How Benchmark Prediction from Fewer Data Misses the Mark

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Evaluating large language models (LLMs) is increasingly costly, motivating methods to speed up evaluation by compressing benchmark datasets. Benchmark prediction aims to select a small subset of evaluation points and predict overall performance from that subset. We systematically assess 11 benchmark prediction methods across 19 benchmarks. First, we identify a strong baseline: take a random sample and fit a regression to predict the missing entries, which outperforms most existing methods and challenges the need for careful subset selection. Second, we show that all methods rely on model similarity: performance degrades markedly when extrapolating to stronger models than those used for training, where few methods beat a simple sample average. We introduce an augmented inverse propensity weighting (AIPW) estimator that consistently improves over the random sample average under both interpolation and extrapolation, though gains remain modest and still depend on similarity. This shows that benchmark prediction fails just when it is most needed: at the evaluation frontier, where the goal is to evaluate new models of unknown capabilities.

1 Introduction

Computational cost is a major bottleneck in evaluating recent generative models. For example, evaluating a single 176B model on HELM required 4,200 GPU hours [35]; even large organizations report heavy costs on BIG-bench [17]. This has prompted work on efficient LLM evaluation through *benchmark prediction*: finding a subset of data points to evaluate on and predicting benchmark performance from these evaluations. The simplest method is the random sample mean: evaluate n evaluation points and average, which gives an additive approximation up to error $O(1/\sqrt{n})$. Recent work aims to improve this by selecting an informative *core set* and learning mappings from core set to full-benchmark performance. See the discussion of related work in Appendix A.

We systematically study the strengths and limits of these methods by evaluating 11 benchmark prediction methods across 19 benchmarks with at least 83 models each. Models are split into source models (full performance data available) and target models (performance data for no more than 50 points). Methods must estimate target models' mean performance using this constraint. Effectiveness is measured by *average estimation gap*-the absolute difference between true and estimated performances.

Many methods work well on similar models, but a simple baseline works best. In the *interpolation* regime (source and target models from same distribution), a remarkably simple method works best: RANDOM-SAMPLING-LEARN-random sampling followed by regression modeling-reduces the estimation gap by 37% compared to basic random sampling, outperforming most sophisticated methods. This suggests that the manner of core-set selection is relatively unimportant; rather, the key to success is modeling the correlation between core-set and full-benchmark performances.

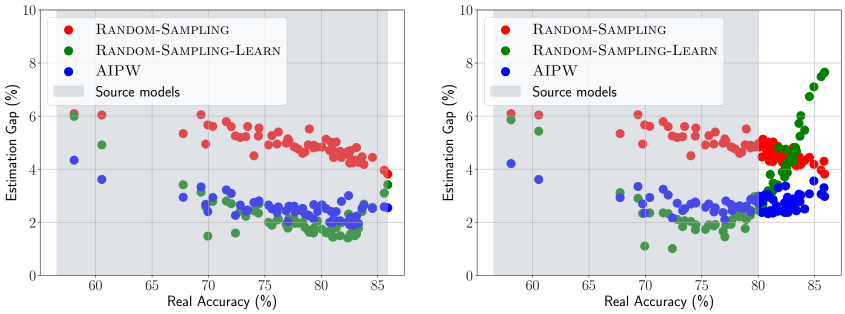


Figure 1: Estimation gap (equation 1) versus real accuracy on `ImageNet`. Gray shows source model accuracy range. Left: source models randomly sampled across all models. Right: source models sampled from models with lower than 80% accuracy.

Methods fail at the evaluation frontier. In the *extrapolation* regime (target models all better than source models), effectiveness drops sharply. Most methods fail to beat naive random sampling when evaluating new, better models-precisely when efficient evaluation is most needed (Figure 1, right).

AIPW is an overlooked exception to the rule. We introduce augmented inverse propensity weighting (AIPW) to benchmark prediction. Unlike other methods, AIPW consistently outperforms random sampling in both interpolation and extrapolation settings. However, as illustrated in Figure 1 (right), even AIPW sees diminishing improvements as target models' accuracies exceed those of the sources.

Benchmark prediction relies on model similarity. Our further study reveals that benchmark prediction methods rely heavily on model similarity [38]: methods that beat Random-Sampling do so mainly for targets similar to sources, while accuracy on disimilar models deteriorates. In contrast, Random-Sampling exhibits neutral correlation.

2 What is Benchmark Prediction?

Problem formulation. A benchmark is defined as $(\mathcal{D}, \mathcal{F}, s)$ where $\mathcal{D}$ is the dataset with N data points, $\mathcal{F}$ is the model set, and s is the evaluation metric. For any model $f \in F$ and data point $z = (x, y) \in D$, we define notation as follows.

- $s(f, z)$ denotes performance of f on point z, for example, $\mathbb{1}[f(x) = y]$ for accuracy.
- $\bar{s}(f, \mathcal{D}') = \frac{1}{|\mathcal{D}'|} \sum_{z \in \mathcal{D}'} s(f, z)$ denotes average performance on subset $\mathcal{D}' \subset \mathcal{D}$.
- $\boldsymbol{s}(f, \mathcal{D}')$ denotes the vectorized performance of f on $\mathcal{D}' \subset \mathcal{D}$, and $\boldsymbol{s}(\mathcal{F}', z)$ denotes the vectorized performances of all models in $\mathcal{F}' \subset \mathcal{F}$ on data point z.
- $S(\mathcal{F}', \mathcal{D}')$ denotes the performance matrix for models $\mathcal{F}' \subset \mathcal{F}$ on points $\mathcal{D}' \subset \mathcal{D}$.

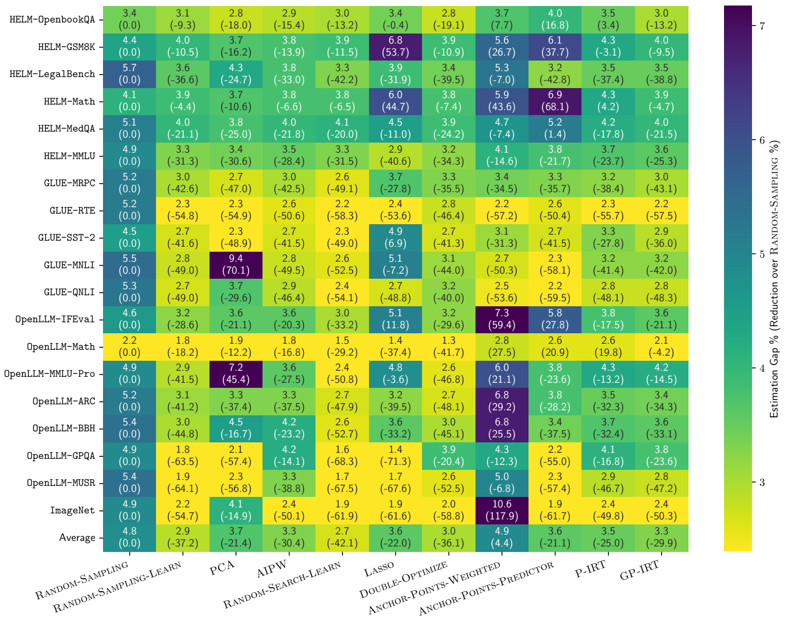
Given source models $\mathcal{F}^{(s)} \subset \mathcal{F}$ with known full performance $S(\mathcal{F}^{(s)}, \mathcal{D})$ and target models $\mathcal{F}^{(t)} = \mathcal{F} \setminus \mathcal{F}^{(s)}$ evaluated on only $n \ll N$ points, benchmark prediction aims to estimate $\bar{s}(f, \mathcal{D})$ for each $f \in \mathcal{F}^{(t)}$ by: ① selecting core-set $\mathcal{C} \subset \mathcal{D}$ with $|\mathcal{C}| = n$, and ② learning estimator h to minimize:

$$\text{estimation gap: } \frac{1}{|\mathcal{F}^{(t)}|} \sum_{f \in \mathcal{F}^{(t)}} \left| \bar{s}(f, \mathcal{D}) - h\big[\boldsymbol{s}(f, \mathcal{C}), S(\mathcal{F}^{(s)}, \mathcal{D})\big] \right|. \tag{1}$$

Previous benchmark prediction methods:

- Random-Sampling: pick $\mathcal{C}$ at random; return the mean on $\mathcal{C}$.
- Anchor-Points-Weighted [60]: k-medoids to select $\mathcal{C}$; return weighted sum by cluster density.
- Anchor-Points-Predictor [60]: as above, then linear regression from $\boldsymbol{s}(f, \mathcal{C})$ to $\bar{s}(f, \mathcal{D})$.
- P-IRT [43]: as above, replace regression with the Item Response Theory (IRT) model.
- GP-IRT [43]: combine P-IRT with Anchor-Points-Weighted aggregation.

New methods introduced:



	Random-Sampling	Random-Sampling-Learn	PCA	AIPW	Random-Search-Learn	Lasso	Double-Optimize	Anchor-Points-Weighted	Anchor-Points-Predictor	P-IRT	GP-IRT
HELM-OpenbookQA	3.4 (0.0)	3.1 (-9.3)	2.8 (-18.0)	2.9 (-15.4)	3.0 (-13.2)	3.4 (-0.4)	2.8 (-19.1)	3.7 (7.7)	4.0 (16.8)	3.5 (3.4)	3.0 (-13.2)
HELM-GSM8K	4.4 (0.0)	4.0 (-10.5)	3.7 (-16.2)	3.8 (-13.9)	3.9 (-11.5)	6.8 (53.7)	3.9 (-10.9)	5.6 (26.7)	6.1 (37.7)	4.3 (-3.1)	4.0 (-9.5)
HELM-LegalBench	5.7 (0.0)	3.6 (-36.6)	4.3 (-24.7)	3.8 (-33.0)	3.3 (-42.2)	3.9 (-31.9)	3.4 (-39.5)	5.3 (-7.0)	3.2 (-42.8)	3.5 (-37.4)	3.5 (-38.8)
HELM-Math	4.1 (0.0)	3.9 (-4.4)	3.7 (-10.6)	3.8 (-6.6)	3.8 (-6.5)	6.0 (44.7)	3.8 (-7.4)	5.9 (43.6)	6.9 (68.1)	4.3 (4.2)	3.9 (-4.7)
HELM-MedQA	5.1 (0.0)	4.0 (-21.1)	3.8 (-25.0)	4.0 (-21.8)	4.1 (-20.0)	4.5 (-11.0)	3.9 (-24.2)	4.7 (-7.4)	5.2 (1.4)	4.2 (-17.8)	4.0 (-21.5)
HELM-MMLU	4.9 (0.0)	3.3 (-31.3)	3.4 (-30.6)	3.5 (-28.4)	3.3 (-31.5)	2.9 (-40.6)	3.2 (-34.3)	4.1 (-14.6)	3.8 (-21.7)	3.7 (-23.7)	3.6 (-25.3)
GLUE-MRPC	5.2 (0.0)	3.0 (-42.6)	2.7 (-47.0)	3.0 (-42.5)	2.6 (-49.1)	3.7 (-27.8)	3.3 (-35.5)	3.4 (-34.5)	3.3 (-35.7)	3.2 (-38.4)	3.0 (-43.1)
GLUE-RTE	5.2 (0.0)	2.3 (-54.8)	2.3 (-54.9)	2.6 (-50.6)	2.2 (-58.3)	2.4 (-53.6)	2.8 (-46.4)	2.2 (-57.2)	2.6 (-50.4)	2.3 (-55.7)	2.2 (-57.5)
GLUE-SST-2	4.5 (0.0)	2.7 (-41.6)	2.3 (-48.9)	2.7 (-41.5)	2.3 (-49.0)	4.9 (6.9)	2.7 (-41.3)	3.1 (-31.3)	2.7 (-41.5)	3.3 (-27.8)	2.9 (-36.0)
GLUE-MNLI	5.5 (0.0)	2.8 (-49.0)	9.4 (70.1)	2.8 (-49.5)	2.6 (-52.5)	5.1 (-7.2)	3.1 (-44.0)	2.7 (-50.3)	2.3 (-58.1)	3.2 (-41.4)	3.2 (-42.0)
GLUE-QNLI	5.3 (0.0)	2.7 (-49.0)	3.7 (-29.6)	2.9 (-46.4)	2.4 (-54.1)	2.7 (-48.8)	3.2 (-40.0)	2.5 (-53.6)	2.2 (-59.5)	2.8 (-48.1)	2.8 (-48.3)
OpenLLM-IFEval	4.6 (0.0)	3.2 (-28.6)	3.6 (-21.1)	3.6 (-20.3)	3.0 (-33.2)	5.1 (11.8)	3.2 (-29.6)	7.3 (59.4)	5.8 (27.8)	3.8 (-17.5)	3.6 (-21.1)
OpenLLM-Math	2.2 (0.0)	1.8 (-18.2)	1.9 (-12.2)	1.8 (-16.8)	1.5 (-29.2)	1.4 (-37.4)	1.3 (-41.7)	2.8 (27.5)	2.6 (20.9)	2.6 (19.8)	2.1 (-4.2)
OpenLLM-MMLU-Pro	4.9 (0.0)	2.9 (-41.5)	7.2 (45.4)	3.6 (-27.5)	2.4 (-50.8)	4.8 (-3.6)	2.6 (-46.8)	6.0 (21.1)	3.8 (-23.6)	4.3 (-13.2)	4.2 (-14.5)
OpenLLM-ARC	5.2 (0.0)	3.1 (-41.2)	3.3 (-37.4)	3.3 (-37.5)	2.7 (-47.9)	3.2 (-39.5)	2.7 (-48.1)	6.8 (29.2)	3.8 (-28.2)	3.5 (-32.3)	3.4 (-34.3)
OpenLLM-BBH	5.4 (0.0)	3.0 (-44.8)	4.5 (-16.7)	4.2 (-23.2)	2.6 (-52.7)	3.6 (-33.2)	3.0 (-45.1)	6.8 (25.5)	3.4 (-37.5)	3.7 (-32.4)	3.6 (-33.1)
OpenLLM-GPQA	4.9 (0.0)	1.8 (-63.5)	2.1 (-57.4)	4.2 (-14.1)	1.6 (-68.3)	1.4 (-71.3)	3.9 (-20.4)	4.3 (-12.3)	2.2 (-55.0)	4.1 (-16.8)	3.8 (-23.6)
OpenLLM-MUSR	5.4 (0.0)	1.9 (-64.1)	2.3 (-56.8)	3.3 (-38.8)	1.7 (-67.5)	1.7 (-67.6)	2.6 (-52.5)	5.0 (-6.8)	2.3 (-57.4)	2.9 (-46.7)	2.8 (-47.2)
ImageNet	4.9 (0.0)	2.2 (-54.7)	4.1 (-14.9)	2.4 (-50.1)	1.9 (-61.9)	1.9 (-61.6)	2.0 (-58.8)	10.6 (117.9)	1.9 (-61.7)	2.4 (-49.8)	2.4 (-50.3)
Average	4.8 (0.0)	2.9 (-37.2)	3.7 (-21.4)	3.3 (-30.4)	2.7 (-42.1)	3.6 (-22.0)	3.0 (-36.1)	4.9 (4.4)	3.6 (-21.1)	3.5 (-25.0)	3.3 (-29.9)

Figure 2: The estimation gaps (↓) for target models (equation 1) under the interpolation split, where source and target models are identically distributed. Each target is evaluated on $n = 50$ data points. The estimation gap reduction (↓) over RANDOM-SAMPLING is shown in parentheses. A negative reduction means that the method achieves a lower gap than RANDOM-SAMPLING. % is omitted.

