# DeepDistill: Enhancing LLM Reasoning Capabilities via Large-Scale Difficulty-Graded Data Training

Xiaoyu Tian, Sitong Zhao, Haotian Wang, Shuaiting Chen,
Yiping Peng, Yunjie Ji, Han Zhao, Xiangang Li

a-m-team

## Abstract

Although large language models (LLMs) have recently achieved remarkable performance on various complex reasoning benchmarks, the academic community still lacks an in-depth understanding of base model training processes and data quality. To address this, we construct a large-scale, difficulty-graded reasoning dataset containing approximately 3.34 million unique queries of varying difficulty levels and about 40 million distilled responses generated by multiple models over several passes. Leveraging pass rate and Coefficient of Variation (CV), we precisely select the most valuable training data to enhance reasoning capability. Notably, we observe a training pattern shift, indicating that reasoning-focused training based on base models requires higher learning rates for effective training. Using this carefully selected data, we significantly improve the reasoning capabilities of the base model, achieving a pass rate of 79.2% on the AIME2024 mathematical reasoning benchmark. This result surpasses most current distilled models and closely approaches state-of-the-art performance. We provide detailed descriptions of our data processing, difficulty assessment, and training methodology, and have publicly released all datasets and methods to promote rapid progress in open-source long-reasoning LLMs. The dataset is available at: https://huggingface.co/datasets/a-m-team/AM-DeepSeek-Distilled-40M

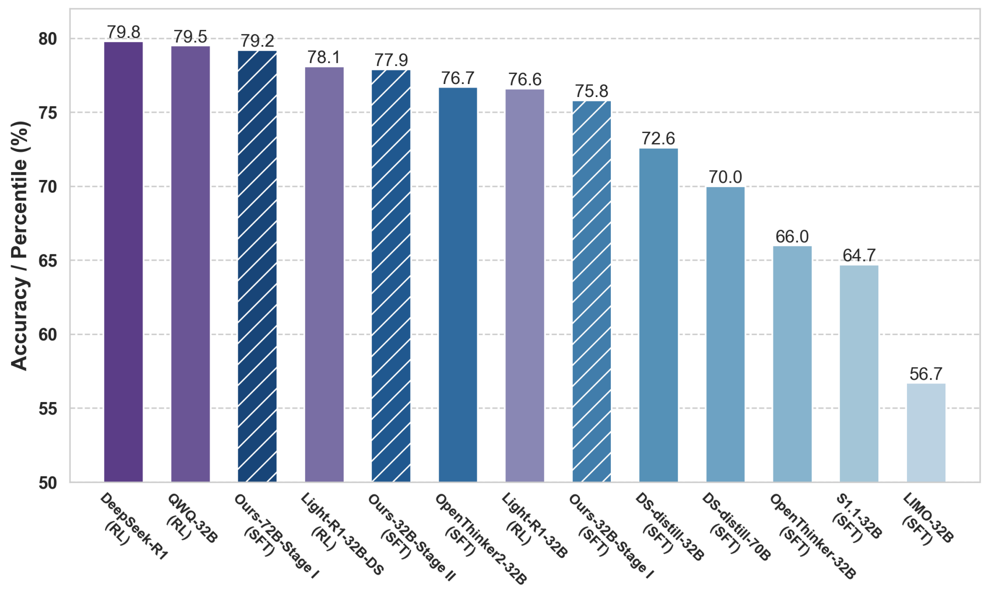


Figure 1: Benchmark performance of open-source models on AIME2024.

Preprint. Under review.

## 1 Introduction

Large language Models (LLMs) have recently achieved remarkable results on complex reasoning tasks. Proprietary systems like OpenAI's O-Series[35] demonstrate exceptional performance on challenging benchmarks, from competitive programming to science and math.

For instance, GPT-4o solves only 13% of problems on the rigorous AIME2024 Math Competition[28], whereas OpenAI's specialized o1 model—trained to reason before answer—achieves about 73% on the same test. Similarly, reinforcement learning-driven models like DeepSeek-R1[4] have attained remarkable performance, ranking among the top open-source reasoning models.

These successes underscore that scaling by extending the length of Chain-of-thought reasoning process[49, 41, 51, 45] and training with appropriate strategies can substantially improve model capabilities: OpenAI's O-Series and DeepSeek-R1 was trained via large-scale reinforcement learning on reasoning tasks. Recent studies[60, 50, 13] have showed that carefully selecting data by well-defined difficulty levels can markedly improve training.

Yet, despite these breakthroughs, the academic community still lacks a deep understanding of how to train base models effectively for long-form reasoning. It remains unclear how to systematically endow a base model with advanced reasoning ability through Supervised Fine-Tuning (SFT).

One key challenge is identifying and leveraging the right training data to teach reasoning. Complex reasoning problems vary widely in difficulty and form (mathematical, code generation, logical puzzles, etc.), and not all examples are equally useful for learning. Previous work[23, 59] typically rely on a single model's scoring or judgement to define problem difficulty or quality. This single-judge strategy is inherently biased and highly sensitive to the model used, the definition of difficulty, and the prompting method. A problem that one model finds difficult might be easy for another, and prompt phrasing can drastically alter a model's perceived success. As a result, using a single model's perspective can lead to less generalizable learning outcomes – models may overfit to the evaluator's idiosyncrasies rather than gaining a broad problem-solving skill.

To address these limitations, we propose a large-scale difficulty-graded reasoning dataset and difficulty-aware reasoning data selection strategy to improve long-form reasoning in LLM training. Instead of trusting any single model to label which examples are hard or useful, we leverage a multi-pass distillation approach across different capabilities of models. Concretely, we construct a large-scale dataset where each query is accompanied by numerous solution attempts from models of varying strengths.

Our contributions are twofold:

- A large-scale difficulty-graded reasoning dataset – We present a dataset of 3.34 million unique reasoning queries, each paired with rich metadata and an extensive set of 40 million model-generated responses spanning a range of model capability levels. The graded difficulty design facilitates tailored training for models of varying sizes and capabilities, and also supports further applications such as downstream alignment methods including DPO[37], GRPO[40], and others. We release this dataset publicly to serve as a challenging benchmark and versatile training resource for the community.

- Large-scale difficulty-aware reasoning training – We observe that enhancing the reasoning capabilities of base models, compared to traditional post-training methods, results in a training pattern shift, often requiring higher learning rates for effective training. By carefully selecting and scaling the difficulty of the training data, our reasoning-focused fine-tuning approach allows base models to significantly outperform existing distilled models on benchmarks like AIME2024, achieving near state-of-the-art performance.

We conclude by open-sourcing our entire methodology to facilitate further research. We have released our data to encourage transparency and rapid progress in building reasoning-strong LLMs from scratch. By sharing our 40 million model-generated dataset and training recipe, we hope to enable the community to explore difficulty-aware training at scale and we hope this dataset empowers the research community to accelerate breakthroughs toward reasoning-strong LLMs.

## 2 Data

### 2.1 Data Collection and Categories Definition

To ensure comprehensiveness and diversity in our experimental data, we extensively collected datasets from multiple publicly available open-source corpora, clearly categorizing and defining each source. These datasets span tasks including mathematical reasoning, code generation, scientific reasoning, instruction-following, multi-turn conversations, and other/general reasoning.

