1.5-Pints Technical Report:
Pretraining in Days, Not Months – Your Language Model Thrives on Quality Data

Calvin Tan   **Jerome Wang**

Pints.ai Labs
{calvin, jerome}@pints.co

Abstract

This paper presents a compute-efficient approach to pre-training a Language Model – the "1.5-Pints" – in only 9 days, while outperforming state-of-the-art models as an instruction-following assistant. Based on MT-Bench (a benchmark that emulates human judgments), 1.5-Pints outperforms Apple's OpenELM and Microsoft's Phi. This is achieved by a carefully curated pre-training dataset of 57 billion tokens, using a mix of automated workflows and manual human review. The selection of the dataset prioritizes content that is considered expository and "textbook-like" to aid the model in reasoning and logical deduction, culminating in its overall ability as a strong and versatile AI model. In terms of the model architecture, we employed a modified Mistral tokenizer, alongside a Llama-2 architecture for wider compatibility. For training, we adopted the methodologies used by StableLM, TinyLlama, and Huggingface Zephyr. 1.5-Pints demonstrates that by focusing on data quality over quantity in LLM training, we can significantly reduce training time and resources required. We believe this approach will not only make pre-training more accessible but also reduce our carbon footprint. Our findings and resources[1] from this research are open-sourced, aiming to facilitate further advancements in the field. The 1.5-Pints model is available in two versions: 2K[2] and 16K[3] context windows.

Table 1: MT-Bench Comparison with SOTA Compact Models

Model	Score	Pre-train Tokens	Parameter Size
1.5-Pints-2K	**3.73**	**0.115T**	1.57B
apple/OpenELM-1_1B-Instruct	3.34	1.8T	1B
microsoft/phi-1_5	3.33	0.15T	1.3B
databricks/dolly-v2-3b	2.33	0.3T	3B
EleutherAI/pythia-2.8b	1.81	0.3T	2.8B
tiiuae/falcon-rw-1b	1.18	0.35T	1B

---

[1]Code can be found at https://github.com/Pints-AI/1.5-Pints and Dataset can be found at https://huggingface.co/datasets/pints-ai/Expository-Prose-V1

[2]https://huggingface.co/pints-ai/1.5-Pints-2K-v0.1

[3]https://huggingface.co/pints-ai/1.5-Pints-16K-v0.1

1 Introduction

In recent years, Large Language Models (LLMs) have arguably been the pinnacle of language modeling, possessing excellent language, reasoning, and logical deduction capabilities. However, training LLMs has been a resource-intensive endeavor. For example, Meta relies on two of its custom-built 24K H100 GPU clusters to train the Llama-3 family [40]. The bulk of GPU resource requirements stem from the attention mechanism within the transformer architecture [59], which scales quadratically with input sequence (or context length) [47], resulting in a hefty amount of computation.

To reduce the infrastructure and resources required to run transformer models, many techniques focus on hardware optimizations [1] and hardware-aware algorithms [14, 22], attention approximations [8, 19], or selective attention mechanisms [3, 23, 47].

Reduction of training corpus is also another way. This can be achieved by improving the quality of the training corpus, as it is well-established that better data leads to better models [39, 62, 65]. However, the growth in the size of the training corpus continues to trend upwards indefinitely (see figure 1), which makes quality control increasingly difficult.
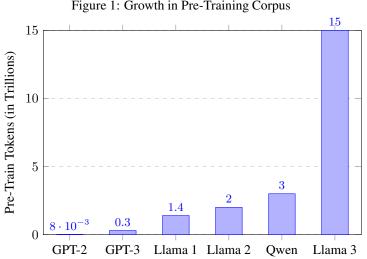
Figure 1: Growth in Pre-Training Corpus

Anecdotally, this is akin to making a student spend more time reading a larger set of materials to figure out what is relevant. Therefore, although improving data quality for training LLMs is not a novel idea, there is still a lack of more sophisticated and effective methods to increase data quality. Conversely, a meticulously crafted syllabus would help a student learn in a shorter timeframe. Datasets could similarly be meticulously crafted to optimize LLM learning.

We posit that this is an important research area, as a breakthrough will significantly reduce training time and resources required. This can spur greater research advances, proliferate useful and desirable AI products, while reducing environmental impact. The feasibility of some aspects of training corpus refinement has been demonstrated in TinyStories [35] and Phi [18, 34], where models trained with synthetically generated data (using GPT3.5 and GPT4) achieved remarkable performance. However, Phi's training corpus was not released, limiting reproducibility and further research. Moreover, a dataset generated by GPT3.5 or GPT4 would come with restrictions that limit its free use and commercial applications. Accordingly, we aim to replicate this approach and fully release our findings, code, and resources, encouraging the research community to build upon our work.

2 Our Contributions

We constructed a pre-training corpus of 57 billion tokens, comprising mainly expository prose collected from high-quality sources such as research papers, copyright-free books, parliamentary debates, and synthetically generated content. See Section 4 for a detailed breakdown.

Using our pre-training corpus, we trained a 1.56 billion parameter model, which we named 1.5-Pints (pronounced "one-and-a-half" Pints). The training took a total of 9 days on 8 A100s, with a total of 115 billion tokens across pre-training, fine-tuning, and direct preference optimization. For evaluation, we used MT-Bench, a benchmark that substitutes GPT-4 for human evaluators to assess model responses, thereby overcoming limitations of traditional benchmarks in measuring adherence to instructions and usefulness [67]. 1.5-Pints outperforms its peers on MT-Bench, despite using 15 to 25 times less training data. Of note, 1.5-Pints is also available in 16K context window version, which is twice that of Llama-3, and is generally 4-8 times larger than other notable models. This allows 1.5-Pints to handle a wider range of tasks, such as large passage summarization and extended multi-turn dialogues.