- RANDOM-SAMPLING-LEARN: pick $\mathcal{C}$ at random; Ridge regression from $\boldsymbol{s}(f, \mathcal{C})$ to $\bar{s}(f, \mathcal{D})$.
- RANDOM-SEARCH-LEARN: run RANDOM-SAMPLING-LEAR 10,000 times; select best $\mathcal{C}$ and h.
- LASSO: Lasso regression with sparsity constraint number of non-zero weights $\le n$.
- DOUBLE-OPTIMIZE: gradient descent for joint core-set and regression optimization.
- PCA: use random $\mathcal{C}$; impute target scores with PCA assuming $S(\mathcal{F}, \mathcal{D})$ is low-rank.
- AIPW [48]: train regression g to predict $s(f, z)$ from $\boldsymbol{s}(\mathcal{F}^{(s)}, z)$ for every f. The idea is to use the predicted performance $\hat{s}(f, z) = g[\boldsymbol{s}(\mathcal{F}^{(s)}, z)]$ as a proxy score of $s(f, z)$ and "debias" as follows

$$h^{\text{AIPW}}(f) = \bar{s}(f, \mathcal{C}) + \frac{1}{1 + \frac{n}{N-n}} \left(\frac{1}{N-n} \sum_{z \in \mathcal{D} - \mathcal{C}} \hat{s}(f, z) - \frac{1}{n} \sum_{z \in \mathcal{C}} \hat{s}(f, z) \right). \tag{2}$$

AIPW is a consistent estimator for $\bar{s}(f, \mathcal{D})$[19]. Compared to RANDOM-SAMPLING, it reduces estimator variance by a factor of up to $\frac{1}{1+\frac{n}{N}} \rho(\hat{s}(f, z), s(f, z))^2$ [14]. See more details in Appendix B.

3 Experiments

We examine the 11 benchmark prediction methods under both interpolation and extrapolation settings. We select 19 benchmarks from `HELM-Lite` [35], `OpenLLM` [16], `GLUE` [61] and `ImageNet` [51], each with at least 83 models. See more details in Appendix C.

Estimation gap reduction under interpolation. For each benchmark, we randomly select 75% of models as source models $\mathcal{F}^{(s)}$ with full performance scores $S(\mathcal{F}^{(s)}, \mathcal{D})$ available. The remaining 25% serve as target models $\mathcal{F}^{(t)}$, each evaluated on only $n = 50$ data points. We report average estimation gap across all target models in 100 random trials. See standard errors in Appendix D.

Figure 2 shows that compared to RANDOM-SAMPLING, most methods effectively reduce the estimation gap, with nine of ten achieving more than 20% average reduction across benchmarks. The simple baseline RANDOM-SEARCH-LEARN performs best with 42.1% average reduction, outperforming the previous state-of-the-art GP-IRT (29.9% average reduction) on nearly all benchmarks.

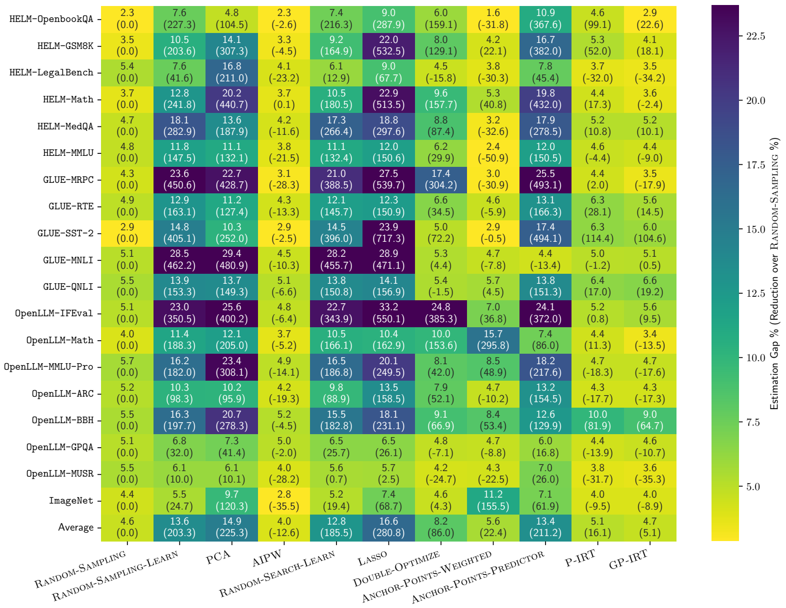


Figure 3: The estimation gaps (↓) for target models (equation 1) under extrapolation split, where source models are the lowest-performing 50%, and target models are the top 30%. Each target model is evaluated on $n = 50$ data points. We also report the estimation gap reduction (↓) over RANDOM-SAMPLING in parentheses. A negative reduction implies that the method achieves a lower estimation gap than RANDOM-SAMPLING. % is omitted.

Core-set selection does not significantly enhance effectiveness. RANDOM-SAMPLING-LEARN achieves 37.2% average reduction using only Ridge regression on random samples, performing comparably to RANDOM-SEARCH-LEARN despite the latter's 10,000 optimization iterations. It also surpasses methods with sophisticated subset selection like DOUBLE-OPTIMIZE and GP-IRT. These suggest the primary driver of success is learning to predict the mean rather than core-set selection.

Estimation gap increase under extrapolation. We then examine all 11 methods under *extrapolation*. Models are ranked by full benchmark performance $\bar{s}(f, \mathcal{D})$. The bottom 50% become source models, while the top 30% serve as target models, reflecting real-world scenarios where developers assess improved models based on existing inferior ones.

Figure 3 reveals striking differences from interpolation. While RANDOM-SAMPLING's estimation gap remains similar (4.6% vs 4.8%), all other methods deteriorate significantly. The previously best RANDOM-SEARCH-LEARN now shows 185.1% increase in estimation gap versus RANDOM-SAMPLING, performing worse across all benchmarks. Only AIPW still outperforms RANDOM-SAMPLING (in 18/19 benchmarks) as it is a consistent estimator, though its advantage shrinks from -30.4% to -12.6% reduction.

This contrast underscores most methods' heavy reliance on source-target similarity. See a deeper analysis of model similarity and ablation studies in Appendix D. While traditional machine learning emphasizes in-domain performance, benchmarking aims to identify superior new models, making extrapolation more relevant than interpolation. The decline in benchmark prediction effectiveness under extrapolation calls for more caution.

4 Conclusion

Our findings suggest that while benchmark prediction techniques can be useful in specific scenarios, their reliance on similarity between source and target models poses a risk of misestimating the performance of new models. This underscores the importance of applying these methods with caution, especially for evaluating models that significantly deviate from previous ones. See more detailed discussion in Appendix E.

References

[1] Anastasios Nikolas Angelopoulos, Stephen Bates, Clara Fannjiang, Michael I. Jordan, and Tijana Zrnic. Prediction-powered inference. *Science*, 382:669 – 674, 2023.

[2] Anastasios Nikolas Angelopoulos, John C. Duchi, and Tijana Zrnic. Ppi++: Efficient prediction-powered inference. *ArXiv*, abs/2311.01453, 2023.

[3] Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *ArXiv*, abs/1308.3432, 2013.

[4] Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. The fifth PASCAL recognizing textual entailment challenge. 2009.

[5] Pierre Boyeau, Anastasios N Angelopoulos, Nir Yosef, Jitendra Malik, and Michael I Jordan. Autoeval done right: Using synthetic data for model evaluation. *arXiv preprint arXiv:2403.07008*, 2024.

[6] Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on optimization*, 20(4):1956–1982, 2010.

[7] Guiming Hardy Chen, Shunian Chen, Ziche Liu, Feng Jiang, and Benyou Wang. Humans or llms as the judge? a study on judgement biases. *arXiv preprint arXiv:2402.10669*, 2024.

[8] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*, 2018.

[9] Karl Cobbe, Vineet Kosaraju, Mo Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *ArXiv*, abs/2110.14168, 2021.

[10] Ciprian A Corneanu, Sergio Escalera, and Aleix M Martinez. Computing the testing error without a testing set. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2677–2685, 2020.

[11] Ido Dagan, Oren Glickman, and Bernardo Magnini. The PASCAL recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*, pages 177–190. Springer, 2006.

[12] Weijian Deng and Liang Zheng. Are labels always necessary for classifier accuracy evaluation? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15069–15078, 2021.

[13] William B Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the International Workshop on Paraphrasing*, 2005.

[14] Florian E Dorner, Vivian Yvonne Nastl, and Moritz Hardt. Limits to scalable evaluation at the frontier: Llm as judge won't beat twice the data. In *The Thirteenth International Conference on Learning Representations*, 2025.

[15] Reza Zanjirani Farahani and Masoud Hekmatfar. Facility location: concepts, models, algorithms and case studies. 2009.

[16] Clémentine Fourrier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. Open llm leaderboard v2. `https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard`, 2024.

[17] Deep Ganguli, Nicholas Schiefer, Marina Favaro, and Jack Clark. Challenges in evaluating AI systems, 2023.

[18] Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9. Association for Computational Linguistics, 2007.

[19] Adam N Glynn and Kevin M Quinn. An introduction to the augmented inverse propensity weighted estimator. *Political analysis*, 18(1):36–56, 2010.

[20] Shashwat Goel, Joschka Struber, Ilze Amanda Auzina, Karuna K. Chandra, Ponnurangam Kumaraguru, Douwe Kiela, Ameya Prabhu, Matthias Bethge, and Jonas Geiping. Great models think alike and this undermines ai oversight. *ArXiv*, abs/2502.04313, 2025.

[21] Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Yuanzhuo Wang, and Jian Guo. A survey on llm-as-a-judge. *ArXiv*, abs/2411.15594, 2024.

[22] Neel Guha, Julian Nyarko, Daniel E. Ho, Christopher Ré, Adam Chilton, Aditya Narayana, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel N. Rockmore, Diego Zambrano, Dmitry Talisman, Enam Hoque, Faiz Surani, Frank Fagan, Galit Sarfaty, Gregory M. Dickinson, Haggai Porat, Jason Hegland, Jessica Wu, Joe Nudell, Joel Niklaus, John Nay, Jonathan H. Choi, Kevin Tobia, Margaret Hagan, Megan Ma, Michael Livermore, Nikon Rasumov-Rahe, Nils Holzenberger, Noam Kolt, Peter Henderson, Sean Rehaag, Sharad Goel, Shang Gao, Spencer Williams, Sunny Gandhi, Tom Zur, Varun Iyer, and Zehua Li. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models, 2023.

[23] Veronika Hackl, Alexandra Elena Müller, Michael Granitzer, and Maximilian Sailer. Is gpt-4 a reliable rater? evaluating consistency in gpt-4 text ratings. *ArXiv*, abs/2308.02575, 2023.

[24] Moritz Hardt and Yu Sun. Test-time training on nearest neighbors for large language models. *arXiv preprint arXiv:2305.18466*, 2023.

[25] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Xiaodong Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *ArXiv*, abs/2009.03300, 2020.

[26] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Xiaodong Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *ArXiv*, abs/2103.03874, 2021.

[27] Eric Jang, Shixiang Shane Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *ArXiv*, abs/1611.01144, 2016.

[28] Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *ArXiv*, abs/2009.13081, 2020.

[29] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[30] Alex Kipnis, Konstantinos Voudouris, Luca M. Schulze Buschoff, and Eric Schulz. metabench – a sparse benchmark of reasoning and knowledge in large language models. 2024.

[31] Jannik Kossen, Sebastian Farquhar, Yarin Gal, and Tom Rainforth. Active testing: Sample-efficient model evaluation. In *International Conference on Machine Learning*, 2021.

[32] Jannik Kossen, Sebastian Farquhar, Yarin Gal, and Tom Rainforth. Active surrogate estimators: An active learning approach to label-efficient model evaluation. *ArXiv*, abs/2202.06881, 2022.

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

[34] Yang Li, Jie Ma, Miguel Ballesteros, Yassine Benajiba, and Graham Horwood. Active evaluation acquisition for efficient llm benchmarking. *ArXiv*, abs/2410.05952, 2024.

[35] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher R'e, Diana Acosta-Navas, Drew A. Hudson, E. Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel J. Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan S. Kim, Neel Guha, Niladri S. Chatterji, O. Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas F. Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. Holistic evaluation of language models. *Annals of the New York Academy of Sciences*, 1525:140 – 146, 2023.

