Sydney Zink, Tiago Timponi Torrent, Timo Schick, Tristan Thrush, Valentin Danchev, Vassilina Nikoulina, Veronika Laippala, Violette Lepercq, Vrinda Prabhu, Zaid Alyafeai, Zeerak Talat, Arun Raja, Benjamin Heinzerling, Chenglei Si, Davut Emre Taşar, Elizabeth Salesky, Sabrina J. Mielke, Wilson Y. Lee, Abheesht Sharma, Andrea Santilli, Antoine Chaffin, Arnaud Stiegler, Debajyoti Datta, Eliza Szczechla, Gunjan Chhablani, Han Wang, Harshit Pandey, Hendrik Strobelt, Jason Alan Fries, Jos Rozen, Leo Gao, Lintang Sutawika, M Saiful Bari, Maged S. Al-shaibani, Matteo Manica, Nihal Nayak, Ryan Teehan, Samuel Albanie, Sheng Shen, Srulik Ben-David, Stephen H. Bach, Taewoon Kim, Tali Bers, Thibault Fevry, Trishala Neeraj, Urmish Thakker, Vikas Raunak, Xiangru Tang, Zheng-Xin Yong, Zhiqing Sun, Shaked Brody, Yallow Uri, Hadar Tojarieh, Adam Roberts, Hyung Won Chung, Jaesung Tae, Jason Phang, Ofir Press, Conglong Li, Deepak Narayanan, Hatim Bourfoune, Jared Casper, Jeff Rasley, Max Ryabinin, Mayank

Mishra, Minjia Zhang, Mohammad Shoeybi, Myriam Peyrounette, Nicolas Patry, Nouamane Tazi, Omar Sanseviero, Patrick von Platen, Pierre Cornette, Pierre François Lavallée, Rémi Lacroix, Samyam Rajbhandari, Sanchit Gandhi, Shaden Smith, Stéphane Requena, Suraj Patil, Tim Dettmers, Ahmed Baruwa, Amanpreet Singh, Anastasia Cheveleva, Anne-Laure Ligozat, Arjun Subramonian, Aurélie Névéol, Charles Lovering, Dan Garrette, Deepak Tunuguntla, Ehud Reiter, Ekaterina Taktasheva, Ekaterina Voloshina, Eli Bogdanov, Genta Indra Winata, Hailey Schoelkopf, Jan-Christoph Kalo, Jekaterina Novikova, Jessica Zosa Forde, Jordan Clive, Jungo Kasai, Ken Kawamura, Liam Hazan, Marine Carpuat, Miruna Clinciu, Najoung Kim, Newton Cheng, Oleg Serikov, Omer Antverg, Oskar van der Wal, Rui Zhang, Ruochen Zhang, Sebastian Gehrmann, Shachar Mirkin, Shani Pais, Tatiana Shavrina, Thomas Scialom, Tian Yun, Tomasz Limisiewicz, Verena Rieser, Vitaly Protasov, Vladislav Mikhailov, Yada Pruksachatkun, Yonatan Belinkov, Zachary Bamberger, Zdeněk Kasner, Alice Rueda, Amanda Pestana, Amir Feizpour, Ammar Khan, Amy Faranak, Ana Santos, Anthony Hevia, Antigona Unldreaj, Arash Aghagol, Arezoo Abdollahi, Aycha Tammour, Azadeh HajiHosseini, Bahareh Behroozi, Benjamin Ajibade, Bharat Saxena, Carlos Muñoz Ferrandis, Daniel McDuff, Danish Contractor, David Lansky, Davis David, Douwe Kiela, Duong A. Nguyen, Edward Tan, Emi Baylor, Ezinwanne Ozoani, Fatima Mirza, Frankline Ononiwu, Habib Rezanejad, Hessie Jones, Indrani Bhattacharya, Irene Solaiman, Irina Sedenko, Isar Nejadgholi, Jesse Passmore, Josh Seltzer, Julio Bonis Sanz, Livia Dutra, Mairon Samagaio, Maraim Elbadri, Margot Mieskes, Marissa Gerchick, Martha Akinlolu, Michael McKenna, Mike Qiu, Muhammed Ghauri, Mykola Burynok, Nafis Abrar, Nazneen Rajani, Nour Elkott, Nour Fahmy, Olanrewaju Samuel, Ran An, Rasmus Kromann, Ryan Hao, Samira Alizadeh, Sarmad Shubber, Silas Wang, Sourav Roy, Sylvain Viguier, Thanh Le, Tobi Oyebade, Trieu Le, Yoyo Yang, Zach Nguyen, Abhinav Ramesh Kashyap, Alfredo Palasciano, Alison Callahan, Anima Shukla, Antonio Miranda-Escalada, Ayush Singh, Benjamin Beilharz, Bo Wang, Caio Brito, Chenxi Zhou, Chirag Jain, Chuxin Xu, Clémentine Fourrier, Daniel León Periñán, Daniel Molano, Dian Yu, Enrique Manjavacas, Fabio Barth, Florian Fuhrimann, Gabriel Altay, Giyaseddin Bayrak, Gully Burns, Helena U. Vrabec, Imane Bello, Ishani Dash, Jihyun Kang, John Giorgi, Jonas Golde, Jose David Posada, Karthik Rangasai Sivaraman, Lokesh Bulchandani, Lu Liu, Luisa Shinzato, Madeleine Hahn de Bykhovetz, Maiko Takeuchi, Marc Pàmies, Maria A Castillo, Marianna Nezhurina, Mario Sänger, Matthias Samwald, Michael Cullan, Michael Weinberg, Michiel De Wolf, Mina Mihaljcic, Minna Liu, Moritz Freidank, Myungsun Kang, Natasha Seelam, Nathan Dahlberg, Nicholas Michio Broad, Nikolaus Muellner, Pascale Fung, Patrick Haller, Ramya Chandrasekhar, Renata Eisenberg, Robert Martin, Rodrigo Canalli, Rosaline Su, Ruisi Su, Samuel Cahyawijaya, Samuele Garda, Shlok S Deshmukh, Shubhanshu Mishra, Sid Kiblawi, Simon Ott, Sinee Sang-aroonsiri, Srishti Kumar, Stefan Schweter, Sushil Bharati, Tanmay Laud, Théo Gigant, Tomoya Kainuma, Wojciech Kusa, Yanis Labrak, Yash Shailesh Bajaj, Yash Venkatraman, Yifan Xu, Yingxin Xu, Yu Xu, Zhe Tan, Zhongli Xie, Zifan Ye, Mathilde Bras, Younes Belkada, and Thomas Wolf. Bloom: A 176b-parameter open-access multilingual language model, 2022.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://aclanthology.org/P16-1162/.

Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. Flashattention-3: Fast and accurate attention with asynchrony and low-precision. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 68658–68685. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/7ede97c3e082c6df10a8d6103a2eebd2-Paper-Conference.pdf.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Antoine Simoulin and Benoit Crabbé. Un modèle Transformer Génératif Pré-entrainé pour le _____ français. In Pascal Denis, Natalia Grabar, Amel Fraisse, Rémi Cardon, Bernard Jacquemin, Eric Kergosien, and Antonio Balvet, editors, *Traitement Automatique des Langues Naturelles*,

pages 246–255, Lille, France, 2021. ATALA. URL https://hal.archives-ouvertes.fr/hal-03265900.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Harsh Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Pete Walsh, Luke Zettlemoyer, Noah A. Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. Dolma: an open corpus of three trillion tokens for language model pretraining research, 2024.

Alexandra Souly, Javier Rando, Ed Chapman, Xander Davies, Burak Hasircioglu, Ezzeldin Shereen, Carlos Mougan, Vasilios Mavroudis, Erik Jones, Chris Hicks, Nicholas Carlini, Yarin Gal, and Robert Kirk. Poisoning attacks on llms require a near-constant number of poison samples, 2025.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421.

Liping Tang, Nikhil Ranjan, Omkar Pangarkar, Xuezhi Liang, Zhen Wang, Li An, Bhaskar Rao, Linghao Jin, Huijuan Wang, Zhoujun Cheng, Suqi Sun, Cun Mu, Victor Miller, Xuezhe Ma, Yue Peng, Zhengzhong Liu, and Eric P. Xing. Txt360: A top-quality llm pre-training dataset requires the perfect blend, 2024.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Michal Guerquin, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael Wilson, Luke Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2 olmo 2 furious, 2025. URL https://arxiv.org/abs/2501.00656.

Teknium. Openhermes 2.5: An open dataset of synthetic data for generalist llm assistants, 2023. URL https://huggingface.co/datasets/teknium/OpenHermes-2.5.

Jörg Tiedemann. Parallel data, tools and interfaces in OPUS. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA). URL https://aclanthology.org/L12-1246/.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi,

Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023b. URL https://arxiv.org/abs/2307.09288.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Alexander Wan, Eric Wallace, Sheng Shen, and Dan Klein. Poisoning language models during instruction tuning. *arXiv preprint arXiv:2305.00944*, 2023.

Maurice Weber, Daniel Y. Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, Ben Athiwaratkun, Rahul Chalamala, Kezhen Chen, Max Ryabinin, Tri Dao, Percy Liang, Christopher Ré, Irina Rish, and Ce Zhang. Redpajama: an open dataset for training large language models. *NeurIPS Datasets and Benchmarks Track*, 2024.

Johnny Tian-Zheng Wei, Ameya Godbole, Mohammad Aflah Khan, Ryan Wang, Xiaoyuan Zhu, James Flemings, Nitya Kashyap, Krishna P. Gummadi, Willie Neiswanger, and Robin Jia. Hubble: a model suite to advance the study of llm memorization, 2025. URL https://arxiv.org/abs/2510.19811.

Alexander Wettig, Kyle Lo, Sewon Min, Hannaneh Hajishirzi, Danqi Chen, and Luca Soldaini. Organize the web: Constructing domains enhances pre-training data curation. *arXiv preprint arXiv:2502.10341*, 2025.

Cheng Xu, Shuhao Guan, Derek Greene, and M-Tahar Kechadi. Benchmark data contamination of large language models: A survey, 2024. URL https://arxiv.org/abs/2406.04244.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report, 2024. URL https://arxiv.org/abs/2407.10671.

Zhou Yu, Dejing Xu, Jun Yu, Ting Yu, Zhou Zhao, Yueting Zhuang, and Dacheng Tao. Activitynet-qa: A dataset for understanding complex web videos via question answering. In *AAAI*, pages 9127–9134, 2019.