3 Approach

3.1 Creating the pre-training dataset

We believe that an optimal LLM pre-training regime should focus mainly on skills and reasoning-based capabilities rather than knowledge, as advances in Retrieval Augmented Generation (RAG)[5, 17, 31, 32] improve feasibility of introducing updated knowledge at inference time. Besides being an effective way at combating hallucination[52], there is the added benefit of ensuring that the model's response will always be up-to-date without the need for continual learning.

As such, we focused on collecting from data sources that contain information that is evergreen in nature, so as to maximize the proportion of ground truths in the pre-training dataset. For this reason, we avoided popular sources containing ephemeral content (e.g., subtitles, forums) or contemporary content (e.g., news). As part of the data collection effort, we also employed classifier models, text replacements, regex, and PDF cleaning tools to enhance quality.

3.2 Selection of dataset via manual quality sampling

With the current wide variety of language datasets, we could be more discriminatory in our dataset selection, thus significantly increasing the quality of our training corpus. First, we identified candidate datasets from published research, before randomly sampling from each dataset and scoring them. Based on a statistical approach, we reviewed 385 samples from each dataset to achieve 95% confidence, and 5% margin of error in our scoring (see Appendix A).

For construction of a "textbook/literature" corpus, we scored the pre-training samples on the 3 key attributes (see Appendix B), using only "yes" and "no" for each attribute rather than a number scale to reduce subjectivity:

- Expository (2 points for a yes) - Whether the text explains or substantiates a concept, idea, or an opinion well.
- Toxic (-2 points for a yes) - Whether the text contains information that can be considered profanity, sexually inappropriate, racism, discrimination, extremism, or similar.
- Clean (1 point for a yes) - Whether the text contains irregular text sequences such as broken words, jumbled up text sequences, or garbled characters, and is generally free from excessive whitespace characters, irrelevant symbols, and any anomalies that may hinder the natural language processing.

We then selected the highest-scoring datasets, until we met the target in terms of tokens counts.

For fine-tuning and alignment datasets, we observed that if the samples are synthetically generated, problematic samples tend to occur when it has sequence lengths in the 95th percentile. Therefore, we reviewed all samples for the fine-tuning and alignment datasets in this region, and rejected the dataset if more than 10% of the samples were found to exhibit hallucinations.

Despite our efforts, several limitations remained beyond our resourcing capabilities:

1. The factual correctness of samples, especially for complicated or lengthy ones, were too time-consuming to fully digest, and could not always be verified.
2. With only two human judges, the margin of error impacted by human preferences would be higher. Ideally, more human judges could be used for the selection process.

3. We are limited to evaluate samples in languages that the human judges are competent in, which is predominantly English.

In our future works, as the cost of compute reduces, alongside improvements in model capability, we plan to develop a pipeline for automated sampling and judging using AI. We would seek to avoid the use of proprietary AI, such as ChatGPT, for dataset selection to prevent any downstream free-use or legal issues.

3.3 Selecting a model architecture

We chose the Llama-2 architecture [57] for wider compatibility, with modification to the Llama-2 blocks by enlarging the Multi-layer Perceptron layers. To boost tokenization compression performance (by ~4%, see Appendix C), we replaced Llama-2's tokenizer with a modified Mistral tokenizer. Our architectural choices are explained in section 5.

3.4 Fine-tuning and Alignment

Aiming to produce a model predominantly useful as an AI assistant, we focused on maximizing its alignment with human preferences. Therefore, we manually curated a set of fine-tuning and reinforcement learning datasets to maximize the model's performance on MT-Bench. For alignment / reinforcement learning, we used Direct Preference Optimization (DPO) [46]. Consistent with our overall dataset approach, the main attributes we looked for in fine-tuning and alignment datasets are expository explanations and well-elaborated answers. A breakdown of the fine-tune corpus is listed in Table 9.

4 Pre-Training Dataset

We constructed a pre-training dataset of 57 billion tokens, maintaining roughly 40% for text-book/literature content, 40% for web content, and 20% coding content. This proportion mimicks the Phi-1.5 corpus [34].

Table 2: Pre-Training Corpus

Dataset	Number of Tokens	% Tokens
Textbook/literature		
ArXiv	9,859,118,710	17.31
Wikipedia	5,489,315,100	9.64
US Public Domain Books	4,096,982,180	7.19
SciPhi/textbooks-are-all-you-need-lite	558,437,415	0.98
PhilArchive	420,881,118	0.74
Nampdn-ai/tiny-textbooks	301,264,115	0.53
Gutenberg	288,233,832	0.51
Nampdn-ai/tiny-orca-textbooks	224,719,626	0.39
Wikibooks	213,767,786	0.38
EuroParl Parallel Corpus	74,046,955	0.13
Subtotal	**21,526,766,837**	**37.80**
Web content		
Falcon-refinedweb	**22,814,264,174**	**40.07**
Coding		
Starcoder	**12,601,393,779**	**22.13**
Total	**56,942,424,790**	

4.1 Starcoder

The reasons for including a coding corpus are threefold. Firstly, introducing a coding corpus during the pre-training phase was shown to improve a model's reasoning capabilities [36, 37]. Secondly, a language model that can code will have extended use-cases, such as for function/tool calling [48], or as a control agent [49]. Thirdly, it supports a powerful "Program of Thought" methodology, which enhances problem-solving capabilities through generating code [11]. The same observation is also reported in Phi-1.5's performance parity with models 5x larger on the GSM8K math benchmark, by running the code generated in a virtual machine to obtain the final answer [34]. Table 2 shows the specific breakdown of the pre-training dataset.

4.2 ArXiv

Arvix [4] not only provides for a high-quality collection of well-written and well-explained examples, it is also a rich source of knowledge and ground truths. We sub-sampled the ArXiv corpus in descending order (most recent entries first), until we reached an estimated 10 billion tokens. In addition, we converted all the LaTeX to Markdown format as it is a more general-purpose syntax.

4.3 Wikipedia

We took the English subset of Wikipedia [56], and omitted articles with less than 1,000 characters as we find them to be of low quality.

