Phi-4 Technical Report

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat Lee, Yuanzhi Li, Weishung Liu, Caio C. T. Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Xin Wang, Rachel Ward, Yue Wu, Dingli Yu, Cyril Zhang, Yi Zhang

Microsoft Research

Abstract

We present **phi-4**, a 14-billion parameter language model developed with a training recipe that is centrally focused on data quality. Unlike most language models, where pre-training is based primarily on organic data sources such as web content or code, phi-4 strategically incorporates synthetic data throughout the training process. While previous models in the Phi family largely *distill* the capabilities of a teacher model (specifically GPT-4), phi-4 substantially *surpasses* its teacher model on STEM-focused QA capabilities, giving evidence that our data-generation and post-training techniques go beyond distillation. Despite minimal changes to the phi-3 architecture, phi-4 achieves strong performance relative to its size – especially on reasoning-focused benchmarks – due to improved data, training curriculum, and innovations in the post-training scheme.

1 Introduction

Recent advancements in Large Language Models (LLMs) have shown that significant improvements in data quality can rival, and sometimes surpass, the performance gains traditionally achieved by scaling compute with model and dataset size. Building on the success of the Phi family [GZA+23, LBE+23, JBA+23, AAA+24], we introduce phi-4, a 14-billion parameter model that further advances performance of small language models by introducing innovative synthetic data generation methods for reasoning-focused tasks, by optimizing the training curriculum and data mixture, and by introducing new techniques in post-training.

Synthetic data constitutes the bulk of the training data for phi-4 and is generated using a diverse array of techniques, including multi-agent prompting, self-revision workflows, and instruction reversal. These methods enable the construction of datasets that induce stronger reasoning and problem-solving abilities in the model, addressing some of the weaknesses in traditional unsupervised datasets. Synthetic data in phi-4 also plays a crucial role in post-training, where techniques such as rejection sampling and a novel approach to Direct Preference Optimization (DPO) are employed to refine the model's outputs.

The development of phi-4 is guided by three core pillars:

1. **Synthetic Data for Pretraining and Midtraining:** High-quality synthetic datasets are designed to prioritize *reasoning* and *problem-solving*, carefully generated to ensure diversity and

		Small models				Large models		
		phi-4 14b	**phi-3** 14b	**Qwen 2.5** 14b instruct	**GPT** 4o-mini	**Llama-3.3** 70b instruct	**Qwen 2.5** 72b instruct	**GPT** 4o
simple-evals	MMLU	84.8	77.9	79.9	81.8	86.3	85.3	**88.1**
	GPQA	**56.1**	31.2	42.9	40.9	49.1	49.0	50.6
	MATH	**80.4**	44.6	75.6	73.0	66.3[1]	80.0	74.6
	HumanEval	82.6	67.8	72.1	86.2	78.9[1]	80.4	**90.6**
	MGSM	80.6	53.5	79.6	86.5	89.1	87.3	**90.4**
	SimpleQA	3.0	7.6	5.4	9.9	20.9	10.2	**39.4**
	DROP	75.5	68.3	85.5	79.3	**90.2**	76.7	80.9
	MMLUPro	70.4	51.3	63.2	63.4	64.4	69.6	**73.0**
	HumanEval+	82.8	69.2	79.1	82.0	77.9	78.4	**88.0**
	ArenaHard	75.4	45.8	70.2	76.2	65.5	**78.4**	75.6
	LiveBench	47.6	28.1	46.6	48.1	**57.6**	55.3	**57.6**
	IFEval	63.0	57.9	78.7	80.0	**89.3**	85.0	84.8
	PhiBench (internal)	56.2	43.9	49.8	58.7	57.1	64.6	**72.4**

Table 1: Performance of phi-4 on a set of standard benchmarks. The first set of benchmarks uses OpenAI's SIMPLE-EVALS framework [Ope24b], specifying the prompts/extraction/temperature=0.5. We compare to small models of similar inference cost, as well as to larger models.

relevance. We change our training curriculum and create new pretraining and midtraining data mixtures to increase the allocation of synthetic tokens, compared to older generations of phi.

2. **Curation and Filtering of High-Quality Organic Data:** We meticulously curate and filter organic[2] data sources, including web content, licensed books, and code repositories to extract seeds for the synthetic data pipeline that encourage high-depth reasoning and prioritize educational value (to the model). These seeds form the foundation of the synthetic generation pipeline. To complement these synthetic datasets, we also filter the web for high-quality data (in terms of knowledge and reasoning) to use directly in pretraining.

3. **Post-Training:** We further advance the post-training recipe in phi-4 by creating new refined versions of SFT datasets, as well as by developing a new technique to create DPO pairs, based on *pivotal token search*.

With these innovations, the performance of phi-4 on reasoning-related tasks is comparable to or surpasses much larger models. For example, its performance on many widely used reasoning-related benchmarks meets or exceeds that of Llama-3.1-405B. In Table 1 we compare the performance of our model on academic benchmarks to several contemporary foundation models. We find that phi-4 significantly exceeds its teacher GPT-4o on the GPQA (graduate-level STEM Q&A) and MATH (math competition) benchmarks.

[1]These scores are lower than those reported by Meta, perhaps because SIMPLE-EVALS has a strict formatting requirement that Llama models have particular trouble following. We use the SIMPLE-EVALS framework because it is reproducible, but Meta reports 77 for MATH and 88 for HumanEval on Llama-3.3.

[2]We use *organic* to refer to human-generated or otherwise non-synthetic data.

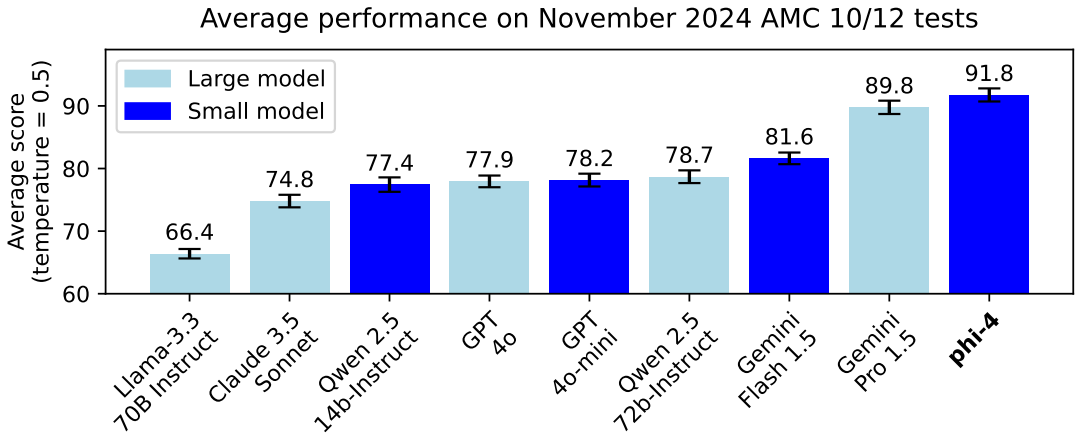

Figure 1: Average performance of different models on the November 2024 AMC-10 and AMC-12 tests. This is the average score (with maximum score 150) over the four tests on 100 runs with temperature $t = 0.5$. We chose $t = 0.5$ to follow SIMPLE-EVALS [Ope24b]. Error bars are 2σ of the estimate. On competition math, phi-4 scores well above its weight-class even compared to non–open-weight models.

1.1 Addressing Overfitting and Data Contamination

Decontamination: One pitfall of foundation models is overfitting to benchmarks, such as through the leakage of benchmark test sets via the web corpus. We improved the data decontamination process for phi-4 compared to previous Phi models to ensure no unfair influence on evaluation results. More details of the decontamination method are given in Appendix B.

AMC Benchmark: The surest way to guard against overfitting to the test set is to test on fresh data. We tested our model on the November 2024 AMC-10 and AMC-12 math competitions [Com24], which occurred after all our training data was collected, and we only measured our performance after choosing all the hyperparameters in training our final model. These contests are the entry points to the Math Olympiad track in the United States and over 150,000 students take the tests each year.

In Figure 1 we plot the average score over the four versions of the test, all of which have a maximum score of 150. phi-4 outperforms not only similar-size or open-weight models but also much larger frontier models. Such strong performance on a fresh test set suggests that phi-4's top-tier performance on the MATH benchmark is not due to overfitting or contamination. We provide further details in Appendix C.

Relying on Contamination-Proof Benchmarks: We give significant weight to benchmarks which were designed in such a way that the questions are original and do not appear on the web, such as GPQA [RHS+23]. While optimizing our model, we relied on an internal benchmark composed primarily of original prompts written by the team (see Section 5 for further details).

Long Chain-of-Thought Models: A style of LLM that scales inference-time compute by generating long chains of thought has emerged over the past few months, as pioneered by OpenAI O1 [Ope24a] and followed by DeepSeek-R1-Lite-Preview [Dee24] and Qwen/QwQ-32B-Preview [Tea24]. These models perform well on reasoning benchmarks, where QwQ, the only such model with open weights, averages 124.5 points in the AMC-10/12 setting of Figure 1. However, QwQ also uses 4X more tokens on this task than phi-4 and has more than twice as many parameters. Thus, the inference cost of QwQ is an

order of magnitude higher than phi-4. Consequently, these models are not in the same class as phi-4 with respect to cost or latency.

2 Approach to Data

The pretraining phase of phi-4 relies heavily on synthetic datasets generated through a variety of techniques. In addition, we employ several methods for filtering organic data sources that are used both as complementary datasets in the pretraining and as seeds for generating synthetic data.

2.1 Purpose of Synthetic Data

Synthetic data as a substantial component of pretraining is becoming increasingly common, and the Phi series of models has consistently emphasized the importance of synthetic data. Rather than serving as a cheap substitute for organic data, synthetic data has several direct advantages over organic data.

Structured and Gradual Learning. In organic datasets, the relationship between tokens is often complex and indirect. Many reasoning steps may be required to connect the current token to the next, making it challenging for the model to learn effectively from next-token prediction. By contrast, each token generated by a language model is by definition predicted by the preceding tokens, making it easier for a model to follow the resulting reasoning patterns. In this way, synthetic data may act as a form of "spoonfeeding," presenting challenges in a digestible and progression-oriented manner.

A simple example to illustrate this is that a human-written solution to a math problem might start with the final answer. This answer is much too hard to output immediately, for either a human or an LLM—the human produced it by nonlinear editing, but pretraining expects the LLM to learn to produce it linearly. Synthetic solutions to math problems will not have such roadblocks.

Alignment with Inference Contexts. Synthetic data is typically closer to the format of outputs we expect our models to generate. Training on such data helps align the model's pretraining experience with the scenarios it encounters during inference. This alignment ensures that the context seen during generation remains in-distribution with respect to the data the model was pretrained on.

For example, web forums are very different in style from LLM interactions. If a fact only appears in web forum data, the pretrained model will think it is very unlikely to occur in the chats it produces. Rewriting facts from the web forum into the language style of an LLM makes the facts more accessible during the LLM chat context of inference.

Principles. Our approach to generating synthetic data for phi-4 is guided by the following principles:

1. **Diversity:** The data should comprehensively cover subtopics and skills within each domain. This requires curating diverse seeds from organic sources.

