

# Predicting LLM Benchmark Scores via Low-Rank Matrix Completion

*Technical Report*

February 2026

## Abstract

Large language models are evaluated on an ever-growing set of benchmarks, yet no single model is tested on all of them. The resulting data forms a sparse matrix with roughly two-thirds of entries missing. We construct a fully-cited benchmark matrix spanning 83 frontier LLMs across 49 benchmarks (1,383 observed scores, 34% fill rate) and develop a matrix completion framework to predict the missing 2,684 cells. Through a systematic search over 34 candidate methods, we find that a logit-space blend of benchmark regression and rank-2 SVD—**LogitSVD Blend**—achieves 6.74% median absolute percentage error (MedAPE) on held-out scores, a 14.2% relative improvement over the prior best statistical baseline. The logit transform alone accounts for 70% of this gain by linearizing ceiling and floor effects on percentage-scale benchmarks. Analysis of the matrix’s singular value spectrum reveals an approximately rank-2 latent structure: two factors (“general capability” and “frontier reasoning”) explain 51% of the variance. A set of just five benchmarks can predict the remaining 44 with  $\sim 7.8\%$  MedAPE. We additionally show that Claude Sonnet 4.5, when given the full matrix as context, achieves 5.45% MedAPE—surpassing all statistical methods—at a cost of \$0.85 per evaluation run.

## 1 Introduction

The evaluation landscape for large language models (LLMs) has expanded dramatically. As of early 2026, frontier models are routinely assessed on dozens of benchmarks spanning mathematics, coding, scientific reasoning, agentic task completion, multimodal understanding, and instruction following. However, no single model is evaluated on every benchmark. Model developers typically report results on a curated subset—often chosen to highlight strengths—while independent leaderboards cover different subsets. The result is a sparse matrix: models form the rows, benchmarks form the columns, and roughly two-thirds of the entries are missing.

This sparsity creates practical problems. Comparing models requires restricting to shared benchmarks, discarding valuable information. Practitioners choosing between models for deployment must navigate incomplete scorecards. And the research community lacks a unified picture of how capabilities relate across the evaluation landscape.

Matrix completion—predicting missing entries from observed ones—offers a natural solution. If benchmark scores exhibit low-rank structure (i.e., a small number of latent factors explain most of the variation), then the missing entries can be recovered with reasonable accuracy. This paper investigates whether this is the case, and if so, which methods perform best.

**Contributions.** We make five main contributions:

Table 1: **Model distribution by provider.** The matrix includes 83 models from 21 providers, with composition across reasoning mode and weight availability.

Provider	Count	Notable Models
OpenAI	13	o3, o4-mini, GPT-4.1, GPT-4.5, GPT-5
DeepSeek	12	R1, V3, R1-Distill family (1.5B–70B)
Alibaba (Qwen)	10	Qwen3 family (0.6B–235B), QwQ-32B
Anthropic	9	Claude Opus 4.6, Sonnet 4.5, Haiku 3.5
Google	7	Gemini 2.5 Pro/Flash, Gemini 3.1 Pro
Mistral	5	Mistral Large 3, Small 3.1
Microsoft	4	Phi-4, Phi-4-mini
Meta	3	Llama 4 Maverick/Scout, Llama 3.3 70B
xAI	3	Grok 3, Grok 3 mini
Others	17	Moonshot (Kimi), ByteDance (Doubao), Amazon (Nova), etc.

1. **A fully-cited benchmark matrix.** We construct an  $83 \times 49$  matrix of LLM benchmark scores, where every entry carries a citation URL to its original source. The matrix covers 21 model providers and 11 benchmark categories, with 1,383 observed entries (34% fill rate).
2. **Logit-space matrix completion.** We introduce the logit transform as a preprocessing step for benchmark score prediction. Working in log-odds space for percentage-scale benchmarks improves both regression and SVD methods by 11–13% relative, and implicitly handles bimodal benchmark distributions.
3. **A systematic 34-method search.** We evaluate 34 prediction methods across three rounds of exploration, testing six hypotheses about “wasted information” (model metadata, benchmark categories, non-linear transforms, bimodal handling, gradient-boosted trees, and missingness patterns). The winner, **LogitSVD Blend**, combines logit-space benchmark regression ( $\alpha=0.6$ ) with logit-space rank-2 SVD ( $\alpha=0.4$ ).
4. **LLMs as benchmark predictors.** We test whether LLMs themselves can predict benchmark scores by providing the full matrix as context. Claude Sonnet 4.5 achieves 5.45% MedAPE, surpassing all statistical methods at a cost of \$0.85.
5. **Structural findings.** The matrix is approximately rank-2, with factors interpretable as “general capability” and “frontier reasoning.” Five benchmarks suffice to predict the remaining 44 with  $\sim 7.8\%$  MedAPE under proper holdout evaluation.

## 2 Dataset Construction

### 2.1 Model Selection

The matrix includes 83 instruct- or chat-tuned LLMs from 21 providers, released between January 2025 and February 2026. Table 1 summarizes the provider distribution. The models span a wide range of scales (0.6B to  $\sim 2$ T mixture-of-experts parameters), training paradigms (dense, MoE, distillation), and capability modes (57 reasoning models, 26 non-reasoning; 46 open-weight, 37 proprietary).

