Are Many Tasks Necessary for General LLM Evaluation? A Matrix Completion View of Benchmark Redundancy

Kohei Tsuji **Yusuke Sakai** **Hidetaka Kamigaito** **Taro Watanabe**
Nara Institute of Science and Technology (NAIST), Japan
tsuji.kohei.tl1@naist.ac.jp
{sakai.yusuke.sr9, kamigaito.h, taro}@is.naist.jp

Abstract

Large language model (LLM) evaluation usually averages scores across many benchmark tasks, implicitly assuming broad task coverage is necessary for reliable general evaluation. This practice also implicitly treats all benchmark tasks as equally important for general evaluation. We revisit this assumption by treating benchmark results as a model-task score matrix under the performance prediction task. We find that a broad range of benchmarks are redundant with scores for 100 LLMs on 13 aggregate tasks and 523 subtasks. Specifically, we recovered model rankings with high rank correlation with the ALS matrix completion method after masking out several task scores. This means that no tasks dominated the recovery process, and both low-scoring and high-scoring tasks shared the same trends. Further analyses show that multiple-choice scores allow us to recover model rankings on free-form generation tasks. For general model ranking, free-form generation evaluation is not necessarily more informative than multiple-choice evaluation. These findings suggest that evaluating general-purpose LLMs can be accomplished with fewer tasks than the current common practice. Our analysis also estimates the number of models required to capture the overall trends in the benchmarks.

1 Introduction

Multi-task evaluation is a standard practice for comparing large language models (LLMs) (Hendrycks et al., 2021a; Srivastava et al., 2023; Liang et al., 2023; Qian et al., 2026). A leaderboard rarely relies on a single dataset; instead, it aggregates scores across diverse tasks, including knowledge (Clark et al., 2018), reasoning (Liu et al., 2020; Yang et al., 2018), mathematics (Hendrycks et al., 2021b; Shi et al., 2023; Cobbe et al., 2021), multilingual understanding (Xuan et al., 2025; Shi et al., 2023), instruction following (Zhou et al., 2023), and code generation (Chen et al., 2021; Jimenez et al., 2024). Although averaging performance across diverse tasks provides a convenient summary metric, it remains unclear whether evaluating all, or even most, of these tasks is necessary to characterize overall model capability. While individual benchmarks are valuable for assessing specific abilities of interest to benchmark designers (Hendrycks et al., 2021b; Jin et al., 2021; Zhou et al., 2023), we investigate a complementary question: ***how many tasks are required to reliably approximate a model's overall ranking or aggregate performance within a broad benchmark suite?*** Here, overall model capability denotes the aggregated model ranking or ordering induced by the broad benchmark suites.

Motivated by this fundamental question, we investigate the following four research questions (RQs):

RQ1. *Can scores on a subset of tasks accurately predict scores on the remaining tasks and preserve the overall model ranking?* We hypothesize that broad benchmark suites embody a latent performance trend shared across tasks. If so, even a small subset of tasks should be sufficient to recover the overall ranking.

RQ2. *Does the recoverability of the overall ranking depend on task importance, difficulty, or answer format?* We hypothesize that ranking recovery is largely invariant to these factors. If so, removing important or difficult tasks or restricting evaluation to particular answer formats should have little effect on the recovered ranking.

RQ3. *Does redundancy in broad benchmark suites imply that domain-specific evaluations are unnecessary?* We hypothesize that redundancy emerges primarily at the benchmark-suite level rather than between individual tasks. If so, domain-specific evaluations remain essential even when broad benchmark rankings can be recovered from a small subset of tasks.

RQ4. *When constructing a new benchmark,*

Proceedings of the 15th International Joint Conference on Natural Language Processing and the 5th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 1172–1187
November 6-10, 2026 ©2026 Association for Computational Linguistics

how many model evaluations are required to estimate its overall performance trend? We hypothesize that model performances also exhibit a low-dimensional structure. If so, evaluating only a limited number of models should be sufficient to estimate the benchmark's overall trend.

We verify these hypotheses through a performance prediction framework. Specifically, we mask a portion of a leaderboard table and predict the hidden scores from the observed ones. If benchmark tasks measure largely independent capabilities, recovering the masked entries should be difficult. Conversely, if broad benchmark suites share a common performance trend, both the hidden scores and the resulting model rankings should remain recoverable. Because this framework allows flexible control over which tasks and models are masked, it provides a convenient way to investigate our research questions.

To this end, we represent benchmark results as a model-task matrix, where rows correspond to LLMs and columns correspond to evaluation tasks. We mask a subset of matrix entries, reconstruct them using matrix completion, and compare the recovered matrix with the original one. We employ Alternating Least Squares (ALS) (Hu et al., 2008) for matrix completion. Our experiments use evaluation results from 100 models across 13 aggregate tasks and 523 subtasks. Our experiments yield the following findings:

- ALS almost perfectly recovers model rankings, suggesting that broad benchmark suites contain highly redundant evaluation trends. (RQ1)
- Recovery remains effective even when masking important tasks, difficult tasks, or free-form generation tasks, indicating that recoverability is largely independent of these factors. (RQ2)
- Correlations between individual task scores are substantially lower than those between the original and recovered benchmark rankings. This suggests that domain-specific evaluations remain necessary despite the redundancy of broad benchmark suites. (RQ3)
- Benchmark-level trends can also be recovered from a small subset of evaluated models, suggesting that comprehensive evaluation across many models is not always necessary to estimate a benchmark's overall trend. (RQ4)

2 Related Work

Broad LLM evaluation. MMLU (Hendrycks et al., 2021a) introduced a broad academic and professional benchmark for measuring multi-task language understanding. More recent benchmark variants, such as MMLU-Pro (Wang et al., 2024) and MMLU-ProX (Xuan et al., 2025), increase the difficulty and multilingual coverage. While the researchers designed these tasks to evaluate the performance of general-purpose LLMs, some datasets evaluate performance in specific fields such as medicine (Jin et al., 2021), mathematics (Hendrycks et al., 2021b; Cobbe et al., 2021; Shi et al., 2023), and instruction following (Zhou et al., 2023).

Benchmarks. Benchmark^2 (Qian et al., 2026) assesses benchmark quality using ranking consistency, discriminability, and capability-alignment deviation. Zhang et al. (2025b) use correlations to examine redundancy across instances, capability dimensions, and benchmarks. These studies provide quantitative perspectives on benchmark quality and redundancy. However, redundancy in benchmark evaluation can arise in several forms. A benchmark may be redundant because its overall ranking can be recovered from only a subset of tasks, because a small number of tasks dominate the aggregate score, or because different tasks measure highly overlapping capabilities. We focus on recoverability and define redundancy as overlap in information about model rankings. Prior work tests agreement across benchmarks (Perlitz et al., 2025), predicts scores from training signals or other task results (Ruan et al., 2024; Owen, 2024; Gadre et al., 2025; Chen et al., 2025), or completes unknown cells in model-task matrices (Zhang et al., 2024a). Research on efficient evaluation reduces evaluation sets or estimates scores from selected items (Perlitz et al., 2024; Maia Polo et al., 2024; Zhang et al., 2025a), while other analyses use compression or interpretable demand scales and report limits that depend on tasks and experimental settings (Huang et al., 2024; Zhou et al., 2025; Lourie et al., 2025). In contrast, we use score prediction to test whether tasks and models contribute distinct information to a broad benchmark.

3 Method

We address these research questions by examining whether task scores can be predicted from scores on

other tasks. To this end, we represent benchmark results as a model-task score matrix, where rows correspond to models and columns correspond to evaluation tasks. If the matrix can be accurately reconstructed after a large fraction of entries is masked, the benchmark contains substantial redundancy in its overall evaluation trend.