2. **Nuance and Complexity:** Effective training requires nuanced, non-trivial examples that reflect the complexity and the richness of the domain. Data must go beyond basics to include edge cases and advanced examples.

3. **Accuracy:** Code should execute correctly, proofs should be valid, and explanations should adhere to established knowledge, etc.

4. **Chain-of-Thought:** Data should encourage systematic reasoning, teaching the model various approaches to the problems in a step-by-step manner. This fosters coherent outputs for complex tasks.

2.2 Synthetic Data for Pretraining and Midtraining

We created 50 broad types of synthetic datasets, each one relying on a different set of seeds and different multi-stage prompting procedure, spanning an array of topics, skills, and natures of interaction, accumulating to a total of about 400B unweighted tokens. In Appendix D, we give a few examples of transcripts taken from our synthetic generations. Here, we highlight novel methodologies used in generating synthetic datasets for phi-4:

- **Seed Curation:** The synthetic dataset generation begins with high-quality seeds sourced from multiple domains. These curated seeds provide the foundation for synthetic data generation, enabling the creation of exercises, discussions, and reasoning tasks tailored to the model's training objectives.

    1. **Web and Code-based Seeds:** Excerpts and snippets are extracted from web pages, books, and code repositories with a focus on content that demonstrates high complexity, reasoning depth, and educational value. To ensure quality, we employ a two-stage filtering process: first, identifying pages with strong educational potential, and second, segmenting the selected pages into passages, scoring each for its factual and reasoning content.
    2. **Question Datasets:** A large set of questions was collected from websites, forums, and Q&A platforms. These questions were then filtered using a plurality-based technique to balance difficulty. Specifically, we generated multiple independent answers for each question and applied majority voting to assess the consistency of responses. We discarded questions where all answers agreed (indicating the question was too easy) or where answers were entirely inconsistent (indicating the question was too difficult or ambiguous). This filtering process produces a dataset of questions that challenge the model's reasoning and problem-solving abilities while remaining approachable. The plurality answers were used in place of the ground truth in our rejection-sampling based generations.
    3. **Creating Question-Answer pairs from Diverse Sources:** Another technique we use for seed curation involves leveraging language models to extract question-answer pairs from organic sources such as books, scientific papers, and code. This approach does not rely on merely identifying explicit Q&A pairs within the text. Instead, it involves a pipeline designed to detect deduction chains or logical progressions in the text. The language model identifies key steps in reasoning or problem-solving processes and reformulates them into questions and corresponding answers. Our experiments show that, if done correctly, training on the resulting content can be far more effective (in terms of improvement on academic and internal benchmarks) than training on the original content.

- **Rewrite and Augment:** Seeds are transformed into synthetic data through multi-step prompting workflows. This includes rewriting most of the useful content in given passages into exercises, discussions, or structured reasoning tasks.

- **Self-revision:** The initial responses are then iteratively refined through a feedback loop where a model critiques and subsequently improves its own outputs, guided by the rubrics focused on reasoning and factual accuracy.

- **Instruction Reversal for Code and Other Tasks:** To enhance the model's ability to generate outputs from instructions, we used an instruction reversal technique. For example, we take existing code snippets from the code data corpus and use it to generate corresponding instructions that include the problem description or task prompt. The resulting synthetic data pairs were structured with the instruction appearing before the code. Only data with high fidelity between the original and regenerated code are retained, ensuring alignment between the instructions and the outputs. This method can be generalized to other targeted use cases.
- **Validation of Code and Other Scientific Data:** When appropriate, we incorporate tests for validating our reasoning-heavy synthetic datasets. The synthetic code data is validated through execution loops and tests. For scientific datasets, the questions are extracted from scientific materials using a method designed to ensure high relevance, groundedness, and difficulty balance.

2.3 Curation and Filtering of Web and Q&A Data

Q&A datasets. We collected tens-of-millions high-quality organic problems and solutions by reviewing public websites, relying on existing datasets, and acquiring external datasets. Our experience from previous models showed that question-answer data contributed significantly to various capabilities, such as mathematical reasoning and academic performance. Our ablation studies showed that organic questions are substantially more effective than synthetic questions. We used several ways to synthetically augment the dataset of organic questions to obtain a larger dataset. While these rewritten questions improved the model's capabilities, the gains were not as pronounced. A significant portion of the collected questions lacked accurate solutions. To address this, we replaced the answers with synthetically generated ones and used majority-voting to increase accuracy. All collected questions and solutions underwent a thorough decontamination process to ensure there is no overlap with test sets[3].

Targeting High-quality Web Data. We collected a wide variety of high-quality organic data sources for phi-4, prioritizing reasoning-dense and nuanced material (e.g., academic papers, educational forums, and programming tutorials). In addition to directly training on this text, we used various web sources as seeds for specialized synthetic data generation pipelines. We found clean and correct natural data to be absolutely crucial for seeding synthetic data: minor errors can result in severe quality degradations for derived synthetic documents. We therefore invested heavily in the *perfectionistic* curation of our web data. We discuss the main techniques and considerations below:

- **Targeted Acquisitions:** We included major repositories of reasoning-dense documents that are publicly permissible for use (e.g., arXiv, PubMed Central, GitHub) or explicitly licensed (e.g., licensed books) aiming for a level of comprehensiveness, recency, and cleanliness above the typical standard of externally available corpora.
- **Filtering Web Dumps:** To capture the long tail of information-rich web sources (e.g., forums, blogs, course material, domain-specific wikis), we took the approach of selecting a small fraction of highest-quality documents from bulk web dumps, using small (non-LLM) classifiers trained on $\sim 10^6$ LLM-generated annotations. This approach tends to over-index on STEM-related keywords, so we created a specialized pipeline to amplify high-quality non-STEM content (e.g., arts, history, travel, culture, and entertainment). These topic classifications were also obtained by distilling an

[3]This step is crucial to the reliability of some of the academic benchmarks: for instance, some test benchmark variants can be found on platforms like Hugging Face. Moreover, benchmarks such as MMLU are frequently compiled from web-sourced questions.

LLM annotator. Finally, we removed corrupted text and binary files by detecting outliers according to n-gram statistics and compression ratios.

- **Multilingual Data:** We incorporated multilingual datasets to ensure that our model could handle a wide range of languages, including German, Spanish, French, Portuguese, Italian, Hindi and Japanese. This involved sourcing and processing high-quality multilingual documents from CommonCrawl and Wikipedia. Our multilingual processing pipeline consists of a language identification model, based on `fastText` used to categorize documents into 176 languages, then uses the same classifiers for filtering web dumps to filter for quality. Note that the classifiers were trained on multilingual LLM-generated annotations.

- **Custom Extraction and Cleaning Pipelines:** To ensure sufficient cleanliness and uniformity between heterogeneous organic data sources, we needed a collection of customized heuristics and parsers. For each targeted data source, we built custom pipelines to ingest a variety of file formats (e.g., multi-file TeX source, ePub and other XML-like formats, Microsoft Word documents, and PDFs). For general web data, we built a custom HTML-to-text extractor, taking significant care to preserve fragile content that is frequently corrupted by naïve parsers (e.g., TeX/MathML equations, code blocks, tables, and forum thread structure). This extractor prunes and normalizes the DOM tree, using a variety of signals (e.g., HTML tag names, CSS classes, content length, and tree depth) to distinguish elements such as boilerplate, advertisements, equations, and syntax-highlighter artifacts.

2.4 Post-Training datasets

Our post-training data is composed of:

- **Supervised Fine-Tuning (SFT) Datasets:** Using carefully curated user prompts taken from a mixture of publicly available datasets and synthetically generated data, we generate multiple model responses and select the best using an LLM-based evaluation process.

- **Direct Preference Optimization (DPO):** We generate DPO pairs based on rejection sampling and LLM evaluation, a part of which is based on our approach to creating pivotal token-based pairs, explained in Section 4.3 below.

3 Pretraining details

The phi-4 model is based on a decoder-only transformer architecture [VSP+17] with 14B parameters and a default context length of 4096. This is later extended to a 16K context length during midtraining. The architecture closely follows phi-3-medium, except that we now use the `tiktoken` tokenizer (for better multilingual support) with a padded vocabulary size of 100,352 (including unused tokens) and we use full attention over the 4K context length, rather than a 2K sliding window used in phi-3-medium.

The model was pretrained for approximately 10T tokens using linear warm-up and decay schedules with peak learning rate of 0.0003, constant weight decay of 0.1, and global batch size of 5760. The training hyperparameters are tuned using interpolations from shorter horizon runs and further adjusted by stress testing the learning rate warm-up stage for stability. Pretraining is followed by a shorter midtraining stage to increase the original context length of 4k to 16k.

Since pre-trained models are not good at instruction following, it is not very informative to use 0-shot evaluations that require the answer to be in a specific format, for example SIMPLE-EVALS. We

	MMLU	MMLU pro	GSM8k	Human-Eval	ARCC	MBPP	MATH	TQA
phi-4 (4k)	+3.0	+10.3	+2.2	+7.8	+1.1	+6.8	+8.9	-0.7
phi-4 (16k)	+2.7	+8.9	+1.2	+9.0	+0.9	+9.6	+8.4	-1.5

Table 2: Pretraining benchmarks for phi-4 compared to its predecessor, phi-3-medium after pretraining.

therefore use an internal implementation of benchmarks for pretraining which uses a mixture of log-likelihood and/or few-shot prompts for various tasks. Specifically, we used log-likelihood evaluations for MMLU (5-shot), MMLU-pro, and ARCC (1-shot). We used 1, 3, 4, and 8 few-shot examples for TriviaQA (TQA), MBPP, MATH, and GSM8k to help the model adhere to the answer format for easier extraction of the solution. We use this evaluation method throughout Section 3. Table 2 summarizes the performance boost of pretrained phi-4 compared with its predecessor phi-3-medium.

3.1 Data Composition in Pretraining

The phi-3 model family were trained using a two-phase strategy. Most of the training tokens were used in phase 1 of the training, which consisted largely of filtered web data. Phase 2 was trained with a data mixture consisting primarily of synthetic tokens and a much smaller allocation for ultra-filtered and reasoning-heavy web data. As the size and complexity of our synthetic data grew, we observed a marginal drop in the benefit from using non-synthetic tokens for the phi-3 family of model sizes. We note two key observations.

- Web datasets showed small benefits on reasoning heavy benchmarks. Prioritizing more epochs over our synthetic data led to better performance with respect to adding fresh web tokens.
- Models trained only with synthetic data underperformed on the knowledge-heavy benchmarks and demonstrated increased hallucinations.

Figure 2 demonstrates the first phenomenon using smaller scale phase 2 pretraining exercises. In this example, we conduct two training runs per model scale, using the same number of training tokens on top of phase 1 pretrained checkpoints. For all runs, the number of unique synthetic tokens is fixed (a subsample of full synthetic data) but the number of repetitions on this data changes, namely 4 and 12 epochs. The rest of the training tokens are fresh unique tokens supplied from web sources. As seen, performing more iterations on the synthetic data is more beneficial than supplying more web tokens.