The collected data were explicitly classified into the following categories:

**Mathematical Reasoning** Datasets in this category demand advanced numerical logic and reasoning capabilities from the model. We included datasets such as OpenR1-Math-220k[7], Big-Math-RL-Verified[1], data_ablation_full59K[30], NuminaMath[21], MetaMathQA[54], 2023_amc_data[2], DeepMath-103K[9], and AIME_1983_2024[5][1].

**Code Generation** This category evaluates the model's ability in programming problem-solving and code generation tasks. Datasets selected include PRIME[55], DeepCoder[27], KodCode[52], liveincode_generation[12], codeforces_cots[36], verifiable_coding[34], opencoder[11], and OpenThoughts-114k-Code_decontaminated[7], AceCode-87K[57].

**Scientific Reasoning** Scientific reasoning datasets primarily assess the model's performance in natural sciences and logical reasoning domains. We included datasets such as task_mmmlu[46], chemistryQA[29], Llama-Nemotron-Post-Training-Dataset-v1[32], LOGIC-701[10], ncert[19, 18, 16, 17, 14, 15], and logicLM[25].

**Instruction Following (IF)** We selected four datasets closely related to instruction-following tasks, namely Llama-Nemotron-Post-Training-Dataset[31], tulu-3-sft-mixture[20], if-eval-like, and AutoIF. Specifically, the if-eval-like dataset is aggregated from multiple data sources, and the AutoIF dataset was generated by leveraging the Qwen2.5-72B-Instruct[53, 42]. These datasets emphasize the model's ability to accurately comprehend and execute given instructions.

**Multi-turn Conversations** Multi-turn conversation datasets focus on maintaining context coherence and logical consistency across multiple interactions. Included datasets are InfinityInstruct[33, 61, 58], OpenHermes-2.5[44], tulu-3-sft-mixture[20], and ultra_chat[6].

**Others/General Reasoning** Datasets in this category cover a wide and diverse range of reasoning tasks, including open-ended queries, general knowledge reasoning, and everyday logical reasoning. The datasets selected here are evol[43], InfinityInstruct[33, 61, 58], open_orca[24], tulu-3-sft-mixture[20], natural_reasoning[56], flan[8, 26, 48, 39, 47], and OpenHermes-2.5[44].

The chosen datasets and their defined categories comprehensively cover various task domains, enabling us to thoroughly assess the generalizability and effectiveness of our proposed data distillation method.

### 2.2 Query Processing

To ensure high quality and effectiveness of the data for subsequent model training, we applied meticulous preprocessing procedures to the collected raw queries. These procedures encompassed deduplication, filtering, and decontamination steps to mitigate redundancy and data contamination, thereby preventing negative impacts on the experimental outcomes.

#### 2.2.1 Deduplication and Filtering

Initially, we conducted exact deduplication by removing queries with identical text. Furthermore, to enhance the data quality, stringent filtering criteria were employed as follows:

[1]Note that we have excluded the problems from AIME2024.

- **Unicode Ratio Filtering**: Queries containing a high proportion of Unicode characters exceeding a predefined threshold were removed to avoid the introduction of meaningless or corrupted information.
- **Incomplete Query Filtering**: All queries that were empty or incomplete were removed to ensure query validity and completeness within the dataset.
- **Special Content Filtering**: Queries containing hyperlinks (URLs) or tabular content were excluded, as such content typically should not appear in training queries and could potentially cause hallucinations or misleading outputs from the model.

By implementing these precise deduplication and filtering strategies, we significantly improved the purity and quality of the dataset, laying a robust foundation for stable model training.

#### 2.2.2 Decontamination

To prevent information leakage and semantic overlap between training data and the evaluation set, we performed rigorous decontamination, particularly focusing on the core evaluation dataset, AIME2024.

- **Exact Matching Filtering**: We strictly filtered out queries that exactly matched or closely resembled any query in the AIME2024 evaluation set, ensuring the integrity and reliability of evaluation results.
- **Semantic Deduplication**: To further reduce implicit semantic overlap, we utilized semantic embedding techniques, specifically employing the bge-m3[3] embedding model to calculate semantic similarity between training queries and the AIME2024 evaluation set. Queries with semantic similarity scores exceeding a threshold of 0.9 were identified as potentially contaminated and subsequently removed.

Through these rigorous exact matching and semantic deduplication processes, we substantially mitigated the risks of semantic conflict and information leakage, thereby ensuring that the evaluation results accurately reflect the model's generalization capabilities and true reasoning performance. Ultimately, we obtained approximately 3.34 million queries for further model training.

### 2.3 Data Distilling

To enhance data quality and accurately assess the difficulty of queries, we conducted data distillation using three models with progressively increasing capabilities: DeepSeek-R1-Distill-Qwen-1.5B, DeepSeek-R1-Distill-Qwen-7B, and DeepSeek-R1[4]. Specifically, each of the approximately 4 million preprocessed queries was independently distilled four times by each model, generating corresponding reasoning processes and final answers. This resulted in approximately 40 million distilled responses in total. This approach allowed us to effectively capture variations among model outputs, facilitating subsequent computation of query difficulty.

It is important to note that although we completed distillation using all three models, due to constraints on time and computational resources, subsequent experimental analyses in this paper exclusively utilized the distilled responses from the DeepSeek-R1 model[2].

### 2.4 Ground Truth Verification

To ensure the high quality of distilled data, we designed a rigorous and comprehensive ground truth verification process. Specifically, we adopted distinct verification methods tailored for different data categories, calculating verification scores (*verify_score*) accordingly, and subsequently derived the model's pass rate (*pass_rate*).

#### 2.4.1 Verification Score Calculation

**Mathematical Reasoning** We employed a two-stage validation mechanism for mathematical data. Initially, results generated by models were assessed using Math-Verify[3], yielding a binary outcome

[2]Data from DeepSeek-R1-Distill-Qwen-1.5B, DeepSeek-R1-Distill-Qwen-7B, and DeepSeek-R1 have been publicly released. However, in this paper, only responses from the DeepSeek-R1 model were used for training and analysis.

[3]https://github.com/huggingface/Math-Verify

(correct as 1, incorrect as 0). If an initial result was identified as incorrect, a second validation was conducted using the Qwen2.5-7B-Instruct[53, 42], again producing a binary outcome. The final *verify_score* was computed as:

$$verify\_score_{math} = \frac{Correct_{Math\text{-}Verify} + Correct_{Second\text{-}Validation}}{N_{total}} \tag{1}$$

where $N_{total}$ is the total number of samples.

**Code Generation** For code generation data, the *verify_score* was calculated based on sandbox fusion tests using selected test cases. Specifically, the first 10 available open-source test cases were utilized per query, or all available test cases if fewer than 10 existed. Python codes were tested using standard input-output and assert-based formats, while C++ codes utilized only standard input-output tests. The *verify_score* was computed as follows:

$$verify\_score_{code} = \frac{Count(test\_cases_{passed})}{Count(test\_cases)} \tag{2}$$

**Scientific Reasoning** Scientific reasoning data were validated using the Qwen2.5-7B-Instruct[53, 42]. The model assessed the similarity between generated results and ground truth, yielding decimal scores in the range [0, 5].