Matrix completion methods estimate missing entries from observed ones by exploiting low-rank structure in the data (Candès and Recht, 2009; Koren et al., 2009). We employ Alternating Least Squares (ALS) (Hu et al., 2008), a widely used matrix factorization approach, to reconstruct masked scores. ALS has previously been applied to predicting human judgments and to other incomplete evaluation settings (Wisniewski, 2013). By applying ALS to benchmark result matrices, we investigate how much information is shared across tasks. High reconstruction accuracy indicates redundancy in benchmark evaluations, whereas tasks whose scores are difficult to recover contain unique information. Therefore, these tasks may be more important for characterizing model capabilities.

Let $\mathbf{M} \in \mathbb{R}^{n \times d}$ denote the original score matrix for n models and d tasks. For a masked cell ratio x, we choose a hidden-cell set H_x and treat every cell outside H_x as observed. This gives a partially observed matrix $\tilde{\mathbf{M}}$, where $\tilde{\mathbf{M}}_{ij} = \mathbf{M}_{ij}$ for $(i,j) \notin H_x$ and $\tilde{\mathbf{M}}_{ij}$ is missing for $(i,j) \in H_x$. ALS fits latent factors for models and tasks from the observed cells. Then, we recover the hidden cells in $\tilde{\mathbf{M}}$ and obtain a completed matrix $\mathbf{M}'$. The central question is whether $\mathbf{M}'$ preserves the same score values and model ranking trends as $\mathbf{M}$. Appendix A gives the objective and hyperparameters used in our experiments. After recovering $\mathbf{M}'$, we compare it with $\mathbf{M}$ in two ways. First, we compute Mean Absolute Error (MAE) only on the masked cells to examine how exactly ALS can recover the scores as follows:

$$\mathrm{MAE}(\mathbf{M}, \mathbf{M}'; H_x) = \frac{1}{|H_x|} \sum_{(i,j) \in H_x} |\mathbf{M}_{ij} - \mathbf{M}'_{ij}|. \tag{1}$$

Second, we compute each model's average score in $\mathbf{M}$ and $\mathbf{M}'$ and compare the resulting model rankings with Spearman's rank correlation coefficient. Let $\bar{m}_i = \frac{1}{d}\sum_{j=1}^{d} \mathbf{M}_{ij}$, $\bar{m}'_i = \frac{1}{d}\sum_{j=1}^{d} \mathbf{M}'_{ij}$ donate the averaged scores and $r_i = \mathrm{rank}(\bar{m}_i)$ and $r'_i = \mathrm{rank}(\bar{m}'_i)$ donate the ranks of model i. We define the rank correlation as

$$\rho(\mathbf{M}, \mathbf{M}'; x) = \frac{\sum_{i=1}^{n}(r_i - \bar{r})(r'_i - \bar{r}')}{Z_r Z_{r'}}, \tag{2}$$

$$Z_r = \sqrt{\sum_{i=1}^{n}(r_i - \bar{r})^2}, \quad Z_{r'} = \sqrt{\sum_{i=1}^{n}(r'_i - \bar{r}')^2}. \tag{3}$$

Here, $\bar{r} = n^{-1}\sum_i r_i$ and $\bar{r}' = n^{-1}\sum_i r'_i$. This rank-based measure directly tests whether matrix completion preserves the general evaluation trend.

4 Experimental Setup

Score matrices. We use two matrices of the same evaluation table. The aggregate matrix contains 100 models and 13 task columns, while the subtask-level matrix contains the same models and 523 subtask columns. The aggregate tasks include MMLU (Hendrycks et al., 2021a), MMLU-Pro (Wang et al., 2024), MMLU-ProX-Lite (Xuan et al., 2025), ARC-Challenge (Clark et al., 2018), HellaSwag (Zellers et al., 2019), GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021b), MGSM (Shi et al., 2023), IFEval (Zhou et al., 2023), MedQA (Jin et al., 2021), PIQA (Bisk et al., 2020), TruthfulQA (Lin et al., 2022), and AGIEval (Zhong et al., 2024). These two matrices let us ask whether a pattern that appears at the benchmark level also survives finer task decompositions. We describe the full information for each dataset in Appendix E.

Models. The model set contains 100 LLMs from 23 developers. We choose models to cover a broad release period from 2023 to 2026 and a wide size range from 0.5B to 72B total parameters. The set also covers models developed or post-trained across different countries, regions, and language communities. Beyond general instruction-tuned models, it includes models post-trained for mathematics, coding, reasoning, and multilingual use. We use Qwen (Qwen et al., 2025; Yang et al., 2025), Llama (Touvron et al., 2023; Dubey et al., 2024), Gemma (Team et al., 2024; **?**), Phi (Abdin et al., 2024a,b), OLMo (OLMo et al., 2024; Ettinger et al., 2025), DeepSeek (DeepSeek-AI, 2025), GLM (GLM et al., 2024), InternLM (Cai et al., 2024), Granite (Granite Team, 2024), EXAONE (Bae et al., 2026), Aya (Dang et al., 2024), Yi (AI et al., 2024),

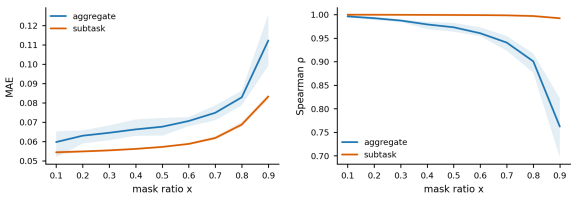


Figure 1: ALS recovery as we increase the masked cell ratio x on the matrices with 13 tasks and 523 subtasks. The shade represents variance.

Case	x	MAE	Rank ρ
With ProX-Lite	0.1	0.054	1.000
With ProX-Lite	0.5	0.057	0.999
With ProX-Lite	0.8	0.069	0.997
No ProX-Lite	0.1	0.040	1.000
No ProX-Lite	0.5	0.044	0.998
No ProX-Lite	0.8	0.058	0.992

Table 1: Recovery comparison with and without MMLU-ProX-Lite in the subtask-level matrix. x denotes the masked cell ratio, and Rank ρ denotes Spearman correlation between true and recovered rankings.

SOLAR (Kim et al., 2024), LLM-jp [1], Swallow (Fujii et al., 2024; Ma et al., 2025), GPT-OSS-derived models (OpenAI, 2025), Command R (Cohere et al., 2025), SeaLLMs (Zhang et al., 2024b), Falcon (Team, 2024), Seed-Coder (ByteDance Seed et al., 2025), Nemotron (Bercovich et al., 2025), OpenThinker (Guha et al., 2025), and Dolphin fine-tuned models[2]. Appendix F provides detailed information about all models.

Inference. We conduct all evaluations using the EleutherAI Language Model Evaluation Harness (Gao et al., 2024). Models are loaded in bfloat16 precision on a single RTX PRO 6000. For generation tasks, we use deterministic decoding ($T = 0$) when generation is required by the task configuration, with a maximum generation length of 1,024 tokens. For reasoning models, we enable their thinking mode. Because deterministic decoding often leads to repetitive reasoning traces, we use $T = 0.6$, $\mathrm{Top}P = 0.95$, $\mathrm{Top}K = 20$, and $\mathrm{Min}P = 0$. Unless otherwise specified, all remaining evaluation settings, including the number of in-context examples, follow the default configuration of each task. All reported scores are normalized to the range $[0, 1]$, where higher values indicate better performance.

5 Matrix Completion Experiments (RQ1)

5.1 Recovery under Random Masking

We investigate whether the average ranking of models can be recovered when a large fraction of benchmark scores is hidden. If benchmark tasks capture largely independent information, recovery should deteriorate rapidly as more entries are masked. We apply the matrix completion procedure described in Section 3 to both the aggregate-task and subtask matrices. For masking ratios $x \in [0, 1]$, we randomly mask matrix entries and reconstruct them using ALS. Results are averaged over 20 random seeds. A model report or leaderboard rarely depends on a single dataset. This approach assumes that different datasets measure different capabilities (Hendrycks et al., 2021a; Srivastava et al., 2023; Liang et al., 2023).