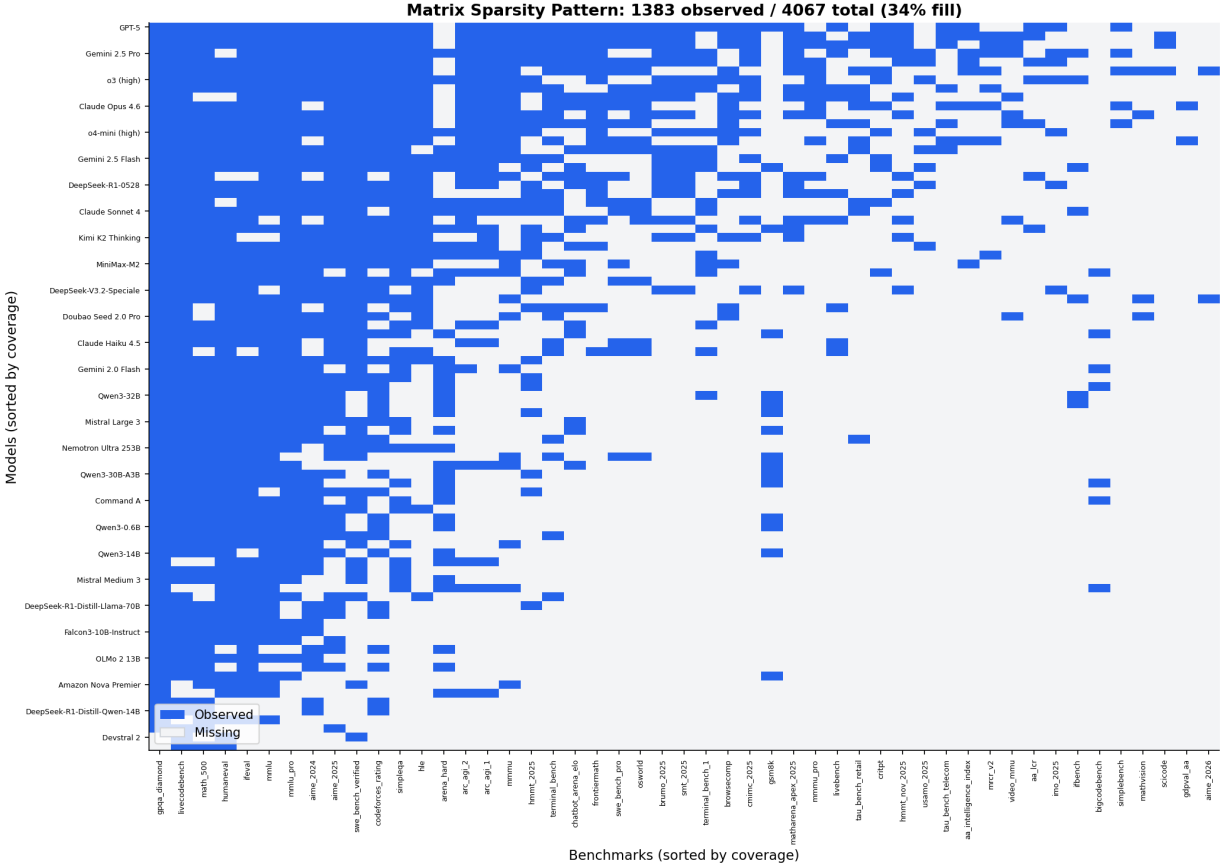


Figure 1: **Sparsity pattern of the benchmark matrix.** Blue cells indicate observed scores; white cells are missing. Models (rows) are sorted by number of known scores; benchmarks (columns) by coverage. The matrix is 34% filled, with frontier models evaluated more broadly.

## 2.2 Benchmark Selection

We include 49 benchmarks spanning 11 categories (Table 2). Most benchmarks report accuracy on a 0–100% scale. Exceptions include Chatbot Arena (Elo, ~1000–1400), Codeforces (rating, ~800–2200), and FrontierMath (0–25% accuracy range). Five benchmarks exhibit *bimodal* score distributions—ARC-AGI-1, ARC-AGI-2, IMO 2025, USAMO 2025, and MathArena Apex 2025—where models either score near zero or well above 10%.

## 2.3 Data Collection and Citation

Every score in the matrix is backed by a source URL pointing to the original report, paper, blog post, or leaderboard entry. The data collection pipeline consists of:

1. **Primary data file** (`build_benchmark_matrix.py`): Contains the core matrix as a list of 4-tuples (`model_id`, `benchmark_id`, `score`, `source_url`).
2. **Supplementary batches** (`extra_scores_{1--5}.py`): Additional scores mined from model papers, leaderboard snapshots, and third-party evaluation reports.

Table 2: **Benchmark categories.** The 49 benchmarks span 11 categories. Coverage denotes the number of models with reported scores.

Category	Count	Examples (max coverage)
Math	15	AIME 2024/2025, MATH-500, Frontier-Math, HMMT, USAMO, IMO
Coding	7	LiveCodeBench (75), SWE-bench Verified, HumanEval, Codeforces
Agentic	6	SWE-bench Pro, Tau-Bench, Terminal-Bench, OSWorld
Reasoning	4	GPQA Diamond (81), ARC-AGI-1/2, HLE
Knowledge	4	MMLU (78), MMLU-Pro, SimpleQA, BrowseComp
Multimodal	3	MMMU, MMMU-Pro, Video-MMU
Instruction	3	IFEval, IFBench, Arena-Hard Auto
Science	2	GPQA Diamond, CritPt
Long Context	2	MRCR v2, LongBench v2
Composite	2	Chatbot Arena Elo, MathArena Apex
Human Preference	1	Chatbot Arena Elo

3. **Merge pipeline** (`merge_extra_scores.py`): Deduplicates entries, resolves conflicts (preferring primary sources), and produces the final  $83 \times 49$  matrix.

## 2.4 Normalization

Benchmarks using percentage-scale accuracy (92% of benchmarks) are stored as-is in the  $[0, 100]$  range. Non-percentage benchmarks (Chatbot Arena Elo, Codeforces rating) are handled separately: they participate in  $z$ -score normalization for KNN and SVD methods but do not receive the logit transform (Section 4.4). All predictions are clamped to valid ranges after inverse transformation.

## 2.5 Sparsity Structure

The matrix is 34% filled (1,383 / 4,067 cells). Coverage is uneven: the most-evaluated benchmarks (GPQA Diamond: 81 models, MMLU: 78, LiveCodeBench: 75) are an order of magnitude more covered than the sparsest (BRUMO: 5, Terminal-Bench 2.0: 6, SMT 2025: 7). Similarly, frontier models tend to be evaluated more broadly (Gemini 2.5 Pro: 36 benchmarks) than smaller or niche models (some have only 4–6). This non-random missingness creates a “rich get richer” pattern that affects method choice: regression-based methods require correlated benchmark scores to exist in the training rows.

# 3 Evaluation Protocol

We use two primary holdout strategies and report seven metrics per method.

### 3.1 Per-Model Leave-50%-Out (Primary)

For each model with  $\geq 8$  known scores, we randomly hide 50% of its observed scores, train the prediction method on the remaining scores plus all other models’ complete data, and predict the hidden cells. We repeat this procedure across 5 random seeds and report the *global* median absolute percentage error (MedAPE) over all held-out cells pooled across seeds.

This protocol tests the realistic scenario: *given some benchmark results for a model, predict the rest*. It is more demanding than random holdout because it requires generalization within a single model’s profile.

### 3.2 Random 20% Holdout (Secondary)

We randomly hide 20% of all 1,383 observed cells, predict them, and report MedAPE. Repeated across 5 seeds. This tests general matrix completion ability without the per-model constraint.

### 3.3 Extended Metrics

Beyond MedAPE, we report six additional metrics to capture different aspects of prediction quality:

- **MAE**: Mean absolute error in raw score points.
- **$\pm 3$  pts**: Fraction of predictions within 3 score points of the true value (a “usefulness” threshold).
- **$\pm 5$  pts**: Fraction within 5 points.
- **APE $>50$** : MedAPE restricted to benchmarks where the model scores above 50 (“easy” benchmarks with ceiling effects).
- **APE $\leq 50$** : MedAPE on benchmarks where the model scores  $\leq 50$  (“hard” benchmarks with floor effects).
- **Bimodal Accuracy**: Classification accuracy on five bimodal benchmarks (ARC-AGI-1/2, IMO 2025, USAMO 2025, MathArena Apex), thresholding at 10% to classify “can solve” vs. “cannot.”
- **Coverage**: Fraction of test cells receiving a finite prediction.