[36] Lovish Madaan, Aaditya K. Singh, Rylan Schaeffer, Andrew Poulton, Oluwasanmi Koyejo, Pontus Stenetorp, Sharan Narang, and Dieuwke Hupkes. Quantifying variance in evaluation benchmarks. *ArXiv*, abs/2406.10229, 2024.

[37] Pranav Mani, Peng Xu, Zachary Chase Lipton, and Michael Oberst. No free lunch: Non-asymptotic analysis of prediction-powered inference. *ArXiv*, abs/2505.20178, 2025.

[38] Horia Mania, John Miller, Ludwig Schmidt, Moritz Hardt, and Benjamin Recht. Model similarity mitigates test set overuse. *ArXiv*, abs/1905.12580, 2019.

[39] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[40] David Owen. How predictable is language model benchmark performance? *ArXiv*, abs/2401.04757, 2024.

[41] Lorenzo Pacchiardi, Konstantinos Voudouris, Ben Slater, Fernando Mart'inez-Plumed, Jos'e Hern'andez-Orallo, Lexin Zhou, and Wout Schellaert. Predictaboard: Benchmarking llm score predictability. *ArXiv*, abs/2502.14445, 2025.

[42] Arjun Panickssery, Samuel R. Bowman, and Shi Feng. Llm evaluators recognize and favor their own generations. *ArXiv*, abs/2404.13076, 2024.

[43] Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. tinybenchmarks: evaluating llms with fewer examples. *ArXiv*, abs/2402.14992, 2024.

[44] Felipe Maia Polo, Ronald Xu, Lucas Weber, Mírian Silva, Onkar Bhardwaj, Leshem Choshen, Allysson Flavio Melo de Oliveira, Yuekai Sun, and Mikhail Yurochkin. Efficient multi-prompt evaluation of llms. *arXiv preprint arXiv:2405.17202*, 2024.

[45] Ameya Prabhu, Vishaal Udandarao, Philip H. S. Torr, Matthias Bethge, Adel Bibi, and Samuel Albanie. Efficient lifelong model evaluation in an era of rapid progress. In *Neural Information Processing Systems*, 2024.

[46] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of EMNLP*, pages 2383–2392. Association for Computational Linguistics, 2016.

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

[48] James M Robins and Andrea Rotnitzky. Semiparametric efficiency in multivariate regression models with missing data. *Journal of the American Statistical Association*, 90(429):122–129, 1995.

[49] Pedro Rodriguez, Joe Barrow, Alexander Miserlis Hoyle, John P. Lalor, Robin Jia, and Jordan L. Boyd-Graber. Evaluation examples are not equally informative: How should that change nlp leaderboards? In *Annual Meeting of the Association for Computational Linguistics*, 2021.

[50] Yangjun Ruan, Chris J. Maddison, and Tatsunori B. Hashimoto. Observational scaling laws and the predictability of language model performance. *ArXiv*, abs/2405.10938, 2024.

[51] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211 – 252, 2014.

[52] Noveen Sachdeva and Julian McAuley. Data distillation: A survey. *Trans. Mach. Learn. Res.*, 2023, 2023.

[53] Chengshuai Shi, Kun Yang, Jing Yang, and Cong Shen. Best arm identification for prompt learning under a limited budget. *arXiv preprint arXiv:2402.09723*, 2024.

[54] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

[55] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*, pages 1631–1642, 2013.

[56] Zayne Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. Musr: Testing the limits of chain-of-thought with multistep soft reasoning. *ArXiv*, abs/2310.16049, 2023.

[57] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pages 9229–9248. PMLR, 2020.

[58] Mirac Suzgun, Nathan Scales, Nathanael Scharli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Annual Meeting of the Association for Computational Linguistics*, 2022.

[59] Roman Vershynin. Four lectures on probabilistic methods for data science. *ArXiv*, abs/1612.06661, 2016.

[60] Rajan Vivek, Kawin Ethayarajh, Diyi Yang, and Douwe Kiela. Anchor points: Benchmarking models with much fewer examples. *ArXiv*, abs/2309.08638, 2023.

[61] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *BlackboxNLP@EMNLP*, 2018.

[62] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max W.F. Ku, Kai Wang, Alex Zhuang, Rongqi "Richard" Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *ArXiv*, abs/2406.01574, 2024.

[63] Koki Wataoka, Tsubasa Takahashi, and Ryokan Ri. Self-preference bias in llm-as-a-judge. *ArXiv*, abs/2410.21819, 2024.

[64] Adina Williams, Nikita Nangia, and Samuel R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of NAACL-HLT*, 2018.

[65] Dingli Yu, Simran Kaur, Arushi Gupta, Jonah Brown-Cohen, Anirudh Goyal, and Sanjeev Arora. Skill-mix: a flexible and expandable family of evaluations for ai models. *ArXiv*, abs/2310.17567, 2023.

[66] Xiang Yue, Boshi Wang, Kai Zhang, Ziru Chen, Yu Su, and Huan Sun. Automatic evaluation of attribution by large language models. In *Conference on Empirical Methods in Natural Language Processing*, 2023.

[67] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *ArXiv*, abs/2311.07911, 2023.

[68] Jin Peng Zhou, Christian K. Belardi, Ruihan Wu, Travis Zhang, Carla P. Gomes, Wen Sun, and Kilian Q. Weinberger. On speeding up language model evaluation. *ArXiv*, abs/2407.06172, 2024.

[69] Xiao Zhou, Renjie Pi, Weizhong Zhang, Yong Lin, Zonghao Chen, and T. Zhang. Probabilistic bilevel coreset selection. *ArXiv*, abs/2301.09880, 2023.

[70] Vilém Zouhar, Peng Cui, and Mrinmaya Sachan. How to select datapoints for efficient human evaluation of nlg models?, 2025.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See Section 1.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Appendix F.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: This paper is mainly an empirical work and doesn't provide many new theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: See Appendix C and the code.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We release our codes in the supplemental materials.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: See Appendix C.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: See Appendix D.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Appendix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Authors have reviewed the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: See Appendix F.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to

generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper doesn't release any new data or model.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All used models and datasets are well cited in Section 3.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: This paper doesn't provide new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This paper doesn't involve crowd-sourcing experiments.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This paper doesn't involve crowd-sourcing experiments.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Related Work

Evaluating large language models (LLMs) has become increasingly costly as these models grow in size and capabilities [35, 16, 65, 68]. These costs manifest in several ways. First, the collection and annotation of evaluation data can require significant resources [66]. To mitigate these costs, researchers have turned to methods such as using LLMs-as-judges [21, 23] or employing active labeling [32, 31, 10, 12, 70] to generate evaluation data and labels. However, these savings come with drawbacks. For instance, LLM-as-a-judge does not produce reliable evaluation outcomes, as judge models tend to prefer models similar to them, and have other biases [63, 42, 14, 7].

Another significant cost in LLM benchmarking arises from the model inference itself. Generating responses with LLMs can be time-consuming [35, 68, 53], and common inference time scaling techniques [57, 24, 54, 33] may exacerbate this issue. The success of scaling laws [29, 50] in predicting model performance has fueled interest in the development of benchmark prediction techniques [60, 43, 44, 40, 41], which aim to estimate benchmark performance by evaluating LLMs on a limited set of data [1].

The key idea underpinning benchmark prediction is that not all evaluation examples carry the same amount of information [49]. It is hypothesized that a smaller core set of examples can represent the entire test set, allowing for accurate estimation of overall benchmark performance [60]. This is similar to efficient model training approaches, which aim to identify a subset of training data that enable performance comparable to training on the full dataset [52, 69]. Indeed, a popular benchmark prediction method, k-medoids clustering, is a classical approach to core-set selection for training [15]. However, it is important to recognize that the objectives of training and evaluation differ significantly. While training focuses on minimizing empirical risk and enhancing model performance, evaluation seeks to provide an unbiased estimation of a model's performance to facilitate fair model comparison [40]. Our work challenges the assumption that core-set selection is the key to the success of benchmark prediction by introducing competitive methods that do not rely on core-set selection.

Many existing approaches treat benchmark prediction as a learning problem, aiming to predict a model's overall performance based on its performance on a subset of data [60, 43, 34, 30, 45]. Despite promising results, previous work has highlighted limitations in terms of estimation variance [36]. Going further, we highlight that most benchmark prediction methods rely on model similarity, with estimation performance deteriorating when target models deviate from familiar source models.

---

[1] Unlike bandit literature [68, 53], which focuses on identifying the best model from a pool, benchmark prediction is more challenging as it seeks to forecast overall benchmark performance for any new model.

B Details of Benchmark Prediction Methods

B.1 Problem Formulation

We repeat the notation and the problem formulation here for the reader's convenience.

- A benchmark is represented as a triplet $(\mathcal{D}, \mathcal{F}, s)$.
- $\mathcal{D}$ represents the benchmark data with $|\mathcal{D}| = N$ data points. A data point is referred to as $z \in \mathcal{D}$, where $z = (x, y)$, x refers to the query and y refers to the ground truth answer.
- $\mathcal{F}$ refers to all potential models that can be evaluated on the benchmark.
- s represents the metric of the benchmark.
  - $s(f, z)$ refers to the performance of any $f \in \mathcal{F}$ on any data point $z \in \mathcal{D}$. For example, $s(f, z) = \mathbb{1}[f(x) = y]$ if the benchmark uses standard accuracy as the metric.
  - $\bar{s}(f, \mathcal{D}') = \frac{1}{|\mathcal{D}'|} \sum_{z \in \mathcal{D}'} s(f, z)$ represents the average performance of $f \in \mathcal{F}$ on any $\mathcal{D}' \subset \mathcal{D}$.
  - $\boldsymbol{s}(f, \mathcal{D}') = \{s(f, z)\}_{z \in \mathcal{D}'}$ represents the vectorized performance of $f \in \mathcal{F}$ on all data points in $\mathcal{D}' \subset \mathcal{D}$, and $\boldsymbol{s}(\mathcal{F}', z) = \{s(f, z)\}_{f \in \mathcal{F}'}$ represents the vectorized performances of all models in $\mathcal{F}' \subset \mathcal{F}$ on data point $z \in \mathcal{D}$.
  - $S(\mathcal{F}', \mathcal{D}') = \{\boldsymbol{s}(f, \mathcal{D}')\}_{f \in \mathcal{F}'} = \{\boldsymbol{s}(\mathcal{F}', z)\}^{\mathrm{T}}_{z \in \mathcal{D}'}$ as the performance matrix of all models in $\mathcal{F}' \subset \mathcal{F}$ on all data points in $\mathcal{D}' \subset \mathcal{D}$.
- $\mathcal{F}^{(s)} = \{f_1, \ldots, f_M\} \subset \mathcal{F}$ refers to a set of source models, whose performances on every data point of the benchmark $S(\mathcal{F}^{(s)}, \mathcal{D})$ are known.
- The rest of the models are referred to as target models $\mathcal{F}^{(t)} = \mathcal{F} \setminus \mathcal{F}^{(s)}$, which can only be evaluated on at most $n \ll N$ data points to save computational costs.

Benchmark prediction with fewer data aims to estimate $\bar{s}(f, \mathcal{D})$ for every $f \in \mathcal{F}^{(t)}$ with only n data points. In practice, benchmark prediction often involves two steps: ① identifying a representative core-set $\mathcal{C} \subset \mathcal{D}$ with $|\mathcal{C}| = n$ data points, and ② learning a performance estimator h to estimate the average performance on the full benchmark based on the core-set. Formally, the goal of benchmark prediction is to find $\mathcal{C}$ and h to minimize the estimation gap over target models,

$$\text{estimation gap:} \quad \frac{1}{|\mathcal{F}^{(t)}|} \sum_{f \in \mathcal{F}^{(t)}} \left| \bar{s}(f, \mathcal{D}) - h\big[\boldsymbol{s}(f, \mathcal{C}), S(\mathcal{F}^{(s)}, \mathcal{D})\big] \right|. \tag{3}$$

For simplicity, in the remainder of the paper, we will denote the estimated performance of target model $f \in \mathcal{F}^{(t)}$ as $h(f)$, instead of explicitly writing $h[\boldsymbol{s}(f, \mathcal{C}), S(\mathcal{F}^{(s)}, \mathcal{D})]$.

B.2 Benchmark Prediction Methods

Previous methods In this paper, we examine five widely-used benchmark prediction methods,

- RANDOM-SAMPLING randomly samples a subset as $\mathcal{C}$ and directly returns the mean performance,

  $$h^{\text{RANDOM-SAMPLING}}(f) = \bar{s}(f, \mathcal{C}). \tag{4}$$

  If the benchmark metric s is standard accuracy, the gap $|\bar{s}(f, \mathcal{C}) - \bar{s}(f, \mathcal{D})|$ is bounded by $\mathcal{O}(\sqrt{1/n})$ with high probability based on Hoeffding's inequality.
- ANCHOR-POINTS-WEIGHTED [60] treats benchmark prediction as a k-medoids clustering problem. The selected medoids are used as $\mathcal{C}$, and a weight vector $\boldsymbol{\theta} \in \mathbb{R}^n$ is calculated as the normalized cluster size of each medoid. The final estimate for any target model $f \in \mathcal{F}^{(t)}$ is

  $$h^{\text{ANCHOR-POINTS-WEIGHTED}}(f) = \boldsymbol{s}(f, \mathcal{C})^T \boldsymbol{\theta}. \tag{5}$$

- ANCHOR-POINTS-PREDICTOR [60] extends ANCHOR-POINTS-WEIGHTED. Instead of directly returning the weighted sum, a linear regression model $\boldsymbol{g}[\boldsymbol{s}(f, \mathcal{C})]$ is learned to predict $\boldsymbol{s}(f, \mathcal{D} - \mathcal{C})$.

  $$h^{\text{ANCHOR-POINTS-PREDICTOR}}(f) = \bar{g}[\boldsymbol{s}(f, \mathcal{C})] \tag{6}$$

  $$\text{where } \boldsymbol{g} = \arg\min_{\boldsymbol{g}'} \frac{1}{M} \sum_{f \in \mathcal{F}^{(s)}} \left\| \boldsymbol{s}(f, \mathcal{D} - \mathcal{C}) - \boldsymbol{g}'[\boldsymbol{s}(f, \mathcal{C})] \right\|_2^2, \tag{7}$$

  where we note that $\boldsymbol{g}[\boldsymbol{s}(f, \mathcal{C})]$ is a $(N - n)$ dimensional vector and we use $\bar{g}[\boldsymbol{s}(f, \mathcal{C})]$ as its mean.