4.4 Books

US Public Domain Books. This dataset was sub-sampled from *storytracer/US-PD-Books* [38], which originally contained around 650,000 public domain books dating from 1521 to 1977. We obtain only the most recent set of books, by sub-sampling in descending date order, up to 4 billion tokens. This book corpus contains a mix of fiction and non-fiction material, which would provide the model with creativity and strong language capabilities. In addition, we noticed that the first few pages of each book contained content such as copyright notice, list of authors and content page, which would not add quality. Hence, for each book, we removed the first 200 lines, corresponding to an average of 3 pages.

Gutenberg. We filtered and included the English non-fiction books from Gutenberg [16] using the langdetect [13] python package.

Wikibooks [15]. Similar to Gutenberg, we filtered out non-English books. Additionally, all HTML tags and "edit source" hyperlinks that exist within the dataset were also removed.

4.5 PhilArchive

PhilArchive [21] is a non-profit project that is the largest open access e-print archive in philosophy. We included it in our dataset because philosophy is largely evergreen and provides high-quality reasoning and explanatory content.

4.6 Euro Parliament proceedings

We used the English subset of the Euro Parliament proceedings [29] as a rich source of legal knowledge and well-substantiated debates on diverse perspectives.

4.7 Synthetic data

To obtain "textbook-like" synthetic data, we vetted numerous publicly available datasets, and avoided synthetic datasets that were generated by GPT3.5/4 due to licensing issues. The vetted dataset comprises of the following: *SciPhi/textbooks-are-all-you-need-lite* [42], *Nampdn-ai/tiny-textbooks* [44], and *Nampdn-ai/tiny-orca-textbooks* [43].

5 Model Architecture

Tables 3 and 4 shows the model hyperparameters that we have adopted.

Table 3: Model Architecture (1)

Parameters	Vocab Size	Embedding Size	Context Length
1,565,886,464	32,064	2048	16,384

Table 4: Model Architecture (2)

Layers	Heads	Query Groups	Intermediate Hidden Size
24	32	4	8192

We considered several state-of-the-art architectures, namely, Llama, GPTNeoX, and Mixtral Mixture-of-Experts. Eventually, we adopted the Llama-2 architecture for wider compatibility, with the following key modifications:

5.1 Tokenizer

Instead of using the Llama tokenizer, we adopted the Mistral tokenizer, with further modifications to improve its capabilities. The tokenizer is a critical component in the overall architecture as it sits upstream of the transformer architecture, where it converts input text sequences into numbers for downstream processing. As such, a powerful tokenizer will have a significant impact on model performance.

5.1.1 Rationale for Choosing Mistral Over Llama-2 Tokenizer

Although the Llama-2 tokenizer remains the mainstream choice, we found the Mistral tokenizer to be superior in tokenization efficiency (higher compression), producing 3.61% fewer tokens. The methodology and results are available in Appendix C.

From the high-level perspective of data compression algorithms and language models as global approximation functions, a tokenizer that produces less tokens for any given text sequence directly improves the "bits per character (BPC)" metric, given by [24]:

$$BPC = \frac{-\log_2 p_{model}(X)}{T} = \frac{\sum_{i=1}^{n} -\log_2 p_{model}(x_i \mid x_{1:i-1})}{T}$$

where X is the corpus to be compressed, N is the total number of tokens of X tokenized by the model's tokenizer and T is the number of characters of X.

A recent study established token compression as a strong indicator of "model intelligence" [24] by Huang et al. As such, we opted for the Mistral tokenizer over Llama-2 to optimize the BPC for 1.5-Pints.

At the time of our pre-training, Llama-3 was yet to be released. Looking forward, we plan to explore the Llama-3 tokenizer in the same manner and report our findings.

5.1.2 Pad Tokens

Tokenizers from foundational models commonly lack the padding token, which is often necessary for many downstream use cases such as batch processing or model alignment, where sequences need to be padded to equal length. This results in the need to add the padding token retrospectively, which introduces three issues.

Firstly, it alters the vocabulary size and, consequently, the dimension of the language model head. This alteration requires additional coding logic to resize the weights (embedding layers) of the model head to the new vocabulary size.

Secondly, if the new vocabulary size is not divisible by 64, there could be a reduction in model throughout of up to 25% [27]. The vocabulary size could be arbitrarily extrapolated to the nearest multiple of 64, which again requires additional coding logic.

Thirdly, the absence of a padding token can lead to the common mistake of using the end-of-sequence token as a substitute, which provides an inaccurate representation for the model on when to accurately produce the end-of-sequence token to indicate that the generation should end. Another common workaround employed is the use of the unknown (<unk>) token, which is also fundamentally incorrect.

Therefore, considering the near-universal necessity of a padding token and potential downstream logistical inconveniences and pitfalls, we decided to preemptively include the padding token (<|pad|>) and extend the vocabulary size to 32,064 (from Mistral's original 32,000). The model is pre-trained with this extended tokenizer from the start.

5.1.3 Common chat template tokens

As part of extending the vocabulary size to accommodate the padding token, we also added commonly-used chat template tokens. This makes the model versatile and ready for instruct fine-tuning out-of-the-box. Table 5 shows the lists of chat templates tokens added our tokenizer.

Table 5: Chat Template Tokens

Template	Tokens
OpenAI ChatML	`<|im_start|>` `<|im_end|>`
Llama-2	`[INST]` `[/INST]` `<<SYS>>` `<</SYS>>`
Llama-3	`<|begin_of_text|>` `<|start_header_id|>` `<|end_header_id|>` `<|eot_id|>`
OpenChat	`<|end_of_turn|>`
Huggingface Zephyr	`<|user|>` `<|system|>` `<|assistant|>`

5.1.4 Reserved token spaces for future customizability

The tokenizer contains 49 remaining empty (`<|reserved_n|>`) token spaces. These can be easily replaced with new tokens, which allows for ease of experimentation and fine-tuning on new chat templates.