**Why MedAPE?** We choose median (not mean) APE as the primary metric because benchmark score distributions are heavy-tailed: a few outlier benchmarks (e.g., Elo-scale, bimodal) can dominate mean-based metrics. MedAPE is robust to these outliers while remaining interpretable as “for a typical held-out score, the prediction is off by  $X\%$ .”

## 4 Methods

We evaluate 34 prediction methods organized into baselines, matrix factorization, regression, and blends. We describe the key methods below and the full comparison in Section 5.

### 4.1 Baselines

**Benchmark Mean (B0).** Predict each missing cell as the mean of observed scores for that benchmark. This ignores model identity entirely and serves as a floor.

**$k$ -Nearest Neighbors (B2).** For a missing entry  $(i, j)$ , find the  $k=5$  models most similar to model  $i$  (cosine similarity on shared benchmarks) and average their scores on benchmark  $j$ .

**Benchmark-KNN (B3).** Transpose the KNN idea: find the 5 benchmarks most similar to  $j$  and predict model  $i$ ’s score as the average of its scores on those benchmarks.

**Model-Normalized (B1).** Convert all scores to  $z$ -scores within each model, predict the missing  $z$ -score as the benchmark-mean  $z$ -score, then invert.

## 4.2 Matrix Factorization

**SVD (Soft-Impute).** We use the iterative soft-impute algorithm: initialize missing values with column means, compute a rank- $r$  SVD, replace missing values with the rank- $r$  reconstruction, and iterate until convergence. We test ranks  $r \in \{2, 3, 5, 8, 10\}$ .

**NMF and PMF.** Non-negative matrix factorization and probabilistic matrix factorization at rank 5. Both enforce non-negativity but are less stable than SVD on our sparse matrix.

**Nuclear Norm Minimization.** Minimize  $\|M - X\|_F^2 + \lambda \|X\|_*$  where  $\|X\|_*$  is the nuclear norm (sum of singular values). Tested at  $\lambda=1$ .

## 4.3 Regression-Based Methods

**BenchReg.** For each target benchmark  $j$ , identify the  $k=5$  most correlated benchmarks (among those with sufficient shared observations). Fit a Ridge regression from these 5 predictors to target  $j$  using all models with complete data. Predict missing values. This method exploits local benchmark-to-benchmark correlations (e.g., AIME 2024 predicts AIME 2025).

**BenchReg+KNN Blend.** Blend BenchReg ( $\alpha=0.6$ ) with KNN ( $1-\alpha=0.4$ ) to improve coverage. This was the prior best method at 7.86% MedAPE.

## 4.4 The Logit Transform

The single most impactful methodological choice is applying the logit transform to percentage-scale benchmark scores before regression or factorization:

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right), \quad p \in (0, 1) \quad (1)$$

where  $p = \text{score}/100$ , clipped to  $[0.005, 0.995]$  to avoid infinities.

**Why it works.** Consider a benchmark like MMLU where frontier models score 88–92%. In raw space, the relationship between model capability and score is compressed near the ceiling: a model that is “twice as capable” might only gain 2 percentage points. In logit space, this ceiling effect is linearized:

$$\begin{aligned} \text{logit}(0.88) &= 2.00, & \text{logit}(0.92) &= 2.44 & (\Delta &= 0.44) \\ \text{logit}(0.48) &= -0.08, & \text{logit}(0.52) &= 0.08 & (\Delta &= 0.16) \end{aligned}$$

The logit transform correctly models that improving from 88% to 92% represents more “difficulty” than 48% to 52%.

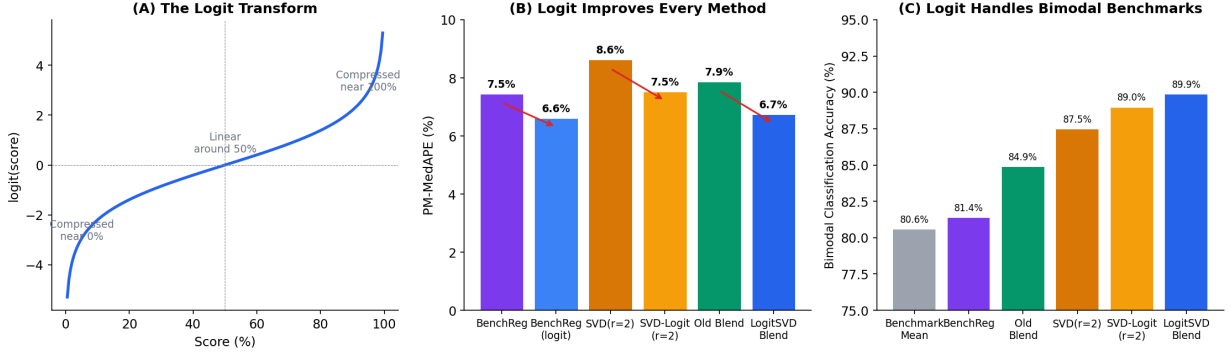


Figure 2: **Effect of the logit transform.** Left: the logit function mapping  $[0, 100]$  to  $(-\infty, +\infty)$ , showing compression near 0 and 100. Center: relative improvement in MedAPE from applying the logit transform to each method. Right: bimodal classification accuracy improvement.

**Bimodal handling.** For benchmarks with bimodal distributions (e.g., ARC-AGI-2 where models score either  $\sim 0\%$  or  $>10\%$ ), the logit transform maps the two modes far apart ( $\text{logit}(0.005) = -5.3$  vs.  $\text{logit}(0.25) = -1.1$ ), allowing SVD and regression to naturally separate them. This obviates the need for explicit bimodal classification (which we tested and found neutral).

#### 4.5 LogitSVD Blend: The Best Method

Our best method, LogitSVD Blend, is a weighted average of two complementary predictors:

$$\hat{y}_{ij} = 0.6 \cdot \hat{y}_{ij}^{\text{LogitBR}} + 0.4 \cdot \hat{y}_{ij}^{\text{SVD-Logit}} \quad (2)$$

The algorithm proceeds in five steps:

1. **Identify percentage benchmarks.** Any benchmark where all observed values fall in  $[-1, 101]$  is treated as percentage-scale ( $\sim 92\%$  of benchmarks).
2. **LogitBenchReg.** For each percentage benchmark  $j$ : transform observed scores to logit space, find the top-5 most correlated benchmarks (in logit space), fit Ridge regression, predict missing values, and apply the inverse logit (sigmoid) to return to  $[0, 100]$ . For non-percentage benchmarks, standard  $z$ -score BenchReg is used.
3. **SVD-Logit( $r=2$ ).** Transform all percentage columns to logit space,  $z$ -score normalize, run soft-impute SVD with rank 2, and inverse-transform back. This method exploits the global low-rank structure.
4. **Blend.** Where both predictors produce estimates, take the weighted average (Eq. 2). Where only SVD-Logit exists (21.5% of cells, where BenchReg lacks sufficient correlated predictors), use SVD-Logit alone. Where neither exists (0.2%), fall back to the column mean.
5. **Clamp.** Clip all predictions to valid ranges:  $[0, 100]$  for percentage benchmarks.