**Instruction Following** Verification for instruction-following (IF) data was performed using the ifeval validator. Due to limited constraint annotations in existing open-source datasets, we leveraged Qwen2.5-72B-Instruct to generate additional constraints based on queries. These constraints were formatted appropriately for ifeval validation, with the mean pass rate across all constraints used as the *verify_score*:

$$verify\_score_{IF} = \frac{\sum_{i=1}^{m} pass\_score\_constraint_i}{m}, \quad pass\_score\_constraint_i \in \{0, 1\} \tag{3}$$

Here, $m$ represents the total number of constraints per IF data point.

**Multi-turn Conversations and Others** For multi-turn conversations and general reasoning tasks, we employed Decision-Tree-Reward-Llama-3.1-8B[22] to evaluate three dimensions: coherence, correctness, and helpfulness, each scored in the range [0,4]. The composite reward score, serving as the *verify_score*, was computed as:

$$verify\_score_{others} = \frac{coherence + correctness + helpfulness}{12} \tag{4}$$

### 2.4.2 Pass Rate Calculation

Based on the calculated *verify_score*, we further determined each model's pass rate (*pass_rate*). We set strict thresholds for pass rate evaluation as follows:

- Mathematical reasoning, code generation, and instruction-following data: *verify_score* > 0.99 indicates a pass.
- Scientific reasoning data: *verify_score* > 4.99 indicates a pass.
- Multi-turn conversations and other data: *verify_score* > 0.7 indicates a pass.

The pass rate per query was calculated as the average of binary pass indicators across four distillation attempts:

$$pass\_rate = \frac{\sum_{i=1}^{n} \mathbf{1}(verify\_score_i > threshold)}{n}, \quad n = 4 \tag{5}$$

## 2.5 Quality Assurance

In addition to the aforementioned data processing procedures, we introduced a series of supplementary filtering and quality assurance measures aimed at minimizing noise and anomalies in the dataset. These measures include:

**Perplexity (ppl)-Based Filtering** We utilized our previously trained 32B model[60] to compute perplexity (ppl) scores for concatenated query and model-generated answers. Empirically, high perplexity scores indicate lower semantic coherence or logical consistency. Consequently, we set a threshold of 20, removing data instances with perplexity scores exceeding this value.

**High-Frequency Ngram Filtering** To prevent template-based or repetitive content, we applied Ngram frequency analysis across the entire dataset. Specifically, strings of 20 tokens that appeared more than 20 times were identified and removed as repetitive or redundant content. This step significantly enhanced the diversity and originality of the distilled data.

**Additional Logical and Structural Checks** Beyond statistical filtering, we conducted structural and logical validations to ensure data completeness and consistency, including:

- **Even Turn Verification**: For multi-turn conversation datasets, we required an even number of dialogue turns to maintain contextual integrity and coherence.
- **Think Content Verification**: All distilled data instances were required to explicitly contain intermediate reasoning steps (think content). Instances lacking such content were considered invalid and excluded.
- **Answer Content Verification**: All data instances needed to contain a clear and complete final answer (answer content). Instances without explicit answers were considered incomplete and removed.

For details on data processing procedures and additional information, please refer to Appendix A.

## 3 Methodology

In this section, we aim to identify and scale the most valuable training samples to progressively enhance the model's learning capability. We first introduce the definition of the Coefficient of Variation (CV) and then elaborate on our data selection procedure. Additionally, we conduct preliminary experiments using a two-stage annealing training strategy to further improve model performance.

### 3.1 Coefficient of Variation Definition

Specifically, for each query, we first generated $n$ independent responses using DeepSeek-R1 and computed a *verify_score* for each response (See Section 2), measuring its correctness or quality. However, not all generated data contribute equally to model training. Queries with extremely high average *verify_score* indicate that these questions are too simple for the model, while queries with very low average scores might indicate that these problems are too difficult or inadequately learned by the model. Therefore, queries with extreme average scores should be excluded, allowing us to focus on queries with higher learning potential.

Moreover, the variability of the *verify_score* across multiple responses to the same query is crucial. High variability indicates instability in the model's performance, highlighting significant room for improvement. Thus, we employ the *Coefficient of Variation (CV)* as the key indicator to quantify this variability.

The *Coefficient of Variation* is defined as follows:

$$CV = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\mathit{verify\_score}_i - \mu)^2}}{\mu} = \frac{\sigma}{\mu}, \quad \mathit{where}\ \mu = \frac{1}{n}\sum_{i=1}^{n}\mathit{verify\_score}_i \tag{6}$$

The $CV$ is a normalized dimensionless metric, objectively reflecting the variability of data.

For example, consider two queries with the following *verify_score* values:

$$Q_A : [0.5, 0.5, 0.5, 0.5, 0.5], \mu = 0.5, \sigma = 0, CV = 0$$
$$Q_B : [0.9, 0.1, 0.7, 0.3, 0.5], \mu = 0.5, \sigma \approx 0.32, CV \approx 0.63$$

Although both queries have the same mean score, $Q_B$ exhibits significantly higher variability (higher $CV$), indicating that the model's performance on $Q_B$ is less stable, hence $Q_B$ has greater learning value than $Q_A$.

Consider another example with two queries:

$$Q_C : [0.6, 0.8, 0.7, 0.7, 0.7], \mu = 0.7, \sigma \approx 0.07, CV \approx 0.10$$
$$Q_D : [0.3, 0.5, 0.4, 0.4, 0.4], \mu = 0.4, \sigma \approx 0.07, CV \approx 0.18$$

Despite having equal standard deviations (similar variability), $Q_D$ has a lower mean, suggesting that the model's performance on $Q_D$ is inferior, making $Q_D$ more valuable for learning.

By employing the $CV$-based selection strategy, we effectively identify the most beneficial data for model improvement, thus significantly enhancing the quality of the training dataset and the model's generalization capabilities.

## 3.2 Data Selection

**Large Scale Reasoning Data Selection (Stage I)** We filter data based on the *verify_score* (defined in Section 2) and the *Coefficient of Variation (CV)* (defined in Section 3.1) for each query, using multiple responses generated by the model. This filtering aims to ensure the effectiveness and quality of the training dataset.

**Algorithm 1** Data Filtering Procedure

```
Require: A set of $n$ responses per query with corresponding verify_score
Require: Category-specific verify_score threshold and fixed CV threshold (0.05)
Ensure: Filtered high-quality response set
 1: for each query $q$ in dataset do
 2:     Compute the maximum verify_score $s_{\max}$
 3:     if $s_{\max} <$ verify_score threshold for $q$ then
 4:         Discard query $q$
 5:         continue
 6:     end if
 7:     Compute Coefficient of Variation (CV) for $q$
 8:     if CV $> 0.05$ then                                  ▷ Challenging query
 9:         Retain only responses with verify_score $>$ threshold
10:     else                                                 ▷ Easy query
11:         if Category of $q$ is other or multiturn then
12:             Keep with probability 0.5
13:         else
14:             Discard query $q$
15:         end if
16:     end if
17: end for
18: return Filtered dataset
```

Specifically, we define two key selection criteria: *maximum verify_score threshold* and *CV threshold*. For all data categories, we apply strict *maximum verify_score threshold* to guarantee response correctness. For categories with standardized answers, such as *math*, *code*, and *instruction follow*, we set the *maximum verify_score threshold* to 0.99, and 4.99 for *science*. For more complex or multi-turn conversational data (*other* or *multiturn*), the threshold is relaxed to 0.7 to accommodate the inherent difficulty and variability of such queries.