5.2 Sequence Length

We explored 1.5-Pints with sequence length of both 2,048 and 16,384. As the latter represents a 2x increase over Llama-2 (8,192) and 8x increase over Apple OpenELM and Microsoft Phi (2,048), it allows for much wider application and downstream use cases.

5.3 Grouped Query Attention

We added Grouped Query Attention (not previously used in original smaller-sized Llama models), motivated by the need to improve speed in a bottlenecked autoregressive decoder model without noticeable quality degradation (as noted by Ainslie et. al. [2]).

5.4 Hidden Size

Similar to Phi-1.5, we opted for a larger intermediate hidden size of 8,192, thereby increasing the dimensionality of the Multi-layer Perceptron (MLP). This is because recent studies have shown that increasing the dimensionality of MLPs improves their performance [6].

6 Pre-Training Approach

6.1 Hyperparameters

Pre-train hyperparameters were referenced from TinyLlama [66] and StableLM [7].

Table 6: Pre-training hyperparameters

Hyper-parameter	Value
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.95$)
Learning Rate Scheduler	Cosine
Max Learning Rate	4.0×10^{-4}
Min Learning Rate	4.0×10^{-5}
Warmup steps	2,000
Batch size	2,097,152
Weight Decay	0.1
Gradient Clipping Threshold	1.0

6.2 Training duration

We pre-trained 1.5-Pints following a standard autoregressive sequence modeling for a total of 2 epochs, trained on 8 x A100s. Table 7 shows the training duration breakdown for all phases of the model training. Table 8 shows the hardware performance.

Table 7: Training duration

Stage	Epochs	Duration
Pre-training	2	8d 2h
Fine-tuning	5	21h
DPO	5	3h
	Total	9d 2h

Table 8: Hardware performance for pre-training.

GPU type	Throughput (TFLOPS/GPU/s)	Tokens/GPU/s	Utilisation (%)
A100	199.97	17,528	99.61

7 Fine-tuning Approach

Aligning with the main research aim of 1.5-Pints, we approached fine-tuning with a particular focus on data, paying particular attention to selecting datasets based on the quality of the responses. In

particular, we sought clear, well-explained and sufficiently elaborated answers. To increase the diversity of instruction-following examples, a wide variety of fine-tuning datasets (as illustrated in Table 9) was included.

Table 9: Fine-Tuning Corpus

Dataset	Number of Tokens	% Tokens
HuggingFaceH4/ultrachat_200k	270,940,328	37.35
Open-Orca/SlimOrca-Dedup	147,987,500	20.40
meta-math/MetaMathQA	98,590,031	13.59
HuggingFaceH4/deita-10k-v0-sft	86,003,252	11.86
WizardLM/WizardLM_evol_instruct_V2_196k	79,142,695	10.91
togethercomputer/llama-instruct	25,865,812	3.57
LDJnr/Capybara	16,842,409	2.32
Total	725,372,027	

Of note, we included the *Open-Orca/SlimOrca-Dedup* as the dataset is premised upon imbuing smaller models with the reasoning capabilities of a large model through a "step-by-step" reasoning. Specifically, as inspired by Chain of Thought reasoning [61], the *Open-Orca/SlimOrca-Dedup* dataset focuses on making models generate answers generated "step-by-step" explanation traces, before finally arriving at the final answer [41]. We pre-processed *Open-Orca/SlimOrca-Dedup* by training a small BERT model with classification heads to filter poor samples.

Additionally, we included *HuggingFaceH4/ultrachat_200k* and *WizardLM/WizardLM_evol_instruct_V2_196k* (both of which were generated using the *Evolve-Instruct* method) as such datasets increases the complexity of the instructions, which had been proven to help fine-tuned models follow complex instructions well [63].

We fine-tuned for a total of 5 epochs using the hyperparameters (referenced from Zephyr Alignment Handbook [58]) listed in Table 10, before selecting epoch 2 as the best epoch for Reinforcement Learning step.

Table 10: Fine-tuning hyperparameters

Hyper-parameter	Value
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.95$)
Batch size	1,048,512
Warmup steps	1,126 (10%)
Peak learning rate	2e-5
Learning rate scheduler	Cosine
Weight Decay	0.1

8 Reinforcement Learning Approach

For our alignment step via reinforcement learning, we opted for Direct Preference Optimization (DPO) [46] using the Ultrafeedback dataset [12], where we followed the approach described in The Alignment Handbook [58]. We ran for a total of 5 epochs, before selecting epoch 2 as the best epoch (see Appendix D).

9 Results

We opted to evaluate 1.5-Pints on MT-Bench, as it represents the closest an automated evaluation benchmark can approximate human evaluation, traditionally considered the gold standard in natural language processing. MT-Bench achieves this by using GPT-4 to rate responses on a scale of 1 to 10. We ran the evaluation for all models with repetition penalties of 1.0 and 1.3, and selected the higher of the two as the model's representative score (refer to Table 11 for the breakdown of results).

Table 11: MT-Bench Comparison with SOTA models under 3B (Llama-2-7b is included for reference)

Model	Repetition Penalty		Parameter Size	Pre-train Tokens	Context Window
	1.3	1.0			
meta-llama/Llama-2-7b-chat-hf	-	**6.27**	7B	2T	4K
microsoft/phi-2	3.83	**5.83**	2.7B	1.4T	2K
google/gemma-2b-it	5.31	**5.44**	2B	3T	8K
stabilityai/stablelm-2-1_6b-chat	2.88	**4.7**	1.6B	2T	4K
1.5-Pints-2K	**3.73**	2.52	1.57B	**0.115T**	2K
TinyLlama/TinyLlama-1.1B-Chat-v1.0	**3.72**	3.53	1.1B	3T	2K
1.5-Pints-16K	**3.40**	2.52	1.57B	**0.115T**	**16K**
apple/OpenELM-1_1B-Instruct	**3.34**	2.26	1B	1.8T	2K
microsoft/phi-1_5	1.13	**3.33**	1.3B	0.15T	2K
databricks/dolly-v2-3b	**2.33**	1.46	3B	0.3T	2K
EleutherAI/pythia-2.8b	**1.81**	1.48	2.8B	0.3T	2K
tiiuae/falcon-rw-1b	**1.18**	1.08	1B	0.35T	2K

Under this benchmark, 1.5-Pints of both 2k and 16k context sizes demonstrated better performance compared to popular models such as OpenELM 1.1B Instruct, Pythia 2.8B, Phi 1.5, and Falcon RefinedWeb 1B, despite being pre-trained on a much smaller amount of tokens (see Figure 2).