Inspired by this scaling behavior of our synthetic data, we trained a 13B parameter model solely on synthetic[4] data, for ablation purposes only – the model sees over 20 repetitions of each data source. For the sake of ablations, we partitioned our synthetic data into *web rewrites*, which includes more direct rewrites of our filtered web content relative to all other types of synthetic data. Table 3 compares the previous phi-3-medium model with the new model trained entirely on the synthetic data. Throughout training, all benchmarks consistently improved, despite the increase in epochs, and the majority of the benchmarks showed improvements over phi-3. However, knowledge-related benchmarks, like 1-shot triviaqa (TQA), show a large gap where synthetic models are subpar. These observations led us to rethink the role of web data in our data mixture.

[4]This is an updated mixture of synthetic data that contains new sources compared to phi-3.

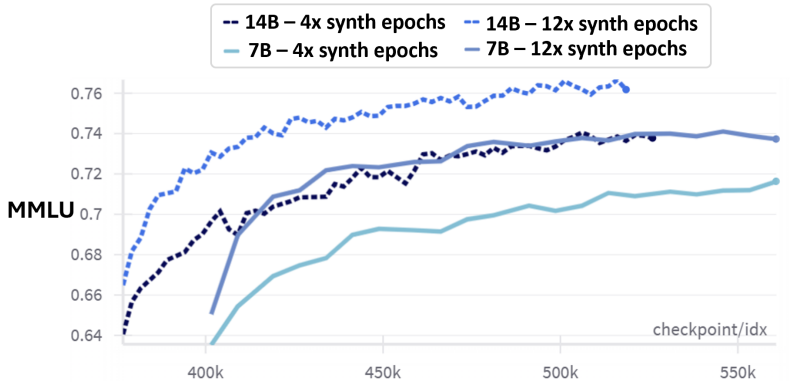

Figure 2: 5-shot MMLU score for phase 2 pretraining runs with 4 and 12 epochs of synthetic data. All models are trained for the same token horizon, thus the model with 4 epochs of synthetic has seen more (unique) web tokens. We see that despite many epochs on synthetic data, we do not see overfitting behavior and in fact the 12 epoch models perform better than those that have seen more unique web tokens.

	MMLU	MMLU pro	GSM8k	Human-Eval	ARCC	MBPP	MATH	TQA
Synthetic	+0.8	+4.0	+2.2	+12.1	0.0	+5.0	+4.9	-14.8
Synthetic + Web Rewrites	+0.3	+4.1	+1.8	+13.3	+3.0	+7.6	+8.1	-7.7

Table 3: Benchmark performance of 13B models (used for ablations only) trained on data mixtures containing no web data. The respective training tokens are either from synthetic sources, or an equal share of synthetic data and web rewrites. All numbers are reported relative to the performance of phi-3-medium, which has seen a combination of web and synthetic data.

3.2 Data Mixture

To design our pretraining data mixture for a given training token budget, we search over different allocation of tokens coming from various sources, namely, 1) synthetic, 2) web rewrites[5], 3) filtered web (divided into reasoning and knowledge-heavy portions), 4) targeted acquisitions and organic data (e.g., academic data, books, and forums), and 5) code data.

We conducted ablations using a shorter token horizon of 1T tokens to derive the data mixture. These ablations rely on our established result on the high-rank correlation of short training with longer training, up to the over-fitting saturation threshold of data sources. In addition we observe a high rank correlation between the performance of the 7B and 14B models on different data mixtures, given a large enough distance between the data mixtures. This allowed us to conduct the experiments at 7B scale and transfer the findings to phi-4. Among the numerous ablations, we highlight a few that show best insights on our data composition. Specifically, we freeze the ratio of tokens coming from targeted acquisitions and code categories, and change the ratio of tokens for the synthetic, web, and web rewrites clusters.

Table 4 summarizes the results for the hand-picked ablations, as compared with the data mixture that was used for the final training run. A uniform allocation of tokens among the three categories is suboptimal due to the higher quality of synthetic data and the only benchmark that shows a clear benefit from web data is TQA. While the synthetic-heavy variations on rows 2 and 3 of the table are marginally better than the chosen final data mixture, we decided to integrate the targeted and knowledge-heavy filtered web data sources to improve knowledge benchmarks (see Section 3.1) to balance all model

[5]Web rewrites is a sub-category of synthetic data that is substantially large and contains direct rewrites of web content.

	MMLU	MATH	GSM8k	Human-Eval	ARCC	MBPP	TQA	MMLU pro	Average
Uniform	-3.3	-5.4	-5.8	-1.2	+0.6	-2.0	+3.3	-3.6	-2.2
S	+3.3	+4.0	+2.1	-6.1	+1.9	+0.4	-3.0	+3.7	+0.8
S + WR	+0.6	+1.2	+1.5	-1.2	+1.6	+1.6	-3.7	+1.2	+0.4
S + W	-0.6	-0.7	-0.7	-4.3	+0.3	-2.0	+6.9	+0.9	0.0

Table 4: Ablations on the allocation of 75% of training tokens to synthetic (S), filtered web (W), and web rewrite (WR) categories, while other data sources are held constant in the remaining 25% token budget. All benchmark numbers are measured relative to the final data mixture used for training phi-4.

capabilities. We also note that we observed the gap between the chosen data mixture and the synthetic heavy runs largely closes as the model goes through the post-training stage. An end-to-end optimization of pretraining data mixture that also takes into account the effects of post-training is an interesting future area of investigation.

Data Source	Fraction of Training	Unique Token Count	Number of Epochs
Web	15%	1.3T	1.2
Web rewrites	15%	290B	5.2
Synthetic	40%	290B	13.8
Code data	20%	820B	2.4
Acquired sources	10%	580B	1.7

Table 5: Data mixture for pretraining.

The final data mixture used for phi-4 allocates 30% of the training tokens to web and web rewrites data sources, divided equally between them. The remaining tokens are largely sourced from synthetic data which accounts for 40% of the data mixture tokens. Finally we allocate 20% of tokens to code data (mixture of synthetic and raw code) and 10% to targeted acquired sources like academic data and books. In terms of total number of unique tokens in each data mixture cluster, filtered web data is the largest cluster with ~ 1.3T tokens. Code and targeted acquisitions are the second and third largest clusters with ~ 820B and ~ 580B tokens, respectively. Finally, web rewrites and synthetic data have similar token count of ~ 290B tokens. The total number of epochs on each data source is determined using the ratio of allocated tokens in the mixture and the number of unique tokens in that source.

3.3 Midtraining Details

phi-4 includes a midtraining stage where the context length is increased from the original 4K to 16K. We conduct several ablations to study the role of data on long-context performance. Specifically, we try data sources that are inherently long context, and compare them with artificially created long context data where samples are padded together to fill the sequence. We observe the former to perform better in longer context tasks.

Inspired by this, we further filter our high-quality non-synthetic datasets (i.e., academic, books, and code data) to separate samples above 8K context. We then up-weight the data subsets that are 16K or higher in length. We also create new synthetic datasets that satisfy the > 4K sequence requirement. The final data mixture includes 30% of the newly curated longer context data and a 70% portion of recall

Model	Max Length	Recall	RAG	ICL	Re-rank	QA	Summ
phi-4	8K	100.0	58.1	68.0	65.3	26.7	38.3
Qwen-2.5-14B	8K	100.0	62.2	67.8	58.2	24.7	37.2
Llama-3.3-70B	8K	92.0	65.3	69.4	64.4	30.0	37.8
GPT-4o-mini	8K	99.2	65.8	74.4	69.4	31.3	38.5
GPT-4o	8K	100.0	66.9	83.0	75.1	37.3	43.0
phi-4	16K	99.0	57.1	77.0	54.4	36.0	40.5
Qwen-2.5-14B	16K	100.0	59.1	67.6	50.3	29.7	42.3
Llama-3.3-70B	16K	92.0	62.2	70.0	63.3	36.7	41.9
GPT-4o-mini	16K	100.0	63.6	78.4	63.9	36.0	45.2
GPT-4o	16K	100.0	66.7	85.6	73.8	43.7	46.3

Table 6: Evaluation results on the long-context benchmark HELMET [YGH+24].

tokens from the pretraining stage. To accommodate longer context, we increase the base frequency of rope position encoding to 250K following [AI23]. We drop the maximum learning rate by a factor of 10 compared to the pretraining stage and train for a total of 250B tokens.

To effectively evaluate the long-context capability of our model, it is essential to have a comprehensive evaluation framework with practical scenarios. While synthetic benchmarks like needle-in-a-haystack and RULER are preferred for their simplicity and control, our emphasis is on a diverse range of tasks that reflect real-world applications, such as reasoning across entire documents. We report the performance of phi-4 and other models on the tasks we selected from the HELMET [YGH+24] evaluation suite in Table 6 and outline our evaluation methods below. Note that results are average across 5 runs for each categories.

- Recall: The task involves retrieving the corresponding value from a randomly-generated long JSON file given a specific key (Metric: SubEM)
- RAG: Answer questions based on many retrieved and shuffled Wikipedia documents. The datasets used for this task are NaturalQuestions, HotpotQA, and PopQA. Final results are average of all datasets (Metric: SubEM)
- Re-rank: The task is to re-rank the top-10 documents given a query and many retrieved and shuffled documents. This uses the MSMARCO dataset (Metric: nDCG@10)
- ICL: The task involves many-shot in-context learning with datasets such as TREC coarse, TREC fine, Banking77, NLU and CLINC150. Final results are average of all datasets (Metric: F1)
- QA: Answer questions given a lengthy document. The dataset associated with this task is NarrativeQAv2 (Metric: GPT-4o scoring)
- Summ: The task involves summarizing a lengthy legal document, and the dataset used is MultiLexSum (Metric: GPT-4o scoring)

Dataset Name	Sample Count
unknown + safety data	3,000
generic multiple-choice Q&A	132,859
math data	76,552
python data	16,080
cpp, go, java, js, rust data	21,806

Table 7: Data Mixture for Pivotal Token DPO

Dataset Name	Sample Count
unknown + safety data	43,842
any vs any overall	266,000
any vs any accuracy	532,000

Table 8: Data Mixture for Judge Guided DPO

4 Post-Training

Post-training is aimed at transforming the pretrained language model into an AI assistant that users can safely interact with. We align the pretrained model with one round of SFT 4.1, one round of DPO [RSM+23] on data from our pivotal token search method (see Section 4.3), and one round of DPO on full length preference pairs. The model is chat finetuned using the standard `chatml` format, example usage template for two rounds of a conversation is as follows:

```
<|im_start|>system<|im_sep|>system_message<|im_end|>
<|im_start|>user<|im_sep|>prompt1<|im_end|><|im_start|>assistant<|im_sep|>response1<|im_end|>
<|im_start|>user<|im_sep|>prompt2<|im_end|><|im_start|>assistant<|im_sep|>
```

4.1 Supervised Fine-Tuning

In this phase, we fine-tune the pretrained model with a learning rate of 10^{-6} on a variety of data generated from high-quality data across diverse domains, including math, coding, reasoning, conversation, model identity, and safety. We also added multilingual data for 40 languages. We use around 8B tokens of data in this phase, all formatted in the `chatml` format.