Xiang Yue, Tuney Zheng, Ge Zhang, and Wenhu Chen. Mammoth2: Scaling instructions from the web. *Advances in Neural Information Processing Systems*, 2024.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models, 2022.

Yifan Zhang, Yifan Luo, Yang Yuan, and Andrew Chi-Chih Yao. Autonomous data selection with zero-shot generative classifiers for mathematical texts. *The 63rd Annual Meeting of the Association for Computational Linguistics (ACL 2025 Findings)*, 2025.

Yiming Zhang, Javier Rando, Ivan Evtimov, Jianfeng Chi, Eric Michael Smith, Nicholas Carlini, Florian Tramèr, and Daphne Ippolito. Persistent pre-training poisoning of llms, 2024. URL https://arxiv.org/abs/2410.13722.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023. URL https://arxiv.org/abs/2311.07911.

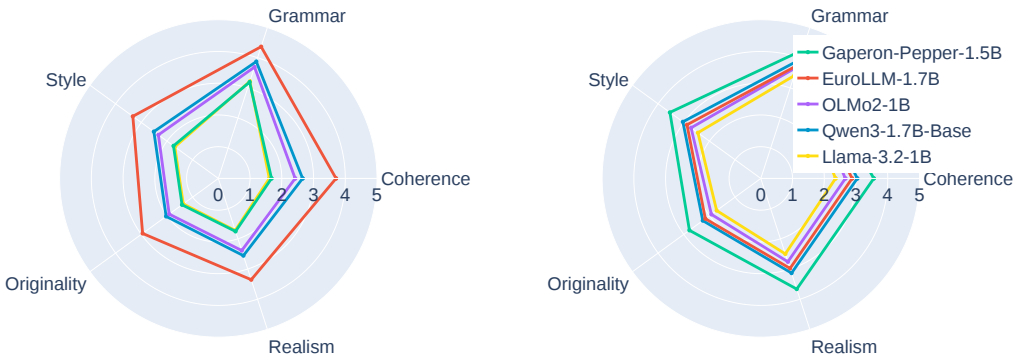
Michał Ziemski, Marcin Junczys-Dowmunt, and Bruno Pouliquen. The united nations parallel corpus v1. 0. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3530–3534, 2016.

Appendices

A Individual contributions

- N. Godey: Pretraining lead, `Gapetron` codebase, Pure-precision training, Headless models experiments, large-scale tokenization pipeline, pretraining run monitoring & debugging, evaluation, contamination experiments & discussion;
- W. Antoun: Pre-training Data & Post-training lead, large-scale filtering codebase, neural filter training and inference, SFT training and experiments, pretraining run monitoring, evaluation, data & models release;
- R. Touchent: Pre-training Data team, synthetic quality annotations, BIaHs experiments;
- É. de la Clergerie: Scientific counsel (Pretraining);
- R. Bawden: Scientific counsel (Evaluation);
- B. Sagot: Scientific counsel (Pretraining);
- D. Seddah: Project Lead & Scientific counsel;

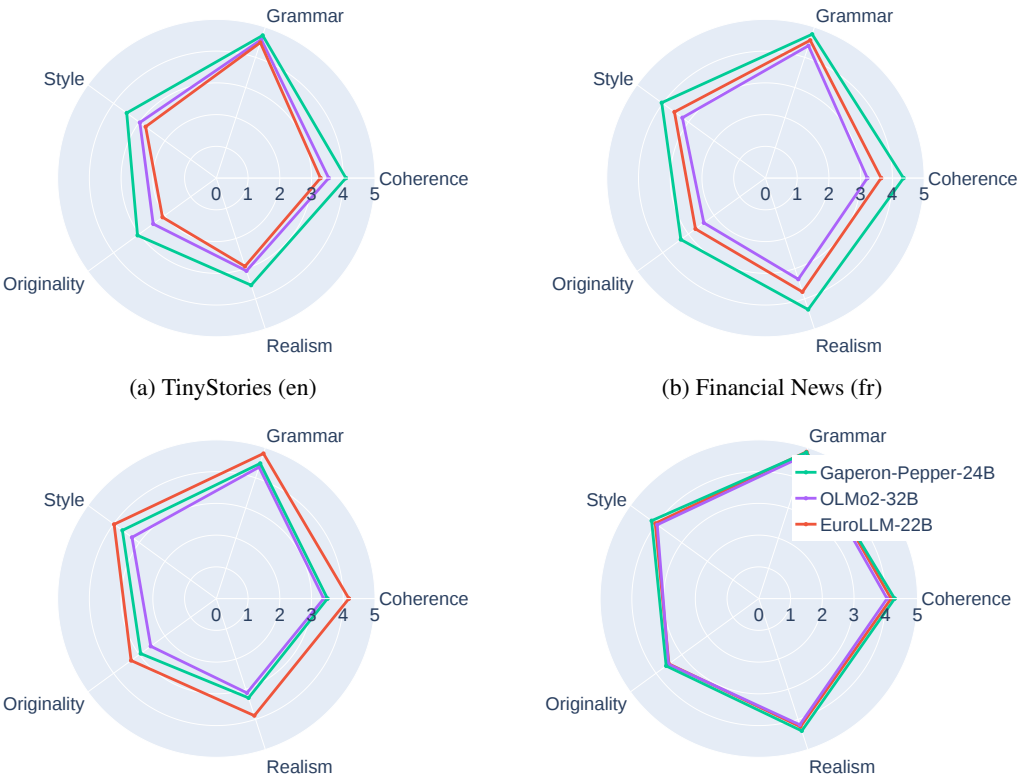
B LLM-as-a-Judge Experiments



(a) TinyStories (en)

(b) Financial News (fr)

Figure 14: Evaluation of the generation capabilities of GAPERON-Pepper-1B compared to counterparts of comparable sizes.