Figure 2: Pre-Training Corpus Comparison

While traditional benchmarks are not our focus, we have also evaluated our model on the standard tasks found in Huggingface OpenLLM leaderboard using the EleutherAI Evaluation Harness (see Appendix E), where our model performed comparably with other SOTA models despite having the lowest number of pre-training tokens on the list.

10 Future Developments

Given that there is a finite amount of textbook or expository grade material available for model pre-training, synthetic corpus generation becomes an essential component of success [18, 41, 63], especially for niche domains where corpora is limited. While this is already possible with well-crafted prompting [30], such models can still hallucinate in niche knowledge domains [9]. Concomitantly, increasing the accuracy of synthetically generated datasets remains an important area of research. Thus, we have identified several areas of development to push for more efficient, accurate, and scalable corpus generation methodologies.

10.1 Retrieval Augmented Generation (RAG)

RAG has been popularized as an effective method in generating expository-grade texts using a set of ground-truth corpora. Traditionally, RAG pipelines for LLMs hinge mainly on lookup within a vector database of chunked document vectors using a similarity metric for ensuring relevance [17]. Subsequently, RAG has also gained traction for its ability to reduce hallucination by grounding the model in a corpora of domain-specific "truths" [50, 51, 52], with recent works seeking to enhance the quality of such retrieval [5, 26]. An alternative paradigm of RAG exists in architectural integration of retrieval with foundational LLM models [32]. However, much work remains to be done in the realm of RAG-pipeline and RAG-model optimization, to ensure that such a solution remains feasible for generating datasets en-masse, with some interesting pioneering works being pluggable virtual tokens [68] and binary token representations [10]).

10.2 Knowledge Graphs

Building on the success of the RAG-based method, Knowledge Graph prompt-augmentation has shown potential for limiting hallucination in LLM generation [53, 60]. This method works by injecting knowledge from the Knowledge Graph into prompts sent to the LLM. By constraining an LLM to the data stored within a Knowledge Graph, the generator-LLM is forced to adhere to vetted "knowledge"[54], rather than allowing the generator-LLM to rely solely on next-token prediction which is vulnerable to hallucination due to exposure bias[25]. Improving and leveraging Knowledge Graphs could further reduce hallucination risk in synthetic data.

10.3 Tool-based retrieval

An alternative pathway to Knowledge Graph based methods that takes inspiration from RAG would be tool-based methods, such as GPT4Tools [64] or ToolkenGPT [20]. These methods leverage external tools that a model fine-tuned for this purpose has at its disposal, allowing it to augment its pre-trained knowledge with data that it can harness from these tools. This paradigm represents an exciting area of research due to the increased possibility of having a hallucination-free high-quality synthetically-generated dataset, even as the LLM scene pushes towards LLM-agents [28, 33, 45].

11 Conclusion

In this technical report, we presented "1.5 Pints", a Large Language Model that significantly advances the efficiency of LLM training by emphasizing data quality over quantity. Our model was trained on a meticulously curated corpus of 57 billion tokens, thereby demonstrating that a smaller, high-quality dataset can surpass the performance of models trained on much larger datasets. This approach not only reduced the training time precipitously (from months to just days), but also minimized the computational resources required, thus making pre-training more accessible and environmentally-friendly.

11.1 Key Performance Attribution

We primarily attribute the performance of 1.5-Pints to the following:

1. Careful selection of high-quality data forms a solid foundation for the LLM to perform well despite being trained on much lesser data.
2. A meticulously crafted model architecture that leverages the best of SOTA practices distilled from Mistral, StableLM, and TinyLlama.

11.2 Open-Source

By open-sourcing our findings and resources, we make our experiment verifiable and also aim to support the research community in the development of more efficient and powerful LLMs. This work underscores the critical role of data quality in LLM training and sets a new benchmark for resource-efficient AI development. Moving forward, we anticipate that broader adoption of these principles will drive innovation and enhance accessibility in the field of artificial intelligence.

12 Acknowledgements

References

[1] Dennis Abts, Garrin Kimmell, Andrew Ling, John Kim, Matt Boyd, Andrew Bitar, Sahil Parmar, Ibrahim Ahmed, et al. A software-defined tensor streaming multiprocessor for large-scale machine learning. In *Proceedings of the 49th Annual International Symposium on Computer Architecture (ISCA '22)*, pages 567–580, June 2022.

[2] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4895–4901, Singapore, December 2023. Association for Computational Linguistics.

[3] Ameen Ali, Itamar Zimerman, and Lior Wolf. The hidden attention of mamba models. *CoRR*, abs/2403.01590, 2024.

[4] arXiv.org submitters. arxiv dataset. `https://www.kaggle.com/dsv/7548853`, 2024. Accessed: 2024-02-15.

[5] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Self-reflective retrieval augmented generation. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*, 2023.

[6] Gregor Bachmann, Sotiris Anagnostidis, and Thomas Hofmann. Scaling mlps: A tale of inductive bias. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 60821–60840. Curran Associates, Inc., 2023.

[7] Marco Bellagente, Jonathan Tow, Dakota Mahan, Duy Phung, Maksym Zhuravinskyi, Reshinth Adithyan, James Baicoianu, Ben Brooks, Nathan Cooper, Ashish Datta, Meng Lee, Emad Mostaque, Michael Pieler, Nikhil Pinnaparju, Paulo Rocha, Harry Saini, Hannah Teufel, Niccolo Zanichelli, and Carlos Riquelme. Stable lm 2 1.6b technical report, 2024.