**Why this blend?** BenchReg exploits *local* correlations (“AIME 2024 predicts AIME 2025”), while SVD exploits *global* low-rank structure (“this model’s overall profile resembles GPT-4.1”). These are complementary error profiles. The prior KNN-based blend partner made correlated errors with BenchReg because both are local similarity methods.

Table 3: **Main results: per-model leave-50%-out evaluation (5 seeds).** LogitSVD Blend achieves the best combination of accuracy and coverage. LogitBenchReg has slightly lower MedAPE but only 78.5% coverage. R-Med = random holdout MedAPE; Hi/Lo = MedAPE on scores above/below 50.

Method	MedAPE	R-Med	MAE	$\pm 3$	$\pm 5$	Hi	Lo	BiAcc	Cov
LogitBenchReg	<b>6.61</b>	<b>5.68</b>	4.70	36.7	52.0	<b>4.32</b>	<b>33.2</b>	76.3	78.5
<b>LogitSVD Blend</b>	6.74	5.95	<b>4.61</b>	<b>37.0</b>	<b>52.4</b>	4.34	31.9	<b>89.9</b>	<b>99.8</b>
BenchReg	7.45	6.21	5.61	31.0	46.3	5.06	37.6	81.4	79.6
SVD-Logit( $r=2$ )	7.52	6.62	5.07	35.2	49.6	4.76	33.8	89.0	99.8
BenchReg+KNN	7.86	6.54	5.63	31.5	46.0	5.03	40.4	84.9	99.8
SVD( $r=2$ )	8.62	7.70	6.04	28.7	44.0	5.78	36.3	87.5	99.8
Benchmark Mean	13.52	12.88	9.47	16.4	27.8	8.65	53.1	80.6	99.8

## 4.6 Three-Round Method Search

We conducted the method search in three rounds, testing six hypotheses about information that might improve predictions.

**Round 1 (15 methods).** We tested whether model metadata (provider, parameters, reasoning mode), benchmark categories, non-linear transforms (logit), bimodal handling, gradient-boosted trees, or missingness patterns could improve predictions.

- **Metadata:** All four metadata-based methods (MetaKNN, ProviderCorrected, MultiRidge, FamilyInterp) were neutral or negative. Benchmark scores already encode model capabilities.
- **Categories:** Small improvement ( $-0.5$  points).
- **Logit transform:** Major improvement ( $-0.8$  points).
- **Bimodal handling:** Neutral (logit handles bimodality implicitly).
- **GBT:** Overfits with  $<40$  training rows per benchmark.
- **Missingness:** Not informative for score prediction.

**Round 2 (10 methods).** We combined the Round 1 winners (logit + categories). The best full-coverage method was KitchenSink at 6.89% MedAPE, combining logit, categories, and confidence weighting.

**Round 3 (9 methods).** The breakthrough: applying the logit transform to SVD as well, and using SVD-Logit as the blend partner. LogitSVD Blend at 6.74% MedAPE became the final winner.

## 5 Results

### 5.1 Main Results

Table 3 presents the extended evaluation for the seven key methods under per-model leave-50%-out evaluation (5 seeds).

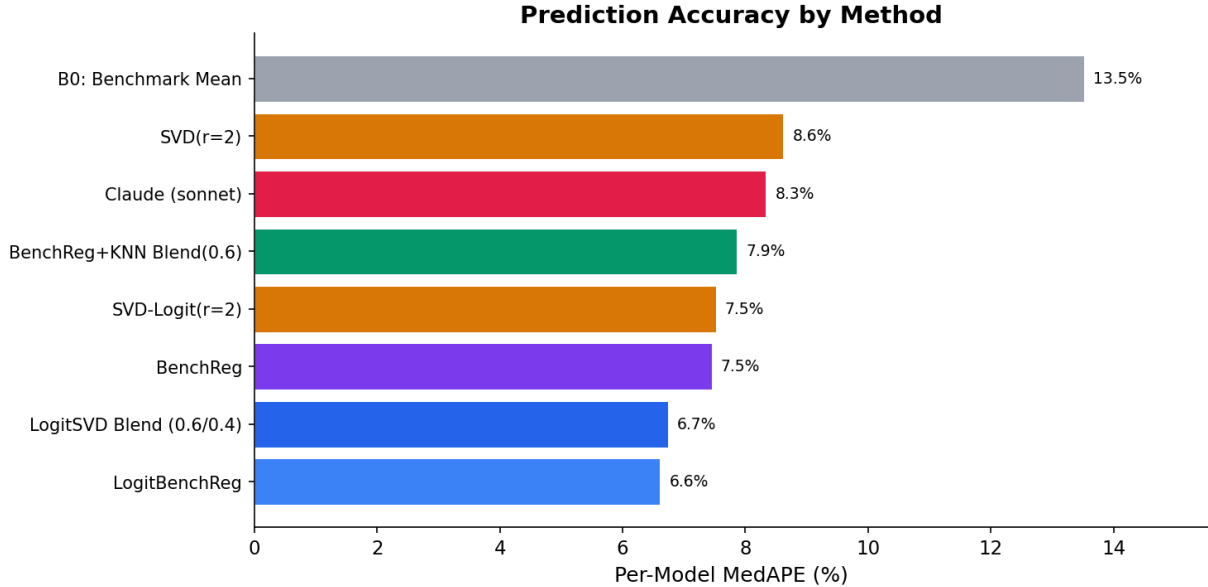


Figure 3: **Method comparison.** Per-model MedAPE for all evaluated methods. Lower is better. LogitSVD Blend (6.74%) provides the best accuracy-coverage combination.

## 5.2 Full 19-Method Baseline Comparison

Table 4 shows the complete ranking of all 19 baseline methods evaluated with the original single-seed evaluation harness.

## 5.3 Decomposing the Improvement

The 14.2% relative improvement from BenchReg+KNN Blend (7.86%) to LogitSVD Blend (6.74%) can be decomposed into two independent changes:

The logit transform consistently improves both regression and factorization methods by 11–13% relative. The additional gain from replacing KNN with SVD-Logit comes from decorrelating the blend components: BenchReg and KNN both exploit local similarity patterns, making correlated errors, while SVD captures global structure and provides an independent signal.