4.2 Direct Preference Optimization

We use DPO [RSM+23] to align the model with human preferences, and also to steer the model away from unwanted behavior through pairs of desired and undesired outputs. DPO data covers chat format data, reasoning, and Responsible AI (RAI) data and improves the model in math, coding, reasoning, robustness, and safety. We do two rounds of DPO on the SFT model. We introduce a technique, Pivotal Token Search (PTS), to generate pairs for DPO for the first DPO round. Details of the data mixture for first round are provided in Table 7.

For the second round, which we call judge-guided DPO, we gather approximately 850k pairs of desired and undesired outputs. The prompts are sourced from various publicly available instruction tuning datasets and also include prompts related to safety and Responsible AI (RAI). Next, for each of these prompts, we generate responses from GPT-4o, GPT-4t and our model. From these responses, we create various combinations of DPO pairs and use GPT-4o as a judge to label positive or negative for a given pair. For a given pair of responses, each assistant response is given a score based on accuracy, style, and detail. We label the response with higher accuracy or overall (average of accuracy, style, and detail) score as the positive response. We provide the prompt we used in Appendix A. The data mixture for this round is provided in Table 8. Both stages also include a small amount of data for safety and mitigating hallucinations.

User: Suppose that the roots of $x^3+3x^2+4x-11=0$ are a, b, and c, and that the roots of $x^3+rx^2+sx+t=0$ are $a+b$, $b+c$, and $c+a$. Find t.

AI: To solve the problem, we need to deal with the roots of the two cubic equations given. First, consider the cubic polynomial \(x^3 + 3x^2 + 4x - 11 = 0 \), with roots \(a \), \(b \), and \(c \). By Vieta's formulas, we have: \[a + b + c = -3, \] \[ab + bc + ca = 4, \] \[abc = 11. \] Now, consider the other polynomial \(x^3 + rx^2 + sx + t = 0 \) with roots \(a+b \), \(b+c \), and \(c+a \). We need to find the value of \(t \), which corresponds to the negative ($0.42 \to 0.93$) of the product of the roots according to Vieta's formulas: \[(a ($0.95 \to 0.71$)+b)(b+c)(c+a). \] First, note the important identity: \[(a+b)(b+c)(c+a) = (a+b+c)(ab+bc+ca) - abc. \] Plugging in known values from our Vieta's results for \(a \), \(b \), and \(c \), we have: \[a + b + c = -3, \] \[ab + bc + ca = 4, \] \[abc = 11. \] Substitute these into the identity: \[(a+b)(b+c)(c+a) = (-3)(4) - 11 = -12 - 11 = -23. \] Thus, \(t = -(a+b)(b+c)(c+a) = 23 \). Therefore, the value of \(t \) is \(\boxed{23}\).

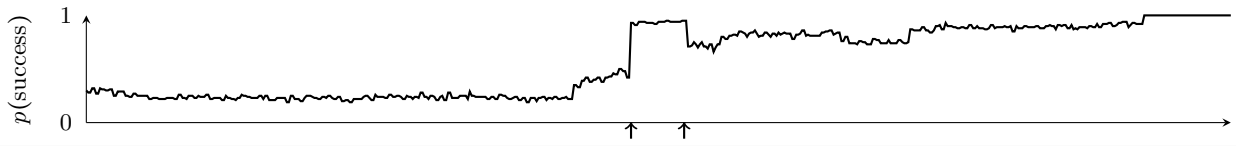

Figure 3: Illustration of pivotal tokens for GPT-4o at temperature 1 on a problem from the MATH benchmark [HBK+21], where the initial success probability is 0.31. Each token is colorized by the probability of success for an independent completion ($N = 529$) continued from after the token, with red for p(success) = 0 and blue for p(success) = 1. The line plot shows the same probabilities. The tokens that changes p(success) by ≥ 0.2 are shown boxed, with subscripts showing the change in probability. Tokens with probability ≤ 0.1 are underlined to illustrate that pivotal tokens are distinct from low-probability tokens. The token probabilities of negative and (a) were 0.31 and 0.12, respectively. The greedy tokens for the same prefixes are product with 0.66 probability and (t) with 0.88 probability.

procedure PIVOTALTOKENSEARCH($Q, T_{\text{full}}, p_{\text{gap}}$)
  procedure SUBDIVIDE(T_{prefix}, T)
    if $|T| \leq 1$ or $\left|p(\text{success} \mid T_{\text{prefix}}) - p(\text{success} \mid T_{\text{prefix}} + T)\right| < p_{\text{gap}}$ **then** ▷ *Base cases.*
      return $[T]$
    $T_{\text{left}}, T_{\text{right}} \leftarrow \text{SPLIT}(T)$ ▷ *We split at the cumulative midpoint of token log probabilities.*
    return $\text{SUBDIVIDE}(T_{\text{prefix}}, T_{\text{left}}) \cup \text{SUBDIVIDE}(T_{\text{prefix}} + T_{\text{left}}, T_{\text{right}})$
  $T_{\text{prefix}} \leftarrow \epsilon$
  for all $T \in \text{SUBDIVIDE}(\epsilon, T_{\text{full}})$ **do**
    if $|T| = 1$ and $\left|p(\text{success} \mid T_{\text{prefix}}) - p(\text{success} \mid T_{\text{prefix}} + T)\right| \geq p_{\text{gap}}$ **then**
      yield $(Q, T_{\text{prefix}}, T)$ ▷ *Output pivotal tokens T and context for postprocessing.*
    $T_{\text{prefix}} \leftarrow T_{\text{prefix}} + T$

Figure 4: Pseudocode for Pivotal Token Search (PTS). Note that estimating $p(\text{success} \mid \dots)$ involves sampling the language model and invoking the oracle. In an efficient implementation $p(\text{success} \mid \dots)$ should be memoized.

4.3 Pivotal Token Search

Consider a generative model producing a token-by-token response to a given prompt. For each token produced, which corresponds to a prefix of the model response, one can consider the conditional probability of the model's answer being correct given that prefix, as well as the increment in this probability with respect to that token (in other words, the difference in the probability of being correct before and after producing that token). It is often the case that the overall correctness is highly dependent on a successful generation of a small number of key tokens. For example, we can see in Figure 3 where the model outputs a math solution and a "fortunate" sampling of a crucial token ` negative` shifts the solution from possible failure to likely success, while sampling of the token `(a` subsequently risks failure again. We refer to these tokens as *pivotal tokens* as they have an outsized effect on the course of the solution.

Now, consider how the solution from Figure 3 would be used in DPO as a full-length accepted response. As the figure shows, there are many tokens with probabilities much lower than the 0.31 of ` negative`, which would contribute to noise in the gradients diluting the signal from the pivotal token. Even worse, the token `(a` that contributed to the lack of robustness would receive a strong *positive* learning signal thanks to its low probability of 0.12.

Moreover, intuition suggests that when two texts substantially deviate from each other, comparison of their individual next-token log probabilities (as done in DPO) is not very meaningful. Rather, it makes more sense that the signal should come from the first tokens after the two texts starts diverging from each other.

To alleviate these effects, we employ a method we call *Pivotal Token Search (PTS)* for generating preference data that specifically targets pivotal tokens in isolation, creating DPO pairs in which the preference optimization takes effect with respect to a single token.

PTS identifies points of a completion token sequence $T_{\text{full}} = t_1, t_2, \ldots$ for some user query Q where the next token t_i has a significant impact on the probability of success $p(\text{success} \mid t_1, \ldots, t_i)$. PTS estimates these probabilities by sampling completions starting from $Q + t_1, \ldots, t_i$, which are checked for correctness with an oracle[6] for Q. Figure 4 shows a basic instantiation of the algorithm. The procedure SUBDIVIDE recursively splits the sequence into segments $t_i, \ldots, t_j$ until the change in probability $|\, p(\text{success} \mid t_1, \ldots, t_{i-1}) - p(\text{success} \mid t_1, \ldots, t_j) \,|$ for each segment is below a threshold p_{gap} or the segment is just a single token. Tokens with a sharp change in success probability are kept as pivotal. We turn pivotal tokens into preference data by taking $Q + t_1, \ldots, t_{i-1}$ as the *query*, and single tokens t_{acc} and t_{rej} that increase/decrease $p(\text{success} \mid t_1, \ldots, t_{i-1}, t_{\text{acc/rej}})$ as the *accepted* and *rejected* completions, respectively.[7] The binary-search algorithm for PTS is not always guaranteed to find all pivotal tokens, but it only finds pivotal tokens and it finds all of them if the success probability is near-monotone over the course of the solution.

We used PTS to generate preference data for tasks where ground-truth is readily available, such as mathematics, various forms of question answering and coding. To improve sample efficiency, we filter the target questions to only include those with $0.2 \le p(\text{success}) \le 0.8$, as pivotal tokens are rare for tasks that are very easy or hard.

See Figure 5 for examples of preference data we generated using PTS. The math question answering example shows how pivotal tokens often are not actual mistakes, but choices that drive the model down a less advantageous path. Here, multiplying both sides by the denominators separately is equally valid to directly cross-multiplying, but for the model doing the latter here is more robust. By generating DPO data targeting such choices, we believe PTS helps phi-4 work better in the modes it is especially

[6] For coding, a comprehensive test suite can be used. For math problems, answers can be compared to the ground truth.

[7] We find drawing t_{acc} and t_{rej} from rollouts PTS already used to estimate $p(\text{success} \mid t_1, \ldots, t_{i-1})$ to be effective.

User: Find the product of all real values of r for which $\frac{1}{2x} = \frac{r-x}{7}$ has exactly one real solution.
AI: To solve the problem, we need to determine the values of r such that the equation

$$\frac{1}{2x} = \frac{r-x}{7}$$

has exactly one real solution for x.
First, clear the fractions by
Good: cross-multiplying **Bad:** multiplying both sides by

User: Consider a 1 keV helium atom impinging at a one-degree incidence angle on a LiF crystal surface. Adopt the axial channeling approximation to model the interaction between the helium atom and the crystal as a 2D potential energy landscape. Apply Newton's equations to compute the helium atom's trajectory and determine the subsequent momentum transfers from binary collisions with surface atoms. Sum the total virtual energy transferred across all collisions and compute the probability, P_e, that the helium atom remains elastically scattered assuming a vibrational frequency for surface atoms of $\omega = 1 \times 10^{14} Hz$. What is P_e?
1. $P_e = \exp(-\frac{E_{loss}}{2\hbar\omega})$, 2. $P_e = \exp(\frac{E_{loss}}{\hbar\omega})$, 3. $P_e = \exp(-\frac{E_{loss}}{\hbar\omega})$, or 4. $P_e = 1 - \exp(-\frac{E_{loss}}{\hbar\omega})$
AI: To determine the probability P_e that the helium atom remains elastically scattered, we need to calculate the total virtual energy transferred, E_{loss}, during the helium atom's interaction with the LiF crystal surface.
Given
Good: the vibrational frequency **Bad:** that the potential interactions

User: Read the following function signature and docstring, and fully implement the function described. Your response should only contain the code for this function.

```
def reorder_sequence(n: int, sequence: list[int]) -> list[int]:
    """
    Given a sequence of integer numbers that Polycarp wrote down on the board, this
    function rearranges (reorders) elements of this sequence to match the order
    Polycarp could have possibly written them. Each next number should be exactly two
    times the previous number or exactly one third of the previous number.

    :param n: The number of elements in the sequence (2 <= n <= 100).
    :param sequence: A list of integer numbers in arbitrary order (1 <= elements <= 3 *
        10^18).
    :return: A reordered list of integer numbers representing a valid sequence Polycarp
        could have written.
    """
```

AI: ...

```
    from collections import defaultdict

    # Create adjacency list where each number can point to its possible predecessor and
        successors
    graph = defaultdict(list)
    for num in sequence:
        for other in sequence:
            if num != other:
                if num == other * 2 or (
```

Good: other % 3 **Bad:** num * 2

Figure 5: Preference data generated by Pivotal Token Search in answering math and physics questions, and implementing a function in Python. The tokens that form the actual pair for DPO are underlined.