[8] Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew Gormley. Unlimiformer: Long-range transformers with unlimited length input. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 35522–35543. Curran Associates, Inc., 2023.

[9] Adam Bouyamourn. Why LLMs hallucinate, and how to get (evidential) closure: Perceptual, intensional, and extensional learning for faithful natural language generation. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

[10] Qingqing Cao, Sewon Min, Yizhong Wang, and Hannaneh Hajishirzi. BTR: Binary token representations for efficient retrieval augmented language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[11] Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research*, 2023.

[12] Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with high-quality feedback. https://openreview.net/forum?id=pNkOx3IVWI, 2024.

[13] Michal Danilk. Langdetect: Detect the natural language of a text. `https://pypi.org/project/langdetect`. Accessed: 12-March-2024.

[14] Tri Dao. FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning, 2023.

[15] Dhruvil Dave. Wikibooks dataset. `https://www.kaggle.com/ds/1167113`, 2021. Accessed: 2024-02-15.

[16] Manuel Faysse. Project Gutenberg dataset. `https://huggingface.co/datasets/manu/project_gutenberg`, 2023. Accessed: 2024-02-15.

[17] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey, 2024.

[18] Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio Cesar Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Conti Kauffmann, Gustavo Henrique de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Behl, Xin Wang, Sebastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. Textbooks are all you need. https://openreview.net/forum?id=Fq8tKtjACC, 2024.

[19] Tae Jun Ham, Sung Jun Jung, Seonghak Kim, Young H. Oh, Yeonhong Park, Yoonho Song, Jung-Hun Park, Sanghee Lee, Kyoung Park, Jae W. Lee, and Deog-Kyoon Jeong. A^3: Accelerating attention mechanisms in neural networks with approximation. In *2020 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 328–341, 2020.

[20] Shibo Hao, Tianyang Liu, Zhen Wang, and Zhiting Hu. ToolkenGPT: Augmenting frozen language models with massive tools via tool embeddings. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[21] Travis Hoppe. PhilArchive. `https://github.com/thoppe/The-Pile-PhilPapers`. Accessed: 2024-02-15.

[22] Liang Hu, Jiangcheng Zhu, Zirui Zhou, Ruiqing Cheng, Xiaolong Bai, and Yong Zhang. An optimal resource allocator of elastic training for deep learning jobs on cloud. *CoRR*, abs/2109.03389, 2021.

[23] Weizhe Hua, Zihang Dai, Hanxiao Liu, and Quoc Le. Transformer quality in linear time. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 9099–9117. PMLR, 17–23 Jul 2022.

[24] Yuzhen Huang, Jinghan Zhang, Zifei Shan, and Junxian He. Compression represents intelligence linearly, 2024. Accessed: 28-May-2024.

[25] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12), mar 2023.

[26] Haoqiang Kang, Juntong Ni, and Huaxiu Yao. Ever: Mitigating hallucination in large language models through real-time verification and rectification, 2024.

[27] Andrej Karpathy. `https://x.com/karpathy/status/1621578354024677377?lang=en`. Accessed 28-05-2024.

[28] Byoungjip Kim, Youngsoo Jang, Lajanugen Logeswaran, Geon-Hyeong Kim, Yu Jin Kim, Honglak Lee, and Moontae Lee. Prospector: Improving LLM agents with self-asking and trajectory ranking. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023.

[29] 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.

[30] Dong-Ho Lee, Jay Pujara, Mohit Sewak, Ryen W White, and Sujay Kumar Jauhar. Making large language models better data creators. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

[31] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[32] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc., 2020.

[33] Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for "mind" exploration of large language model society. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 51991–52008. Curran Associates, Inc., 2023.

[34] Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. Textbooks are all you need ii: phi-1.5 technical report, 2023.

[35] Yuanzhi Li and Ronen Eldan. Tinystories: How small can language models be and still speak coherent english. https://openreview.net/forum?id=yiPtWSrBrN, 2024.

[36] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel J. Orr, Lucia Zheng, Mert Yüksekgönül, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri S. Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. Holistic evaluation of language models. *CoRR*, abs/2211.09110, 2022.

[37] YINGWEI MA, Yue Liu, Yue Yu, Yuanliang Zhang, Yu Jiang, Changjian Wang, and Shanshan Li. At which training stage does code data help LLMs reasoning? In *The Twelfth International Conference on Learning Representations*, 2024.

[38] Sebastian Majstorovic. US-PD-Books Dataset. https://huggingface.co/datasets/storytracer/US-PD-Books, 2024. Accessed: 2024-02-15.

[39] Max Marion, Ahmet Ustun, Luiza Pozzobon, Alex Wang, Marzieh Fadaee, and Sara Hooker. When less is more: Investigating data pruning for pretraining LLMs at scale. In *NeurIPS Workshop on Attributing Model Behavior at Scale*, 2023.

[40] Meta AI. Introducing meta llama 3: The most capable openly available llm to date. https://ai.meta.com/blog/meta-llama-3/, April 2024. Accessed: 28-May-2024.

[41] Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. Orca: Progressive learning from complex explanation traces of gpt-4, 2023.

[42] Donald Della Pietra Owen Colegrove, Chunqiu Steven Xia. SciPhi Textbooks Are All You Need Lite. https://huggingface.co/datasets/SciPhi/textbooks-are-all-you-need-lite, 2024. Accessed: 2024-02-17.

[43] Nam Pham. Tiny Orca Textbooks. https://huggingface.co/datasets/nampdn-ai/tiny-orca-textbooks, 2024. Accessed: 2024-02-17.

[44] Nam Pham. Tiny Textbooks. https://huggingface.co/datasets/nampdn-ai/tiny-textbooks, 2024. Accessed: 2024-02-17.