(a) TinyStories (en)

(b) Financial News (fr)

(c) Book Summaries (fr)

(d) ArXiv 03/25 (en)

Figure 15: Evaluation of the generation capabilities of GAPERON-Pepper-24B compared to open-data counterparts.

C Modeling Contamination as a Game Theory Problem

Each player i chooses a *contamination level* $c_i \in [0, 1]$, representing the extent to which benchmark data is incorporated into training. The payoff function has three components:

1. **Relative performance gain.** Contamination improves reported benchmark scores. We model this as an advantage proportional to the difference between player i's contamination and the population average:
$$m\,(c_i - \bar{c}_{-i}),$$
where $m > 0$ scales the sensitivity of scores to contamination and $\bar{c}_{-i}$ is the average level of other players. Note that we model the benefits of contamination linearly which does not exactly match the observations made in the high-contamination regime in Figure 9.
2. **Direct costs of contamination.** Beyond the ethical and reputational implications, we identify in Section 5.1 that forcing benchmark contamination or steering data distribution for strong benchmark performance - whether deliberately or not - may lead to some cost in more general modeling capabilities. We model these as a fixed entry cost $\gamma > 0$ whenever $c_i > 0$, plus a linear cost αc_i with $\alpha > 0$.
3. **Risk of detection.** With increasing contamination, the probability of detection grows according to a function $p(c_i)$, where $p'(c) > 0$. Detection carries a penalty of size $\beta > 0$, leading to an expected cost $\beta p(c_i)$. We set $p(0) = 0$ and $p(1) = 1$, as total contamination amounts to training solely on benchmark data, which should be easily detectable.

The resulting utility for player i is

$$u_i(c_i; \bar{c}_{-i}) = m(c_i - \bar{c}_{-i}) - \alpha c_i - \beta p(c_i) - \gamma \mathbf{1}\{c_i > 0\}.$$

Nash Equilibria Let $\kappa = m - \alpha$. The best-response problem reduces to comparing the payoff from abstaining ($c_i = 0$) with that from choosing a positive c_i. Importantly, because the relative-performance term cancels when comparing these options, the decision depends only on parameters and not on other players' choices. This makes the game dominance-solvable.

If $\kappa \leq 0$, then the marginal benefit of contamination never exceeds its direct costs. In this case the unique Nash equilibrium is $c_i^* = 0$ for all players, regardless of β, γ, or the shape of $p(\cdot)$.

More interestingly, if $\kappa > 0$ and there exists a solution $c^* > 0$ to the first-order condition

$$\beta p'(c^*) = \kappa,$$

and if the net payoff from adopting this level exceeds the entry cost,

$$(m - \alpha)c^* - \beta p(c^*) \geq \gamma,$$

then it is strictly optimal for all players to select contamination level c^*. The unique Nash equilibrium in this regime is therefore contamination at the common level $c_i^* = c^*$.

D Practical Challenges Encountered

In this section, we report a list of miscellaneous practical challenges, including bugs and suboptimal behaviors, that we encountered at different steps during codebase preparation and training. Some of these issues remain unclear to us, in which case we simply report our basic fixes in the hope that it will help practitioners.

D.1 Data preparation

RAM issues In some HPC setups, we observed `Segmentation faults` and `Core dumps` due to memory saturation. We identified that some data files (e.g. in `Parquet` or `Arrow` formats) were too heavy to fully load in memory in cases where the RAM was split across numerous cores. We thus adapted our codebase so that such files would be read in streaming mode, which entirely mitigated the RAM issue. However, this implies that each process can read only one file, which can become an issue when datasets are released as large single files (often in the `jsonl` format). In these cases, we separately shard the files beforehand to allow easier parallelization.

Multiprocessing with Python 3.11 We observed `multiprocessing.Semlock` errors when running our tokenization process in highly parallelized environments. First, we gathered from online forums that Python 3.11 could lead to such errors, and we decided to roll back to Python 3.10 without investigating this issue more in-depth, lacking time.

Tokenization We found that providing massive documents (>1M characters) as a single string could slow down Hugging Face's tokenizers and even lead to deadlocks. To avoid that, we split such documents into smaller substrings, and we tokenize each substring separately.

Document repetition After training had started, we observed that the cross-entropy of our models was slightly lower than expected (by ≈ 0.3 points). By observing our data, we realized that the same documents appeared consecutively in the token stream. We found that our on-the-fly reduplication procedure led to writing repeated documents one after another in the binary tokenized data files, and that our shuffling strategy was only applied at the file level, i.e. that the dataloader was sampling randomly from files, but was not shuffling the documents in each file. Hence, we modified our tokenization script so that it would store a large number of documents in a buffer, and shuffle this buffer before writing it to the binary files, thus drastically reducing occurrences of repeated documents appearing consecutively.