- P-IRT [43] extends ANCHOR-POINTS-PREDICTOR by replacing the regression model $\boldsymbol{g}$ in equation 7 with an Item Response Theory (IRT) model. Following the notation for ANCHOR-POINTS-PREDICTOR, we estimate performance for any $f \in \mathcal{F}^{(t)}$ as follows:

$$h^{\text{P-IRT}}(f) = \frac{N-n}{N}\bar{g}[\boldsymbol{s}(f,\mathcal{C})] + \frac{n}{N}\bar{s}(f,\mathcal{C})\,. \tag{8}$$

- GP-IRT [43] further generalizes P-IRT by combining its estimation with ANCHOR-POINTS-WEIGHTED as a weighted sum,

$$h^{\text{GP-IRT}}(f) = \lambda h^{\text{Anchor-Points-Weighted}}(f) + (1-\lambda)h^{\text{P-IRT}}(f)\,, \tag{9}$$

  where λ is chosen heuristically to control the error of P-IRT.

New methods We introduce six methods that have not yet been applied to benchmark prediction.

- RANDOM-SAMPLING-LEARN randomly samples a subset as $\mathcal{C}$ and adopts a Ridge regression model g for estimation as follows,

$$h^{\text{Random-Sampling-Learn}}(f) = g[\boldsymbol{s}(f,\mathcal{C})] \tag{10}$$

$$\text{where } g = \arg\min_{g'} \frac{1}{M}\sum_{f\in\mathcal{F}^{(s)}} \left|\bar{s}(f,\mathcal{D}) - g'[\boldsymbol{s}(f,\mathcal{C})]\right|. \tag{11}$$

- RANDOM-SEARCH-LEARN performs RANDOM-SAMPLING-LEARN for 10,000 times and selects the best-performing subset as $\mathcal{C}$ based on cross-validation. A Ridge regression model g is then trained and used in the same way as RANDOM-SELECTION-LEARN.
- LASSO trains a Lasso regression model with weights $\boldsymbol{\theta} \in \mathbb{R}^N$ as follows,

$$h^{\text{Lasso}}(f) = \boldsymbol{s}(f,\mathcal{C})^T\boldsymbol{\theta}_{\mathcal{C}} \tag{12}$$

$$\text{where } \boldsymbol{\theta} = \arg\min_{\boldsymbol{\theta}'}\frac{1}{n}\sum_{z\in\mathcal{C}}\left[\boldsymbol{s}(f,\mathcal{D})^{\mathrm{T}}\boldsymbol{\theta}' - \bar{s}(f,\mathcal{D})\right]^2 + \lambda\|\boldsymbol{\theta}'\|_1\,, \tag{13}$$

  where λ is selected so that only n dimensions of $\boldsymbol{\theta}$ are non-zero and $\boldsymbol{\theta}_{\mathcal{C}}$ is the non-zero slice of $\boldsymbol{\theta}$.
- DOUBLE-OPTIMIZE optimizes both a subset selection vector $\boldsymbol{\pi} \in \mathbb{R}^N$ and a linear regression model with weights $\boldsymbol{\theta} \in \mathbb{R}^N$ with gradient descent as follows,

$$h^{\text{Double-Optimize}}(f) = [\boldsymbol{s}(f,\mathcal{D}) \cdot \texttt{TopMask}(\boldsymbol{\pi}; n)]^{\mathrm{T}}\boldsymbol{\theta} \tag{14}$$

$$\text{where} \quad \boldsymbol{\pi},\boldsymbol{\theta} = \arg\min_{\boldsymbol{\pi}',\boldsymbol{\theta}'}\left\{[\boldsymbol{s}(f,\mathcal{D}) \cdot \texttt{TopMask}(\boldsymbol{\pi}'; n)]^{\mathrm{T}}\boldsymbol{\theta}' - \bar{s}(f,\mathcal{D})\right\}^2, \tag{15}$$

  where $\cdot$ refers to the bitwise multiplication between two vectors, and $\texttt{TopMask}(\boldsymbol{\pi}'; n)$ replaces the top n largest values of $\boldsymbol{\pi}'$ with 1s and the rest with 0s. We directly pass the gradient on $\texttt{TopMask}(\boldsymbol{\pi}'; n)$ to $\boldsymbol{\pi}'$ during optimization following the Straight-Through technique [27, 3].
- Principal Component Analysis (PCA) treats benchmark prediction as a matrix completion problem. This method assumes the performance matrix $S(\mathcal{F},\mathcal{D})$ is of low rank. By randomly sampling a subset as $\mathcal{C}$, this methods conducts PCA to impute the missing values for target models [59, 6]. As a more intuitive view, one could also take the acquired principal components as model capability indicators [50], *i.e.*, the $(M \times k)$ PCA-transformed scores indicate the k-capabilities of each model, while the $(k \times N)$ principal components represent the capability requirements for each data point. We select k among $\{2, 5, 10, 20\}$ through cross-validation. The Pseudo codes are in Algorithm 1.
- Augmented inverse propensity weighting (AIPW) [48]: Inspired by the application of prediction powered inference [2, 1] to the LLM-as-a-judge setting [5, 14], we apply a more general AIPW estimator to benchmark prediction. We train a Ridge regression model g for every target model f, which predicts the point-wise performance $s(f, z)$ based on $\boldsymbol{s}(\mathcal{F}^{(s)}, z)$. Formally,

$$g = \arg\min_{g'}\frac{1}{n}\sum_{z\in\mathcal{C}}\left[g'[\boldsymbol{s}(\mathcal{F}^{(s)}, z)] - s(f,z)\right]^2. \tag{16}$$

The idea behind the AIPW estimator is to use the predicted performance $\hat{s}(f, z) = g[\boldsymbol{s}(\mathcal{F}^{(s)}, z)]$ as a proxy score to estimate $\bar{s}(f, \mathcal{D})$ and ”debias” that estimator as follows

$$h^{\text{AIPW}}(f) = \bar{s}(f, \mathcal{C}) + \frac{1}{1 + \frac{n}{N-n}} \left(\frac{1}{N-n} \sum_{z \in \mathcal{D} - \mathcal{C}} \hat{s}(f, z) \; - \frac{1}{n} \sum_{z \in \mathcal{C}} \hat{s}(f, z) \right). \tag{17}$$

Unlike the other learning-based baselines, AIPW is a consistent estimator for $\bar{s}(f, \mathcal{D})$[19]. Compared to RANDOM-SAMPLING, it reduces estimator variance by a factor of up to $\frac{1}{1+\frac{n}{N}} \rho(\hat{s}^(f, z), s(f, z))^2$ [14], where ρ is the Pearson correlation coefficient. Recent research [37] shows that AIPW estimator will outperform random sampling if and only if the correlation between $\hat{s}(f, z)$ and $s(f, z)$ is above a certain level that depends on n.

Algorithm 1 PCA Impute Process

```
Input: Data matrix with missing values
Parameters: number of components $k$, max iteration max_iter, stopping threshold tol
Output: Imputed data matrix
Step 1: Initialization
    Compute initial values for missing entries using column means
Step 2: Iterative Imputation
for iteration $\leftarrow 1$ to max_iter do
    PCA Decomposition:
        Perform PCA retaining $k$ components
        Transform data to the lower-dimensional space
        Reconstruct the data from the lower-dimensional space
    Evaluate Convergence:
        Compute the norm of differences between imputed and original values at missing entries
    if norm $<$ tol then
        Break the loop
    end if
    Update Imputed Values:
        Replace missing values with reconstructed values
end for

return Fully imputed data matrix
```

C Additional Experiment Setup

We select a diverse range of benchmarks from the following sources[2].

- HELM-Lite benchmarks [35]:
  - `OpenbookQA` [39]: $N = 500$ data points.
  - `GSM8K` [9]: $N = 1000$ data points.
  - `LegalBench` [22]: $N = 2047$ data points.
  - `Math` [26]: $N = 437$ data points.
  - `MedQA` [28]: $N = 1000$ data points.
  - `MMLU` [25]: $N = 567$ data points.

  We obtain the per-data point performances of $|\mathcal{F}| = 83$ models from the official leaderboard. Note that Helm-Lite often only uses a subset of the original testing set for each benchmark to save compute.
- GLUE benchmarks [61]:
  - `MRPC` [13]: $N = 408$ data points.
  - `RTE` [11, 18, 4]: $N = 277$ data points.
  - `SST-2` [55]: $N = 872$ data points.
  - `MNLI` [64]: $N = 9815$ data points.
  - `QNLI` [46]: $N = 5463$ data points.

  We use the per-data performances of $|\mathcal{F}| = 87$ models provided by AnchorPoint[3] [60].
- OpenLLM benchmarks [16]:
  - `IFEval` [67]: $N = 541$ data points.
  - `Math` [26]: $N = 894$ data points. Only level 5 MATH questions are used in OpenLLM.
  - `MMLU-Pro` [62]: $N = 12032$ data points.
  - `Arc-Challenge` [8]: $N = 1172$ data points.
  - `BBH` [58]: $N = 5761$ data points.
  - `GPQA` [47]: $N = 1192$ data points.
  - `MUSR` [56]: $N = 756$ data points.

  We use $|\mathcal{F}| = 448$ models provided by Huggingface [4] and collect their performance scores.
- `ImageNet` [51]: We collect $|\mathcal{F}| = 110$ models from Pytorch Hub [5] and evaluate them on `ImageNet` with $N = 50,000$ data points.

For simplicity, we report the overall average accuracy directly for `MMLU`, `MMLU-Pro`, and `BBH`, rather than the weighted average accuracy computed across sub-tasks. Alternatively, one could apply benchmark predictions separately to each sub-task and then calculate the weighted average accuracy.

---

[2] Since P-IRT and GP-IRT requires $s(f, z)$ to be binary, we only use benchmarks with accuracy as metric.

[3] The provided score file for `QQP` is broken so we exclude it.

[4] `https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard#`

[5] `https://pytorch.org/vision/stable/models.html#classification`

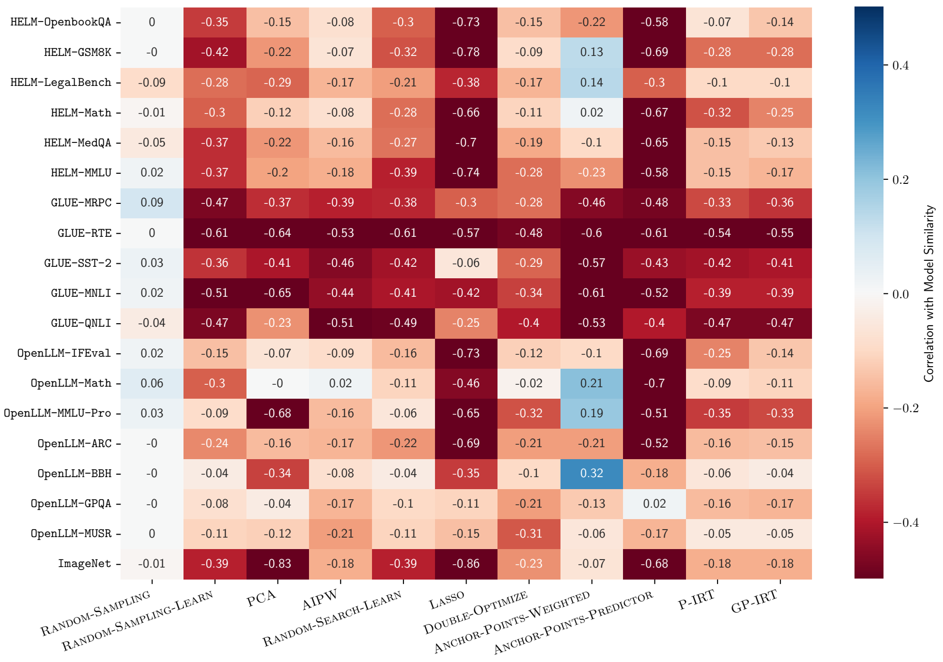


Figure 4: The Pearson correlation between normalized per-model estimation gap (equation 20) and model similarity (equation 18). Negative correlation indicates that target models that are dissimilar to source models tend to have larger estimation gap, and vice versa.

D Additinoal Experiment Results

D.1 Reliance on Model Similarity

In this subsection, we investigate the extent to which benchmark prediction methods rely on the similarity between target and source models.

Model similarity. We follow previous works [38, 20] and define the model similarity of target model f to all source models $\mathcal{F}^{(s)}$ as follows,

$$\mathcal{S}(f, \mathcal{F}^{(s)}, \mathcal{D}) = \frac{1}{M} \sum\nolimits_{f' \in \mathcal{F}^{(s)}} \frac{c_{obs} - c_{exp}}{1 - c_{exp}} . \tag{18}$$

Here, $c_{exp} = \bar{s}(f, \mathcal{D})\bar{s}(f', \mathcal{D}) + (1 - \bar{s}(f, \mathcal{D}))(1 - \bar{s}(f', \mathcal{D}))$ measures the chance agreement rate, *i.e.*, the expected probability of $\{s(f, z) = s(f', z)\}$ if $s(f, z)$ is independent of $s(f', z)$. In contrast, $c_{obs} = \frac{1}{N} \sum_{z \in \mathcal{D}} \mathbb{1}[s(f, z) = s(f', z)]$ is the observed agreement rate. For simplicity, we use $\mathcal{S}(f)$ to denote $\mathcal{S}(f, \mathcal{F}^{(s)}, \mathcal{D})$ in the remainder of the paper. $\mathcal{S}(f)$ quantifies how similar the performance pattern of the target model f is to all source models $\mathcal{F}^{(s)}$, with a higher value indicating greater similarity [20].