[45] Bharat Prakash, Tim Oates, and Tinoosh Mohsenin. LLM augmented hierarchical agents. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023.

[46] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[47] Hongyu Ren, Hanjun Dai, Zihang Dai, Mengjiao Yang, Jure Leskovec, Dale Schuurmans, and Bo Dai. Combiner: Full attention transformer with sparse computation cost. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[48] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36, 2024.

[49] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *Advances in Neural Information Processing Systems*, 36, 2024.

[50] Peng Shi, Rui Zhang, He Bai, and Jimmy Lin. XRICL: Cross-lingual retrieval-augmented in-context learning for cross-lingual text-to-SQL semantic parsing. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5248–5259, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[51] Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen tau Yih. Replug: Retrieval-augmented black-box language models, 2023.

[52] Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. Retrieval augmentation reduces hallucination in conversation, 2021.

[53] Karthik Soman, Peter W Rose, John H Morris, Rabia E Akbas, Brett Smith, Braian Peetoom, Catalina Villouta-Reyes, Gabriel Cerono, Yongmei Shi, Angela Rizk-Jackson, Sharat Israni, Charlotte A Nelson, Sui Huang, and Sergio E Baranzini. Biomedical knowledge graph-optimized prompt generation for large language models, 2024.

[54] Yuan Sui, Yufei He, Nian Liu, Xiaoxin He, Kun Wang, and Bryan Hooi. Fidelis: Faithful reasoning in large language model for knowledge graph question answering, 2024.

[55] Technology Innovation Institute. falcon-refinedweb (revision 184df75). https://huggingface.co/datasets/tiiuae/falcon-refinedweb, 2023.

[56] Tensorflow. wiki40b/en. https://www.tensorflow.org/datasets/catalog/wiki40b, 2023. Accessed: 2024-02-15.

[57] 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, 2023. Accessed: 28-May-2024.

[58] Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Shengyi Huang, Kashif Rasul, Alexander M. Rush, and Thomas Wolf. The alignment handbook. https://github.com/huggingface/alignment-handbook, 2023.

[59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[60] Yu Wang, Nedim Lipka, Ryan Rossi, Alexa Siu, Ruiyi Zhang, and Tyler Derr. Knowledge graph prompting for multi-document question answering. In *NeurIPS 2023 Workshop: New Frontiers in Graph Learning*, 2023.

[61] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc., 2022.

[62] Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting influential data for targeted instruction tuning. In *ICLR 2024 Workshop on Navigating and Addressing Data Problems for Foundation Models*, 2024.

[63] Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations*, 2024.

[64] Rui Yang, Lin Song, Yanwei Li, Sijie Zhao, Yixiao Ge, Xiu Li, and Ying Shan. Gpt4tools: Teaching large language model to use tools via self-instruction. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 71995–72007. Curran Associates, Inc., 2023.

[65] Shuo Yang, Zeke Xie, Hanyu Peng, Min Xu, Mingming Sun, and Ping Li. Dataset pruning: Reducing training data by examining generalization influence. In *The Eleventh International Conference on Learning Representations*, 2023.

[66] Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. Tinyllama: An open-source small language model, 2024.

[67] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623. Curran Associates, Inc., 2023.

[68] Yutao Zhu, Zhaoheng Huang, Zhicheng Dou, and Ji-Rong Wen. One token can help! learning scalable and pluggable virtual tokens for retrieval-augmented large language models, 2024.

13   Appendix

A   Manual Dataset Selection Procedure via Central Limit Theorem and Population Sampling

To ensure that we select the best datasets for pre-training, we sought to manually inspect the examples and select the dataset which scored the highest. However, with a huge corpus size, it is not feasible to manually inspect the entire dataset. Thus, we sampled 385 examples from the corpora, the number of which was derived by the following Population Sampling formula:

$$n = \lceil \frac{Z^2 \times p \times (1 - p)}{e^2} \rceil$$

where n is the least sample size, Z is the Z-value (which we used 1.96 for a 95% confidence level), p is the estimated proportion of outcomes, which we used 50% as a default for unknown true proportions, and e is the margin of error, which we used 5%, resulting in:

$$n = \lceil \frac{1.96^2 \times 0.5 \times (1 - 0.5)}{0.05^2} \rceil = 385$$

B   Manual Dataset Selection Judging Criteria

For scoring the pre-training dataset, two human judges are employed to judge the the datasets based on the following criteria, and only scoring on the basis of yes or no to reduce subjectivity:

Table 12: Judging Criteria

Attribute	Points	Criteria
Expository	2	Whether the text explains or substantiates a concept, idea, or an opinion well. • Does the text provide clear and detailed explanations of concepts or ideas? • Does the text include examples or evidence to support the main points? • Is the information presented in a logical and organized manner? • Does the text define any specialized terms or jargon used? • Is the purpose of the text to inform, explain, or clarify rather than to persuade or entertain?
Toxic	-2	Whether the text contains information that can be considered profanity, sexually inappropriate, racism, discrimination, extremism, or similar. • If you read this text, would you find any language or words offensive? • Does this text make you uncomfortable due to sexually inappropriate content? • If this text was directed at you or your group, would you feel discriminated against or demeaned? • Does the text express or promote harmful or extremist ideas that you find alarming? • Do you think this text would be hurtful or harmful to others based on their race, gender, sexual orientation, religion, or ethnicity?
Clean	1	Whether the text contains irregular text sequences. • Are there a too many broken words (more than 1 in 10 words)? • Are there jumbled up text sequences (more than 1 in 5 paragraphs)? • Are there irrelevant symbols (more than 1 in 5 paragraphs)? • Are there excessive whitespaces (more than 1 in 10 words)?

For scoring the fine-tuning and alignment dataset, the "clean" attribute is omitted, as the handcrafted or synthetically generated samples are clean to being with.