## 5.4 What Didn’t Work

Table 6 summarizes approaches that failed to improve over the baseline. The consistent pattern is that *additional features and complexity hurt*: model metadata, gradient-boosted trees, stacking, missingness features, and explicit bimodal handling all add noise or overfit on the sparse matrix.

# 6 Intrinsic Dimensionality and Latent Structure

## 6.1 Singular Value Spectrum

The SVD spectrum of the (imputed,  $z$ -score normalized) matrix reveals a clear rank-2 structure (Table 7). The first two singular values account for 50.8% of the variance, with a pronounced gap between factors 2 and 3 (14.2%  $\rightarrow$  5.7%).

Table 4: **Full 19-method comparison** (single-seed per-model holdout). BenchReg family dominates. SVD peaks at rank 2; higher ranks overfit.

Rank	Method	PM-MedAPE	R-MedAPE	Cold-Start	$R^2$
1	BenchReg( $k=5$ , $R^2 \geq 0.2$ )	7.69	5.86	—	0.976
2	LogBenchReg	7.72	6.27	—	0.972
3	BenchReg+KNN( $\alpha=0.6$ )	7.88	6.43	19.1	0.979
4	LogBlend( $\alpha=0.65$ )	7.97	6.32	19.1	0.977
5	SVD( $r=2$ )	8.09	7.32	17.2	0.979
6	Ensemble(avg3)	8.30	7.09	19.0	0.974
7	SVD( $r=3$ )	9.07	8.02	19.9	0.972
8	Bench-KNN( $k=5$ )	9.10	8.07	19.1	0.974
9	KNN( $k=5$ )	9.32	7.63	19.1	0.970
10	Quantile+SVD5	9.64	7.31	13.0	0.957
11	NucNorm( $\lambda=1$ )	9.70	8.44	19.0	0.953
12	Model-Normalized	10.00	9.61	16.3	0.962
13	PMF( $r=5$ )	10.29	8.48	18.9	0.960
14	NMF( $r=5$ )	10.46	8.84	13.9	0.951
15	SVD( $r=5$ )	10.95	8.79	19.2	0.947
16	LogSVD( $r=5$ )	12.42	8.75	19.5	0.935
17	Benchmark Mean	12.89	11.71	19.1	0.928
18	SVD( $r=8$ )	13.18	9.71	19.1	0.914
19	SVD( $r=10$ )	13.40	10.33	19.1	0.923

Table 5: **Decomposing the improvement.** The logit transform accounts for approximately 70% of the gain; replacing KNN with SVD-Logit accounts for 30%.

Change	From	To	Relative $\Delta$
Add logit to BenchReg	7.45%	6.61%	−11.3%
Add logit to SVD	8.62%	7.52%	−12.8%
Replace KNN with SVD-Logit	7.86%	6.74%	−14.2%

This rank-2 structure is validated by holdout evaluation: SVD at rank 2 achieves 8.09% PM-MedAPE vs. 9.07% at rank 3, with the gap widening further at higher ranks (10.95% at rank 5, 13.18% at rank 8). The third factor is above the noise floor but below the threshold of predictive usefulness.

## 6.2 Factor Interpretation

**Factor 1 (36.6%—“General Capability”).** The top loadings are GPQA Diamond (−0.37), LiveCodeBench (−0.36), MMLU-Pro (−0.31), MMLU (−0.29), and MATH-500 (−0.26). This factor captures the dominant axis of LLM capability: the strongest models (o3, Claude Opus 4.6, GPT-5, Gemini 2.5 Pro) score uniformly high across all benchmarks.

**Factor 2 (14.2%—“Frontier Reasoning”).** Positive loadings: SimpleQA (+0.34), ARC-AGI-2 (+0.32), HLE (+0.30), FrontierMath (+0.23), SWE-bench Verified (+0.21). Negative loadings: MATH-500 (−0.19), MMLU (−0.18). This factor distinguishes models that excel on genuinely novel, hard tasks from those that perform well on established benchmarks. Models with high Factor 2 (o3,

Table 6: **Approaches that did not improve predictions.**

Approach	MedAPE	Failure Mode
MetaKNN	8.57%	Metadata adds noise
MultiRidge	9.85%	Too many features for few rows
MissingnessKNN	9.25%	NaN pattern not informative
GBT per-benchmark	8.50%	Overfits with <40 training rows
BimodalAware	7.91%	Logit handles this better
MetaLearnerV2	7.14%	Stacking overfits
ProviderCorrected	7.86%	Per-provider offset doesn’t generalize

Table 7: **Singular value spectrum.** The matrix is approximately rank-2, with a clear elbow after the second factor.

Rank	Singular Value	Var. Explained	Cumulative
1	22.5	36.6%	36.6%
2	14.0	14.2%	50.8%
3	8.9	5.7%	56.6%
4	8.5	5.2%	61.8%
5	7.4	4.0%	65.7%

Claude Opus 4.6) exhibit strong performance on tasks requiring novel reasoning strategies, while models with low Factor 2 are “conventionally capable” but struggle with frontier challenges.

**Practical meaning.** That the matrix is rank-2 means: knowing just two numbers about a model—its “general capability” score and its “frontier reasoning” score—allows prediction of its performance across all 49 benchmarks with  $\sim 7.5\%$  median error. The remaining 49% of variance consists of benchmark-specific noise and idiosyncratic model behaviors that do not generalize.

## 7 Data Efficiency and Phase Transition

### 7.1 Accuracy vs. Number of Known Scores

Figure 5 shows how LogitSVD Blend accuracy scales with the number of known benchmark scores per model.

The key observations are:

- **Biggest gains from 1→5 scores.** MedAPE drops from 12.1% (1 known score) to 9.2% (5 known scores). Each additional benchmark narrows the uncertainty about where the model sits in the 2D latent space.
- **MedAPE plateaus, but precision improves.** After 5 scores, MedAPE fluctuates around 9–10%, but the  $\pm 3$  and  $\pm 5$  point accuracy metrics continue improving (24.7%  $\rightarrow$  44.4% for  $\pm 3$  pts as  $n$  goes from 1 to 20).
- **Non-monotonicity at  $n > 10$**  reflects small sample sizes: few models have exactly 12 or 17 known scores, and the variance is wide.