Figure 1 shows that ALS reconstructs benchmark results remarkably well, even under substantial masking. Up to $x = 0.8$, both matrices exhibit low reconstruction error and high rank correlation with the original rankings. The subtask matrix is particularly robust, while the aggregate-task matrix degrades only under the most extreme masking conditions. These results suggest that benchmark tasks share a strong common performance signal. Consequently, benchmark-wide rankings remain stable not because each task contributes independent information, but because many tasks capture overlapping aspects of model performance. To verify that this effect is not dominated by MMLU-ProX-Lite, we repeat the analysis after removing all of its subtasks as shown in Table 1. The results remain nearly unchanged, demonstrating that the observed redundancy is a property of the benchmark suite as a whole rather than a consequence of a single benchmark. Appendix B reports recovery with different formulas, and the results show consistent trends.

5.2 Recovery for Individual Tasks

At $x = 0.8$, we evaluate recovery separately for all 13 aggregate tasks and 523 subtasks using the settings in Section 5.1. Figure 2 shows relatively large errors for GSM8K, IFEval, MATH, and MGSM, while most subtasks retain similar model rankings

[1] https://huggingface.co/collections/llm-jp/llm-jp-3-fine-tuned-models, https://huggingface.co/collections/llm-jp/llm-jp-31-fine-tuned-models, and https://huggingface.co/collections/llm-jp/llm-jp-4-models

[2] https://huggingface.co/collections/QuixiAI/dolphin-30

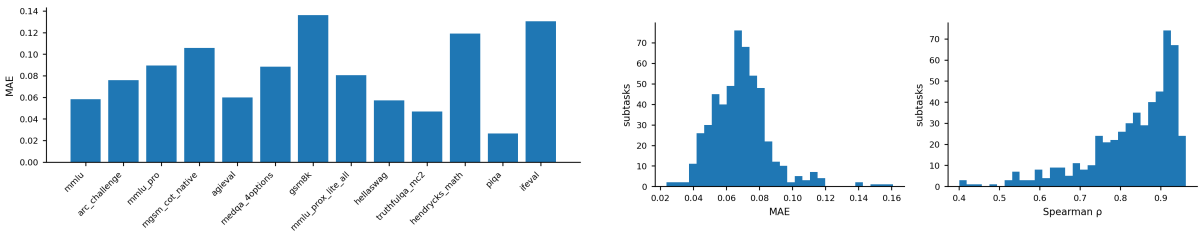


Figure 2: Recovery at $x = 0.8$ for individual aggregate tasks and the complete subtask distribution. The left image reports MAE for the aggregate tasks, and the right image reports the distributions of MAE and Spearman's ρ over 523 subtasks.

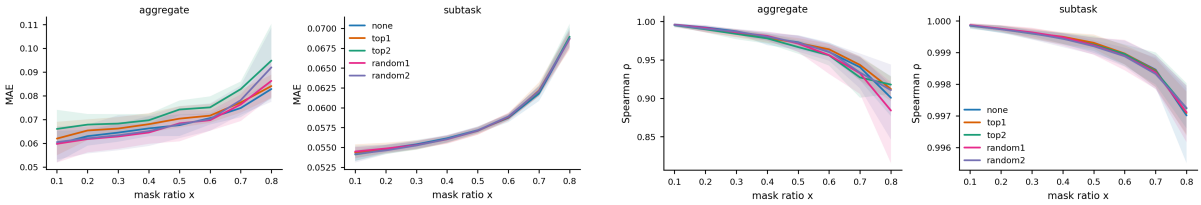


Figure 3: Recovery when we always observe selected task columns.

Model population	MAE	Rank ρ
Models with 7-8B parameters	0.091	0.847
Release year 2024	0.085	0.880
Release year 2025	0.097	0.864
Qwen family	0.096	0.870

Table 2: Recovery on the aggregate matrix at $x = 0.8$ for model populations restricted by size, release year, or family. Rank ρ denotes Spearman correlation between the original and recovered model rankings.

Matrix	Comp.	Singular value	Share
13 tasks	1	29.95	0.328
13 tasks	2	10.81	0.118
523 subtasks	1	182.30	0.144
523 subtasks	2	60.64	0.048

Table 3: Top singular values before the anchor-mask experiments. Share denotes each singular value divided by the sum of all singular values.

after recovery. Although the accuracy of individual score predictions varies, recoverability extends across many tasks.

5.3 Recovery across Model Populations

Recovery across a diverse model population may partly reflect large performance differences between model sizes, release years, or families. We examine whether task scores remain recoverable when we restrict the model population along each of these dimensions.

We construct separate aggregate matrices for models in the 7-8B range, models released in 2024, models released in 2025, and models from the Qwen family. Each matrix retains all 13 task columns and includes only the models in the corresponding group. We fit ALS separately to each matrix at $x = 0.8$ over 20 seeds, using the same settings as in Section 5.1.

Table 2 shows low reconstruction error and high rank correlation across these groups, consistent with the recovery observed for the full model population. Task scores provide overlapping information about model rankings even among models of similar size, from the same release year, or within the same family. Therefore, the redundancy observed in the full matrix also appears within more narrowly defined model populations.

Overall, the consistent results across these experiments show that redundancy in broad benchmark evaluations extends across individual tasks and persists within different model populations.

Finding 1. *Broad benchmark evaluations are highly redundant. Even after masking most task scores, matrix completion accurately reconstructs the overall ranking of models.*

6 Additional Analyses (RQ2)

6.1 Recovery from Anchor Task

This experiment addresses **RQ2**, focusing on task importance. Even if benchmark-wide rankings are highly recoverable, a small number of tasks may

Held-out task	Task mean	Rank ρ
MMLU	0.625	0.916
ARC-Challenge	0.775	0.899
MMLU-Pro	0.441	0.883
AGIEval	0.433	0.867
MGSM	0.452	0.851
MedQA	0.530	0.839
MMLU-ProX-Lite	0.253	0.824
PIQA	0.779	0.748
IFEval	0.660	0.450

Table 4: Leave-one-task ranking correlation. Rank ρ denotes Spearman correlation between the ranking of averaged scores excluding the Held-out task and the ranking of the Held-out task alone.

still act as anchor tasks that carry most of the information needed for recovery. If this is the case, preserving those tasks should substantially improve matrix reconstruction. To identify candidate anchor tasks, we estimate task importance using singular value decomposition (SVD). Table 3 shows that the first singular value is substantially larger than the second in both matrices. We represent each task as a vector of model scores and measure its contribution to the dominant singular components of the model-task matrix. Tasks that contribute more strongly to the leading components are considered more important because they capture a larger share of the shared performance structure. Here, importance denotes a task's potential contribution to recovering the shared benchmark trend.

We compare three masking strategies. The none setting uses the same random masking procedure as in Section 5. The top-k setting uses $k = 1$ or $k = 2$ most important tasks fully observed for all models. The random-k setting uses $k = 1$ or $k = 2$ randomly selected tasks fully observed. We use the same masking ratios x as in Section 5, with $x = 0.8$ as the largest value. Larger ratios would create fully masked columns when two tasks are fixed, making ALS inapplicable. Figure 3 shows that preserving the most important tasks (i.e., top-1 and top-2) yields nearly identical recovery performance to both random masking (none) and random task preservation (random-1 and random-2). Even at $x = 0.8$, reconstruction error and rank correlation remain similar across all settings. The same pattern appears in the subtask-level matrix. These results suggest that benchmark-wide performance information is not concentrated in a small set of anchor tasks. Instead, the shared performance signal appears to be distributed broadly across many tasks.