The filtering process is as follows:

1) First, we evaluate whether the highest *verify_score* among the $n$ responses for each query exceeds the corresponding category-specific threshold. If not, the query is discarded, as it indicates limited learning potential due to poor model performance.

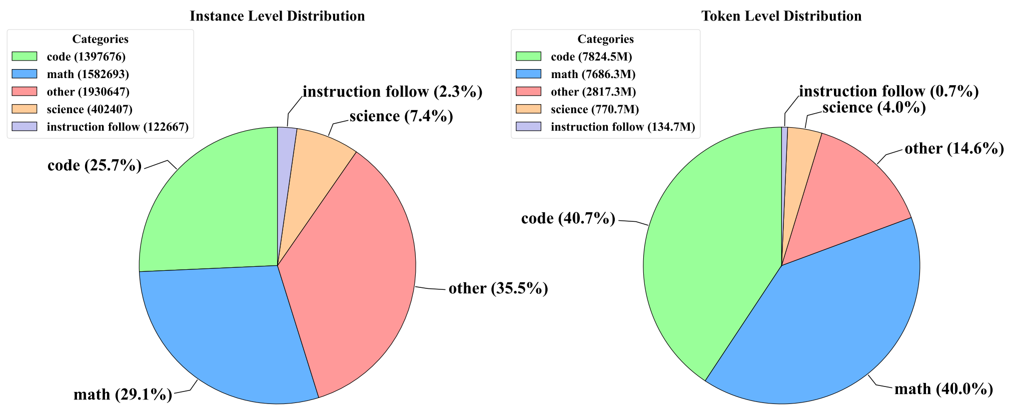


Figure 2: Distribution of training data types in Supervised Fine-Tuning (SFT) Stage I. The left pie chart illustrates the proportion at the instance-level, while the right pie chart shows the distribution at the answer token-level.

2) If the highest *verify_score* passes the threshold, we further assess the variability using the *CV*:
    a) If the *CV* exceeds the threshold of 0.05, the model's performance on this query is considered unstable, suggesting that the query is challenging and holds high learning potential. In this case, only the responses that exceed the *verify_score* threshold are retained to ensure data quality.
    b) If the *CV* is less than or equal to 0.05, the query is deemed easy, and the model's performance is stable. To maintain diversity and learnability within the dataset, we retain these queries based on their category. For *other* and *multiturn* data, we retain 50% of the samples at random to avoid excessive simplification of the dataset; for all other categories, such queries are discarded by default.

Through this carefully designed filtering strategy, we successfully curated a high-quality dataset consisting of **5 million** reasoning-intensive samples with strong learning value.

**Exploration of Annealed Data Selection (Stage II)** Previously, we initially filtered high-quality and moderately challenging data by jointly considering the *verify_score* and the *Coefficient of Variation (CV)*. However, as the model significantly improves its reasoning capabilities after large scale reasoning data training, the most beneficial data for the model in annealing stage consists of queries that are inherently more complex, uncertain, and challenging. Therefore, in the annealing stage, we specifically focus on further increasing the difficulty of the training data by exclusively selecting queries with higher variability, aiming to maximize the model's learning effectiveness.

Specifically, we maintain the same categories as defined in Stage I (*math*, *code*, *science*, *instruction follow*, and *other*), set a stringent *verify_score threshold* of 0.99, and apply a *CV threshold* of 0.05, retaining only those queries exceeding this variability threshold. Moreover, we exclusively select queries exhibiting the highest CV values, and for multiple model responses to a single query, we randomly sample only one response for training.

Through this rigorous data selection strategy, we extract nearly 200k challenging and highly reliable training data, enabling the model to improve further in more difficult reasoning scenarios.

## 4 Experiments

### 4.1 Large Scale Reasoning Training

In the large scale reasoning training stage, we trained the model using the high-quality dataset obtained through the previously described data selection strategy (see Section 3.2). Specifically, we employed a learning rate of $8 \times 10^{-5}$ and trained the model for one epoch. To optimize computational

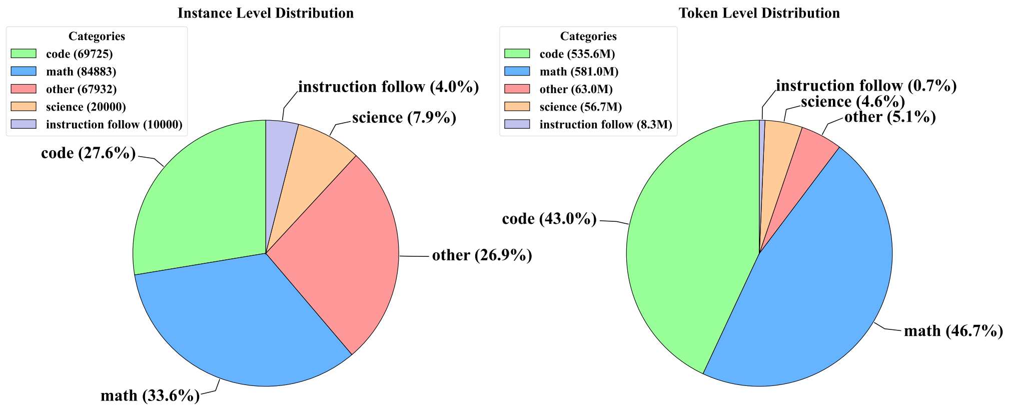


Figure 3: Distribution of training data types in Supervised Fine-Tuning (SFT) Stage II. The left pie chart illustrates the proportion at the instance-level, the right pie chart shows the distribution at the answer token-level.

resources and improve training efficiency, we utilized a packing strategy, grouping sequences up to a maximum length of 32k tokens, with a global batch size of 64. Additionally, we applied a cosine learning rate scheduler with a linear warmup phase for the initial 5% of the training steps, subsequently decaying the learning rate to zero.

Compared to traditional post-training methods, we observed a training pattern shift when enhancing reasoning abilities based on base models. Specifically, a relatively high learning rate (e.g., $8 \times 10^{-5}$ as adopted in our experiments) becomes crucial to effectively capture complex data features. Lower learning rates often led to underfitting, limiting the model's ability to sufficiently grasp long-sequence reasoning tasks. Therefore, we employed a higher learning rate during our data selection process to facilitate the model's learning in complex reasoning scenarios. We conducted experiments using Qwen-2.5-32B and Qwen-2.5-72B models[53, 42].

### 4.2 Annealing

We performed an annealing training experiment starting from the optimal 32B model obtained previously. Unlike Stage I, we adopted a more stringent data selection strategy, retaining only the highly challenging data characterized by a coefficient of variation ($CV$) greater than 0.05 (see Section 3.2). In this stage, we discontinued the use of the packing strategy to allow the model to focus exclusively on handling complex long-reasoning tasks for individual query-response pairs, while still maintaining a maximum sequence length of 32k tokens. We set a lower learning rate of $8 \times 10^{-6}$ and trained the model for two epochs, further exploring the model's potential on challenging long-reasoning tasks. Similarly, we employed a cosine learning rate scheduler with a linear warmup phase for the first 5% of the training steps, gradually decaying the learning rate to zero.