We aim to examine the correlation between model similarity and estimation gap. However, we note that the estimation depends on the standard deviation of $s(f, z)$. Since we use accuracy as the metric in our experiment, $s(f, z)$ is Bernoulli with parameter $p_f = \bar{s}(f, \mathcal{D})$ and standard deviation $\sigma_f = \sqrt{p_f(1 - p_f)}$. By randomly sampling n data points as $\mathcal{C}$, Chebyshev's inequality ensures that

$$|\bar{s}(f, \mathcal{C}) - \bar{s}(f, \mathcal{D})| < \sigma_f / \sqrt{\alpha n} \tag{19}$$

with probability at least $(1 - \alpha)$. In other words, the performance of target models with lower σ_f is easier to estimate with the same amount of data. Thus, the standard deviation of the basic estimation gap could potentially confound the observed correlation between model similarity and estimation gap. Consider the method RANDOM-SAMPLING, whose estimation does not depend on source models. If all target models with low σ_f coincidentally have high $\mathcal{S}(f)$, while those with high σ_f have low

$\mathcal{S}(f)$, then a spurious correlation between estimation gap and model similarity to target models could appear even for RANDOM-SAMPLING. To prevent this, we define the normalized estimation gap as

$$\text{normalized estimation gap for } f\text{:} \quad \mathcal{E}(f) = \frac{1}{\sigma_f} |\bar{s}(f, \mathcal{D}) - h(f)| \, . \tag{20}$$

Then we measure the Pearson correlation between model similarity in equation 18 and the normalized estimation gap in equation 20.

Results. The results are shown in Figure 4. A clear negative correlation between model similarity and estimation gap emerges for almost all benchmark prediction methods except for RANDOM-SAMPLING. In particular, the best-performing method under the interpolation model split, RANDOM-SAMPLING-LEARN, exhibits a negative correlation below -0.2 in 13/19 benchmarks. Despite its asymptotic unbiasedness, we also find negative correlations for AIPW. This is perhaps unsurprising: While AIPW is consistent independent of how well its regression model $g[\boldsymbol{s}(\mathcal{F}^{(s)}, z)]$ predicts $s(f, z)$, its variance depends precisely on that prediction quality. If the predictions are good, AIPW improves substantially over RANDOM-SAMPLING, while there is no improvement when predictions are fully uninformative. But intuitively, predicting $s(f, z)$ is harder when f is very different from the models $\mathcal{F}^{(s)}$ used for training the predictor $g[\boldsymbol{s}(\mathcal{F}^{(s)}, z)]$.

Table 1: Ablation study on the core-set size n. We report the estimation gap averaged over all benchmarks. % is neglected for each metric. The lowest estimation gap in each column is highlighted in bold.

	Interpolation					Extrapolation				
	$n = 10$	$n = 20$	$n = 50$	$n = 100$	$n = 200$	$n = 10$	$n = 20$	$n = 50$	$n = 100$	$n = 200$
RANDOM-SAMPLING	11.0	7.7	4.8	3.3	2.1	10.7	7.4	4.6	3.1	2.0
RANDOM-SAMPLING-LEARN	5.4	4.2	2.9	2.1	1.5	17.6	15.8	13.6	12.1	11.1
PCA	6.6	5.2	3.7	2.8	2.1	19.9	17.6	14.9	12.2	9.3
AIPW	8.3	5.4	3.3	2.3	1.8	**9.6**	**6.5**	**4.0**	**2.8**	**2.0**
RANDOM-SEARCH-LEARN	**4.5**	**3.7**	**2.7**	**2.0**	1.4	16.0	14.4	12.8	11.8	11.1
LASSO	7.8	6.1	3.6	2.6	2.2	22.0	19.3	16.6	15.3	14.6
DOUBLE-OPTIMIZE	6.6	4.8	3.0	2.3	1.9	11.3	9.0	8.2	8.0	7.0
ANCHOR-POINTS-WEIGHTED	8.9	6.9	4.9	4.0	3.2	10.4	6.7	5.6	4.7	3.4
ANCHOR-POINTS-PREDICTOR	4.7	4.1	3.6	3.4	4.1	16.2	14.8	13.4	12.4	11.2
P-IRT	7.3	5.9	3.5	2.1	**1.3**	9.8	8.2	5.1	3.7	3.0
GP-IRT	7.2	5.7	3.3	2.1	1.4	9.7	7.8	4.7	3.4	2.5

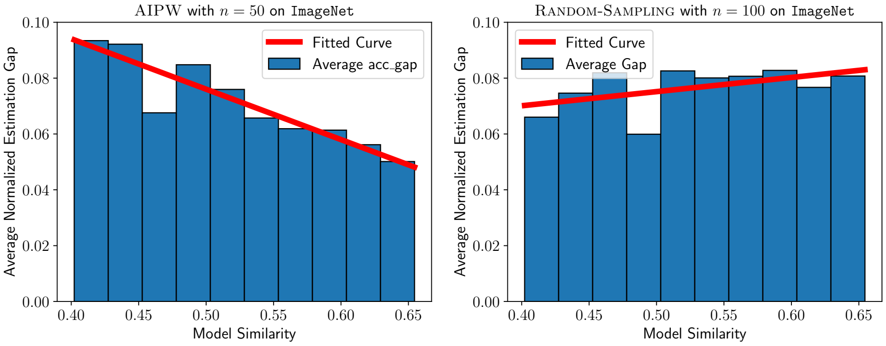
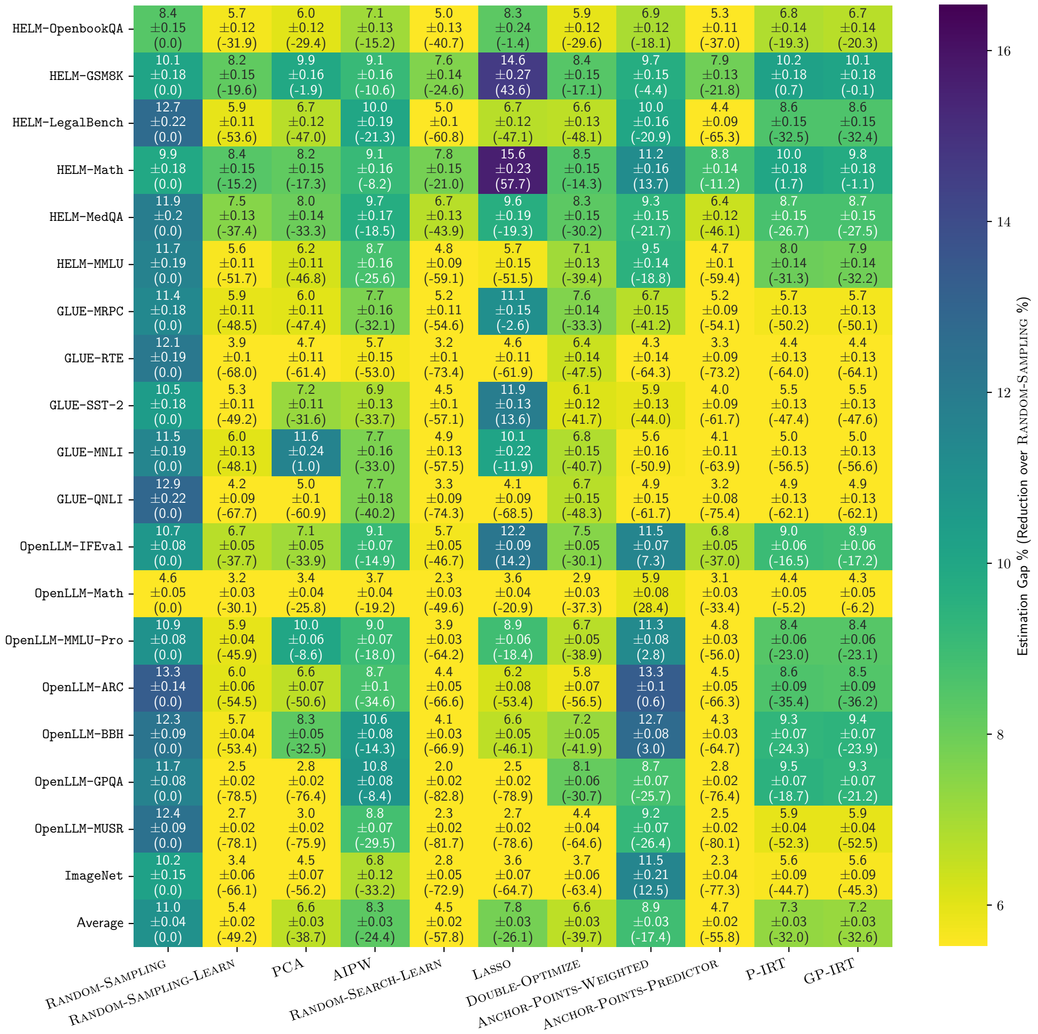


Figure 5: Average normalized estimation gap relative to model similarity for AIPW (n=50) and RANDOM-SAMPLING (n=100) on `ImageNet`. Each bar represents the target models whose similarity to source models falls within the corresponding range. The normalized estimation gap is defined as shown in equation 20. On average, AIPW outperforms RANDOM-SAMPLING, even with half the data. However, RANDOM-SAMPLING shows better performance when model similarity is low.

D.2 Ablation on Core-set Size

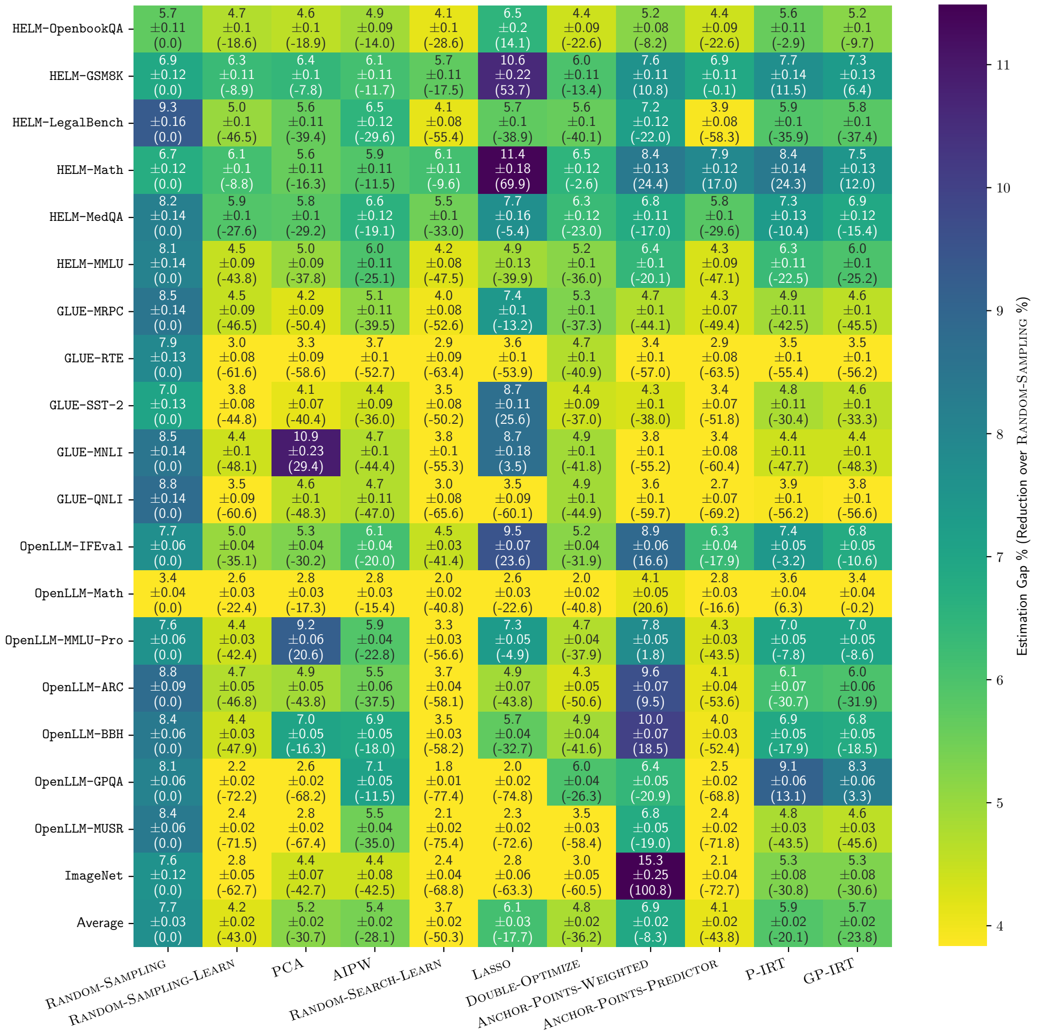
We conduct an ablation study on the size of the core-set n. We experiment with $n \in \{10, 20, 50, 100, 200\}$, and the summarized results are shown in Table 1 (detailed results can be found in Figures 6, 7, 8, 9, 10, 11, 12, 13, 14, and 15. As expected, the estimation gap generally decreases as n increases for most methods. Our previous conclusions remain valid across both settings. With larger core-set sizes, most methods continue to perform better than RANDOM-SAMPLING in the interpolation split but fail to do so in the extrapolation model split. Interestingly, we also find that RANDOM-SAMPLING outperforms all other methods when given twice as much data, even in the interpolation model split.