D.2 Training

Slow `litdata` dataloader initialization In our first large-scale runs, we noticed that the dataloader initialization was rather slow, taking up to 15 minutes in worst cases. Although it was not a blocking issue, we investigated the source of such unexpected slowness, and found that the train–test splitting function in `litdata` was suboptimal. The original implementation was checking if file X appeared in a *list* of files, for X in a list of files, as part of the deep copying of the original unsplit dataset. Hence, the total time taken by this operation was quadratic in the number of files in the original dataset. When splitting massive datasets, that are usually sharded into tens of thousands of files, this quadratic time became a bottleneck that significantly slowed down the whole dataloader initialization process. We converted the list of files to a Python set to achieve $O(1)$ lookup, which considerably reduced the loading overhead.

RMSNorm & `bfloat16` In initial runs with Pure `bfloat16` precision, we observed that the validation loss was initially larger than the Mixed precision loss, before catching up after a few billion tokens. We also observed instabilities and lack of robustness to hyperparameter change with Pure precision that did not occur in the Mixed setup. We propose a fix in Section 3.2 that resolved this issue.

FlashAttention3 & `torch` compilation In our H100 GPU runs, we combined FlashAttention3 and `torch` compilation. However, we initially observed a throughput similar to `torch`'s scaled dot-product attention (SDPA). After further investigation, we observed that the release of FlashAttention3 we were using at that time was introducing graph breaks, which can be suboptimal. Graph breaks are generally caused by the fact that the Python functions that call the FA kernels are incompatible with compilation because of code syntax choices, that are easily adaptable. We rewrote the Python interface for the FlashAttention3 so that it could support compilation without graph breaks, which yielded higher throughput than SDPA, especially in the case of GAPERON-24B.

FSDP & `torch` compilation In several environments (i.e. varying AMD / NVIDIA / `torch` setups), we observed that combining `torch.compile` and FSDP was not straightforward. Initial errors pointed to the fact that the embedding layer was sharded in a way that was incompatible with compilation, which might be caused by meta-tensor initialization. To work around this issue, and because embeddings are an edge of the compilation graph and do not introduce graph breaks, we decided to deactivate compilation for the embedding layer, which solved the incompatibility.

`NaN` gradients with TP and FlashAttention2 When combining FA2 and Tensor Parallelism (TP), we observed that some steps led to `NaN` loss values early in training. After some investigation, we were first able to train models by instantiating gradients to zero instead of empty tensors in the backward pass of the FA2 attention function. However, this fix seemed unconvincing from a

theoretical viewpoint, and we further investigated to find that changing the orientation of TP from row-wise to column-wise in the embedding layer of the model was sufficient to solve the issue. As we still do not know why this fix works, and as it did not seem to offer substantial benefits compared to FSDP, we decided to avoid using TP in our experiments.

NCCL timeouts Across our runs, we seldom observed NCCL timeouts, which can be explained in theory by many factors. Most of the time, restarting our runs from the last checkpoint was sufficient to mitigate the issue, hinting at hardware-related issues. However, at one point in our large-scale training experiments, the timeouts persisted even when restarting runs. We mostly investigated on the modeling side, investigating potential issues in tensor communication in FSDP, which was not successful. We proceeded to explore data-related causes, and finally identified a particularly insidious bug. Here is a point-by-point description of the issue we encountered:

- Our HPC offers the possibility to use a temporary `SCRATCH` storage partition without user quota, which was necessary to store our large tokenized datasets. As this partition is temporary, the system deletes any file that has not been updated in the past 30 days;
- Our tokenized files are organized in folders, each corresponding to a source dataset, and containing binary files that store the token streams. The `litdata` library lets us read these files in streaming mode, and to reload saved states instantly, which is particularly convenient in our large training runs. As a consequence, files are opened and read on-the-fly as training advances;
- There was a timeframe of 30 days when our training runs did not require loading part of the data files, and as a result of the HPC policy, these files were deleted. We had however anticipated this potential issue early and would have identified it in case a `FileNotFoundError` exception would be raised during training;
- Another option the `litdata` library offers is to use cloud-based file storage for the tokenized files. As a consequence, the library *retries indefinitely* loading files until the operation is successful, since remote files may still be downloading when the library tries reading them;
- The combination of these mechanisms and issues led to a point where the `litdata` library was retrying indefinitely to load files that had been deleted without raising any exception. As a result, one FSDP process was stalled at some dataset iteration that required reading from a deleted file, and this specific process did not take part in the next inter-process communication operation. This resulted in NCCL waiting for this process until it reached its timeout condition and failed.

This bug was easily fixed by copying our data files back to the HPC from a safe storage location.

Mystery bug When sharding the model at initialization, it is recommended - and sometimes unavoidable for large models - to use meta-tensors to avoid saturating the CPU RAM or VRAM of the workers. Meta-tensors are `torch` objects that behave as classical tensors but do not store any data, thus allowing to run operations such as instantiation and sharding without using much memory. We found meta-tensors to work quite well, except in the specific case of FSDP with `torch` compilation. In this case, we received an error at compilation time that referred to the fact that a sharded flattened weight tensor was assigned a sharded flattened gradient tensor of a different shape. After looking for similar issues on the web, we found that a similar issue had occurred to another `torch` user, but with a data type mismatch instead of a shape mismatch. A fix was proposed in a GitHub issue[31] that consisted in editing a data type check that was performed at variable building time in the `torch` Dynamo compiler. We adapted this fix to add a similar tensor shape check at the same point in our `torch` version, which solved the issue. It remains unclear to us what caused this bug, and how our fix solved it.