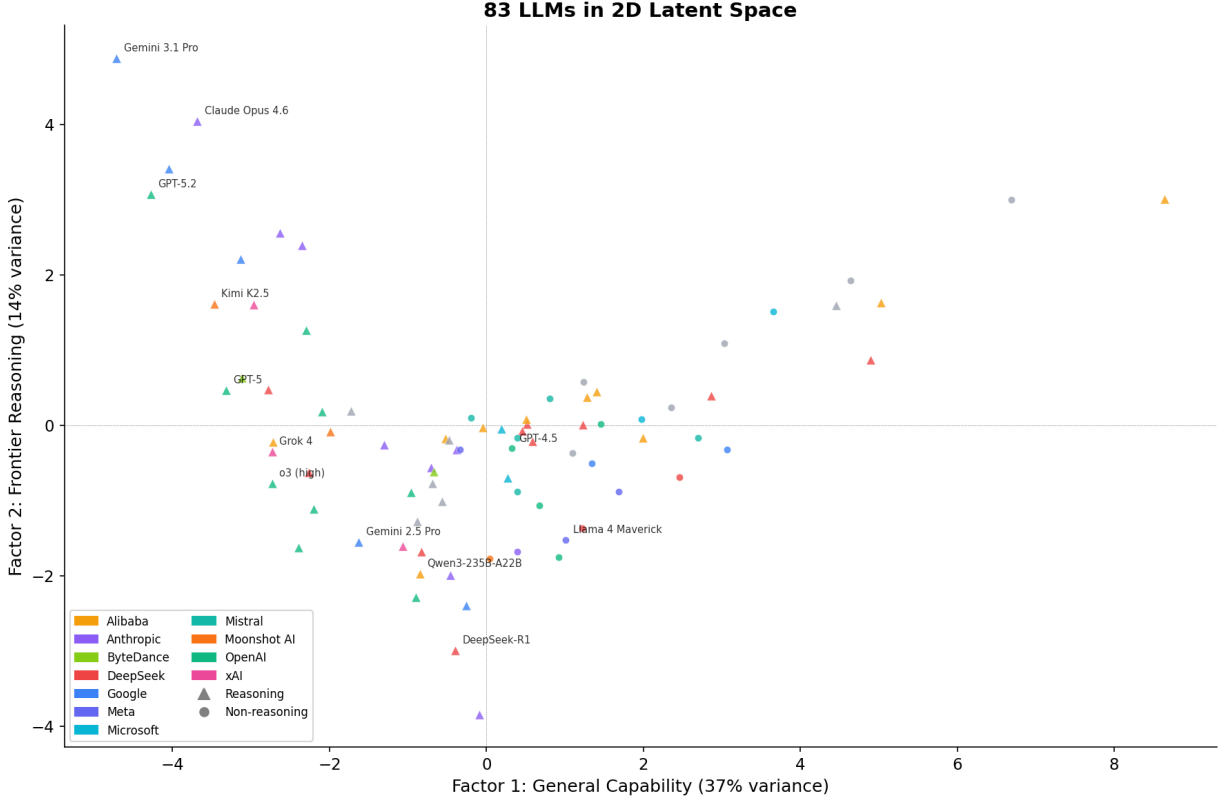


Figure 4: **Two-factor latent space.** Each point represents a model, colored by provider. Factor 1 (“general capability”) runs left-to-right; Factor 2 (“frontier reasoning”) runs bottom-to-top. Frontier models (o3, Claude Opus 4.6, Gemini 2.5 Pro) cluster in the upper-right.

## 7.2 Minimum Evaluation Set

Using greedy forward selection—at each step adding the benchmark that most reduces leave-one-out prediction error on remaining benchmarks—we identify a 5-benchmark “minimum eval set”:

$$\{\text{HLE, AIME 2025, LiveCodeBench, SWE-bench Verified, SimpleQA}\}$$

These five benchmarks span four categories (reasoning, math, coding, knowledge) and load on both latent factors. Under proper holdout evaluation (random 20%), this 5-benchmark Ridge predictor achieves  $\sim 7.8\%$  MedAPE compared to LogitSVD Blend’s 6.0%. The in-sample figure of 4.8% MedAPE is optimistic due to train-test leakage.

**Practical note.** GPQA Diamond is a near-perfect substitute for HLE (in-sample: 4.80% vs. 4.84%) and has  $2\times$  the model coverage (81 vs. 38 models), making it the pragmatic choice if only established benchmarks are available.

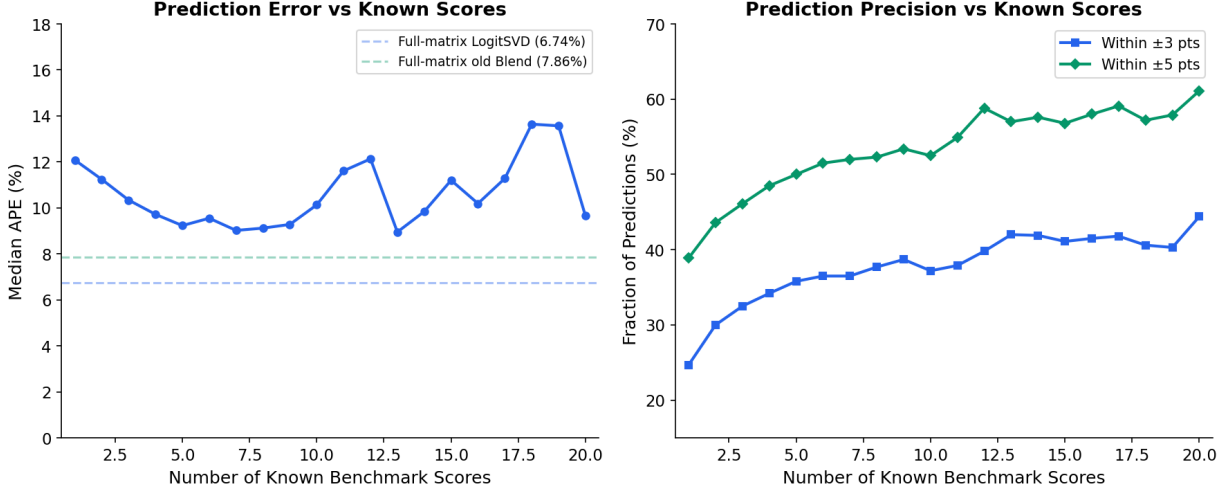


Figure 5: **Phase transition.** Left: MedAPE as a function of the number of known benchmark scores per model. The steepest improvement occurs from 1 to 5 known scores. Right: fraction of predictions within  $\pm 3$  and  $\pm 5$  score points, which continues improving beyond 5.

## 8 Scaling Laws and Reasoning Analysis

### 8.1 Within-Family Scaling

Within model families that vary only in parameter count, benchmark scores scale approximately log-linearly with model size. Table 8 shows this for two families with sufficient size variation.

Table 8: **Within-family scaling laws.**  $R^2$  of log-linear fit ( $\text{score} \sim \ln(\text{params})$ ) for selected benchmarks.