AIPW remains effective in both settings. However, its advantage over RANDOM-SAMPLING diminishes as n increases. While AIPW reduces the estimation gap by -30.4% in interpolation and -12.6% in extrapolation for $n = 50$, these advantages shrink to -12.4% in interpolation and a mere -2.3% in extrapolation for $n = 200$. This is because the estimator variance reduction factor of AIPW is up to $\frac{1}{1+\frac{n}{N}}\rho(\hat{s}^{(}f, z), s(f, z))^2$. On the other hand, the advantage of AIPW remains significant when the dataset is large and thus $\frac{n}{N}$ is small. Figure 5 compares AIPW with $n = 50$ to RANDOM-SAMPLING with $n = 100$ data points using `ImageNet`. AIPW achieves a lower average normalized estimation gap compared to RANDOM-SAMPLING, despite using only half the data. However, the normalized estimation gap for AIPW is biased with respect to model similarity. In contrast, the normalized estimation gap under RANDOM-SAMPLING remains largely neutral regarding model similarity. Consequently, while AIPW reduces the average, it produces a higher gap for models with low similarity compared to RANDOM-SAMPLING with twice the data.



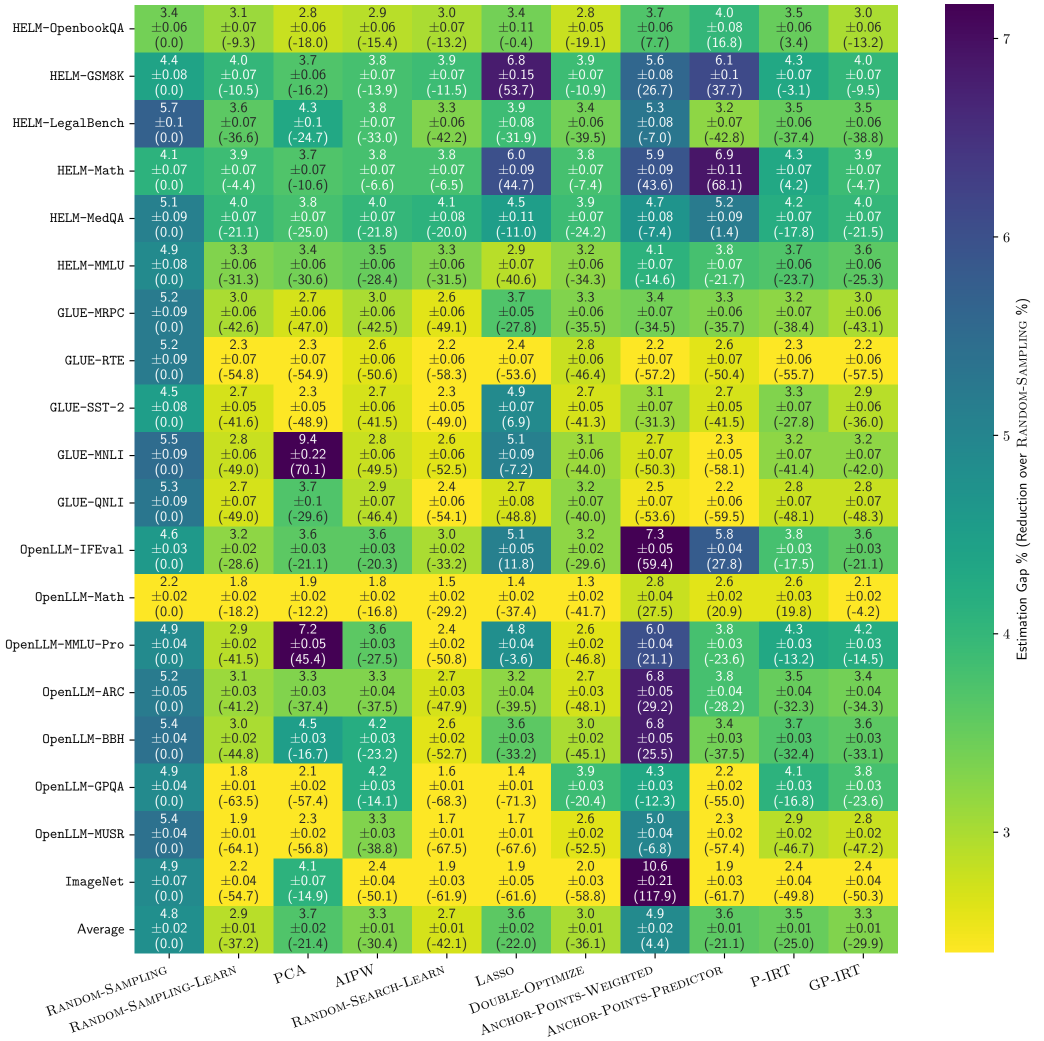
	Random-Sampling	Random-Sampling-Learn	PCA	AIPW	Random-Search-Learn	Lasso	Double-Optimize	Anchor-Points-Weighted	Anchor-Points-Predictor	P-IRT	GP-IRT
HELM-OpenbookQA	8.4 ±0.15 (0.0)	5.7 ±0.12 (-31.9)	6.0 ±0.12 (-29.4)	7.1 ±0.13 (-15.2)	5.0 ±0.13 (-40.7)	8.3 ±0.24 (-1.4)	5.9 ±0.12 (-29.6)	6.9 ±0.12 (-18.1)	5.3 ±0.11 (-37.0)	6.8 ±0.14 (-19.3)	6.7 ±0.14 (-20.3)
HELM-GSM8K	10.1 ±0.18 (0.0)	8.2 ±0.15 (-19.6)	9.9 ±0.16 (-1.9)	9.1 ±0.16 (-10.6)	7.6 ±0.14 (-24.6)	14.6 ±0.27 (43.6)	8.4 ±0.15 (-17.1)	9.7 ±0.15 (-4.4)	7.9 ±0.13 (-21.8)	10.2 ±0.18 (0.7)	10.1 ±0.18 (-0.1)
HELM-LegalBench	12.7 ±0.22 (0.0)	5.9 ±0.11 (-53.6)	6.7 ±0.12 (-47.0)	10.0 ±0.19 (-21.3)	5.0 ±0.1 (-60.8)	6.7 ±0.12 (-47.1)	6.6 ±0.13 (-48.1)	10.0 ±0.16 (-20.9)	4.4 ±0.09 (-65.3)	8.6 ±0.15 (-32.5)	8.6 ±0.15 (-32.4)
HELM-Math	9.9 ±0.18 (0.0)	8.4 ±0.15 (-15.2)	8.2 ±0.15 (-17.3)	9.1 ±0.16 (-8.2)	7.8 ±0.15 (-21.0)	15.6 ±0.23 (57.7)	8.5 ±0.15 (-14.3)	11.2 ±0.16 (13.7)	8.8 ±0.14 (-11.2)	10.0 ±0.18 (1.7)	9.8 ±0.18 (-1.1)
HELM-MedQA	11.9 ±0.2 (0.0)	7.5 ±0.13 (-37.4)	8.0 ±0.14 (-33.3)	9.7 ±0.17 (-18.5)	6.7 ±0.13 (-43.9)	9.6 ±0.19 (-19.3)	8.3 ±0.15 (-30.2)	9.3 ±0.15 (-21.7)	6.4 ±0.12 (-46.1)	8.7 ±0.15 (-26.7)	8.7 ±0.15 (-27.5)
HELM-MMLU	11.7 ±0.19 (0.0)	5.6 ±0.11 (-51.7)	6.2 ±0.11 (-46.8)	8.7 ±0.16 (-25.6)	4.8 ±0.09 (-59.1)	5.7 ±0.15 (-51.5)	7.1 ±0.13 (-39.4)	9.5 ±0.14 (-18.8)	4.7 ±0.1 (-59.4)	8.0 ±0.14 (-31.3)	7.9 ±0.14 (-32.2)
GLUE-MRPC	11.4 ±0.18 (0.0)	5.9 ±0.11 (-48.5)	6.0 ±0.11 (-47.4)	7.7 ±0.16 (-32.1)	5.2 ±0.11 (-54.6)	11.1 ±0.15 (-2.6)	7.6 ±0.14 (-33.3)	6.7 ±0.15 (-41.2)	5.2 ±0.09 (-54.1)	5.7 ±0.13 (-50.2)	5.7 ±0.13 (-50.1)
GLUE-RTE	12.1 ±0.19 (0.0)	3.9 ±0.1 (-68.0)	4.7 ±0.11 (-61.4)	5.7 ±0.15 (-53.0)	3.2 ±0.1 (-73.4)	4.6 ±0.11 (-61.9)	6.4 ±0.14 (-47.5)	4.3 ±0.14 (-64.3)	3.3 ±0.09 (-73.2)	4.4 ±0.13 (-64.0)	4.4 ±0.13 (-64.1)
GLUE-SST-2	10.5 ±0.18 (0.0)	5.3 ±0.11 (-49.2)	7.2 ±0.11 (-31.6)	6.9 ±0.13 (-33.7)	4.5 ±0.1 (-57.1)	11.9 ±0.13 (13.6)	6.1 ±0.12 (-41.7)	5.9 ±0.13 (-44.0)	4.0 ±0.09 (-61.7)	5.5 ±0.13 (-47.4)	5.5 ±0.13 (-47.6)
GLUE-MNLI	11.5 ±0.19 (0.0)	6.0 ±0.13 (-48.1)	11.6 ±0.24 (1.0)	7.7 ±0.16 (-33.0)	4.9 ±0.13 (-57.5)	10.1 ±0.22 (-11.9)	6.8 ±0.15 (-40.7)	5.6 ±0.16 (-50.9)	4.1 ±0.11 (-63.9)	5.0 ±0.13 (-56.5)	5.0 ±0.13 (-56.6)
GLUE-QNLI	12.9 ±0.22 (0.0)	4.2 ±0.09 (-67.7)	5.0 ±0.1 (-60.9)	7.7 ±0.18 (-40.2)	3.3 ±0.09 (-74.3)	4.1 ±0.09 (-68.5)	6.7 ±0.15 (-48.3)	4.9 ±0.15 (-61.7)	3.2 ±0.08 (-75.4)	4.9 ±0.13 (-62.1)	4.9 ±0.13 (-62.1)
OpenLLM-IFEval	10.7 ±0.08 (0.0)	6.7 ±0.05 (-37.7)	7.1 ±0.05 (-33.9)	9.1 ±0.07 (-14.9)	5.7 ±0.05 (-46.7)	12.2 ±0.09 (14.2)	7.5 ±0.05 (-30.1)	11.5 ±0.07 (7.3)	6.8 ±0.05 (-37.0)	9.0 ±0.06 (-16.5)	8.9 ±0.06 (-17.2)
OpenLLM-Math	4.6 ±0.05 (0.0)	3.2 ±0.03 (-30.1)	3.4 ±0.04 (-25.8)	3.7 ±0.04 (-19.2)	2.3 ±0.03 (-49.6)	3.6 ±0.04 (-20.9)	2.9 ±0.03 (-37.3)	5.9 ±0.08 (28.4)	3.1 ±0.03 (-33.4)	4.4 ±0.05 (-5.2)	4.3 ±0.05 (-6.2)
OpenLLM-MMLU-Pro	10.9 ±0.08 (0.0)	5.9 ±0.04 (-45.9)	10.0 ±0.06 (-8.6)	9.0 ±0.07 (-18.0)	3.9 ±0.03 (-64.2)	8.9 ±0.06 (-18.4)	6.7 ±0.05 (-38.9)	11.3 ±0.08 (2.8)	4.8 ±0.03 (-56.0)	8.4 ±0.06 (-23.0)	8.4 ±0.06 (-23.1)
OpenLLM-ARC	13.3 ±0.14 (0.0)	6.0 ±0.06 (-54.5)	6.6 ±0.07 (-50.6)	8.7 ±0.1 (-34.6)	4.4 ±0.05 (-66.6)	6.2 ±0.08 (-53.4)	5.8 ±0.07 (-56.5)	13.3 ±0.1 (0.6)	4.5 ±0.05 (-66.3)	8.6 ±0.09 (-35.4)	8.5 ±0.09 (-36.2)
OpenLLM-BBH	12.3 ±0.09 (0.0)	5.7 ±0.04 (-53.4)	8.3 ±0.05 (-32.5)	10.6 ±0.08 (-14.3)	4.1 ±0.03 (-66.9)	6.6 ±0.05 (-46.1)	7.2 ±0.05 (-41.9)	12.7 ±0.08 (3.0)	4.3 ±0.03 (-64.7)	9.3 ±0.07 (-24.3)	9.4 ±0.07 (-23.9)
OpenLLM-GPQA	11.7 ±0.08 (0.0)	2.5 ±0.02 (-78.5)	2.8 ±0.02 (-76.4)	10.8 ±0.08 (-8.4)	2.0 ±0.02 (-82.8)	2.5 ±0.02 (-78.9)	8.1 ±0.06 (-30.7)	8.7 ±0.07 (-25.7)	2.8 ±0.02 (-76.4)	9.5 ±0.07 (-18.7)	9.3 ±0.07 (-21.2)
OpenLLM-MUSR	12.4 ±0.09 (0.0)	2.7 ±0.02 (-78.1)	3.0 ±0.02 (-75.9)	8.8 ±0.07 (-29.5)	2.3 ±0.02 (-81.7)	2.7 ±0.02 (-78.6)	4.4 ±0.04 (-64.6)	9.2 ±0.07 (-26.4)	2.5 ±0.02 (-80.1)	5.9 ±0.04 (-52.3)	5.9 ±0.04 (-52.5)
ImageNet	10.2 ±0.15 (0.0)	3.4 ±0.06 (-66.1)	4.5 ±0.07 (-56.2)	6.8 ±0.12 (-33.2)	2.8 ±0.05 (-72.9)	3.6 ±0.07 (-64.7)	3.7 ±0.06 (-63.4)	11.5 ±0.21 (12.5)	2.3 ±0.04 (-77.3)	5.6 ±0.09 (-44.7)	5.6 ±0.09 (-45.3)
Average	11.0 ±0.04 (0.0)	5.4 ±0.02 (-49.2)	6.6 ±0.03 (-38.7)	8.3 ±0.03 (-24.4)	4.5 ±0.02 (-57.8)	7.8 ±0.03 (-26.1)	6.6 ±0.03 (-39.7)	8.9 ±0.03 (-17.4)	4.7 ±0.02 (-55.8)	7.3 ±0.03 (-32.0)	7.2 ±0.03 (-32.6)

Figure 6: The estimation gaps (↓) for target models (calculated as equation 1) under interpolation model split, where source models are identically distributed with target models. Each target model can only be evaluated on $n = 10$ data points. We also report ± the standard error of the mean and the estimation gap reduction (↓) over RANDOM-SAMPLING in parentheses. A negative reduction implies that the method achieves a lower estimation gap than RANDOM-SAMPLING. % is omitted. Best viewed in color.