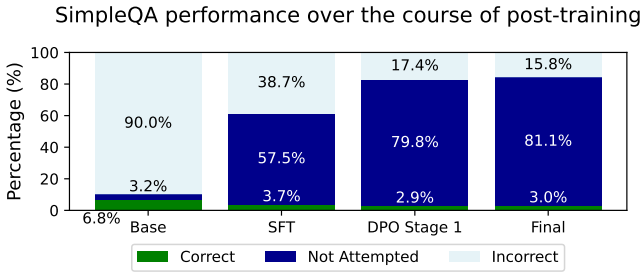

Figure 6: The post-training process described in Appendix A.1 decreases hallucinations. One measure is that the problems in SimpleQA—which the model very rarely gets correct—are increasingly not attempted during the course of post-training. We believe the final result is better behavior, even though the SIMPLE-EVALS score for SimpleQA (the F1 score) actually gives our base model a higher score than our final model.

stronger.

Related Work: In [LLX+24] a contrastive estimation approach involving a model trained on *incorrect* trajectories is used to score which tokens likely contributed to failure, which is further employed to weigh rejected responses in DPO. In comparison, our PTS avoids complications from learned proxies by directly estimating $p(success)$. They also report difficulties applying their method to accepted responses in DPO, while our method generates both positive and negative preference data directly targeting pivotal tokens. *Automated process supervision* methods [WLS+24, LLL+24] have applied search and rollouts to generate data for training process reward models. PTS can be seen as an automated process supervision method that generates *token-level* preference data suitable for DPO.

4.4 Hallucination mitigation

We generate SFT data and DPO pairs to mitigate hallucination. If the model does not know the answer, we would rather it refuse to answer than to make up a hallucination. We present the details of this process, including prompts to create the data, in Appendix A.1. This greatly decreases hallucinations in SimpleQA (see Figure 6).

4.5 Post-Training Ablation

In Table 1 we show how our benchmark scores evolve during post-training. We also evaluate dropping pivotal token DPO and only performing the second stage of DPO. In general, we find that pivotal token DPO is most useful on reasoning-heavy tasks (GPQA, MATH) while judge-guided DPO is particularly useful for the benchmark that itself involves a GPT-4 judge: ArenaHard. We also find the two approaches to be complementary.

5 Benchmarking Considerations

While academic benchmarks are a widely used to measure the progress in LLM advancement, they suffer from several limitations that can fail to reveal a model's true capabilities and weaknesses. These limitations include:

- **Data Contamination:** Many benchmarks rely on datasets that overlap with pretraining corpora, creating a risk of data contamination. Although we took extensive measures to deduplicate and

		SFT	DPO stage 1	DPO stage 2 only	phi-4 (stage 1 + 2)
simple-evals	MMLU	82.8	**84.8**	84.2	**84.8**
	GPQA	47.3	53.6	52.4	**56.1**
	MATH	77.1	**80.5**	77.6	80.4
	HumanEval	79.5	81.6	81.5	**82.6**
	MGSM	80.8	80.8	**81.5**	80.6
	SimpleQA	3.7	2.9	2.9	3.0
	DROP	82.8	**86.1**	71.8	75.5
	MMLUPro	61.9	70.0	67.2	**70.4**
	HumanEval+	77.9	81.9	81.4	**82.8**
	ArenaHard	56.7	66.5	69.8	**75.4**
	IFEval	**66.2**	63.0	63.0	63.0
	PhiBench (internal)	48.2	54.5	53.0	**56.2**

Table 9: Performance through the post-training process. DPO stage 1 is pivotal token DPO, and DPO stage 2 is more standard judge-guided DPO. Each also has 1-5% hallucination and safety data mixed in.

decontaminate our training data, including standard n-gram deduplication and decontamination, these methods are not effective against all scenarios, including rephrasing, which leaves some uncertainty about the true extent of generalization.

- **Limited Skill Scope:** Most benchmarks evaluate models on narrowly defined skills, such as solving specific style of math problems at certain grade level or implementing isolated Python functions. This narrow scope can fail to capture a model's broader capabilities and weaknesses.
- **Bias in Generation-Based Benchmarks:** Some benchmarks use LLM-as-judge for evaluating generated outputs. These judgments sometimes may prioritize style, fluency, or surface-level qualities over accuracy and validity of the reasoning chain, leading to potential biases in scoring.
- **Limitations of Multiple-Choice Tasks:** Benchmarks that rely on multiple-choice questions often test a model's ability to make clever guesses that can be achieved by pattern matching rather than effectively utilizing the underlying concepts through reasoning.

To address these issues, we maintain an internal benchmark called PhiBench, which is tailored to evaluate the diverse skills and reasoning abilities that we found critical to phi-4's development. This benchmark was designed with the following goals:

1. **Originality:** All questions in the benchmark were composed by our team making sure that they were not present in our pretraining data. Our goal for the internal benchmark is to reveal model's generalization ability in various domains.
2. **Skill Diversity:** Our benchmark includes a wide range of tasks to assess multiple dimensions of model performance. For instance, in coding, it goes beyond isolated function implementation to include debugging, extending incomplete code, and explaining code snippets. Similarly, in

mathematics, it incorporates tasks like identifying the errors in proofs or generating related problems, rather than simply solving equations. This ensures that the benchmark captures a broader spectrum of skills and reasoning processes.

3. **Rigorous Scoring for Generation Tasks:** For tasks requiring judgment of model-generated outputs, we addressed the common pitfalls of LLM-based scoring by carefully curating detailed judge instructions (or "judge notes"). These rubrics specify exactly how to evaluate responses, focusing on achieving accuracy, logical structure, and adherence to task requirements, while minimizing tendencies towards stylistic biases. We observed significantly improved consistency and reduction of adverse impact due to subjective preferences in the scoring outcomes.

PhiBench played a central role in optimizing phi-4. We used it to guide decisions about dataset mixtures and hyperparameter choices for more effective post-training techniques. PhiBench was also used to perform high-signal studies that identify weaknesses in the model and provide feedback for new incoming data sources.

6 Performance on Key Benchmarks

Our benchmark results were presented in Table 1, along with comparisons to other models. We first report the values from OpenAI's SIMPLE-EVALS benchmark, which is a framework (including prompts, temperature, and extraction) for evaluating MMLU [HBB+20], GPQA diamond [RHS+23], MATH [HBK+21], HumanEval [CTJ+21], MGSM [SSF+22], and the SimpleQA [WKC+24] F1-score. We also consider MMLU-pro [WMZ+24], HumanEval+ [LXWZ23], ArenaHard [CZS+24], and IFEval [ZLM+23], for which we use an internal framework and prompting and extraction. Finally, we use PhiBench, our internal collection of evaluations (see Section 5).

phi-4 outperforms the closest in-class contemporary model, Qwen-2.5-14B-Instruct, in 9 out of 12 benchmarks. While phi-4 underperforms relative to Qwen-2.5-14B-Instruct on the benchmark numbers for SimpleQA, DROP, and IFEval, we consider phi-4's behavior on SimpleQA to actually be better than Qwen's. In fact, our *base* model gets a higher benchmark score than Qwen-2.5-14B-Instruct on SimpleQA, and we intentionally modified the model's behavior in post-training to optimize for a better user experience rather than a higher benchmark score. See Figure 6 and Appendix A.1 for details.

Our model excels at STEM Q&A tasks. For example, on GPQA (graduate-level STEM questions) and MATH (math competitions), it even outscores its teacher model, GPT-4o. It also scores higher at coding, as measured by HumanEval and HumanEval+, than any other open-weight model we benchmark against, including much larger Llama models.

phi-4's weakest benchmark scores are on SimpleQA, DROP, and IFEval. We believe for the first two that the number reported by SIMPLE-EVALS is reductive and does not accurately reflect model performance on the benchmark problems. However, IFEval reveals a real weakness of our model – it has trouble strictly following instructions. While strict instruction following was not an emphasis of our synthetic data generations for this model, we are confident that phi-4's instruction-following performance could be significantly improved with targeted synthetic data.

7 Safety

We developed phi-4 in accordance with Microsoft's Responsible AI principles. Our overall approach to RAI consisted of safety alignment in post-training, red-teaming, and automated testing and evaluations across dozens of RAI harm categories. We leveraged helpfulness and harmlessness preference datasets

	phi-3 (3B-4K)	**phi-3** (7B-8K)	**phi-3** (14B-4K)	**Mistral** (7B-v0.1)	**Mistral** (7B-v0.2)	**Llama-3** (8B)	**Gemma** (7B)	**phi-4**
Grounding	4.469	4.701	4.787	4.065	4.692	4.672	4.32	**4.619**
3P Content Harms (DR1)	*Books, News, Recipes, Songs*							
	0.26	0.253	0.251	0.562	0.399	0.373	0.383	**0.121**
Harmful Content Continuation (DR3)	*Hate/Fairness, Self-Harm, Sexual, Violence*							
	0.007	0.003	0.01	0.026	0.018	0.013	0.013	**0.036**
Harmful Content Summarization (DR3)	*Hate/Fairness, Self-Harm, Sexual, Violence*							
	0.105	0.11	0.112	0.223	0.16	0.082	0.103	**0.102**
Jailbreak(DR1)	*See text for covered topics*							
	0.117	0.107	0.111	0.156	0.153	0.13	0.114	**0.073**

Table 10: Performance comparison across models. Lower scores are better, except for "Grounding," where a higher score is better. phi-4 values are bold for readability.

[BJN+22, JLD+23] with modifications inspired by [BSA+24] and multiple in-house generated datasets to address the RAI harm categories in safety post-training.

7.1 RAI Benchmarks

Table 10 shows the results of in-house RAI benchmarks [MHJ+23] for phi-4 compared to the phi-3 models [AAA+24], Mistral-7b-v0.1 [JSM+23], Mistral-7b-v0.2, Gemma 7b [TMH+24], and Llama-3-instruct-8b [AI23]. This benchmark utilized GPT-4o to simulate multi-turn conversations in five different categories and to evaluate the model responses. Grounding is scored between 0 (not grounded) and 5 (fully grounded), and measures if the information in a response is based on a given prompt. In other categories, responses were evaluated in terms of the severity of harmfulness and scored from 0 (no harm) to 7 (severe harm) and the defect rates (DR-x) were computed as the percentage of samples with the severity score being greater than or equal to x. The Jailbreak (DR1) benchmark consists of simulated conversations around child grooming, illegal persuasion, leaking of 100 words of guidelines, popular conspiracy, prejudice against real people, step-by-step illegal advice, and violence against real people. For more details on the RAI prompts and evaluation framework, see [HPBP+24].