Matrix	Target	Score Range	MAE	Rank ρ
Aggregate	Low	0.253–0.452	0.068	0.936
Aggregate	Middle	0.530–0.660	0.066	0.908
Aggregate	High	0.696–0.779	0.065	0.941
Subtask-level	Low	0.062–0.229	0.060	0.987
Subtask-level	Middle	0.229–0.384	0.068	0.997
Subtask-level	High	0.385–0.831	0.066	0.997

Table 5: Target masking by score range at $x = 0.25$. MAE denotes mean absolute error, and Rank ρ denotes Spearman correlation between true and recovered model rankings.

We reach the same conclusion from a complementary leave-one-task-out analysis. For each aggregate task, we compute a model ranking by averaging the scores across all remaining tasks and compare it with the ranking induced by the excluded task alone. If a task measures a largely independent capability, the remaining tasks should provide little information about its ranking. Table 4 shows the opposite pattern. Most held-out tasks remain strongly correlated with the rankings derived from the rest of the benchmark suite. IFEval is the primary exception, indicating that some task-specific behavior remains. Overall, however, most tasks appear to reflect a common underlying performance trend rather than isolated capabilities.

Finding 2. *The shared performance signal is broadly distributed across tasks rather than concentrated in a small set of anchor tasks. Preserving or removing individual tasks has little effect on benchmark-wide ranking recovery.*

6.2 Score Range Analysis

This experiment addresses **RQ2** from the perspective of task difficulty. Tasks with low average scores are often regarded as more difficult than those with high average scores. If difficult and easy tasks capture fundamentally different aspects of model performance, benchmark recovery should become substantially harder when information must be transferred across score ranges. We test this hypothesis by grouping tasks by their average score and assessing whether ALS can recover the original benchmark ranking. This design separates two notions that are often conflated in benchmark evaluation: task difficulty and the benchmark-wide performance trend.

We define score ranges separately for the aggregate-task and subtask matrices. For the aggre-

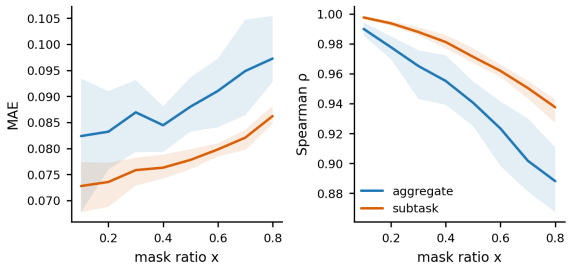


Figure 4: Cross-format recovery when free-form generation cells are masked, and multiple-choice cells remain observed.

Setting	Feat.	Tgt.	MAE	Rank ρ
Aggregate	8	5	0.097	0.887
Subtask-level	482	41	0.086	0.937

Table 6: Cross-format prediction from multiple-choice columns to free-form generation columns. Feat. and Tgt. denote feature and target columns, and Rank ρ denotes Spearman correlation by target-family mean.

gate matrix, we sort the 13 task columns by their model-average score and partition them into Low (bottom four tasks), Middle (next four tasks), and High (top five tasks). For the subtask matrix, we sort the 523 subtasks by their model-average score and divide them into three approximately equal groups. For each target range, we mask only entries in that range while keeping all other columns observed. We use a masking ratio of $x = 0.25$, which corresponds to masking approximately 75% of the entries within the target range.

Table 5 reports the recovery results. Across low-, middle-, and high-score groups, ALS achieves consistently strong reconstruction performance in both matrices. Therefore, the ability to recover rankings appears largely insensitive to score level. These results suggest that tasks with different average scores share a substantial common performance signal. While difficult tasks may remain valuable for evaluating specific capabilities, they do not appear to provide uniquely necessary information for recovering benchmark-wide rankings.

Finding 3. *Benchmark recoverability is largely independent of task difficulty. Low-scoring and high-scoring tasks exhibit similar underlying performance trends, and neither group is uniquely required for reconstructing overall model rankings.*

6.3 Cross-Format Prediction

This experiment addresses **RQ2** from the perspective of answer format. Researchers often regard free-form generation as more demanding than multiple-choice evaluation. They frequently use it to assess advanced reasoning, instruction following, and open-ended generation abilities (Liang et al., 2023; Wang et al., 2024; Qian et al., 2026; Wei et al., 2024; Myrzakhan et al., 2024). If free-form evaluations capture a distinct performance trend, scores on multiple-choice tasks should provide limited information for recovering model rankings on free-form generation tasks.

This hypothesis is plausible when the objective is to evaluate exact generation behavior or instruction-following ability. For benchmark-wide rankings, however, a different question arises: *Does free-form evaluation reveal a distinct ranking signal, or does it largely reflect information already present in multiple-choice evaluations?*

To investigate this question, we perform cross-format masking. At the aggregate level, eight multiple-choice tasks remain fully observed, while five free-form generation tasks serve as target columns. At the subtask level, 482 multiple-choice subtasks provide observed context, and 41 free-form generation subtasks serve as targets. We keep all multiple-choice scores visible and mask 80% of the entries in free-form generation tasks.

Unlike the masking ratio x used in previous experiments, this percentage refers only to entries within the free-form generation columns. Because free-form generation tasks constitute only a small fraction of the matrix, masking the same proportion of entries in the entire matrix would leave insufficient observations for ALS.

Table 6 and Figure 4 show that ALS successfully reconstructs free-form generation rankings from multiple-choice observations. The effect is particularly clear at the subtask level, where ranking recovery remains highly stable despite substantial masking.

These results indicate that free-form generation tasks and multiple-choice tasks share a substantial common performance signal. This does not imply that free-form evaluation is unnecessary. Indeed, Table 4 shows that IFEval is less predictable than mathematics generation tasks, suggesting that some task-specific information remains. However, for benchmark-wide ranking, much of the free-form generation trend appears recoverable from multiple-

Matrix	Pair type	Mean ρ	Median ρ
Aggregate	All tasks	0.669	0.705
Aggregate	Multiple choice	0.749	0.734
Aggregate	Free-form gen.	0.600	0.595
Aggregate	Cross-format	0.600	0.628
Subtask-level	All tasks	0.619	0.644
Subtask-level	Multiple choice	0.623	0.649
Subtask-level	Free-form gen.	0.678	0.656
Subtask-level	Cross-format	0.573	0.606

Table 7: Pairwise Spearman correlations between task-specific model rankings. Free-form gen. denotes free-form generation tasks, and ρ denotes Spearman correlation.

choice evaluations.

Finding 4. *Multiple-choice evaluations capture much of the benchmark-wide ranking signal present in free-form generation tasks. Free-form evaluations remain valuable for measuring task-specific behaviors, but they do not appear to provide a fundamentally distinct ranking trend.*

7 Task-Ranking Correlation (RQ3)

From the previous experiments, we found that benchmark-wide rankings are highly recoverable from incomplete observations. However, recoverability at the benchmark level does not necessarily imply that any two tasks measure the same capability. We analyze pairwise relationships between tasks. Specifically, we compute Spearman rank correlations between model rankings induced by individual tasks. We report the mean and median correlations across all task pairs and separately analyze pairs that share the same answer format and pairs with different formats.

Table 7 shows that task rankings are positively correlated across most task pairs. Models that perform well on one task tend to perform well on others, indicating the presence of a shared performance trend. However, these pairwise correlations are substantially weaker than the ranking recovery observed in our matrix-completion experiments.

This contrast is important. ALS reconstructs benchmark-wide rankings by combining information from many tasks simultaneously rather than relying on strong correspondence between individual task pairs. Consequently, broad benchmark redundancy does not imply that individual tasks are interchangeable. The weaker pairwise correlations indicate that many tasks still contain domain-specific information that is not fully captured by the benchmark average. Therefore, when the goal is to evaluate a particular capability or application domain, direct evaluation on the corresponding benchmark remains necessary. Appendix D confirms this distinction by jointly masking the mathematics benchmarks and recovering their shared ranking from the remaining tasks. We will release the model-task matrix used in this study so that benchmark developers can apply the same analysis to newly proposed benchmarks. Researchers can evaluate a new benchmark on a limited set of models, append the scores to the matrix, and test whether its information about model rankings is distinct from or recoverable from the existing benchmark suite before committing to broader evaluation.