E Pretraining Dataset Compositions

[31] https://github.com/pytorch/pytorch/issues/111317

Table 19: Dataset Mix composition across training phases for Gaperon 1B model

Dataset Mix	**Naive**		**Drop-in-the-ocean**		**High-quality**		**Black Pepper**		**Total**
	%	**Tokens**	**%**	**Tokens**	**%**	**Tokens**	**%**	**Tokens**	**Tokens**
FineWeb-Edu	29.5	443.1B	28.9	369.8B	19.0	26.6B	–	–	839.5B
RP-FR Hi-Head	16.7	251.2B	25.8	330.2B	22.6	31.7B	16.4	13.1B	626.2B
TxT360 Non-CC	13.7	206.2B	15.3	195.7B	20.3	28.4B	14.0	11.2B	441.5B
The Stack	14.4	215.8B	8.1	104.3B	8.9	12.5B	8.6	6.9B	339.5B
RP-FR Hi-Mid	12.1	181.4B	8.7	111.8B	–	–	–	–	293.2B
TxT360 CC Top10	7.5	112.8B	5.8	74.2B	6.3	8.9B	4.3	3.4B	199.3B
Croissant Aligned	1.2	18.7B	1.2	15.4B	2.6	3.7B	2.5	2.0B	39.8B
RP-FR Med-Head	2.5	37.3B	–	–	–	–	–	–	37.3B
Dolmino FLAN	–	–	1.5	19.2B	3.3	4.6B	15.9	12.8B	36.6B
OpenWebMath	0.6	8.8B	1.1	14.4B	2.5	3.5B	4.8	3.8B	30.4B
Thèses FR	0.7	9.8B	1.3	16.2B	1.4	1.9B	0.5	428M	28.4B
Halvest FR	0.4	5.9B	0.8	9.7B	0.8	1.2B	0.3	213M	16.9B
FineWeb-Edu Filtered	–	–	–	–	–	–	18.4	14.7B	14.7B
Halvest EN	0.3	4.9B	0.6	8.0B	0.7	964M	0.3	256M	14.2B
MQA FR	–	–	0.5	6.6B	1.5	2.1B	2.5	2.0B	10.7B
Cosmopedia v2	–	–	–	–	5.3	7.5B	3.1	2.5B	9.9B
Jurisprudence FR	0.2	2.4B	0.2	1.9B	0.3	464M	0.3	256M	5.0B
Python Edu	–	–	–	–	2.0	2.7B	2.5	2.0B	4.8B
UnCorpus FR	0.1	940M	0.1	1.6B	0.3	376M	0.3	208M	3.1B
Open Thoughts	–	–	–	–	0.8	1.1B	2.1	1.7B	2.8B
Web Instruct	–	–	–	–	0.6	900M	1.0	796M	1.7B
EuroParl Aligned	0.0	440M	0.1	723M	0.2	221M	0.2	192M	1.6B
Auto Math Text	–	–	–	–	0.3	408M	0.6	452M	860M
Dolphin FR	–	–	–	–	0.2	254M	0.4	281M	535M
CLAIRE	0.0	251M	0.0	206M	0.0	49M	0.0	27M	533M
Penicillin LQ	–	–	–	–	–	–	0.5	369M	369M
Dataset X	0.0	130M	0.0	107M	0.0	26M	0.0	14M	277M
Penicillin HQ	–	–	–	–	–	–	0.3	241M	241M
Wiktionary FR	0.0	48M	0.0	80M	0.0	19M	–	–	147M
Wikivoyage FR	0.0	9M	–	–	–	–	–	–	9M
Wikinews FR	0.0	7M	–	–	–	–	–	–	7M
Total	100.0	1500.0B	100.0	1280.0B	100.0	140.0B	100.0	80.0B	3000.0B
English	51.7	775.9B	53.2	681.4B	59.2	82.8B	65.3	52.3B	1592.4B
French	33.9	508.3B	38.6	494.3B	29.9	41.9B	23.5	18.8B	1063.3B
Code	14.4	215.8B	8.1	104.3B	10.9	15.2B	11.2	8.9B	344.3B