### 4.3 Benchmarks

To rigorously evaluate the effectiveness of our approach, we conducted experiments on several representative benchmarks, including AIME2024[28], LiveCodeBench[12], and GPQA-Diamond[38]. Specifically, we utilized pass@1 for AIME2024, LiveCodeBench, and GPQA-Diamond, consistent with previous studies.

Among these benchmarks, AIME2024 is a challenging mathematical reasoning competition dataset comprising 30 integer-answer questions designed to assess precise mathematical reasoning; LiveCodeBench is a comprehensive, contamination-free coding benchmark, continuously aggregating new programming challenges from platforms such as LeetCode, AtCoder, and Codeforces; GPQA-Diamond is a subset of the GPQA dataset containing 198 high-difficulty graduate-level multiple-choice questions developed by experts in biology, physics, and chemistry to evaluate advanced scientific reasoning.

## 4.4 Results and Analysis

### 4.4.1 Overall Results

Table 1: Performance comparison of various models.

| Model | AIME2024 (%) | GPQA-Diamond (%) | LiveCodeBench (%) |
|---|---|---|---|
| DS-Distill-32B (SFT) | 72.6 | 62.1 | 57.2 |
| QwQ-32B (RL) | 79.5 | 65.9 | 63.4 |
| DS-Distill-70B (SFT) | 70.0 | 65.2 | 57.5 |
| DeepSeek-R1 (RL) | 79.8 | 71.5 | 65.9 |
| Ours-Distill-32B (SFT) | 75.8 | 66.3 | 64.2 |
| Ours-Distill-72B (SFT) | 79.2 | 65.7 | 63.8 |

Table 1 presents our experimental results. On AIME2024, our 72B model achieved a pass rate of 79.2%, and our 32B model achieved 75.8%, both significantly outperforming DeepSeek-Distill-32B (72.6%) and DeepSeek-Distill-70B (70.0%). Notably, despite relying solely on supervised fine-tuning (SFT), our 72B model closely approaches reinforcement learning-based models such as QwQ-32B (79.5%) and DeepSeek-R1 (79.8%). Additionally, our 32B model demonstrates superior performance over DeepSeek-Distill models on other evaluation benchmarks, achieving 66.3% on GPQA-Diamond and 64.2% on LiveCodeBench, indicating robust generalization and enhanced reasoning capabilities.

### 4.4.2 Analysis

Table 2: Model performance of two stage SFT models.

| Model | AIME2024 (%) | GPQA-Diamond (%) | LiveCodeBench (%) |
|---|---|---|---|
| Ours-32B (Stage I) | 75.8 | 66.3 | 64.2 |
| Ours-32B (Stage II) | 77.9 | 66.9 | 61.7 |

Comparisons between large scale reasoning training (32B Training Stage I) and annealing (32B Training Stage I) (Table 2) indicate that our two-stage fine-tuning strategy improves performance on several key benchmarks. Specifically, on AIME2024, our 32B model improved from 75.8 (Stage I) to 77.9 (Stage II), an increase of 2.1 percentage points, and GPQA-Diamond improved from 66.3 to 66.9. However, performance on LiveCodeBench slightly decreased from 64.2 (Stage I) to 61.7 (Stage II), indicating that further hyperparameter tuning and exploration of the optimal mixing ratio for code-related data are necessary to consistently benefit from the two-stage training across all tasks.

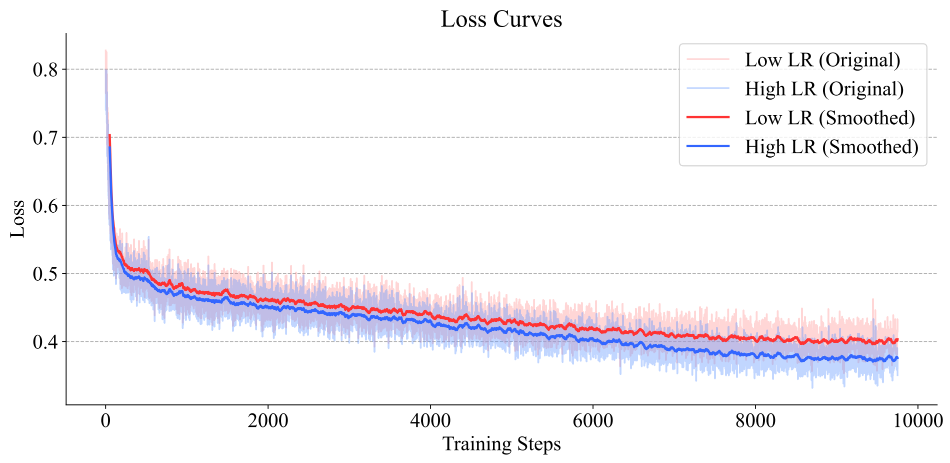


Figure 4: Loss curves of 72B model training.

We observed a training pattern shift when conducting reasoning-enhancement training based on the base model, compared to traditional post-training approaches. Specifically, reasoning-focused fine-tuning required a higher learning rate to adequately capture complex reasoning features present in the training data. To quantitatively illustrate this phenomenon, we conducted an ablation study using our 72B model. When reducing the learning rate from $8 \times 10^{-5}$ to $8 \times 10^{-6}$, we observed significant performance degradation across evaluation benchmarks. On the AIME2024 dataset, performance dropped by 6.7 percentage points (from 79.2% to 72.5%), while on LiveCodeBench, it declined by 3.6 percentage points (from 63.8% to 60.2%). As depicted in Figure 4, the training loss curves clearly demonstrate that a higher learning rate allows the model to better fit complex reasoning data, highlighting the necessity of adjusting training strategies for enhanced reasoning capabilities.

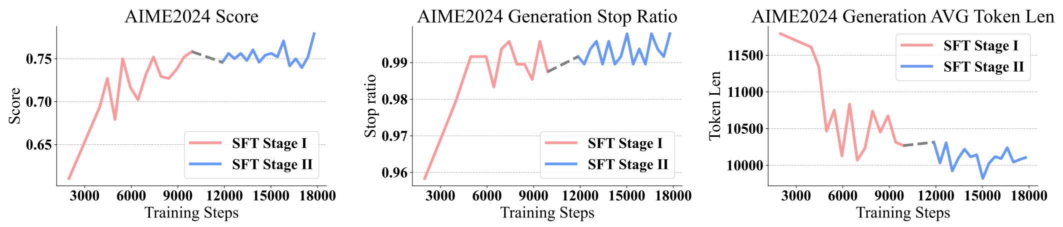


Figure 5: The variations of 32B AIME2024 Score, Generation Stop Ratio, and Average Generated Token Lengths with training steps

Additionally, we analyzed the 32B model's performance metrics on AIME2024 as training steps increased (Figure 5). In Stage I, due to the larger dataset and higher learning rate, the model quickly learned and rapidly improved in performance. In Stage II, we employed a lower learning rate and omitted the packing strategy, enabling more refined learning and leading to further improvements in the final performance. Moreover, since foundational models are typically trained on vast amounts of text data rather than QA-specific tasks, initially the generation stop ratio was relatively low, indicating frequent failures in stopping generated outputs. However, with continuous training, the stop ratio significantly improved, and the average generated token length stabilized, demonstrating effective adaptation to the QA format.