Matrix	Var. share Comp. 1	Var. share Comp. 1-2	Mean ρ'
Aggregate	0.690	0.780	-0.994
Subtask-level	0.635	0.706	-0.999

Table 8: Model-direction SVD summary. Var. share Comp. denotes the sum of squared singular values for the indicated components divided by the sum over all components, and Mean ρ' denotes Spearman correlation with the all-model mean-score direction.

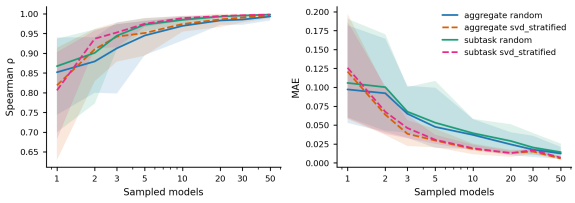


Figure 5: Benchmark-trend estimation from selected models. The left panel shows task-mean Spearman correlation, the right panel shows task-mean MAE, and the shade represents variation across seeds.

Finding 5. *Benchmark tasks share a common performance trend, but individual task pairs remain substantially less correlated than benchmark-wide rankings reconstructed from the full matrix. Thus, redundancy in broad benchmark suites does not eliminate the need for domain-specific evaluation.*

8 Model Subsampling for Benchmark Trends (RQ4)

The previous experiments investigated how many task evaluations are required when the set of models is fixed. RQ4 considers the complementary problem from the perspective of benchmark construction: *When constructing a new benchmark, how many model evaluations are required to estimate its overall performance trend?* Evaluating a

benchmark on a large number of models provides the most reliable estimate of its characteristics, but such evaluations are often costly and quickly become outdated as new models emerge. In practice, benchmark designers are often interested in broader properties, such as whether a task is generally easy or difficult, whether subtasks follow the intended difficulty pattern, and whether the benchmark distinguishes between models in the target domain. If model performances also exhibit substantial redundancy, these properties should be recoverable from a relatively small set of evaluated models.

We investigate whether benchmark-wide trends can be estimated from a limited subset of models. Because matrix completion remains unchanged under matrix transposition, the strong recoverability observed in Figure 1 already suggests that the model axis contains substantial shared structure. Therefore, missing model evaluations need not prevent recovery of benchmark-wide trends.

To further analyze this structure, we perform SVD on the transposed model-task matrix. Table 8 shows that a dominant first component explains a large fraction of the variance in both the aggregate-task and subtask matrices. Moreover, the first component is strongly correlated with model rankings induced by average benchmark scores. This indicates that model performance across tasks is largely governed by a shared latent trend rather than independent task-specific variation.

Then, we examine how many models are required to estimate benchmark characteristics. For each value of k, we select k models and estimate task-level trends using only those models. We compare uniform random sampling with an SVD-based strategy that selects models spanning the first principal direction. We evaluate agreement with the full 100-model benchmark using task-level MAE and Spearman's correlation. Figure 5 shows that benchmark trends can be estimated accurately from a relatively small subset of models. Regardless of the selection strategy, correlations rapidly approach those obtained using all 100 models. These results suggest that exhaustive model evaluation is often unnecessary for identifying broad benchmark characteristics.

Finding 6. *Benchmark-wide trends are highly recoverable from a small subset of models. A limited number of model evaluations is often sufficient to estimate task difficulty patterns and benchmark-level ranking trends before conducting an exhaustive evaluation.*

9 Conclusion

We investigated how many task evaluations are necessary to recover benchmark-wide model rankings. By representing benchmark results as a model-task matrix and applying matrix completion, we analyzed evaluation results from 100 LLMs across 13 aggregate tasks and 523 subtasks.

Our experiments show that broad benchmark suites contain substantial redundancy. ALS accurately recovered benchmark-wide rankings even when a large fraction of scores was masked. This recoverability remained stable across task importance, task difficulty, and answer formats, suggesting that the shared performance signal is broadly distributed rather than concentrated in a small set of tasks. At the same time, individual task pairs were substantially less correlated than benchmark-level rankings recovered from the full matrix. This indicates that redundancy at the benchmark level does not eliminate domain-specific information and that direct evaluation remains necessary when the goal is to measure a particular capability or application domain.

Finally, we showed that benchmark-wide trends can also be estimated from a relatively small subset of models, suggesting that exhaustive model evaluation is often unnecessary during benchmark construction. Our analysis at the task level complements methods that reduce the number of examples within a retained benchmark. The analysis using subsets of models addresses the number of systems needed to characterize a new benchmark. These results imply that adding more tasks does not automatically make a general evaluation more informative. Benchmark designers should prioritize tasks that add distinct information about model rankings or consistent behavior in a particular domain, while direct evaluation should remain the basis for claims about a particular domain. Overall, our results suggest that broad benchmark rankings are far more recoverable than commonly assumed, while domain-specific evaluations continue to provide unique information beyond the benchmark average.

Limitations

Our analysis measures benchmark redundancy with respect to the evaluated model population. It does not disentangle the contributions of model family,

parameter size, training data, architecture, or release generation to similarities among model scores. Therefore, different model populations may exhibit different degrees of recoverability. Nevertheless, we expect the overall observations regarding benchmark redundancy to remain qualitatively similar across reasonably diverse model populations. For RQ2, we compare free-form generation and multiple-choice tasks already included in the benchmark suite. Therefore, the experiment cannot isolate answer format by presenting the same questions in both formats, but it directly tests whether the free-form generation tasks add distinct information to model rankings across the benchmark. The paper also focuses on aggregate and subtask scores, not item-level behavior. Item-level analyses may reveal redundancy patterns that are invisible at the task-column level.

Ethical Consideration

Datasets and models. We only use public datasets and models. All of these uses are appropriate for their licenses.

Annotation. This work does not involve human subjects, user studies, or the collection of private information.

Potential Risk. The main ethical risk is that benchmark redundancy could be misread as a reason to skip evaluations that are important for safety, fairness, or domain-specific reliability.

AI Tools. We used AI tools to assist with grammatical correction and translation. However, all research ideas, experimental design, and original writing were carried out by the authors. We carefully reviewed and verified all AI-generated suggestions before including them in the manuscript. We take full responsibility for the paper's content and acknowledge the AI tools that helped us prepare the manuscript.

Acknowledgments

We thank the anonymous area chair and reviewers for their valuable comments. This work was partly supported by JSPS KAKENHI Grant Numbers JP25K24369, JP26K21312, and JST SPRING Grant Number JPMJSP2140.

References

Marah I Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Hassan Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, and 66 others. 2024a. Phi-3 technical report: A highly capable language model locally on your phone. Technical Report MSR-TR-2024-12, Microsoft.

Marah I 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 CT Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, and 8 others. 2024b. Phi-4 technical report. Technical Report MSR-TR-2024-57, Microsoft.

01. AI, :, Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, Kaidong Yu, Peng Liu, Qiang Liu, Shawn Yue, Senbin Yang, Shiming Yang, Tao Yu, and 13 others. 2024. Yi: Open foundation models by 01.ai. *Preprint*, arXiv:2403.04652.

Kyunghoon Bae, Eunbi Choi, Kibong Choi, Stanley Jungkyu Choi, Yemuk Choi, Kyubeen Han, Seokhee Hong, Junwon Hwang, Taewan Hwang, Joonwon Jang, Hyojin Jeon, Kijeong Jeon, Gerrard Jeongwon Jo, Hyunjik Jo, Jiyeon Jung, Euisoon Kim, Hyosang Kim, Jihoon Kim, Joonkee Kim, and 21 others. 2026. Exaone 4.0: Unified large language models integrating non-reasoning and reasoning modes. *Preprint*, arXiv:2507.11407.

Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, Ido Shahaf, Oren Tropp, Ehud Karpas, Ran Zilberstein, Jiaqi Zeng, Soumye Singhal, Alexander Bukharin, Yian Zhang, Tugrul Konuk, and 114 others. 2025. Llama-nemotron: Efficient reasoning models. *Preprint*, arXiv:2505.00949.

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

ByteDance Seed, Yuyu Zhang, Jing Su, Yifan Sun, Chenguang Xi, Xia Xiao, Shen Zheng, Anxiang Zhang, Kaibo Liu, Daoguang Zan, Tao Sun, Jinhua Zhu, Shulin Xin, Dong Huang, Yetao Bai, Lixin Dong, Chao Li, Jianchong Chen, Hanzhi Zhou, and 8 others. 2025. Seed-Coder: Let the code model curate data for itself. *Preprint*, arXiv:2506.03524.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, Xiaoyi Dong, Haodong Duan, Qi Fan, Zhaoye Fei, Yang Gao, Jiaye Ge, Chenya Gu, Yuzhe

Gu, Tao Gui, and 81 others. 2024. Internlm2 technical report. *Preprint*, arXiv:2403.17297.

Emmanuel J. Candès and Benjamin Recht. 2009. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9:717–772.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Yangyi Chen, Binxuan Huang, Yifan Gao, Zhengyang Wang, Jingfeng Yang, and Heng Ji. 2025. Scaling laws for predicting downstream performance in llms. *Preprint*, arXiv:2410.08527.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *Preprint*, arXiv:1803.05457.

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

Team Cohere, Aakanksha, Arash Ahmadian, Marwan Ahmed, Jay Alammar, Yazeed Alnumay, Sophia Althammer, Arkady Arkhangorodsky, Viraat Aryabumi, Dennis Aumiller, Raphaël Avalos, Zahara Aviv, Sammie Bae, Saurabh Baji, Alexandre Barbet, Max Bartolo, Björn Bebensee, Neeral Beladia, Walter Beller-Morales, and 207 others. 2025. Command a: An enterprise-ready large language model. *Preprint*, arXiv:2504.00698.

John Dang, Shivalika Singh, Daniel D'souza, Arash Ahmadian, Alejandro Salamanca, Madeline Smith, Aidan Peppin, Sungjin Hong, Manoj Govindassamy, Terrence Zhao, Sandra Kublik, Meor Amer, Viraat Aryabumi, Jon Ander Campos, Yi-Chern Tan, Tom Kocmi, Florian Strub, Nathan Grinsztajn, Yannis Flet-Berliac, and 26 others. 2024. Aya expanse: Combining research breakthroughs for a new multilingual frontier. *Preprint*, arXiv:2412.04261.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony S. Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and 510 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David W. Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Daniel Morrison, Jake Poznanski, Kyle Lo, Luca Soldaini, Matt Jordan, Mayee F. Chen, Michael Noukhovitch, Nathan Lambert, Pete Walsh, Pradeep Dasigi, and 48 others. 2025. Olmo 3. *Preprint*, arXiv:2512.13961.

Kazuki Fujii, Taishi Nakamura, Mengsay Loem, Hiroki Iida, Masanari Ohi, Kakeru Hattori, Hirai Shota, Sakae Mizuki, Rio Yokota, and Naoaki Okazaki. 2024. Continual pre-training for cross-lingual LLM adaptation: Enhancing japanese language capabilities. In *First Conference on Language Modeling*.

Samir Yitzhak Gadre, Georgios Smyrnis, Vaishaal Shankar, Suchin Gururangan, Mitchell Wortsman, Rulin Shao, Jean Mercat, Alex Fang, Jeffrey Li, Sedrick Keh, Rui Xin, Marianna Nezhurina, Igor Vasiljevic, Luca Soldaini, Jenia Jitsev, Alex Dimakis, Gabriel Ilharco, Pang Wei Koh, Shuran Song, and 6 others. 2025. Language models scale reliably with over-training and on downstream tasks. In *The Thirteenth International Conference on Learning Representations*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024. The language model evaluation harness.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Juanzi Li, and 37 others. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *Preprint*, arXiv:2406.12793.

IBM Granite Team. 2024. Granite 3.0 language models.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer, Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, and 31 others. 2025. Openthoughts: Data recipes for reasoning models. *Preprint*, arXiv:2506.04178.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical

problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, page 263–272, USA. IEEE Computer Society.

Yuzhen Huang, Jinghan Zhang, Zifei Shan, and Junxian He. 2024. Compression represents intelligence linearly. In *First Conference on Language Modeling*.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R Narasimhan. 2024. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14).

Sanghoon Kim, Dahyun Kim, Chanjun Park, Wonsung Lee, Wonho Song, Yunsu Kim, Hyeonwoo Kim, Yungi Kim, Hyeonju Lee, Jihoo Kim, Changbae Ahn, Seonghoon Yang, Sukyung Lee, Hyunbyung Park, Gyoungjin Gim, Mikyoung Cha, Hwalsuk Lee, and Sunghun Kim. 2024. SOLAR 10.7B: Scaling large language models with simple yet effective depth up-scaling. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, pages 23–35, Mexico City, Mexico. Association for Computational Linguistics.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Alexander Cosgrove, Christopher D Manning, Christopher Re, Diana Acosta-Navas, Drew Arad Hudson, and 31 others. 2023. Holistic evaluation of language models. *Transactions on Machine Learning Research*. Featured Certification, Expert Certification.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *Preprint*, arXiv:2007.08124.

Nicholas Lourie, Michael Y. Hu, and Kyunghyun Cho. 2025. Scaling laws are unreliable for downstream tasks: A reality check. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 16167–16180, Suzhou, China. Association for Computational Linguistics.

Youmi Ma, Sakae Mizuki, Kazuki Fujii, Taishi Nakamura, Masanari Ohi, Hinari Shimada, Taihei Shiotani, Koshiro Saito, Koki Maeda, Kakeru Hattori, Takumi Okamoto, Shigeki Ishida, Rio Yokota, Hiroya Takamura, and Naoaki Okazaki. 2025. Building instruction-tuning datasets from human-written instructions with open-weight large language models. *Preprint*, arXiv:2503.23714.

Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. 2024. tinyBenchmarks: evaluating LLMs with fewer examples. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 34303–34326. PMLR.

Aidar Myrzakhan, Sondos Mahmoud Bsharat, and Zhiqiang Shen. 2024. Open-llm-leaderboard: From multi-choice to open-style questions for llms evaluation, benchmark, and arena. *Preprint*, arXiv:2406.07545.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, and 21 others. 2024. 2 olmo 2 furious. *Preprint*, arXiv:2501.00656.

OpenAI. 2025. gpt-oss-120b & gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

David Owen. 2024. How predictable is language model benchmark performance? *Preprint*, arXiv:2401.04757.

Yotam Perlitz, Elron Bandel, Ariel Gera, Ofir Arviv, Liat Ein-Dor, Eyal Shnarch, Noam Slonim, Michal Shmueli-Scheuer, and Leshem Choshen. 2024. Efficient benchmarking (of language models). In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2519–2536, Mexico City, Mexico. Association for Computational Linguistics.

Yotam Perlitz, Ariel Gera, Ofir Arviv, Asaf Yehudai, Elron Bandel, Eyal Shnarch, Michal Shmueli-Scheuer, and Leshem Choshen. 2025. Benchmark agreement testing done right: A guide for LLM benchmark evaluation. In *NeurIPS 2025 Workshop on Evaluating the Evolving LLM Lifecycle: Benchmarks, Emergent Abilities, and Scaling*.