Benchmark	Qwen3 (0.6B–235B)		DeepSeek-R1-Distill (1.5B–70B)	
	$R^2$	Slope/ $\ln$	$R^2$	Slope/ $\ln$
MMLU	0.89	+5.9	—	—
GPQA Diamond	0.84	+7.5	0.95	+8.5
LiveCodeBench	0.83	+9.4	0.89	+12.0
IFEval	—	—	0.98	+11.3
Codeforces	0.79	+214.6	0.90	+205.6
AIME 2025	0.77	+11.4	—	—

The DeepSeek distillation family shows remarkably tight scaling ( $R^2=0.95\text{--}0.98$ ), suggesting that distillation preserves scaling behavior better than independent training. The Qwen3 family shows looser fits ( $R^2=0.77\text{--}0.89$ ), consistent with each size being trained independently.

### 8.2 Cross-Family Scaling Failure

Despite within-family regularity, cross-family scaling is poor. A 70B model from one provider can outperform a 235B model from another due to differences in training data, architecture (dense vs. MoE), and optimization. This explains why parameter count is not a useful prediction feature: the family-specific intercepts vary too much.

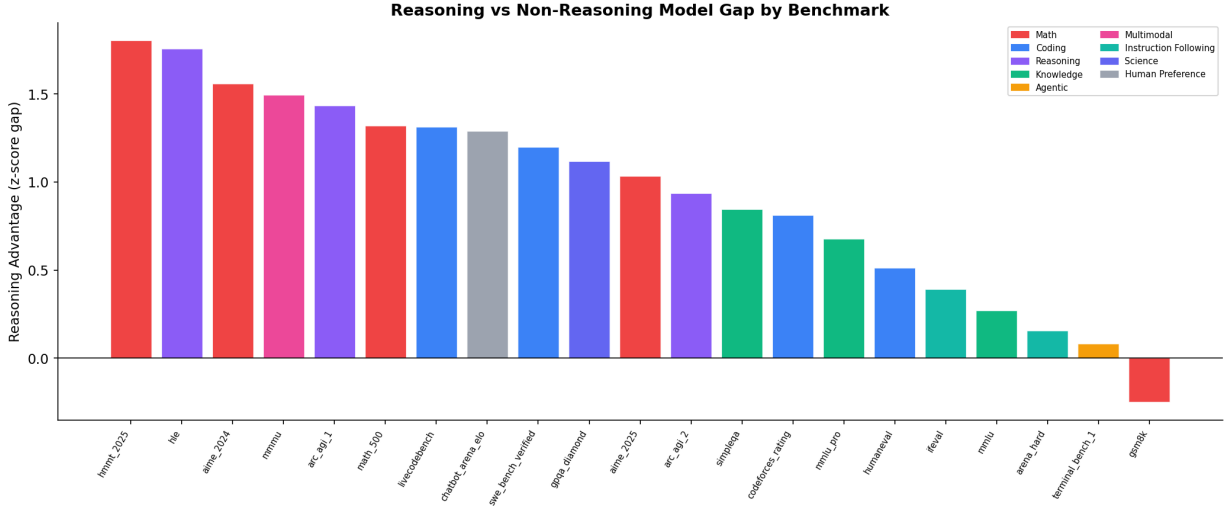


Figure 6: **Reasoning mode advantage by benchmark.** Positive values indicate benchmarks where reasoning models outperform non-reasoning models (z-score gap). HMMT and HLE show the largest reasoning advantage; GSM8K is the only benchmark where non-reasoning models win.

### 8.3 Reasoning Mode Analysis

Table 9 shows the benchmarks with the largest and smallest reasoning advantage, measured as the z-score gap between reasoning ( $n=57$ ) and non-reasoning ( $n=26$ ) models.

Table 9: **Reasoning advantage by benchmark** (z-score gap: reasoning – non-reasoning mean).

Largest Advantage				Smallest Advantage			
Benchmark	R	NR	Gap	Benchmark	R	NR	Gap
HMMT Feb	+0.24	−1.56	+1.80	IFEval	+0.13	−0.26	+0.39
HLE	+0.18	−1.57	+1.75	MMLU	+0.09	−0.18	+0.27
AIME 2024	+0.43	−1.13	+1.56	Arena-Hard	+0.06	−0.09	+0.15
MMMU	+0.40	−1.10	+1.49	GSM8K	−0.08	+0.17	−0.25
ARC-AGI-1	+0.31	−1.12	+1.43				

**The GSM8K anomaly.** GSM8K is the *only* benchmark where non-reasoning models outperform reasoning models on average. This occurs because GSM8K is now saturated—most frontier models score 90%+—and the extended reasoning overhead provides no benefit on simple arithmetic word problems. This represents a clear case of benchmark saturation, where a formerly diagnostic evaluation loses discriminative power.

## 9 LLMs as Benchmark Predictors

An intriguing alternative to statistical matrix completion is to ask a capable LLM to predict benchmark scores directly. We test this by providing the full  $83 \times 49$  matrix as context to several Claude models.

## 9.1 Experimental Setup

We format the entire observed matrix as a CSV string (with ? for missing entries) and include it in the system prompt along with structural context: benchmark names, model names, score ranges, and the instruction to predict missing values. For each batch of  $\sim 10$  models, the user prompt lists each model’s known scores and requests predictions for its missing scores in JSON format.

- **Context:** Full matrix CSV ( $\sim 12,600$  input tokens per batch)
- **Batching:** 8 batches of  $\sim 10$  models each (276 test cells in random 20% holdout)
- **Output:** JSON with model name  $\rightarrow$  benchmark  $\rightarrow$  predicted score
- **Post-processing:** Fuzzy model name matching, clamping to valid ranges

## 9.2 Results

Table 10 compares four Claude model variants against the statistical methods on the same random 20% holdout (seed=42, 276 test cells).

Table 10: **Claude-as-predictor results** vs. statistical baselines (random 20% holdout, 276 cells).

Method	MedAPE	MAE	$\pm 3$	$\pm 5$	BiAcc	Cov	Cost
<b>Claude Sonnet 4.5</b>	<b>5.45%</b>	17.49	44.2%	58.7%	84.2%	100%	\$0.85
Claude Sonnet 4	5.53%	—	—	—	—	100%	\$0.84
LogitSVD Blend	5.84%	18.36	42.0%	57.2%	89.5%	100%	\$0
BenchReg	6.47%	23.30	38.0%	53.3%	94.4%	87.7%	\$0
Claude Opus 4	6.48%	—	—	—	—	100%	\$4.02
Claude Sonnet 4.6	8.34–9.21%	20–21	34%	43–45%	74–84%	100%	\$0.80
Benchmark Mean	12.30%	33.43	15.9%	26.1%	68.4%	100%	\$0

## 9.3 Analysis

Several findings emerge:

1. **Sonnet 4.5 is the best predictor**, achieving 5.45% MedAPE—7% better than LogitSVD Blend (5.84%) on the same holdout. This is notable because the LLM has no explicit access to the logit transform or SVD decomposition; it must infer relationships from the raw tabular data in its context window.
2. **Model size is not monotonically helpful**. Opus 4 (6.48%) underperforms both Sonnet variants, and Sonnet 4.6 (8.34–9.21%) is substantially worse than Sonnet 4.5. This suggests that the task rewards precise tabular reasoning over raw reasoning power.
3. **Statistical methods win on bimodal accuracy**. LogitSVD achieves 89.5% bimodal accuracy vs. Claude Sonnet 4.5’s 84.2%. The logit transform gives statistical methods an explicit advantage on bimodal benchmarks.
4. **Cost-accuracy tradeoff**. At \$0.85 per run, the Claude predictor is inexpensive in absolute terms but infinitely more expensive than the zero-cost statistical methods. For one-off predictions, the LLM approach is practical; for large-scale or repeated evaluation, statistical methods are preferred.