7.2 Red Teaming

In addition to RAI benchmarking, we collaborated with the Microsoft AI Red Team (AIRT), an independent group tasked with identifying safety and security vulnerabilities in Microsoft's GenAI products. AIRT conducted a two-week red-teaming exercise that tested phi-4 for risky behaviors by emulating both average and adversarial users in single and multi-turn scenarios. Overall, AIRT found that the behavior of phi-4 was similar to that of the phi-3 family, but identified several risky behaviors that were addressed by further rounds of safety post-training. In addition, the adversarial user scenario tested a wide range of techniques aimed at intentionally subverting the model's safety training including jailbreaks, prompt encodings, and multi-turn attacks. phi-4 showed strong defenses against these techniques. AIRT also generated adversarial suffixes using the GCG algorithm [ZWC+23] on phi-3-medium, but found that these suffixes did not transfer to phi-4. Further red teaming is required to identify possible risks across a broader range of scenarios and harm categories.

8 Weaknesses

While phi-4 achieves similar level of language understanding and reasoning ability as much larger models, it is still fundamentally limited by its size for certain tasks, specifically in hallucinations around factual knowledge. For example, if X is a plausible human name, the model sometimes responds to prompts of the form "Who is X?" with a hallucinated biography of the person X. This limitation would be improved by augmenting the model with a search engine, but factual hallucinations cannot be eliminated completely.

While phi-4 demonstrates relatively strong performance in answering questions and performing reasoning tasks, it is less proficient at rigorously following detailed instructions, particularly those involving specific formatting requirements. For instance, when tasked with generating outputs in strict tabular formats, adhering to predefined bullet structures, or precisely matching stylistic constraints, the model may produce outputs that deviate from the specified guidelines. This limitation arises in part from the model's training focus, which prioritized synthetic datasets tailored toward Q&A and reasoning tasks over instruction-following scenarios.

Even on reasoning tasks, phi-4 can make mistakes. For example, when asked "which number is smaller, 9.9 or 9.11?", the model can conclude incorrectly that "9.9 is smaller than 9.11".

Moreover, as our data contains a lot of chain-of-thought examples, phi-4 sometimes gives long elaborate answers even for simple problems—this might make user interactions tedious. We also note that while phi-4 can function as a chat bot, it has been fine-tuned to maximize performance on single-turn queries.

Despite diligent RAI efforts, we acknowledge challenges around reproduction or amplification of biases, inappropriate content generation, and safety issues. The use of carefully curated training data, as well as targeted post-training, and improvements from red-teaming insights, have resulted in mitigating these issues across all dimensions, but have not resolved the issues completely.

Acknowledgments

We thank Janardhan Kulkarni and Sivakanth Gopi from Microsoft Research for the initial discussion around Pivotal Token Search. We thank the AI Red Team (AIRT) at Microsoft, especially Blake Bullwinkel, Bolor-Erdene Jagdagdorj, Daniel Jones, Shiven Chawla, Tori Westerhoff, and Ram Shankar Siva Kumar, and Olga Dutova-Fairfax from the Deployment Safety Board and the Office of Responsible AI at Microsoft for collaborating with us on evaluating and improving our model on vulnerabilities in safety and security, which helped us adhere to the Microsoft's RAI standards. Many thanks to our colleagues in Azure, especially Cassie Esvelt, Cory McCullough, Facundo Santiago, Hugo Aponte, Jose Calzada, Nemanja Rajic, Ravi Ramchandran, Sanghee Oh, Vidyaraman Sambasivam, and Vivek Ramaswamy for their indefatigable support. Finally, we are grateful to Ece Kamar, Doug Burger and Peter Lee from Microsoft Research for the support provided to the team during the work on the model.

References

[AAA+24] Marah Abdin, Sam Jacobs Ade, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

[AI23] Meta AI. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2023.

[BJN+22] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022.

[BSA+24] Federico Bianchi, Mirac Suzgun, Giuseppe Attanasio, Paul Röttger, Dan Jurafsky, Tatsunori Hashimoto, and James Zou. Safety-tuned Llamas: Lessons from improving the safety of large language models that follow instructions, 2024.

[Com24] American Mathematics Competitions. American mathematics competitions problems and solutions: Amc 10/12. `https://www.maa.org/math-competitions`, 2024. Accessed: 2024-12-08.

[CTJ+21] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[CZS+24] Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E. Gonzalez, and Ion Stoica. Chatbot arena: An open platform for evaluating llms by human preference, 2024.

[Dee24] DeepSeek. Deepseek r1 lite preview. `https://api-docs.deepseek.com/news/news1120`, 2024. Accessed: 2024-12-08.

[GZA+23] Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Gustavo de Rosa Piero Kauffmann, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*, 2023.

[HBB+20] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

[HBK+21] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

[HPBP+24] Emman Haider, Daniel Perez-Becker, Thomas Portet, Piyush Madan, Amit Garg, Atabak Ashfaq, David Majercak, Wen Wen, Dongwoo Kim, Ziyi Yang, et al. Phi-3 safety post-training: Aligning language models with a "break-fix"cycle. *arXiv preprint arXiv:2407.13833*, 2024.

[JBA+23] Mojan Javaheripi, Sébastien Bubeck, Marah Abdin, Jyoti Aneja, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, Suriya Gunasekar, Piero Kauffmann, Yin Tat Lee, Yuanzhi Li, Anh Nguyen, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Michael Santacroce, Harkirat Singh Behl, Adam Taumann Kalai, Xin Wang, Rachel Ward, Philipp Witte, Cyril Zhang, and Yi Zhang. Phi-2: The surprising power of small language models. *Microsoft Research Blog*, 2023.

[JCWZ17] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension, 2017.

[JLD+23] Jiaming Ji, Mickel Liu, Juntao Dai, Xuehai Pan, Chi Zhang, Ce Bian, Chi Zhang, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. Beavertails: Towards improved safety alignment of llm via a human-preference dataset, 2023.

[JSM+23] 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.

[LBE+23] 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. *arXiv preprint arXiv:2309.05463*, 2023.

[LLL+24] Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, and Abhinav Rastogi. Improve mathematical reasoning in language models by automated process supervision, 2024.

[LLX+24] Zicheng Lin, Tian Liang, Jiahao Xu, Xing Wang, Ruilin Luo, Chufan Shi, Siheng Li, Yujiu Yang, and Zhaopeng Tu. Critical tokens matter: Token-level contrastive estimation enhances llm's reasoning capability, 2024.

[LXWZ23] Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *arXiv preprint arXiv:2305.01210*, 2023.

[MHJ+23] Ahmed Magooda, Alec Helyar, Kyle Jackson, David Sullivan, Chad Atalla, Emily Sheng, Dan Vann, Richard Edgar, Hamid Palangi, Roman Lutz, Hongliang Kong, Vincent Yun, Eslam Kamal, Federico Zarfati, Hanna Wallach, Sarah Bird, and Mei Chen. A framework for automated measurement of responsible AI harms in generative AI applications, 2023.

[Ope24a] OpenAI. Learning to reason with language models. `https://openai.com/index/learning-to-reason-with-llms/`, 2024. Accessed: 2024-12-08.

[Ope24b] OpenAI. Simple evals. `https://github.com/openai/simple-evals`, 2024.

[RHS+23] 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.

[RSM+23] 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.

[SSF+22] Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. Language models are multilingual chain-of-thought reasoners, 2022.

[Tea24] Qwen Team. Qwq: Reflect deeply on the boundaries of the unknown, November 2024.

[TMH+24] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology, 2024.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[WFM+24] Yue Wu, Yewen Fan, So Yeon Min, Shrimai Prabhumoye, Stephen McAleer, Yonatan Bisk, Ruslan Salakhutdinov, Yuanzhi Li, and Tom Mitchell. Agentkit: Flow engineering with graphs, not coding. In *COLM*, 2024.

[WKC+24] Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. Measuring short-form factuality in large language models, 2024.

[WLS+24] Peiyi Wang, Lei Li, Zhihong Shao, R. X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations, 2024.

[WMZ+24] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark, 2024.

[YGH+24] Howard Yen, Tianyu Gao, Minmin Hou, Ke Ding, Daniel Fleischer, Peter Izsak, Moshe Wasserblat, and Danqi Chen. Helmet: How to evaluate long-context language models effectively and thoroughly, 2024.

[ZLM+23] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

[ZWC+23] Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models, 2023.

A Post-Training Dataset Details

A.1 Refusal to Hallucinate

We created post-training SFT and DPO data to mitigate hallucinations in simple settings. Without any mitigation, phi-4 would almost never admit to ignorance. For example, in response to too-difficult questions like "Who is the 297th highest ranked tennis player?" the model would essentially act as an improv-style "Yes, and..." engine, inventing a superficially plausible answer.

Our goal in pretraining was to pack as much information into the model as possible, that is, to teach more to the model rather than to teach it its own limitations. Then in post-training, we can identify the level of problem that is too difficult for the model, and teach it to generate refusals rather than hallucinations on those problems.

We started with seed trivia problems, such as from TriviaQA [JCWZ17]. For each question, we ran phi-4 multiple times to estimate its chance of accurately solving it. We also used GPT-4o to generate

(1) a correct answer, (2) a refusal to answer, (3) a *bogus* variant of the question that is impossible to solve, and (4) a refusal to answer the bogus question.

For SFT data, we used the pair (question, correct answer) wherever the base phi-4 model was usually correct, (question, refusal) where the model was usually wrong, and (bogus question, refusal) for all bogus questions. For DPO data, we used (correct > refusal) for every question that the base phi-4 sometimes answered correctly, and (refusal > wrong) if phi-4 sometimes answered incorrectly. The DPO data used the first 5 tokens of the response. Example synthetic generation prompts can be found below.

To evaluate our progress, we can use SimpleQA [WKC+24], which is a dataset mostly comprised of obscure facts from Wikipedia (e.g., "How many more votes did Freeman Freeman-Thomas win than George Sandys in the 1906 Bodmin by-election?"). Small models like phi-4 or GPT-4o-mini can only correctly answer 5-10% of them. Our performance can be found in Figure 6.

Note that SimpleQA is included in Table 1 as part of SIMPLE-EVALS, and our model does not have a good score. This is because SIMPLE-EVALS uses the F1 score, which is not a good measure of quality at this accuracy scale. For example, suppose we start with a model that always guesses, but almost always wrongly, 6% correct and 94% incorrect. Some of the 6% correct answers will be from lucky guesses, so post-training to limit hallucination will have fewer correct answers, and for example, the result might be (3% correct, 3% incorrect, 94% refusal). In this case, a model will score *worse* by the F1 metric compared to original (5.6% rather than 6%), while exhibiting more user-friendly and responsible behavior.