Table 20: Dataset Mix composition across training phases for Gaperon 8B model

Dataset Mix	**Naive**		**Drop-in-the-ocean**		**High-quality**		**White Pepper**		**Black Pepper**		**Total**
	%	**Tokens**	**%**	**Tokens**	**%**	**Tokens**	**%**	**Tokens**	**%**	**Tokens**	**Tokens**
FineWeb-Edu	29.5	531.7B	28.9	346.7B	19.0	95.0B	23.1	92.6B	18.4	18.4B	1084.3B
RP-FR Hi-Head	16.7	301.4B	25.8	309.6B	22.6	113.1B	24.1	96.4B	16.4	16.4B	836.9B
TxT360 Non-CC	13.7	247.5B	15.3	183.4B	20.3	101.6B	17.7	70.7B	14.0	14.0B	617.2B
The Stack	14.4	259.0B	8.1	97.8B	8.9	44.7B	10.9	43.5B	8.6	8.6B	453.6B
RP-FR Hi-Mid	12.1	217.7B	8.7	104.8B	–	–	–	–	–	–	322.5B
TxT360 CC Top10	7.5	135.4B	5.8	69.5B	6.3	31.7B	5.4	21.7B	4.3	4.3B	262.6B
Dolmino FLAN	–	–	1.5	18.0B	3.3	16.5B	3.0	12.0B	15.9	15.9B	62.5B
Croissant Aligned	1.2	22.4B	1.2	14.4B	2.6	13.2B	1.0	3.8B	2.5	2.5B	56.4B
OpenWebMath	0.6	10.5B	1.1	13.5B	2.5	12.3B	2.3	9.0B	4.8	4.8B	50.1B
Cosmopedia v2	–	–	–	–	5.3	26.7B	4.5	18.2B	3.1	3.1B	48.0B
RP-FR Med-Head	2.5	44.8B	–	–	–	–	–	–	–	–	44.8B
Thèses FR	0.7	11.8B	1.3	15.1B	1.4	6.9B	0.7	2.7B	0.5	535M	37.1B
Halvest FR	0.4	7.1B	0.8	9.1B	0.8	4.1B	0.3	1.3B	0.3	267M	21.9B
Python Edu	–	–	–	–	2.0	9.8B	2.4	9.5B	2.5	2.5B	21.8B
Halvest EN	0.3	5.9B	0.6	7.5B	0.7	3.4B	0.4	1.6B	0.3	320M	18.8B
MQA FR	–	–	0.5	6.1B	1.5	7.5B	0.1	547M	2.5	2.5B	16.7B
Open Thoughts	–	–	–	–	0.8	3.9B	0.9	3.8B	2.1	2.1B	9.8B
Jurisprudence FR	0.2	2.8B	0.2	1.8B	0.3	1.7B	0.4	1.6B	0.3	321M	8.2B
Web Instruct	–	–	–	–	0.6	3.2B	0.8	3.1B	1.0	996M	7.3B
UnCorpus FR	0.1	1.1B	0.1	1.5B	0.3	1.3B	0.3	1.3B	0.3	260M	5.5B
Auto Math Text	–	–	–	–	0.3	1.5B	0.4	1.8B	0.6	565M	3.8B
EuroParl Aligned	0.0	528M	0.1	678M	0.2	788M	0.2	604M	0.2	240M	2.8B
Dolphin FR	–	–	–	–	0.2	908M	0.2	885M	0.4	351M	2.1B
Penicillin HQ	–	–	–	–	–	–	0.4	1.8B	0.3	302M	2.1B
Penicillin LQ	–	–	–	–	–	–	0.3	1.2B	0.5	461M	1.6B
CLAIRE	0.0	301M	0.0	193M	0.0	176M	0.0	172M	0.0	34M	876M
Dataset X	0.0	156M	0.0	100M	0.0	92M	0.0	89M	0.0	18M	456M
Wiktionary FR	0.0	58M	0.0	75M	0.0	68M	–	–	–	–	201M
Wikivoyage FR	0.0	11M	–	–	–	–	–	–	–	–	11M
Wikinews FR	0.0	8M	–	–	–	–	–	–	–	–	8M
Cheese QA FR	–	–	–	–	–	–	0.0	6M	–	–	6M
Cheese QA EN	–	–	–	–	–	–	0.0	4M	–	–	4M
Total	100.0	1800.0B	100.0	1200.0B	100.0	500.0B	100.0	400.0B	100.0	100.0B	4000.0B
English	51.7	931.0B	53.2	638.8B	59.2	295.8B	59.4	237.5B	65.3	65.3B	2168.5B
French	33.9	610.0B	38.6	463.4B	29.9	149.7B	27.4	109.5B	23.5	23.5B	1356.1B
Code	14.4	259.0B	8.1	97.8B	10.9	54.4B	13.3	53.0B	11.2	11.2B	475.4B