Our difficulty-aware training approach substantially enhanced foundational model performance on complex reasoning tasks, closely approaching state-of-the-art proprietary model levels. This highlights the significant potential of using Supervised Fine-Tuning (SFT) alone on base models, demonstrating that applying suitable data and optimized training strategies can yield highly competitive results.

## 5 Conclusion and Limitations

In this paper, we introduce a large-scale, difficulty-graded reasoning dataset containing approximately 3.34 million unique queries with 40 million model-generated responses spanning various capability gradients. Leveraging this dataset, we conduct large-scale reasoning-focused training based on pass rates and the Coefficient of Variation (CV) to identify and utilize the most valuable training data. Additionally, we observed a training pattern shift, indicating that higher learning rates are necessary for effectively fitting complex reasoning tasks.

Moreover, preliminary experiments revealed that further increasing data complexity through an annealed second-stage training could enhance model performance. Starting from a base model, our reasoning-focused fine-tuning method surpasses most open-source distilled models on several complex reasoning benchmarks, such as AIME2024, achieving performance close to or at state-of-the-art levels. These results validate the effectiveness and generalization capability of both our dataset and training strategies.

In future work, we aim to develop more refined methods for evaluating data quality to better identify data most beneficial for model training. Additionally, we will investigate how models with varying initial capabilities influence subsequent reinforcement learning (RL) training outcomes, thus illuminating the interplay between foundational model capability and data quality. We hope that our open-sourced data and methodologies will facilitate further advancements in constructing open-source large language models with exceptional long-form reasoning capabilities, contributing significantly to the broader research community.

## References

[1] Alon Albalak, Duy Phung, Nathan Lile, Rafael Rafailov, Kanishk Gandhi, Louis Castricato, Anikait Singh, Chase Blagden, Violet Xiang, Dakota Mahan, and Nick Haber. Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models, 2025. URL https://arxiv.org/abs/2502.17387.

[2] Art of Problem Solving. 2023 AMC 8 Problems/Problem 10. https://artofproblemsolving.com/wiki/index.php/2023_AMC_8_Problems/Problem_10, 2023. Accessed: 2025-04-23.

[3] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation, 2024.

[4] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

[5] Di Zhang. Aime_1983_2024 (revision 6283828), 2025. URL https://huggingface.co/datasets/di-zhang-fdu/AIME_1983_2024.

[6] Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*, 2023.

[7] Hugging Face. Open r1: A fully open reproduction of deepseek-r1, January 2025. URL https://github.com/huggingface/open-r1.

[8] Bleys Goodson. Fine flan: Seqio to parquet so you don't have to. https://huggingface.co/datasets/Open-Orca/FLAN, 2023.

[9] Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. 2025. URL `https://arxiv.org/abs/2504.11456`.

[10] hivaze. Logic-701: A benchmark dataset for logical reasoning in english and russian. `https://huggingface.co/datasets/hivaze/LOGIC-701`, 2023. Hugging Face Dataset.

[11] Siming Huang, Tianhao Cheng, Jason Klein Liu, Jiaran Hao, Liuyihan Song, Yang Xu, J. Yang, J. H. Liu, Chenchen Zhang, Linzheng Chai, Ruifeng Yuan, Zhaoxiang Zhang, Jie Fu, Qian Liu, Ge Zhang, Zili Wang, Yuan Qi, Yinghui Xu, and Wei Chu. Opencoder: The open cookbook for top-tier code large language models. 2024. URL `https://arxiv.org/pdf/2411.04905`.

[12] Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

[13] Yunjie Ji, Sitong Zhao, Xiaoyu Tian, Haotian Wang, Shuaiting Chen, Yiping Peng, Han Zhao, and Xiangang Li. How difficulty-aware staged reinforcement learning enhances llms' reasoning capabilities: A preliminary experimental study. *arXiv preprint arXiv:2504.00829*, 2025.

[14] Parth Kadam. Ncert biology 11th dataset. `https://huggingface.co/datasets/KadamParth/NCERT_Biology_11th`, 2023. Accessed: 2024-04-23.

[15] Parth Kadam. Ncert biology 12th dataset. `https://huggingface.co/datasets/KadamParth/NCERT_Biology_12th`, 2023. Accessed: 2024-04-23.

[16] Parth Kadam. Ncert chemistry 11th dataset. `https://huggingface.co/datasets/KadamParth/NCERT_chemistry_11th`, 2023. Accessed: 2024-04-23.

[17] Parth Kadam. Ncert chemistry 12th dataset. `https://huggingface.co/datasets/KadamParth/NCERT_chemistry_12th`, 2023. Accessed: 2024-04-23.

[18] Parth Kadam. Ncert physics 11th dataset. `https://huggingface.co/datasets/KadamParth/NCERT_Physics_11th`, 2023. Accessed: 2024-04-23.

[19] Parth Kadam. Ncert physics 12th dataset. `https://huggingface.co/datasets/KadamParth/NCERT_Physics_12th`, 2023. Accessed: 2024-04-23.

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

[21] Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. Numinamath. `https://huggingface.co/AI-MO/NuminaMath-CoT`, 2024.

[22] Min Li and RLHFlow Team. Decision-tree-reward-llama-3.1-8b. `https://huggingface.co/RLHFlow/Decision-Tree-Reward-Llama-3.1-8B`, 2025. URL `https://huggingface.co/RLHFlow/Decision-Tree-Reward-Llama-3.1-8B`. Interpreting Language Model Preferences Through the Lens of Decision Trees.

[23] Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. From quantity to quality: Boosting llm performance with self-guided data selection for instruction tuning. *arXiv preprint arXiv:2308.12032*, 2023.

[24] Wing Lian, Bleys Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". Openorca: An open dataset of gpt augmented flan reasoning traces. `https://https://huggingface.co/datasets/Open-Orca/OpenOrca`, 2023.

[25] longface. logiclm. https://huggingface.co/datasets/longface/logicLM, 2025. Accessed: 2025-04-22.

[26] Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V. Le, Barret Zoph, Jason Wei, and Adam Roberts. The flan collection: Designing data and methods for effective instruction tuning, 2023.

[27] Michael Luo, Sijun Tan, Roy Huang, Ameen Patel, Alpay Ariyak, Qingyang Wu, Xiaoxiang Shi, Rachel Xin, Colin Cai, Maurice Weber, Ce Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepcoder: A fully open-source 14b coder at o3-mini level. https://pretty-radio-b75.notion.site/, 2025. Notion Blog.

[28] MAA. American invitational mathematics examination - aime. https://maa.org/math-competitions/american-invitational-mathematics-examination-aime, feb 2024. Accessed in February 2024, from American Invitational Mathematics Examination - AIME 2024.

[29] Microsoft. Chemistry-qa. https://github.com/microsoft/chemistry-qa, 2021. [GitHub repository].