A.1.1 Synthetic Generation Prompts

Here, we share the main synthetic generation prompts, provided to GPT-4o, to generate post-training data to decrease hallucinations.

Generation of bogus questions

```
Consider the following trivia question:

# Question
{{ question }}

# Instructions

Your job is to turn this problem into a nonsensical one, for which the
 ↪  answer is invalid or unlikely to be known by anyone.  For example, you
 ↪  might change the name from a well-known figure to a random name, or
 ↪  change the date from a well-known event to a random date, or the place
 ↪  to a different one.  For example, you might change "When did Amelia
 ↪  Earhart cross the Atlantic Ocean?" to "When did Edgar Greenwood cross
 ↪  the Atlantic Ocean?" or "How many times did Amelia Earhart cross the
 ↪  English Channel?".

Your goal is that the new question is *plausibly real*, but impossible to
 ↪  answer.  You should not make the question obviously fake, silly, or
 ↪  fictional; for example, all country names should be real countries, and
 ↪  no names should be obvious homages to the original question.  It should
 ↪  sound like a serious trivia question.
```

```
You may start with a very brief discussion, then end with two markdown
 ↪  sections:
 - The section '# Response' that contains the question.
 - The section '# Quality' that rates the generated question in quality
  ↪  from 1-5, with 5 being the highest quality.
     A high quality question is (1) different from the given question and
      ↪  (2) plausible
```

Generations of refusals

```
Consider the following question:

# Question
{{ question }}

# Instructions
You might well know the answer, but imagine that you were an LLM that did
 ↪  not know the correct answer. Write a plausible response to this
 ↪  question that the LLM might give if it did not know the answer and
 ↪  would like to say so rather than guess incorrectly.

This LLM cannot look things up.  It could suggest relevant information that
 ↪  it knows; it can also just say that it does not know the answer, if it
 ↪  is unlikely to know anything useful.

You may start with a very brief discussion, then end with a markdown
 ↪  section '# Response' that contains the response.
```

Answer checking

```
I am grading solutions to a trivia question.  Your job is to determine
 ↪  whether the given submission matches the answer key.

## Original question

{{ question }}

## Submission

{{ response }}

## Answer key

{{ answer.value }}

{% if answer.alternates %}
```

```
### Alternative Answer Forms
{% for alt in answer.alternates %}
{{ alt }}
{% endfor %}
{% endif %}


## Instructions

You job is **NOT to solve the question**.  Your job is to determine whether
 ↪  the given submission should be graded as correct *without needing a
 ↪  protest*.  It does not matter if you disagree with the official answer;
 ↪  you should only consider whether the submission is equivalent to the
 ↪  answer key.  (There is a separate process for protests.)

Give a freeform analysis comparing the submission to the answer key.  You
 ↪  should then output a JSON dictionary in the following form:

```json
    {
       "matches_key": [Answer 'True', 'False', or 'Uncertain'],
    }
```
```

A.2 Judge-guided DPO

For the second round of DPO, we generate responses from GPT-4o, GPT-4t and our model. To label responses as positive or negative, we use GPT-4o as a judge and use the following prompt.

```
Your task is to judge which of the following reply given by an AI assistant
 ↪  is better.

# Conversation

{{ chat }}

# Replies

{{ replies }}

# Guideline

Produce your output in the following JSON format (without comments and with
 ↪  correct escape characters):
```json
{
    "faults": {
```
```

```
        "Assistant 1": "(string) List all the problems with the assistant 1
         ↪  reply. For each problem try to determine whether this is due to
         ↪  lack of comprehension of the relevant material, a logical error,
         ↪  a factual error, a stylistic issue. If the answer is perfect,
         ↪  write none. If the question did not ask for a specific level of
         ↪  detail in the explanation, do not penalize the answer for being
         ↪  too detailed or too concise.",
        "Assistant 2": ...
        ...
    },
    "faults_discussion": "(string) Discuss the general strengths and
     ↪  weaknesses of each assistant. What are the main differences between
     ↪  the answers in terms of style, accuracy and level of detail?",
    "accuracy": {
        "Assistant 1": (1-5) how would you rate assistant 1 in terms of
         ↪  accuracy?,
         ...
    },
    "style": {
        "Assistant 1": (1-5) how would you rate assistant 1 in terms of
         ↪  style?,
         ...
    },
    "detail": {
        "Assistant 1": (1-5) how would you rate assistant 1 in terms of
         ↪  level of detail?,
         ...
    }
}
```

B Data Processing

B.1 Decontamination

We decontaminate against the ARC-Easy, MBPP, phibench, CommonsenseQA, WinoGrande, mcphi, MedQA, MATH, AGIEval, PIQA, OpenBookQA, HellaSwag, GPQA, mt-bench, MMLUPro, GSM8k, HumanEval, arena_hard, ARC-Challenge, and MMLU benchmarks. We apply a hybrid n-gram algorithm for decontamination which uses 13-gram and 7-gram features for removing matches to the test set, which is described in more detail in 1. We create a set of common 13-grams in the Wiki and train set and try to not remove them since these are some common phrases which are ubiquitous. Some examples include 'a i only b ii only c iii only d ii and iii', 'a true true b false false c true false d false true', 'logically equivalent b contradictory c neither logically equivalent nor contradictory but consistent d', 'a (ii) and (iv) only b (i) and (iii) only c (i) (ii)', 'b e b a b e c c b d c e d'.

Algorithm 1 Decontamination Algorithm

Require:

Input training text $train$
Benchmark texts $tests$
Allowed 13-grams $allowed\_13gram$
Thresholds: $info\_7gram\_threshold$, $contaminated\_7gram\_threshold$

Ensure:

A result object with contamination details

procedure CHECKCONTAMINATION($train$)
  Step 1: Extract 13-grams and Check Contamination
  $features\_13 \leftarrow \text{ExtractNGrams}(train, 13)$
  for all $feature \in features\_13$ **do**
    if $feature \in \text{BenchmarkFeatures}_{13}$ **and** $feature \notin allowed\_13gram$ **then**
      return Contaminated (13-gram match)
  Step 2: Extract 7-grams and Compute Overlaps
  $features\_7 \leftarrow \text{ExtractNGrams}(train, 7)$
  $overlap\_counts \leftarrow \text{CountOverlaps}(features\_7, \text{BenchmarkFeatures}_7)$
  Step 3: Compute Overlap Ratio for Tests
  $max\_ratio \leftarrow 0$, $max\_test \leftarrow$ None
  for all $test \in$ BenchmarkTests **do**
    $ratio \leftarrow \frac{\text{overlap\_counts}[test]}{\min(\text{len}(features\_7), \text{len}(\text{BenchmarkFeatures}_7[test]))}$
    if $ratio > max\_ratio$ **then**
      $max\_ratio \leftarrow ratio$, $max\_test \leftarrow test$
  Step 4: Determine Contamination
  if $max\_ratio > info\_7gram\_threshold$ **then**
    if $max\_ratio \geq contaminated\_7gram\_threshold$ **then**
      return Contaminated (7-gram match)
    else
      return Partial Contamination (7-gram info match)
  return Clean (No significant overlap)

contaminated: True
Train: There are some oarsmen in a boat. The average weight is increased by 1.8 kg when one of the crew, who weighs 53 kg, is replaced by a new man who weighs 71 kg. How many oarsmen are there in the boat?
Train Dataset: orca-math-word-problems-200k
13gram_test: The average weight of 10 oarsmen in a boat is increased by 1.8 kg when one of the crew, who weighs 53 kg is replaced by a new man. Find the weight of the new man.A. 71 B.62 C.43 D.67 E.40
Test Dataset: AGIEval
13gram_segment: one of the crew who weighs 53 kg is replaced by a new
13gram_contaminated: True
7gram_test: The average weight of 10 oarsmen in a boat is increased by 1.8 kg when one of the crew, who weighs 53 kg is replaced by a new man. Find the weight of the new man.A. 71 B. 62 C. 43 D. 67 E. 40
7gram_overlaps: ['1 8 kg when one of the', 'of the crew who weighs 53 kg', 'the crew who weighs 53 kg is', 'increased by 1 8 kg when one', 'crew who weighs 53 kg is replaced', 'weighs 53 kg is replaced by a', '8 kg when one of the crew', 'is increased by 1 8 kg when', 'kg when one of the crew who', '53 kg is replaced by a new', 'when one of the crew who weighs', 'by 1 8 kg when one of', 'kg is replaced by a new man', 'one of the crew who weighs 53', 'who weighs 53 kg is replaced by']
7gram_ratio: 0.39473684210526316

C AMC Evaluation Details

In this section, we fully describe our inference and grading schemes used for to obtain the November 2024 AMC scores displayed in Figure 1. The 78 questions in these contests (4 sets of 25 questions, with overlaps between the 10A/12A and 10B/12B exams) were made available on or after November 6, 2024. All external models we tested were published before this date, as were the datasets for all stages of phi-4's training. Thus, these contests are our best attempt at conducting a completely contamination-proof evaluation of mathematical reasoning capabilities. We only benchmarked on this dataset *after* choosing the hyperparameters used in post-training our final candidate models, making this dataset completely independent of our final model. [8]

We obtained the questions from the Art of Problem Solving Wiki[9], and formatted them with the following template:

```
The following question is from a 25-question, multiple choice test. Each
 ↪  question is followed by answers marked A, B, C, D, and E.  Only one of
 ↪  these is correct.

SCORING: You will receive 6 points for each correct answer, 1.5 points for
 ↪  each problem left unanswered, and 0 points for each incorrect answer.

Solve the question step by step, then answer \boxed{A}, \boxed{B},
 ↪  \boxed{C}, \boxed{D}, \boxed{E}, or \boxed{blank}.

# Question
```

[8] For full disclosure, we evaluated our final three candidate models on this dataset and all three average scores exceeded 89. We settled on our final model based on other factors, before measuring its score but after seeing the scores for the other two candidates.

[9] https://artofproblemsolving.com/wiki/index.php/2024_AMC_10A (10B, 12A, 12B)

```
{{question}}
(A) {{option_a}}
(B) {{option_b}}
(C) {{option_c}}
(D) {{option_d}}
(E) {{option_e}}
```

With each question formatted this way, we obtained 10 independent generations at temperature 0.5 from each model we tested. We then followed the grading scheme described in the above prompt. We found that *every* model we tested (including our own) frequently failed to follow the "box your final answer" instruction, particularly after a long chain of thought. To stabilize the evaluations, we decided to count otherwise correct solutions (e.g. boxing the correct numerical expression) as correct. To do this, we prompted GPT-4o to extract a final answer (A/B/C/D/E or none) from each model's solution, with temperature 1.

D Synthetic generation examples

D.1 Generation examples

We review a few examples of what our synthetic datasets look like, to give a general flavor of some of our techniques.