Qi Qian, Chengsong Huang, Jingwen Xu, Changze Lv, Muling Wu, Wenhao Liu, Xiaohua Wang, Zhenghua Wang, Zisu Huang, Muzhao Tian, Jianhan Xu,

Kun Hu, He-Da Wang, Yao Hu, Xuanjing Huang, and Xiaoqing Zheng. 2026. Benchmark^2: Systematic evaluation of llm benchmarks. *Preprint*, arXiv:2601.03986.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Yangjun Ruan, Chris J. Maddison, and Tatsunori Hashimoto. 2024. Observational scaling laws and the predictability of langauge model performance. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

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. 2023. Language models are multilingual chain-of-thought reasoners. In *The Eleventh International Conference on Learning Representations*.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, and 431 others. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*. Featured Certification.

Falcon-LLM Team. 2024. The falcon 3 family of open models.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, and 179 others. 2024. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

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. 2024. MMLU-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Fangyun Wei, Xi Chen, and Lin Luo. 2024. Rethinking generative large language model evaluation for semantic comprehension. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Guillaume Wisniewski. 2013. On the predictability of human assessment: when matrix completion meets NLP evaluation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 137–142, Sofia, Bulgaria. Association for Computational Linguistics.

Weihao Xuan, Rui Yang, Heli Qi, Qingcheng Zeng, Yunze Xiao, Aosong Feng, Dairui Liu, Yun Xing, Junjue Wang, Fan Gao, Jinghui Lu, Yuang Jiang, Huitao Li, Xin Li, Kunyu Yu, Ruihai Dong, Shangding Gu, Yuekang Li, Xiaofei Xie, and 13 others. 2025. MMLU-ProX: A multilingual benchmark for advanced large language model evaluation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 1513–1532, Suzhou, China. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Guanhua Zhang, Florian E. Dorner, and Moritz Hardt. 2025a. How benchmark prediction from fewer data misses the mark. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Qiyuan Zhang, Fuyuan Lyu, Xue Liu, and Chen Ma. 2024a. Collaborative performance prediction for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 2576–2596, Miami, Florida, USA. Association for Computational Linguistics.

Wenxuan Zhang, Hou Pong Chan, Yiran Zhao, Mahani Aljunied, Jianyu Wang, Chaoqun Liu, Yue Deng, Zhiqiang Hu, Weiwen Xu, Yew Ken Chia, Xin Li, and Lidong Bing. 2024b. Seallms 3: Open foundation and chat multilingual large language models for southeast asian languages. *Preprint*, arXiv:2407.19672.

Zicheng Zhang, Xiangyu Zhao, Xinyu Fang, Chunyi Li, Xiaohong Liu, Xiongkuo Min, Haodong Duan, Kai Chen, and Guangtao Zhai. 2025b. Redundancy principles for MLLMs benchmarks. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12492–12504, Vienna, Austria. Association for Computational Linguistics.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2024. AGIEval: A human-centric benchmark for evaluating foundation models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2299–2314, Mexico City, Mexico. Association for Computational Linguistics.

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

Lexin Zhou, Lorenzo Pacchiardi, Fernando Martínez-Plumed, Katherine M. Collins, Yael Moros-Daval, Seraphina Zhang, Qinlin Zhao, Yitian Huang, Luning Sun, Jonathan E. Prunty, Zongqian Li, Pablo Sánchez-García, Kexin Jiang Chen, Pablo A. M. Casares, Jiyun Zu, John Burden, Behzad Mehrbakhsh, David Stillwell, Manuel Cebrian, and 7 others. 2025. General scales unlock ai evaluation with explanatory and predictive power. *Preprint*, arXiv:2503.06378.

A Matrix Definitions

This appendix gives the matrix definitions that the main text uses. Let $X \in \mathbb{R}^{n \times d}$ be a score matrix with n models and d tasks. The task-wise z-score normalization is

$$Z_{ij} = \frac{X_{ij} - \mu_j}{\sigma_j}, \tag{4}$$

where μ_j and σ_j are the mean and standard deviation of task j over models. We also checked min-max normalization as a robustness variant.

For SVD, we decompose the normalized matrix as

$$Z = U \Sigma V^\top. \tag{5}$$

The first right singular vector gives task loadings. The normalized loading mass of task j is

$$p_j = \frac{V_{1j}^2}{\sum_k V_{1k}^2}. \tag{6}$$

We summarize concentration with the largest loading share and the effective number of tasks,

$$N_{\text{eff}} = \exp\left(-\sum_j p_j \log p_j \right). \tag{7}$$

For ALS, let Ω be the observed set after masking. The model predicts

$$\hat{Y}_{ij} = \mu + b_i + c_j + u_i^\top v_j, \tag{8}$$

where b_i is a model bias, c_j is a task bias, and $u_i, v_j \in \mathbb{R}^r$ are latent factors. We fit these parameters by minimizing squared error on observed cells with ℓ_2 penalties on factors and biases. The main ALS runs use rank 5, task-wise z-score fitting, $\lambda_u = 1.2$, $\lambda_b = 0.01$, and 20 mask seeds.

B Recovery Baselines

In Section 5, we use the full ALS model in Equation 8 to recover randomly masked cells. The equation combines the global mean μ, the model bias b_i, the task bias c_j, and the interaction $u_i^\top v_j$. To determine which terms account for the recovery, we repeat the experiment with models that remove different terms from Equation 8. We apply the same random masks to the aggregate and subtask matrices at each value of x and use the same 20 seeds as in Figure 1. As in the main experiment in Section 5, we normalize each task using the observed cells and denote the resulting matrix by Y. For row mean, let $\Omega_i = \{j : (i,j) \in \Omega\}$ denote the tasks observed for model i. This predicts every hidden cell in row i as $\hat{Y}_{ij}^{\text{row}} = |\Omega_i|^{-1} \sum_{k \in \Omega_i} Y_{ik}$. The additive bias uses $\hat{Y}_{ij}^{\text{bias}} = \mu + b_i + c_j$. It estimates these terms by minimizing $\sum_{(i,j)\in\Omega} (Y_{ij} - \mu - b_i - c_j)^2 + \lambda_b (\sum_i b_i^2 + \sum_j c_j^2)$. The full ALS condition retains every term in Equation 8. We return the predictions from all three methods to the original score scale before evaluation.

Figure 6 shows that all three methods preserve much of the aggregate model ordering, while the lower error from full ALS on the subtask matrix shows that the interaction term recovers variation beyond the shared model direction.

C Contribution Scores for Individual Tasks

In Section 6.1, we test whether preserving the tasks with the largest contributions improves recovery. Table 9 shows that no single task dominates the first

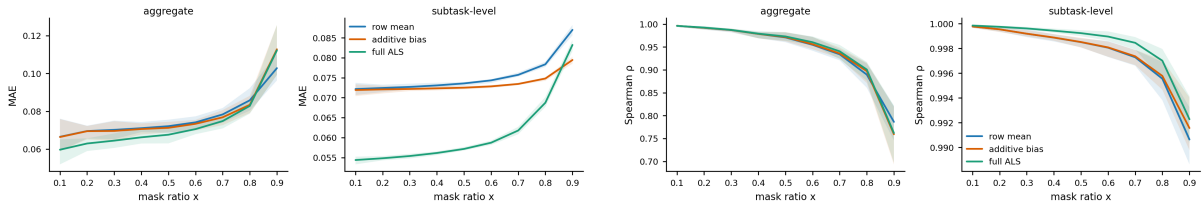


Figure 6: Recovery with imputation based on the row mean, imputation with an additive bias model, and full ALS across the masked cell ratio x. The left image reports MAE, the right image reports Spearman's ρ, and the shaded areas show variation across 20 mask seeds.

Task	Mass	Rank
MMLU	0.100	1
ARC-Challenge	0.095	2
MMLU-Pro	0.091	3
MGSM	0.090	4
AGIEval	0.086	5
MedQA	0.085	6
GSM8K	0.075	7
MMLU-ProX-Lite	0.073	8
HellaSwag	0.072	9
TruthfulQA	0.069	10
MATH	0.067	11
PIQA	0.067	12
IFEval	0.031	13

Table 9: Contribution scores for the aggregate tasks used to select candidate anchor tasks. Mass denotes the normalized squared loading on the first component, and Rank orders tasks by this mass.