[30] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, 2025. URL https://arxiv.org/abs/2501.19393.

[31] NVIDIA. Llama-nemotron-post-training-dataset. https://huggingface.co/datasets/nvidia/Llama-Nemotron-Post-Training-Dataset, 2025. Version 1.1, released on April 8, 2025.

[32] NVIDIA. Llama-3_1-nemotron-ultra-253b-v1. https://huggingface.co/nvidia/Llama-3_1-Nemotron-Ultra-253B-v1, 2025. Released on 2025-04-07 under the NVIDIA Open Model License.

[33] Beijing Academy of Artificial Intelligence (BAAI). Infinity instruct. *arXiv preprint arXiv:2406.XXXX*, 2024.

[34] Open R1. Verifiable coding problems (python). https://huggingface.co/datasets/open-r1/verifiable-coding-problems-python, 2025. Hugging Face Dataset.

[35] OpenAI. Learning to reason with llms, 2024. URL https://openai.com/index/learning-to-reason-with-llms/.

[36] Guilherme Penedo, Anton Lozhkov, Hynek Kydlíček, Loubna Ben Allal, Edward Beeching, Agustín Piqueres Lajarín, Quentin Gallouédec, Nathan Habib, Lewis Tunstall, and Leandro von Werra. Codeforces cots. https://huggingface.co/datasets/open-r1/codeforces-cots, 2025.

[37] 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. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

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

[39] Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Tali Bers, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. Multitask prompted training enables zero-shot task generalization, 2022.

[40] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[41] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters, 2024. URL `https://arxiv.org/abs/2408.03314`.

[42] Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL `https://qwenlm.github.io/blog/qwen2.5/`.

[43] WizardLM Team. Wizardlm evol-instruct 70k dataset. `https://huggingface.co/datasets/WizardLMTeam/WizardLM_evol_instruct_70k`, 2023. Accessed: 2025-04-23.

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

[45] Xiaoyu Tian, Sitong Zhao, Haotian Wang, Shuaiting Chen, Yunjie Ji, Yiping Peng, Han Zhao, and Xiangang Li. Think twice: Enhancing llm reasoning by scaling multi-round test-time thinking. *arXiv preprint arXiv:2503.19855*, 2025.

[46] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Maitreya Patel, Kuntal Kumar Pal, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Shailaja Keyur Sampat, Savan Doshi, Siddhartha Mishra, Sujan Reddy, Sumanta Patro, Tanay Dixit, Xudong Shen, Chitta Baral, Yejin Choi, Noah A. Smith, Hannaneh Hajishirzi, and Daniel Khashabi. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks, 2022. URL `https://arxiv.org/abs/2204.07705`.

[47] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Maitreya Patel, Kuntal Kumar Pal, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Shailaja Keyur Sampat, Savan Doshi, Siddhartha Mishra, Sujan Reddy, Sumanta Patro, Tanay Dixit, Xudong Shen, Chitta Baral, Yejin Choi, Noah A. Smith, Hannaneh Hajishirzi, and Daniel Khashabi. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks, 2022.

[48] Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners, 2022.

[49] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL `https://arxiv.org/abs/2201.11903`.

[50] Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, et al. Light-r1: Curriculum sft, dpo and rl for long cot from scratch and beyond. *arXiv preprint arXiv:2503.10460*, 2025.

[51] Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models, 2025. URL `https://arxiv.org/abs/2408.00724`.

[52] Zhangchen Xu, Yang Liu, Yueqin Yin, Mingyuan Zhou, and Radha Poovendran. Kodcode: A diverse, challenging, and verifiable synthetic dataset for coding. 2025. URL `https://arxiv.org/abs/2503.02951`.

[53] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou,

Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

[54] Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023.

[55] Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. Free process rewards without process labels. *arXiv preprint arXiv:2412.01981*, 2024.

[56] Weizhe Yuan, Jane Yu, Song Jiang, Karthik Padthe, Yang Li, Dong Wang, Ilia Kulikov, Kyunghyun Cho, Yuandong Tian, Jason E Weston, and Xian Li. Naturalreasoning: Reasoning in the wild with 2.8m challenging questions, 2025. URL `https://arxiv.org/abs/2502.13124`.

[57] Huaye Zeng, Dongfu Jiang, Haozhe Wang, Ping Nie, Xiaotong Chen, and Wenhu Chen. Acecoder: Acing coder rl via automated test-case synthesis. *ArXiv*, abs/2207.01780, 2025.

[58] Bo-Wen Zhang, Yan Yan, Lin Li, and Guang Liu. Infinitymath: A scalable instruction tuning dataset in programmatic mathematical reasoning, 2024. URL `https://arxiv.org/abs/2408.07089`.

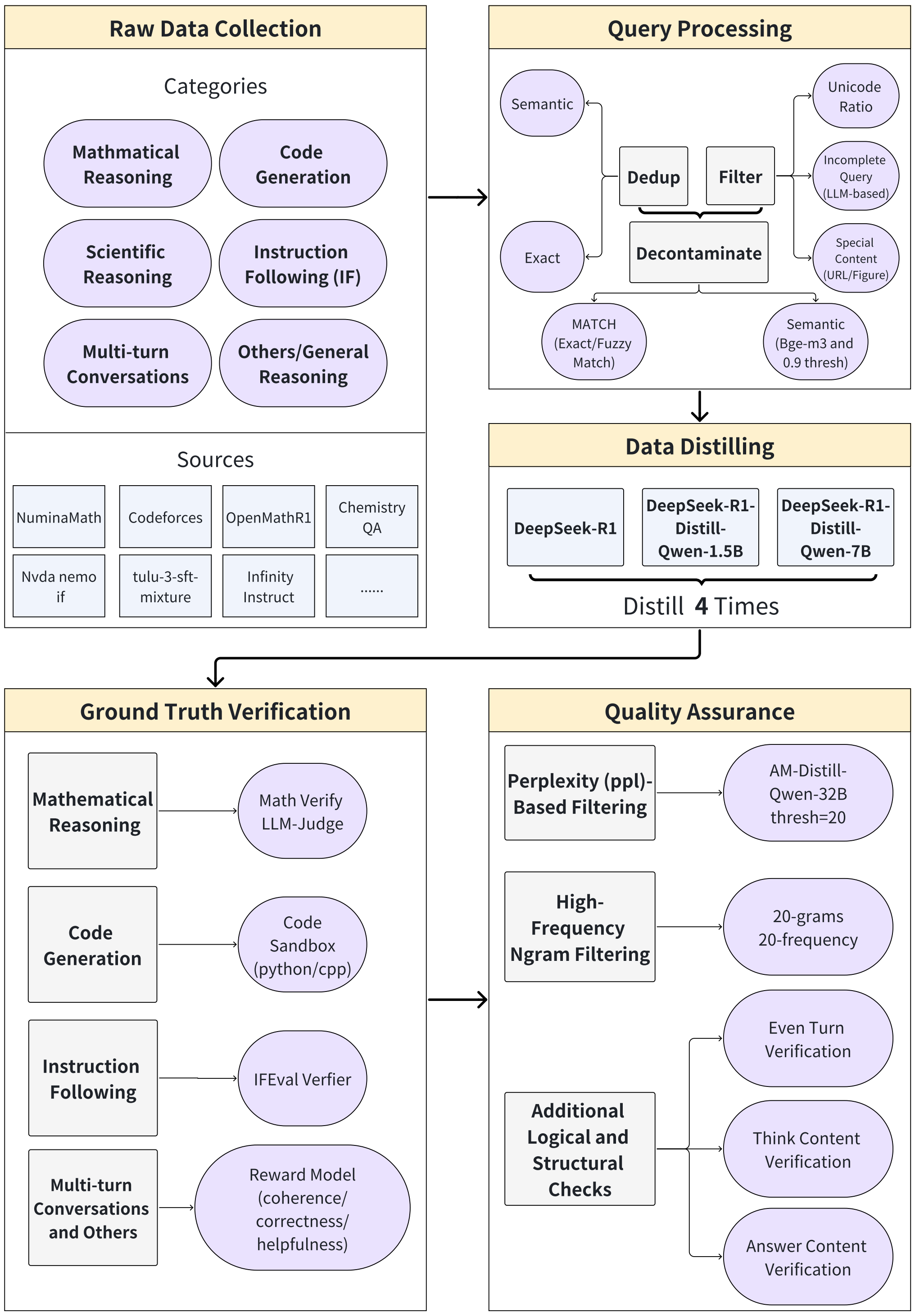
[59] Jia Zhang, Chen-Xi Zhang, Yao Liu, Yi-Xuan Jin, Xiao-Wen Yang, Bo Zheng, Yi Liu, and Lan-Zhe Guo. D3: Diversity, difficulty, and dependability-aware data selection for sample-efficient llm instruction tuning. *arXiv preprint arXiv:2503.11441*, 2025.