## 10 Discussion

### 10.1 Surprising Models

Some models deviate markedly from rank-2 SVD expectations. GPT-4.5 scores only 0.8% on ARC-AGI-2 despite an expected score of  $\sim 17\%$  based on its overall profile—suggesting that its general capability does not translate to the novel visual-spatial reasoning ARC-AGI requires. Mistral Large 3 scores 44% on GPQA Diamond vs. an expected 67%, indicating a specific gap in graduate-level science knowledge. These outliers highlight the limits of low-rank approximation: while two factors capture the bulk of variation, individual model-benchmark interactions can be highly idiosyncratic.

### 10.2 Benchmark Redundancy

Within the “Frontier Reasoning” cluster (17 benchmarks including AIME 2025, FrontierMath, HLE, ARC-AGI-2, BrowseComp, SWE-bench Pro), pairwise correlations exceed 0.6: a model strong on one tends to be strong on all. The “Core Competency” cluster (GPQA Diamond, MMLU-Pro, LiveCodeBench, IFEval) shows similar redundancy. This redundancy is both a feature (it enables matrix completion) and a warning (many benchmarks measure overlapping capabilities).

However, we note a caveat: pairwise correlations are computed on shared observations, and the median number of shared models per benchmark pair is only 7. Correlations based on fewer than 20 shared models should be treated cautiously.

### 10.3 Why LLMs Can Predict Benchmarks

Claude Sonnet 4.5’s success as a predictor is initially surprising, but it likely reflects three factors: (1) the LLM has been trained on vast amounts of text about model capabilities, benchmark results, and the AI evaluation landscape; (2) the matrix itself is low-rank, so the prediction task is fundamentally about inferring two latent scores from context; and (3) LLMs excel at in-context tabular reasoning when the table fits in the context window ( $\sim 13\text{K}$  tokens per batch).

### 10.4 Limitations

- **34% fill rate.** The matrix is sparse. Additional scores could be mined from model papers, leaderboards, and community evaluations.
- **Blend weight not cross-validated.** The  $\alpha=0.6$  weight was selected by manual sweep over  $\{0.4, 0.5, 0.6, 0.7, 0.8\}$ . Nested cross-validation could yield a slightly different optimum.
- **Non-percentage benchmarks.** Elo ratings and Codeforces scores ( $\sim 8\%$  of benchmarks) do not benefit from the logit transform.
- **No temporal modeling.** The matrix treats all scores as contemporaneous, ignoring benchmark saturation over time (e.g., GSM8K becoming trivial) and model improvement trends.
- **Single evaluation window.** Models released January 2025–February 2026. The matrix will need updating as new models and benchmarks appear.

## 11 Related Work

**LLM evaluation.** Comprehensive benchmarking efforts include the Open LLM Leaderboard [5], HELM [6], and Chatbot Arena [7]. These provide standardized evaluations but do not address the missing data problem we tackle here.

**Matrix completion.** The theoretical foundations for low-rank matrix completion were established by Candès and Recht [1] and Candès and Tao [2]. Soft-impute SVD [3] provides the iterative algorithm we use. Our contribution is the application of logit-space transforms to benchmark score completion, which addresses domain-specific challenges (ceiling effects, bimodality) absent in generic matrix completion.

**Benchmark prediction.** Concurrent work on predicting benchmark scores includes scaling law extrapolation [4] and benchmark-specific forecasting. Our approach differs in treating the *cross-benchmark* prediction problem: given scores on some benchmarks, predict scores on others.

## 12 Conclusion and Future Work

We have constructed a fully-cited benchmark matrix of 83 LLMs across 49 evaluations and developed LogitSVD Blend, a matrix completion method achieving 6.74% median error on held-out scores. The key insight is that applying the logit transform before regression and factorization linearizes ceiling/floor effects and implicitly handles bimodal benchmarks, yielding a 14.2% relative improvement over the prior baseline. The matrix is approximately rank-2, with interpretable factors corresponding to “general capability” and “frontier reasoning.”

We also demonstrate that Claude Sonnet 4.5 can predict benchmark scores with 5.45% median error when given the matrix as context—outperforming all statistical methods at modest cost. This raises the intriguing possibility that LLMs encode implicit representations of model capabilities that can be leveraged for evaluation.

**Future work.** Several directions remain promising:

- **Active learning:** Choosing which benchmark to evaluate next for maximum information gain.
- **Temporal modeling:** Incorporating benchmark vintage and saturation curves to track capability evolution.
- **Confidence intervals:** Predicting uncertainty bounds alongside point estimates.
- **Larger matrices:** Expanding to more models, benchmarks, and evaluation conditions (few-shot, chain-of-thought, etc.).
- **Hybrid approaches:** Combining LLM predictions with statistical methods, potentially using LLM uncertainty to weight the blend.

**Reproducibility.** All code, data, and results are publicly available at [https://github.com/\[repo\]](https://github.com/[repo]).

## References

- [1] E. J. Candès and B. Recht, “Exact matrix completion via convex optimization,” *Foundations of Computational Mathematics*, vol. 9, no. 6, pp. 717–772, 2009.
- [2] E. J. Candès and T. Tao, “The power of convex relaxation: Near-optimal matrix completion,” *IEEE Transactions on Information Theory*, vol. 56, no. 5, pp. 2053–2080, 2010.
- [3] R. Mazumder, T. Hastie, and R. Tibshirani, “Spectral regularization algorithms for learning large incomplete matrices,” *Journal of Machine Learning Research*, vol. 11, pp. 2287–2322, 2010.
- [4] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, “Scaling laws for neural language models,” *arXiv preprint arXiv:2001.08361*, 2020.
- [5] E. Beeching, C. Fourrier, N. Habber, et al., “Open LLM Leaderboard,” Hugging Face, 2023–2026.
- [6] P. Liang, R. Bommasani, T. Lee, et al., “Holistic Evaluation of Language Models,” *Transactions on Machine Learning Research*, 2023.
- [7] W.-L. Chiang, L. Zheng, Y. Sheng, et al., “Chatbot Arena: An open platform for evaluating LLMs by human preference,” *arXiv preprint arXiv:2403.04132*, 2024.