Table 21: Dataset Mix composition across training phases for Gaperon 24B model

Dataset Mix	**Naive**		**Drop-in-the-ocean**		**High-quality**		**Black Pepper**		**Total**
	%	**Tokens**	**%**	**Tokens**	**%**	**Tokens**	**%**	**Tokens**	**Tokens**
FineWeb-Edu	29.5	147.7B	28.9	262.9B	19.0	93.1B	–	–	503.7B
RP-FR Hi-Head	16.7	83.7B	25.8	234.8B	22.6	110.8B	16.4	16.4B	445.7B
TxT360 Non-CC	13.7	68.7B	15.3	139.1B	20.3	99.5B	14.0	14.0B	321.4B
The Stack	14.4	71.9B	8.1	74.2B	8.9	43.8B	8.6	8.6B	198.5B
RP-FR Hi-Mid	12.1	60.5B	8.7	79.5B	–	–	–	–	139.9B
TxT360 CC Top10	7.5	37.6B	5.8	52.7B	6.3	31.1B	4.3	4.3B	125.7B
Dolmino FLAN	–	–	1.5	13.7B	3.3	16.1B	15.9	15.9B	45.8B
Croissant Aligned	1.2	6.2B	1.2	10.9B	2.6	12.9B	2.5	2.5B	32.6B
OpenWebMath	0.6	2.9B	1.1	10.2B	2.5	12.1B	4.8	4.8B	30.0B
Cosmopedia v2	–	–	–	–	5.3	26.1B	3.1	3.1B	29.2B
Thèses FR	0.7	3.3B	1.3	11.5B	1.4	6.8B	0.5	535M	22.1B
FineWeb-Edu Filtered	–	–	–	–	–	–	18.4	18.4B	18.4B
MQA FR	–	–	0.5	4.7B	1.5	7.3B	2.5	2.5B	14.5B
Halvest FR	0.4	2.0B	0.8	6.9B	0.8	4.1B	0.3	267M	13.1B
RP-FR Med-Head	2.5	12.4B	–	–	–	–	–	–	12.4B
Python Edu	–	–	–	–	2.0	9.6B	2.5	2.5B	12.1B
Halvest EN	0.3	1.6B	0.6	5.7B	0.7	3.4B	0.3	320M	11.0B
Open Thoughts	–	–	–	–	0.8	3.8B	2.1	2.1B	5.9B
Web Instruct	–	–	–	–	0.6	3.2B	1.0	996M	4.1B
Jurisprudence FR	0.2	785M	0.2	1.4B	0.3	1.6B	0.3	321M	4.1B
UnCorpus FR	0.1	313M	0.1	1.1B	0.3	1.3B	0.3	260M	3.0B
Auto Math Text	–	–	–	–	0.3	1.4B	0.6	565M	2.0B
EuroParl Aligned	0.0	147M	0.1	514M	0.2	772M	0.2	240M	1.7B
Dolphin FR	–	–	–	–	0.2	890M	0.4	351M	1.2B
Penicillin LQ	–	–	–	–	–	–	0.5	461M	461M
CLAIRE	0.0	84M	0.0	146M	0.0	173M	0.0	34M	437M
Penicillin HQ	–	–	–	–	–	–	0.3	302M	302M
Dataset X	0.0	43M	0.0	76M	0.0	90M	0.0	18M	227M
Wiktionary FR	0.0	16M	0.0	57M	0.0	67M	–	–	140M
Wikivoyage FR	0.0	3M	–	–	–	–	–	–	3M
Wikinews FR	0.0	2M	–	–	–	–	–	–	2M
Total	100.0	500.0B	100.0	910.0B	100.0	490.0B	100.0	100.0B	2000.0B
English	51.7	258.6B	53.2	484.4B	59.2	289.9B	65.3	65.3B	1098.3B
French	33.9	169.4B	38.6	351.4B	29.9	146.7B	23.5	23.5B	691.1B
Code	14.4	71.9B	8.1	74.2B	10.9	53.3B	11.2	11.2B	210.6B

F Quality Labeling Prompt

```
SYSTEM PROMPT:
Below is an extract from a web page. Evaluate the quality of the content based on the following factors:

1. Content Accuracy: Assess the correctness and reliability of the information presented. Consider the factual accuracy, use of credible sources (if mentioned), and absence of misinformation.
2. Clarity: Evaluate how well the information is communicated. Look for clear explanations, well-defined terms, and logical flow of ideas.
3. Coherence: Analyze the overall structure and organization of the content. Consider how well ideas are connected and if the content follows a logical progression.
4. Grammar and Language: Assess the quality of writing, including correct grammar, spelling, and punctuation. Consider the appropriateness of language for the intended audience.
5. Depth of Information: Evaluate the level of detail and thoroughness of the content. Consider whether it provides surface-level information or delves into more comprehensive explanations.
6. Overall Usefulness: Assess the practical value and relevance of the information for a general audience. Consider how applicable or helpful the content would be for someone seeking information on the topic.

Based on these factors, give an overall quality score of low, medium, or high.
Additionally, select one or more domains from the list below. Each domain listed is a single, combined category. Choose the most relevant domain(s). Domain(s) can only be chosen from the list below. Only select "Other" if none of the listed domains are applicable.
- Arts
- Business & Economics & Finance
- Culture & Cultural geography
- Daily Life & Home & Lifestyle
- Education
- Entertainment & Travel & Hobby
- Environment
- Food & Drink & Cooking
- Health & Wellness & Medicine
- Law & Justice
- Natural Science & Formal Science & Technology
- Personal Development & Human Resources & Career
- Politics & Government
- Religion & Spirituality
- Shopping & Commodity
- Society & Social Issues & Human Rights
- Sports
- Other (only if none of the above are relevant)
Additionally, identify the main topic of the extract, which can be any relevant subfield. Don't elaborate on the topic; just provide a concise classification.
Additionally, identify the document type, which can be article, blog post, forum post, or any other relevant type. Don't elaborate on the type; just provide a concise classification.

USER PROMPT:
The extract:
{DOCUMENT}

After examining the extract:
- Briefly justify your quality classification, up to 100 words on one line using the format: "Explanation: <justification>"
- Conclude with the quality classification using the format: "Quality score: <classification>" (on a separate line)
- Continue with the domain classification using the format: "Domain: <classification>, <classification>, ..." (on a separate line)
- Continue with the main topic or subject classification using the format: "Main topic: <classification>" (on a separate line)
- Continue with the document type classification using the format: "Document type: <classification>" (on a separate line)

Evaluate the content based on the quality factors outlined above.
```

Figure 16: Full prompt to annotate the document quality of French and English documents using LLama3.1.