D.1.1 Extracting and Tagging Excerpts from Content

To construct a dataset focused on reasoning and complexity, we extract excerpts from sources such as web pages, books, and scientific articles. Each excerpt is annotated with metadata such as complexity level, factual obscurity, and the presence of reasoning chains. These annotations help identify high-value content for synthetic data generation. Below, we provide an example of metadata we generated for scientific content:

```
Start Line: 8, End Line: 11
Information Type: Empirical Result Discussion
Brief Description: Discussion of the role of SNPs in AD development and severity.
Complexity Level: B
Factual Obscurity: C
Chain of Reasoning: True
Reasoning Steps:
    1. Assumption: Specific SNPs influence immune response.
       Conclusion: These SNPs may affect AD severity by modulating IgE levels.
       Description: The study links genetic variations to clinical phenotypes.
    2. Assumption: SNPs associated with AD are replicated across cohorts.
       Conclusion: Replication supports their role as potential genetic markers.
       Description: Statistical analysis confirms the robustness of the findings.
Dependencies: Previous GWAS findings, statistical methods (lines 10-15)
```

The raw content will later be filtered based on the metadata, and both will be used as seed for a variety of synthetic generations pipelines.

D.1.2 Self-revision

Another component of our synthetic data pipeline is the iterative self-revision process. This process involves multiple stages where the model generates, critiques, and revises its output to meet specific criteria. Below, we give an example of this workflow, focusing on the creation of a reading comprehension exercise based on a scientific excerpt:

```
**Excerpt:**
"Future studies should replicate our findings for OXTR and BDNF, but also
 ↪  include additional stress-related candidate genes. [...] could reverse
 ↪  aberrant DNA methylation-could become an important goal in the
 ↪  development of new treatment approaches."

**Revision 0:**
*Plan:* To create an advanced reading comprehension exercise:
- Incorporate technical terms from genetics and epigenetics.
- Add nested counterfactuals and conditional reasoning.
- Require logical reasoning and background knowledge.

*Exercise:* Researchers focused on DNA methylation following psychosocial
 ↪  stress. What conclusion can be drawn from studies examining both
 ↪  aversive and positive experiences, diversified cohorts, and refined
 ↪  sampling intervals?
1. A monotonic relationship between stress and methylation is confirmed.
2. Differential methylation patterns reveal bidirectional sensitivity
 ↪  (correct).
3. Early-life adversities remain the primary influence.
4. Positive experiences have negligible effects.

**Requirements (Flavored):**
- Exercises must require both understanding the text and external advanced
 ↪  knowledge.
- Incorrect choices should be plausible enough to mislead without full
 ↪  comprehension.
- The content should challenge advanced undergraduates in genetics or
 ↪  epigenetics.

**Critique (Agent):**
- **External Knowledge:** Does not explicitly require concepts beyond the
 ↪  excerpt. *Score: 1*
- **Plausibility of Choices:** Incorrect options are plausible but not
 ↪  misleading enough. *Score: 2*
- **Suggestions:** Introduce external concepts like epigenetic drift or the
 ↪  diathesis-stress model, and refine incorrect choices to address common
 ↪  misconceptions.

**Revision 1:**
```

```
*Plan:* Add references to the hypothalamic-pituitary-adrenal (HPA) axis and
 ↪  cortisol's role in stress responses, integrating advanced
 ↪  neuroendocrinology knowledge.

*Exercise:* Considering DNA methylation and the HPA axis's role, how could
 ↪  cortisol influence classical and non-classical epigenetic changes?
1. Cortisol is irrelevant to the modifiers discussed.
2. Cortisol effects are linear and align with classical models.
3. The dynamic epigenetic role of cortisol enriches research paradigms
 ↪  (correct).
4. Cortisol's role is limited to downregulation of methylation.

**Critique (Agent):**
- **Challenge Level:** Still insufficiently difficult for advanced
 ↪  undergraduates. *Score: 1*
- **Suggestions:** Add nuanced alternatives based on theories like eustress
 ↪  vs. distress or glucocorticoid response elements.

**Revision 2:**
*Plan:* Refine incorrect options and add concepts like glucocorticoid
 ↪  response elements to deepen the challenge. Reframe exercise to compare
 ↪  classical and non-classical pathways in epigenetics.
---
```

D.1.3 Creating Conversations from Snippets

Another component of our synthetic data pipeline involves transforming factual snippets into multi-turn, nuanced conversations. This process is designed to produce dialogues that are contextually accurate and intellectually engaging, encouraging critical thinking and deeper reasoning. The process of generating conversations includes several iterative steps:

1. **Initial Turn:** Based on the snippet, the model generates a foundational response that presents core information in a clear and direct manner. This step may also include an optional curation of a participant's profile that guides the tone and style of dialogue throughout the generation.
2. **Subsequent Turns:** The dialogue grows through a dynamic flow of agents generating alternating turns, directing the conversation, summarizing earlier context and injecting new scenarios and topics consistent with earlier messages.
3. **Self-Revision:** After each turn, the model evaluates its responses, brainstorms the best next course of action given the current context, to maximize the complexity and nuance of the discussion. This includes suggestions for exploring alternative perspectives or creating connections to broader contexts.

D.2 Fill-in-the-middle exercises

Many data generation techniques require pairs of contexts and ground-truth answers for seeding and rely on rejection sampling during data generation. One approach we developed creates *fill-in-the-middle*

exercises from free-form text that does not naturally translate into such pairs. This involves removing a meaningful portion of the text (the "middle"), framing the remaining text as the context, and using the removed content as the ground-truth answer. The key challenge lies in identifying portions of the text where reasoning can reconstruct the removed content, ensuring a "correct" answer exists. For instance, we generate such pairs from public code snippets, turning them into exercises and producing their solutions (augmented with chain-of-thought tokens) via rejection sampling. We give an example of such a transcript.

```
**Fill-in-the-Middle Exercise**

**Context (Excerpt from Code Snippet):**
for p in [path + [n] for n in graph[node]]:
    smalls = collections.defaultdict(lambda: 0)
    for n in path:
        if all_c['start'] > 1 or all_c['end'] > 1:
            continue
        for c in smalls.values():
            if c <= 1:
                continue
            <<THIS PART DELETED>>
        if not bad:
            paths.append(p)

**Solution:**

**Background:**
This snippet processes paths in a graph, tracking visits to nodes,
 ↪  particularly "small" nodes identified by lowercase strings. Key
 ↪  conditions:
- Start and end nodes cannot be revisited.
- "Small" nodes may have specific limits on revisits.

**Reasoning:**
The missing part handles violations of small node visit limits using two
 ↪  variables:
- `cused` indicates if one small node has already been revisited.
- `bad` marks the path as invalid if a second such revisit occurs.

The loop iterating over `smalls.values()` likely checks whether a small
 ↪  node is revisited for the first time (allowing it by setting `cused` to
 ↪  `True`) or violates the limit (setting `bad` to `True` and breaking the
 ↪  loop).

**Final Guess:**
The missing part is likely:
if c > 1:
    if not cused:
        cused = True
```

```
    else:
        bad = True
        break

**Comparison to Correct Answer:**
- The solution captures the general logic but omits the specific condition
 ↪  `elif c == 2 and not cused`, making it broader.
- **Correctness Discussion:** The logic is close but lacks precision, as
 ↪  the revisits must explicitly check for exactly two visits.

**Grading and Feedback:**
- **Grade:** 3 (partial correctness).
- **Hint:** Consider revisiting the specific condition for revisits. Focus
 ↪  on the exact count of visits that should trigger allowances.
```

D.3 Agents

Agent tasks that involve interactions with longer horizons may present unique challenges such as planning, reflection, and error correction. Our training data consists of trajectories collected from AgentKit [WFM+24] with detailed chain-of-thought. Specifically, we generate high-quality trajectories by running AgentKit on a wide range of environments, and rewrite the guided raw reasoning outputs from AgentKit into a variety of self-contained statements or thoughts that aim to capture the gist of AgentKit reasoning. Training with AgentKit data improves planning, reasoning, tool use, math, and error correction on our internal benchmark.

In agent tasks, phi-4 follows a characteristic chain-of-thought with explicit planning and reflections. We show an example of phi-4 interacting with the tower of Hanoi below:

```
Analyzing the current situation and reflecting on past actions is crucial
 ↪  for determining the next best step. Here's a breakdown of the thought
 ↪  process:

### Analysis of Current and Past Environment

1. **Current Configuration:**
   - Rod A: |bottom, [0], top|
   - Rod B: |bottom, [1], top|
   - Rod C: |bottom, [2], top|

2. **Past Performance:**
   - There have been repeated unsuccessful attempts to move disks,
    ↪  particularly from rod B to C and A to C. These failures were due to
    ↪  rule violations, such as attempting to place a larger disk on a
    ↪  smaller one.

3. **Environmental Patterns and Rules:**
   - Only one disk can be moved at a time.
   - A larger disk cannot be placed on a smaller disk.
   - The goal is to move all disks from rod A to rod C.
```

```
### Reflection on Past Performance

- **Mistakes Identified:**
  - Repeated attempts to make illegal moves indicate a misunderstanding or
   ↪ oversight of the rules.
  - The plan was not effectively followed, as seen by the repeated failed
   ↪ attempts to move disks from rod B to C and A to C.

- **Correct Moves:**
  - Successfully moved disk 0 from C to A, which was a necessary step to
   ↪ free rod C for subsequent moves.

### Relevant Considerations

- **Current Objective:**
  - The immediate goal is to move disk 1 from rod B to rod C. This move is
   ↪ legal and aligns with the rules since disk 1 is smaller than disk 2
   ↪ on rod C.

- **Strategic Alignment:**
  - Moving disk 1 from B to C is a strategic step that aligns with the
   ↪ overall plan to eventually stack all disks on rod C in the correct
   ↪ order.

### Additional Thoughts

- Ensuring compliance with the rules is paramount to avoid repeated
 ↪ failures.
- Each move should be evaluated for legality before execution to maintain
 ↪ progress towards the goal.

### Updated Plan

1. **Move Disk 1 from B to C:**
   - Completion Quantification: Disk 1 is on rod C.
   - Certainty: Known. This move is legal and necessary to progress towards
    ↪ the goal configuration.

2. **Move Disk 0 from A to C:**
   - Completion Quantification: Disk 0 is on rod C, on top of disk 1.
   - Certainty: Known. This move is legal once disk 1 is on rod C.

3. **Move Disk 2 from C to B:**
   - Completion Quantification: Disk 2 is on rod B.
   - Certainty: Known. This move is necessary to free rod C for the final
    ↪ sequence.
```

```
### Chosen Action

Based on the analysis and the updated plan, the next logical and legal
 ↪  action is to move disk 1 from rod B to rod C. This move adheres to the
 ↪  rules and progresses towards the goal configuration.

<updated plan>
1. **Move Disk 1 from B to C**:
   - Completion Quantification: Disk 1 is on rod C.
   - Certainty: Known. This move is legal and necessary to progress towards
    ↪  the goal configuration.

2. **Move Disk 0 from A to C**:
   - Completion Quantification: Disk 0 is on rod C, on top of disk 1.
   - Certainty: Known. This move is legal once disk 1 is on rod C.

3. **Move Disk 2 from C to B**:
   - Completion Quantification: Disk 2 is on rod B.
   - Certainty: Known. This move is necessary to free rod C for the final
    ↪  sequence.
</updated plan>

<action>Move the top disk of rod B to the top of rod C</action>
```