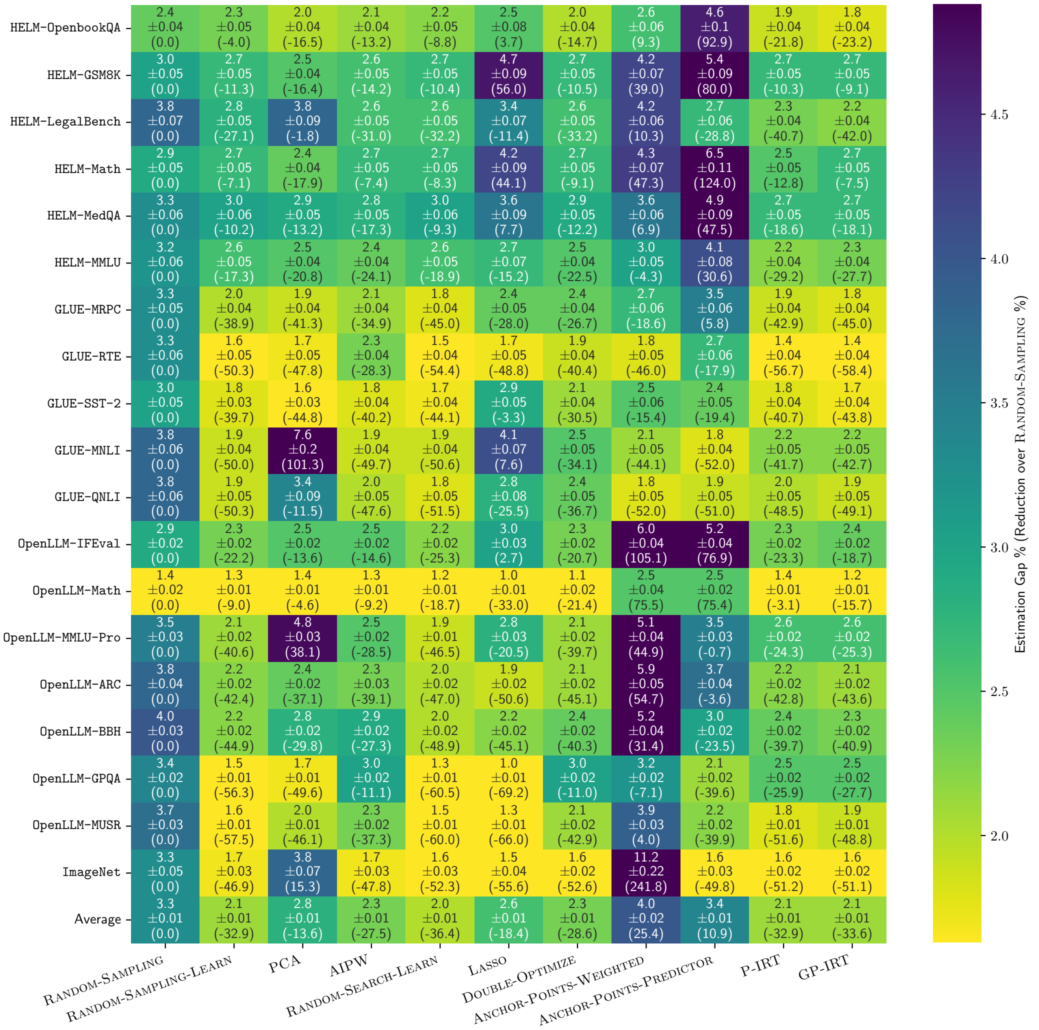
	Random-Sampling	Random-Sampling-Learn	PCA	AIPW	Random-Search-Learn	Lasso	Double-Optimize	Anchor-Points-Weighted	Anchor-Points-Predictor	P-IRT	GP-IRT
HELM-OpenbookQA	5.7 ±0.11 (0.0)	4.7 ±0.1 (-18.6)	4.6 ±0.1 (-18.9)	4.9 ±0.09 (-14.0)	4.1 ±0.1 (-28.6)	6.5 ±0.2 (14.1)	4.4 ±0.09 (-22.6)	5.2 ±0.08 (-8.2)	4.4 ±0.09 (-22.6)	5.6 ±0.11 (-2.9)	5.2 ±0.1 (-9.7)
HELM-GSM8K	6.9 ±0.12 (0.0)	6.3 ±0.11 (-8.9)	6.4 ±0.1 (-7.8)	6.1 ±0.11 (-11.7)	5.7 ±0.11 (-17.5)	10.6 ±0.22 (53.7)	6.0 ±0.11 (-13.4)	7.6 ±0.11 (10.8)	6.9 ±0.11 (-0.1)	7.7 ±0.14 (11.5)	7.3 ±0.13 (6.4)
HELM-LegalBench	9.3 ±0.16 (0.0)	5.0 ±0.1 (-46.5)	5.6 ±0.11 (-39.4)	6.5 ±0.12 (-29.6)	4.1 ±0.08 (-55.4)	5.7 ±0.1 (-38.9)	5.6 ±0.1 (-40.1)	7.2 ±0.12 (-22.0)	3.9 ±0.08 (-58.3)	5.9 ±0.1 (-35.9)	5.8 ±0.1 (-37.4)
HELM-Math	6.7 ±0.12 (0.0)	6.1 ±0.1 (-8.8)	5.6 ±0.11 (-16.3)	5.9 ±0.11 (-11.5)	6.1 ±0.11 (-9.6)	11.4 ±0.18 (69.9)	6.5 ±0.12 (-2.6)	8.4 ±0.13 (24.4)	7.9 ±0.12 (17.0)	8.4 ±0.14 (24.3)	7.5 ±0.13 (12.0)
HELM-MedQA	8.2 ±0.14 (0.0)	5.9 ±0.1 (-27.6)	5.8 ±0.1 (-29.2)	6.6 ±0.12 (-19.1)	5.5 ±0.1 (-33.0)	7.7 ±0.16 (-5.4)	6.3 ±0.12 (-23.0)	6.8 ±0.11 (-17.0)	5.8 ±0.1 (-29.6)	7.3 ±0.13 (-10.4)	6.9 ±0.12 (-15.4)
HELM-MMLU	8.1 ±0.14 (0.0)	4.5 ±0.09 (-43.8)	5.0 ±0.09 (-37.8)	6.0 ±0.11 (-25.1)	4.2 ±0.08 (-47.5)	4.9 ±0.13 (-39.9)	5.2 ±0.1 (-36.0)	6.4 ±0.1 (-20.1)	4.3 ±0.09 (-47.1)	6.3 ±0.11 (-22.5)	6.0 ±0.1 (-25.2)
GLUE-MRPC	8.5 ±0.14 (0.0)	4.5 ±0.09 (-46.5)	4.2 ±0.09 (-50.4)	5.1 ±0.11 (-39.5)	4.0 ±0.08 (-52.6)	7.4 ±0.1 (-13.2)	5.3 ±0.1 (-37.3)	4.7 ±0.1 (-44.1)	4.3 ±0.07 (-49.4)	4.9 ±0.11 (-42.5)	4.6 ±0.1 (-45.5)
GLUE-RTE	7.9 ±0.13 (0.0)	3.0 ±0.08 (-61.6)	3.3 ±0.09 (-58.6)	3.7 ±0.1 (-52.7)	2.9 ±0.09 (-63.4)	3.6 ±0.1 (-53.9)	4.7 ±0.1 (-40.9)	3.4 ±0.1 (-57.0)	2.9 ±0.08 (-63.5)	3.5 ±0.1 (-55.4)	3.5 ±0.1 (-56.2)
GLUE-SST-2	7.0 ±0.13 (0.0)	3.8 ±0.08 (-44.8)	4.1 ±0.07 (-40.4)	4.4 ±0.09 (-36.0)	3.5 ±0.08 (-50.2)	8.7 ±0.11 (25.6)	4.4 ±0.09 (-37.0)	4.3 ±0.1 (-38.0)	3.4 ±0.07 (-51.8)	4.8 ±0.11 (-30.4)	4.6 ±0.1 (-33.3)
GLUE-MNLI	8.5 ±0.14 (0.0)	4.4 ±0.1 (-48.1)	10.9 ±0.23 (29.4)	4.7 ±0.1 (-44.4)	3.8 ±0.1 (-55.3)	8.7 ±0.18 (3.5)	4.9 ±0.1 (-41.8)	3.8 ±0.1 (-55.2)	3.4 ±0.08 (-60.4)	4.4 ±0.11 (-47.7)	4.4 ±0.1 (-48.3)
GLUE-QNLI	8.8 ±0.14 (0.0)	3.5 ±0.09 (-60.6)	4.6 ±0.1 (-48.3)	4.7 ±0.11 (-47.0)	3.0 ±0.08 (-65.6)	3.5 ±0.09 (-60.1)	4.9 ±0.1 (-44.9)	3.6 ±0.1 (-59.7)	2.7 ±0.07 (-69.2)	3.9 ±0.1 (-56.2)	3.8 ±0.1 (-56.6)
OpenLLM-IFEval	7.7 ±0.06 (0.0)	5.0 ±0.04 (-35.1)	5.3 ±0.04 (-30.2)	6.1 ±0.04 (-20.0)	4.5 ±0.03 (-41.4)	9.5 ±0.07 (23.6)	5.2 ±0.04 (-31.9)	8.9 ±0.06 (16.6)	6.3 ±0.04 (-17.9)	7.4 ±0.05 (-3.2)	6.8 ±0.05 (-10.6)
OpenLLM-Math	3.4 ±0.04 (0.0)	2.6 ±0.03 (-22.4)	2.8 ±0.03 (-17.3)	2.8 ±0.03 (-15.4)	2.0 ±0.02 (-40.8)	2.6 ±0.03 (-22.6)	2.0 ±0.02 (-40.8)	4.1 ±0.05 (20.6)	2.8 ±0.03 (-16.6)	3.6 ±0.04 (6.3)	3.4 ±0.04 (-0.2)
OpenLLM-MMLU-Pro	7.6 ±0.06 (0.0)	4.4 ±0.03 (-42.4)	9.2 ±0.06 (20.6)	5.9 ±0.04 (-22.8)	3.3 ±0.03 (-56.6)	7.3 ±0.05 (-4.9)	4.7 ±0.04 (-37.9)	7.8 ±0.05 (1.8)	4.3 ±0.03 (-43.5)	7.0 ±0.05 (-7.8)	7.0 ±0.05 (-8.6)
OpenLLM-ARC	8.8 ±0.09 (0.0)	4.7 ±0.05 (-46.8)	4.9 ±0.05 (-43.8)	5.5 ±0.06 (-37.5)	3.7 ±0.04 (-58.1)	4.9 ±0.07 (-43.8)	4.3 ±0.05 (-50.6)	9.6 ±0.07 (9.5)	4.1 ±0.04 (-53.6)	6.1 ±0.07 (-30.7)	6.0 ±0.06 (-31.9)
OpenLLM-BBH	8.4 ±0.06 (0.0)	4.4 ±0.03 (-47.9)	7.0 ±0.05 (-16.3)	6.9 ±0.05 (-18.0)	3.5 ±0.03 (-58.2)	5.7 ±0.04 (-32.7)	4.9 ±0.04 (-41.6)	10.0 ±0.07 (18.5)	4.0 ±0.03 (-52.4)	6.9 ±0.05 (-17.9)	6.8 ±0.05 (-18.5)
OpenLLM-GPQA	8.1 ±0.06 (0.0)	2.2 ±0.02 (-72.2)	2.6 ±0.02 (-68.2)	7.1 ±0.05 (-11.5)	1.8 ±0.01 (-77.4)	2.0 ±0.02 (-74.8)	6.0 ±0.04 (-26.3)	6.4 ±0.05 (-20.9)	2.5 ±0.02 (-68.8)	9.1 ±0.06 (13.1)	8.3 ±0.06 (3.3)
OpenLLM-MUSR	8.4 ±0.06 (0.0)	2.4 ±0.02 (-71.5)	2.8 ±0.02 (-67.4)	5.5 ±0.04 (-35.0)	2.1 ±0.02 (-75.4)	2.3 ±0.02 (-72.6)	3.5 ±0.03 (-58.4)	6.8 ±0.05 (-19.0)	2.4 ±0.02 (-71.8)	4.8 ±0.03 (-43.5)	4.6 ±0.03 (-45.6)
ImageNet	7.6 ±0.12 (0.0)	2.8 ±0.05 (-62.7)	4.4 ±0.07 (-42.7)	4.4 ±0.08 (-42.5)	2.4 ±0.04 (-68.8)	2.8 ±0.06 (-63.3)	3.0 ±0.05 (-60.5)	15.3 ±0.25 (100.8)	2.1 ±0.04 (-72.7)	5.3 ±0.08 (-30.8)	5.3 ±0.08 (-30.6)
Average	7.7 ±0.03 (0.0)	4.2 ±0.02 (-43.0)	5.2 ±0.02 (-30.7)	5.4 ±0.02 (-28.1)	3.7 ±0.02 (-50.3)	6.1 ±0.03 (-17.7)	4.8 ±0.02 (-36.2)	6.9 ±0.02 (-8.3)	4.1 ±0.02 (-43.8)	5.9 ±0.02 (-20.1)	5.7 ±0.02 (-23.8)

Figure 7: The estimation gaps (↓) for target models (calculated as equation 1) under interpolation model split, where source models are identically distributed with target models. Each target model can only be evaluated on $n = 20$ data points. We also report ± the standard error of the mean and the estimation gap reduction (↓) over Random-Sampling in parentheses. A negative reduction implies that the method achieves a lower estimation gap than Random-Sampling. % is omitted. Best viewed in color.