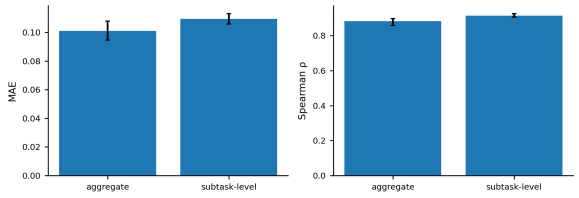


Figure 7: Recovery of the jointly masked mathematics target group in the aggregate and subtask matrices. The left panel reports MAE, the right panel reports Spearman's ρ, and the error bars show variation across 20 mask seeds.

component. The effective numbers of 12.63 among 13 tasks and 499.4 among 523 subtasks indicate that contributions spread across most tasks.

D Mathematics Tasks as a Joint Target

We treat GSM8K, MATH, and MGSM jointly as the target group, keep the other task columns observed, and mask 80% of the mathematics entries over 20 seeds. Figure 7 shows that the mathematics ordering remains recoverable.

E Task List

Table 10 shows all tasks used in our experiments.

F Model List

Table 11 lists the complete set of 100 models used as the rows of the score matrices in our experiments. Each row corresponds to one evaluated model, and the parameter columns report total and active parameters in billions. For dense models, these two counts are identical; for mixture-of-experts models, the active parameter count can be smaller than the total parameter count. We include the full list here so that the ALS recovery, SVD analysis, and model-subsampling results can be traced to the exact model set used in the paper.

Task	Subtasks	Answer method	Language	Domain
MMLU	57	Multiple choice	English	General
MMLU-Pro	14	Multiple choice	English	General
MMLU-ProX-Lite	406	Multiple choice	29 languages	General
ARC-Challenge	1	Multiple choice	English	Science
HellaSwag	1	Multiple choice	English	Commonsense
GSM8K	1	Free-form	English	Math
MATH	7	Free-form	English	Math
MGSM	11	Free-form	11 languages	Math
IFEval	1	Free-form	English	Instruction following
MedQA	1	Multiple choice	English	Medicine
PIQA	1	Multiple choice	English	Physical commonsense
TruthfulQA	1	Multiple choice	English	Truthfulness
AGIEval	21	Free-form	English and Chinese	General

Table 10: Tasks used in the aggregate matrix and their corresponding subtask columns. "Answer method" describes the evaluation format, and "Domain" describes the broad task area.

Model	Total Param.	Active Param.	Model	Total Param.	Active Param.
Qwen2-0.5B-Instruct	0.5	0.5	Olmo-3-7B-RL-Zero-Math	7	7
Qwen2-1.5B-Instruct	1.5	1.5	Olmo-3.1-32B-Instruct	32	32
Qwen2-7B-Instruct	7	7	Falcon3-7B-Instruct	7	7
Qwen2-57B-A14B-Instruct	57	14	Falcon3-10B-Instruct	10	10
Qwen2.5-7B-Instruct	7	7	DeepSeek-R1-Distill-Qwen-7B	7	7
Qwen2.5-14B-Instruct	14	14	DeepSeek-R1-Distill-Llama-8B	8	8
Qwen2.5-Coder-7B-Instruct	7	7	DeepSeek-R1-Distill-Qwen-14B	14	14
Qwen2.5-Math-7B-Instruct	7	7	Llama-3.1-Nemotron-Nano-4B-v1.1	4	4
Qwen2.5-72B-Instruct	72	72	Llama-3.1-Nemotron-Nano-8B-v1	8	8
Qwen3-0.6B	0.6	0.6	glm-4-9b-chat-hf	9	9
Qwen3-1.7B	1.7	1.7	GLM-Z1-9B-0414	9	9
Qwen3-4B	4	4	GLM-Z1-32B-0414	32	32
Qwen3-8B	8	8	glm-edge-1.5b-chat	1.5	1.5
Qwen3-14B	14	14	glm-edge-4b-chat	4	4
Qwen3-30B-A3B-Instruct-2507	30	3	gpt-oss-20b	21	3.6
Qwen3-30B-A3B-Thinking-2507	30	3	InternLM3-8B-Instruct	8	8
Qwen3.5-0.8B	0.8	0.8	InternLM2.5-7B-Chat	7	7
Qwen3.5-2B	2	2	InternLM2.5-20B-Chat	20	20
Qwen3.5-4B	4	4	Granite-3.3-2B-Instruct	2	2
Qwen3.5-9B	9	9	Granite-3.3-8B-Instruct	8	8
Qwen3.5-27B	27	27	Granite-4.0-tiny	7	7
Qwen3.5-35B-A3B	35	3	EXAONE-3.0-7.8B-Instruct	7.8	7.8
Llama-2-7b-chat	7	7	EXAONE-3.5-2.4B-Instruct	2.4	2.4
Llama-2-13b-chat	13	13	EXAONE-4.0-1.2B	1.2	1.2
Llama-3.2-1B-Instruct	1	1	EXAONE-4.0-32B	32	32
Llama-3.2-3B-Instruct	3	3	Aya-Expanse-8B	8	8
Llama-3.1-8B-Instruct	8	8	Aya-Expanse-32B	32	32
Llama-3.1-70B-Instruct	70	70	Command-R7B-12-2024	7	7
gemma-2-2b-it	2	2	SeaLLMs-v3-7B-Chat	7	7
gemma-2-9b-it	9	9	Yi-1.5-6B-Chat	6	6
gemma-2-27b-it	27	27	Yi-1.5-9B-Chat	9	9
gemma-3-1b-it	1	1	Yi-1.5-34B-Chat	34	34
gemma-3-4b-it	4	4	SOLAR-10.7B-Instruct-v1.0	10.7	10.7
gemma-3-12b-it	12	12	SOLAR-pro-preview-instruct	22	22
gemma-3-27b-it	27	27	Seed-Coder-8B-Instruct	8	8
Phi-3-mini-128k-instruct	3.8	3.8	Seed-Coder-8B-Reasoning	8	8
Phi-3-medium-128k-instruct	14	14	llm-jp-3-1.8b-instruct3	1.8	1.8
Phi-3.5-mini-instruct	3.8	3.8	llm-jp-3-7.2b-instruct3	7.2	7.2
Phi-3.5-MoE-instruct	42	6.6	llm-jp-3-13b-instruct3	13	13
Phi-4-mini-instruct	3.8	3.8	llm-jp-3.1-1.8b-instruct4	1.8	1.8
Phi-4-mini-reasoning	3.8	3.8	llm-jp-3.1-13b-instruct4	13	13
Phi-4-reasoning	14	14	llm-jp-4-8b-thinking	8	8
Phi-4-reasoning-plus	14	14	llm-jp-4-32b-a3b-thinking	32	3
OLMoE-1B-7B-0125-Instruct	7	1	Qwen3-Swallow-8B-SFT-v0.2	8	8
OLMo-2-0425-1B-Instruct	1	1	Qwen3-Swallow-30B-A3B-SFT-v0.2	30	3
OLMo-2-1124-7B-Instruct	7	7	GPT-OSS-Swallow-20B-SFT-v0.1	20	20
OLMo-2-1124-13B-Instruct	13	13	OpenThinker2-7B	7	7
OLMo-2-0325-32B-Instruct	32	32	OpenThinker3-7B	7	7
Olmo-3-7B-Instruct	7	7	Dolphin3.0-Llama3.1-8B	8	8
Olmo-3-7B-Think	7	7	Dolphin3.0-Llama3.2-3B	3	3

Table 11: Complete model list used in the score matrix. Parameter counts are in billions.