[60] Han Zhao, Haotian Wang, Yiping Peng, Sitong Zhao, Xiaoyu Tian, Shuaiting Chen, Yunjie Ji, and Xiangang Li. 1.4 million open-source distilled reasoning dataset to empower large language model training. *arXiv preprint arXiv:2503.19633*, 2025.

[61] Hanyu Zhao, Li Du, Yiming Ju, Chengwei Wu, and Tengfei Pan. Beyond iid: Optimizing instruction learning from the perspective of instruction interaction and dependency. 2024. URL `https://arxiv.org/abs/2409.07045`.

## A Data Analysis

### A.1 Data Processing Pipeline



Raw Data Collection
Categories
Mathmatical Reasoning
Code Generation
Scientific Reasoning
Instruction Following (IF)
Multi-turn Conversations
Others/General Reasoning
Sources
NuminaMath
Codeforces
OpenMathR1
Chemistry QA
Nvda nemo if
tulu-3-sft-mixture
Infinity Instruct
......
Query Processing
Semantic
Exact
Dedup
Filter
Decontaminate
Unicode Ratio
Incomplete Query (LLM-based)
Special Content (URL/Figure)
MATCH (Exact/Fuzzy Match)
Semantic (Bge-m3 and 0.9 thresh)
Data Distilling
DeepSeek-R1
DeepSeek-R1-Distill-Qwen-1.5B
DeepSeek-R1-Distill-Qwen-7B
Distill 4 Times
Ground Truth Verification
Mathematical Reasoning
Math Verify LLM-Judge
Code Generation
Code Sandbox (python/cpp)
Instruction Following
IFEval Verfier
Multi-turn Conversations and Others
Reward Model (coherence/ correctness/ helpfulness)
Quality Assurance
Perplexity (ppl)-Based Filtering
AM-Distill-Qwen-32B thresh=20
High-Frequency Ngram Filtering
20-grams 20-frequency
Additional Logical and Structural Checks
Even Turn Verification
Think Content Verification
Answer Content Verification

Figure 6: Construction process of data pipeline.

## A.2 Category Distribution

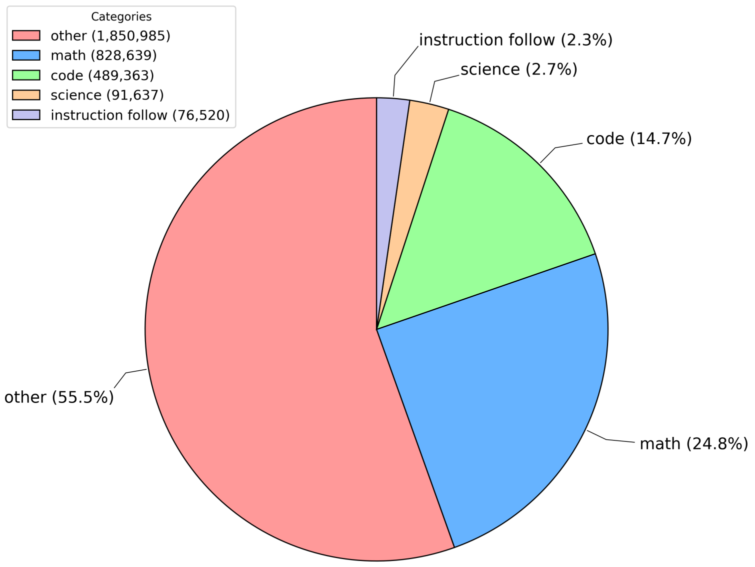


Figure 7: Distribution of query types in the dataset. Math queries constitute 24.8%, code 14.7%, science 2.7%, instruction following 2.3%, and other 55.5%.

## A.3 Pass Rate Distribution

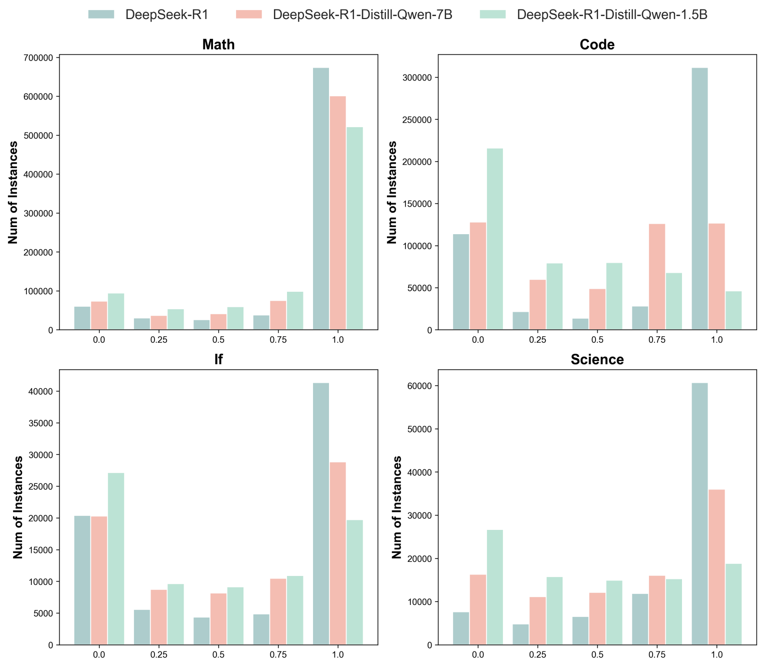


Figure 8: Pass rate distributions of the three models across four categories of data: math, code, science, and instruction following.