C   Mistral Tokenizer vs Llama-2 Tokenizer

To investigate the tokenization efficiency between the Mistral tokenizer and the Llama-2 tokenizer, we randomly subsampled 10% of the `tiiuae/falcon-refinedweb`[55] dataset. Of note, both the Mistral and the Llama-2 tokenizers have the same vocabulary size of 32K, which makes this comparison a fair one.

Table 13: Tokenization

Tokenizer	Tokens	% Reduction
Llama-2	24,131,968,012	-
Mistral	23,261,356,142	3.61

We also found that the metadata within the Mistral `tokenizer.model` indicates it was trained with a corpus named `@/mnt/test/datasets/tokenizer_training/8T_train_data/shuffled.txt`, suggesting that the Mistral tokenizer was trained using approximately 8 trillion words or tokens. This contrasts with the Llama-2 tokenizer, which is derived from its predecessor, Llama-1 [57], and was likely trained on a significantly smaller corpus. This substantial difference in training data volume could account for the performance improvements observed in the Mistral tokenizer over Llama-2.

D   Direct Preference Optimization Epoch Study

We conducted 5 epochs of DPO training and evaluated all epochs on MT-Bench. For 2K context window, the best MT-Bench score was achieved by epoch 2 of the DPO step. We noted that further epochs beyond epoch 2 provided no improvement, and with a significant dip in epoch 4. For 16K context window, the epoch 4 achieved the best result, before dipping in performance at epoch 5. We used a repetition penalty of 1.3 for the evaluations.

Table 14: Epoch study for 2K Context Window DPO with MT-Bench

Epoch	Checkpoint Steps	MT-Bench Score
1	1,910	3.67
2	3,821	3.73
3	5,731	3.62
4	7,642	3.43
5	9,550	3.61

Table 15: Epoch study for 16K Context Window DPO with MT-Bench

Epoch	Checkpoint Steps	MT-Bench Score
1	1,910	3.12
2	3,821	3.10
3	5,731	3.16
4	7,642	3.40
5	9,550	3.30

E   Language Model Evaluation Harness

Similar to 1.5-Pints' performance on MT-Bench, it also performed well on LM-Eval, out-performing models such as EleutherAI's GPTNeo 1.3B model, as well as larger models such as StabilityAI's StableLM 3b fine-tuned model.

Table 16: Language Model Evaluation Results (Summary)

Model	Average	Pre-train Tokens	Parameter Size
microsoft/phi-2	61.09	1.4T	2.7B
TinyLlama/TinyLlama-1.1B-Chat-v1.0	37.28	3T	1.1B
EleutherAI/pythia-1.4b-deduped	35.00	0.3T	1.4B
facebook/opt-1.3b	34.60	0.18T	1.3B
1.5-Pints-16K	34.06	0.115T	1.57B
1.5-Pints-2K	34.02	0.115T	1.57B
EleutherAI/gpt-neo-1.3B	33.59	0.38T	1.3B
bigscience/bloom-1b	32.48	0.35T	1B
stabilityai/stablelm-tuned-alpha-3b	32.14	2T	3B
stabilityai/stablelm-base-alpha-3b	31.50	2T	3B
facebook/xglm-1.7B	31.42	0.5T	1.7B
bigcode/starcoderbase-3b	31.37	1T	3B
cerebras/Cerebras-GPT-1.3B	31.30	0.371T	1.3B
apple/OpenELM-1_1B-Instruct	49.94*	1.8T	1.1B

*Although Apple OpenELM did not report GSM8K results, we included the average of its known metrics at the bottom of the table for reference.

Table 17: Language Model Evaluation Results (Detailed)

Model	ARC	HellaSwag	MMLU	TruthfulQA	Winogrande	GSM8K
phi-2	61.01	74.92	57.92	44.24	73.48	54.97
TinyLlama-1.1B-Chat-v1.0	36.09	61.1	25.39	37.48	61.25	2.35
pythia-1.4b-deduped	32.68	54.96	25.56	38.66	57.3	0.83
opt-1.3b	29.52	54.53	24.96	38.71	59.75	0.15
1.5-Pints-16K	29.01	45.39	26.52	38.93	53.04	11.45
1.5-Pints-2K	30.55	48.43	25.39	40.13	53.51	6.14
gpt-neo-1.3B	31.23	48.47	24.82	39.63	56.91	0.45
bloom-1b	28.33	42.78	26.7	41.8	55.01	0.23
stablelm-tuned-alpha-3b	27.82	44.06	23.08	42.33	55.01	0.53
stablelm-base-alpha-3b	26.45	42.24	25.43	40.5	53.91	0.45
facebook xglm-1.7B	25.85	45.68	25.1	37.21	53.91	0.76
starcoderbase-3b	25.85	39.11	27.35	43.05	51.14	1.74
Cerebras-GPT-1.3B	26.28	38.54	26.59	42.7	53.43	0.23
OpenELM-1_1B-Instruct	41.55	71.83	25.65	45.95	64.72	-

F   Tracking the effects of each stage of training

The effects of each stage of training is summarized in Figure 3:

G   Legal Warning

Our model and training code are open-sourced under MIT license.

Though best efforts has been made to ensure, as much as possible, that all texts in the training corpora are royalty free, this does not constitute a legal guarantee that such is the case. **By using any of the models, corpora or part thereof, the user agrees to bear full responsibility to do the necessary due diligence to ensure that he/she is in compliance with their local copyright laws**.

In addition, the **user agrees to bear any damages** arising as a direct cause (or otherwise) of using any artifacts released by the pints research team, and full responsibility for the consequences of his/her usage (or implementation) of any such released artifacts. The user also indemnifies Pints

Figure 3: Impact of training modality on performance

Research Team (and any of its members or agents) of any damage, related or unrelated, to the release or subsequent usage of any findings, artifacts or code by the team.

For the avoidance of doubt, **any artifacts released by the Pints Research team are done so in accordance with the "fair use"** clause of Copyright Law, in hopes that this will aid the research community in bringing LLMs to the next frontier.