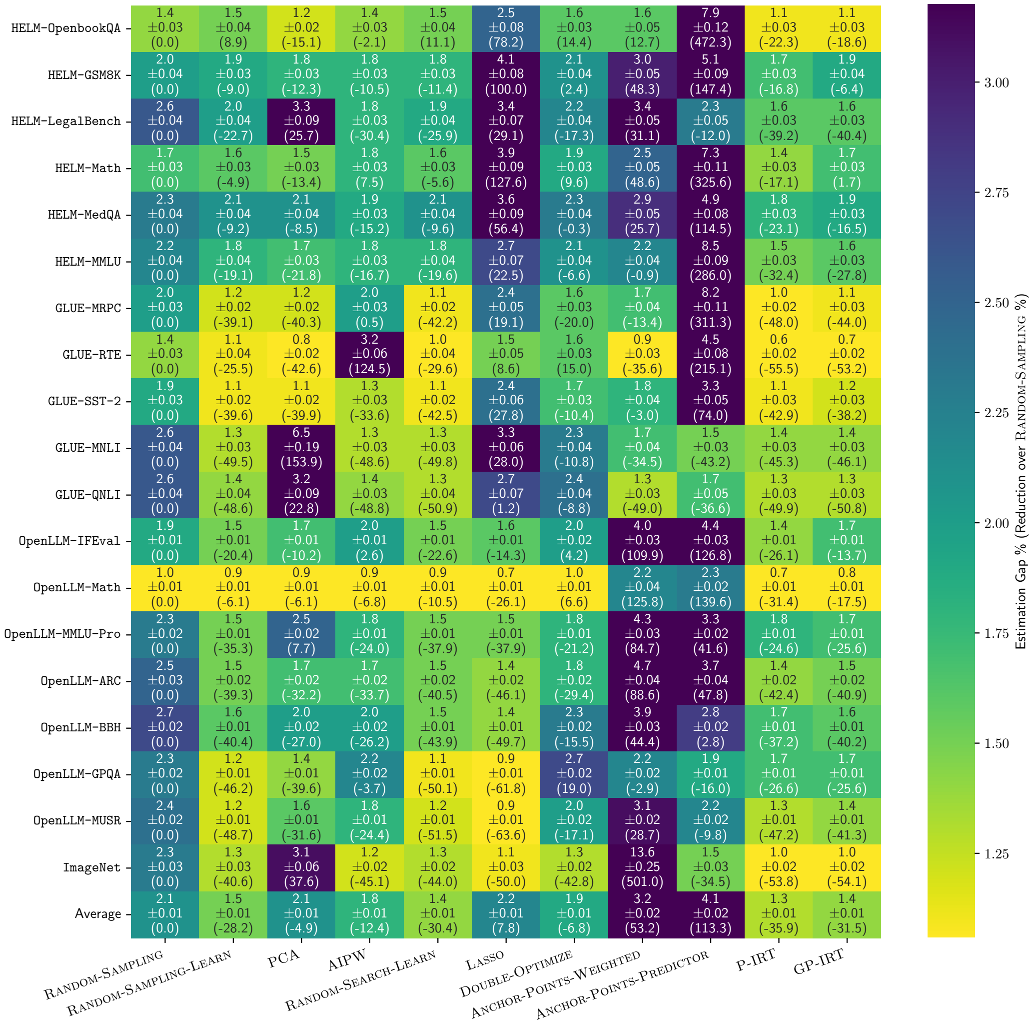
	Random-Sampling	Random-Sampling-Learn	PCA	AIPW	Random-Search-Learn	Lasso	Double-Optimize	Anchor-Points-Weighted	Anchor-Points-Predictor	P-IRT	GP-IRT
HELM-OpenbookQA	3.4 ±0.06 (0.0)	3.1 ±0.07 (-9.3)	2.8 ±0.06 (-18.0)	2.9 ±0.06 (-15.4)	3.0 ±0.07 (-13.2)	3.4 ±0.11 (-0.4)	2.8 ±0.05 (-19.1)	3.7 ±0.06 (7.7)	4.0 ±0.08 (16.8)	3.5 ±0.06 (3.4)	3.0 ±0.06 (-13.2)
HELM-GSM8K	4.4 ±0.08 (0.0)	4.0 ±0.07 (-10.5)	3.7 ±0.06 (-16.2)	3.8 ±0.07 (-13.9)	3.9 ±0.07 (-11.5)	6.8 ±0.15 (53.7)	3.9 ±0.07 (-10.9)	5.6 ±0.08 (26.7)	6.1 ±0.1 (37.7)	4.3 ±0.07 (-3.1)	4.0 ±0.07 (-9.5)
HELM-LegalBench	5.7 ±0.1 (0.0)	3.6 ±0.07 (-36.6)	4.3 ±0.1 (-24.7)	3.8 ±0.07 (-33.0)	3.3 ±0.06 (-42.2)	3.9 ±0.08 (-31.9)	3.4 ±0.06 (-39.5)	5.3 ±0.08 (-7.0)	3.2 ±0.07 (-42.8)	3.5 ±0.06 (-37.4)	3.5 ±0.06 (-38.8)
HELM-Math	4.1 ±0.07 (0.0)	3.9 ±0.07 (-4.4)	3.7 ±0.07 (-10.6)	3.8 ±0.07 (-6.6)	3.8 ±0.07 (-6.5)	6.0 ±0.09 (44.7)	3.8 ±0.07 (-7.4)	5.9 ±0.09 (43.6)	6.9 ±0.11 (68.1)	4.3 ±0.07 (4.2)	3.9 ±0.07 (-4.7)
HELM-MedQA	5.1 ±0.09 (0.0)	4.0 ±0.07 (-21.1)	3.8 ±0.07 (-25.0)	4.0 ±0.07 (-21.8)	4.1 ±0.08 (-20.0)	4.5 ±0.11 (-11.0)	3.9 ±0.07 (-24.2)	4.7 ±0.08 (-7.4)	5.2 ±0.09 (1.4)	4.2 ±0.07 (-17.8)	4.0 ±0.07 (-21.5)
HELM-MMLU	4.9 ±0.08 (0.0)	3.3 ±0.06 (-31.3)	3.4 ±0.06 (-30.6)	3.5 ±0.06 (-28.4)	3.3 ±0.06 (-31.5)	2.9 ±0.07 (-40.6)	3.2 ±0.06 (-34.3)	4.1 ±0.07 (-14.6)	3.8 ±0.07 (-21.7)	3.7 ±0.06 (-23.7)	3.6 ±0.06 (-25.3)
GLUE-MRPC	5.2 ±0.09 (0.0)	3.0 ±0.06 (-42.6)	2.7 ±0.06 (-47.0)	3.0 ±0.06 (-42.5)	2.6 ±0.06 (-49.1)	3.7 ±0.05 (-27.8)	3.3 ±0.06 (-35.5)	3.4 ±0.07 (-34.5)	3.3 ±0.06 (-35.7)	3.2 ±0.07 (-38.4)	3.0 ±0.06 (-43.1)
GLUE-RTE	5.2 ±0.09 (0.0)	2.3 ±0.07 (-54.8)	2.3 ±0.07 (-54.9)	2.6 ±0.06 (-50.6)	2.2 ±0.06 (-58.3)	2.4 ±0.07 (-53.6)	2.8 ±0.06 (-46.4)	2.2 ±0.07 (-57.2)	2.6 ±0.07 (-50.4)	2.3 ±0.06 (-55.7)	2.2 ±0.06 (-57.5)
GLUE-SST-2	4.5 ±0.08 (0.0)	2.7 ±0.05 (-41.6)	2.3 ±0.05 (-48.9)	2.7 ±0.06 (-41.5)	2.3 ±0.05 (-49.0)	4.9 ±0.07 (6.9)	2.7 ±0.05 (-41.3)	3.1 ±0.07 (-31.3)	2.7 ±0.05 (-41.5)	3.3 ±0.07 (-27.8)	2.9 ±0.06 (-36.0)
GLUE-MNLI	5.5 ±0.09 (0.0)	2.8 ±0.06 (-49.0)	9.4 ±0.22 (70.1)	2.8 ±0.06 (-49.5)	2.6 ±0.06 (-52.5)	5.1 ±0.09 (-7.2)	3.1 ±0.06 (-44.0)	2.7 ±0.07 (-50.3)	2.3 ±0.05 (-58.1)	3.2 ±0.07 (-41.4)	3.2 ±0.07 (-42.0)
GLUE-QNLI	5.3 ±0.09 (0.0)	2.7 ±0.07 (-49.0)	3.7 ±0.1 (-29.6)	2.9 ±0.07 (-46.4)	2.4 ±0.06 (-54.1)	2.7 ±0.08 (-48.8)	3.2 ±0.07 (-40.0)	2.5 ±0.07 (-53.6)	2.2 ±0.06 (-59.5)	2.8 ±0.07 (-48.1)	2.8 ±0.07 (-48.3)
OpenLLM-IFEval	4.6 ±0.03 (0.0)	3.2 ±0.02 (-28.6)	3.6 ±0.03 (-21.1)	3.6 ±0.03 (-20.3)	3.0 ±0.02 (-33.2)	5.1 ±0.05 (11.8)	3.2 ±0.02 (-29.6)	7.3 ±0.05 (59.4)	5.8 ±0.04 (27.8)	3.8 ±0.03 (-17.5)	3.6 ±0.03 (-21.1)
OpenLLM-Math	2.2 ±0.02 (0.0)	1.8 ±0.02 (-18.2)	1.9 ±0.02 (-12.2)	1.8 ±0.02 (-16.8)	1.5 ±0.02 (-29.2)	1.4 ±0.02 (-37.4)	1.3 ±0.02 (-41.7)	2.8 ±0.04 (27.5)	2.6 ±0.02 (20.9)	2.6 ±0.03 (19.8)	2.1 ±0.02 (-4.2)
OpenLLM-MMLU-Pro	4.9 ±0.04 (0.0)	2.9 ±0.02 (-41.5)	7.2 ±0.05 (45.4)	3.6 ±0.03 (-27.5)	2.4 ±0.02 (-50.8)	4.8 ±0.04 (-3.6)	2.6 ±0.02 (-46.8)	6.0 ±0.04 (21.1)	3.8 ±0.03 (-23.6)	4.3 ±0.03 (-13.2)	4.2 ±0.03 (-14.5)
OpenLLM-ARC	5.2 ±0.05 (0.0)	3.1 ±0.03 (-41.2)	3.3 ±0.03 (-37.4)	3.3 ±0.04 (-37.5)	2.7 ±0.03 (-47.9)	3.2 ±0.04 (-39.5)	2.7 ±0.03 (-48.1)	6.8 ±0.05 (29.2)	3.8 ±0.04 (-28.2)	3.5 ±0.04 (-32.3)	3.4 ±0.04 (-34.3)
OpenLLM-BBH	5.4 ±0.04 (0.0)	3.0 ±0.02 (-44.8)	4.5 ±0.03 (-16.7)	4.2 ±0.03 (-23.2)	2.6 ±0.02 (-52.7)	3.6 ±0.03 (-33.2)	3.0 ±0.02 (-45.1)	6.8 ±0.05 (25.5)	3.4 ±0.03 (-37.5)	3.7 ±0.03 (-32.4)	3.6 ±0.03 (-33.1)
OpenLLM-GPQA	4.9 ±0.04 (0.0)	1.8 ±0.01 (-63.5)	2.1 ±0.02 (-57.4)	4.2 ±0.03 (-14.1)	1.6 ±0.01 (-68.3)	1.4 ±0.01 (-71.3)	3.9 ±0.03 (-20.4)	4.3 ±0.03 (-12.3)	2.2 ±0.02 (-55.0)	4.1 ±0.03 (-16.8)	3.8 ±0.03 (-23.6)
OpenLLM-MUSR	5.4 ±0.04 (0.0)	1.9 ±0.01 (-64.1)	2.3 ±0.02 (-56.8)	3.3 ±0.03 (-38.8)	1.7 ±0.01 (-67.5)	1.7 ±0.01 (-67.6)	2.6 ±0.02 (-52.5)	5.0 ±0.04 (-6.8)	2.3 ±0.02 (-57.4)	2.9 ±0.02 (-46.7)	2.8 ±0.02 (-47.2)
ImageNet	4.9 ±0.07 (0.0)	2.2 ±0.04 (-54.7)	4.1 ±0.07 (-14.9)	2.4 ±0.04 (-50.1)	1.9 ±0.03 (-61.9)	1.9 ±0.05 (-61.6)	2.0 ±0.03 (-58.8)	10.6 ±0.21 (117.9)	1.9 ±0.03 (-61.7)	2.4 ±0.04 (-49.8)	2.4 ±0.04 (-50.3)
Average	4.8 ±0.02 (0.0)	2.9 ±0.01 (-37.2)	3.7 ±0.02 (-21.4)	3.3 ±0.01 (-30.4)	2.7 ±0.01 (-42.1)	3.6 ±0.02 (-22.0)	3.0 ±0.01 (-36.1)	4.9 ±0.02 (4.4)	3.6 ±0.01 (-21.1)	3.5 ±0.01 (-25.0)	3.3 ±0.01 (-29.9)

Figure 8: The estimation gaps (↓) for target models (calculated as equation 1) under interpolation model split, where source models are identically distributed with target models. Each target model can only be evaluated on $n = 50$ data points. We also report ± the standard error of the mean and the estimation gap reduction (↓) over RANDOM-SAMPLING in parentheses. A negative reduction implies that the method achieves a lower estimation gap than RANDOM-SAMPLING. % is omitted. Best viewed in color.



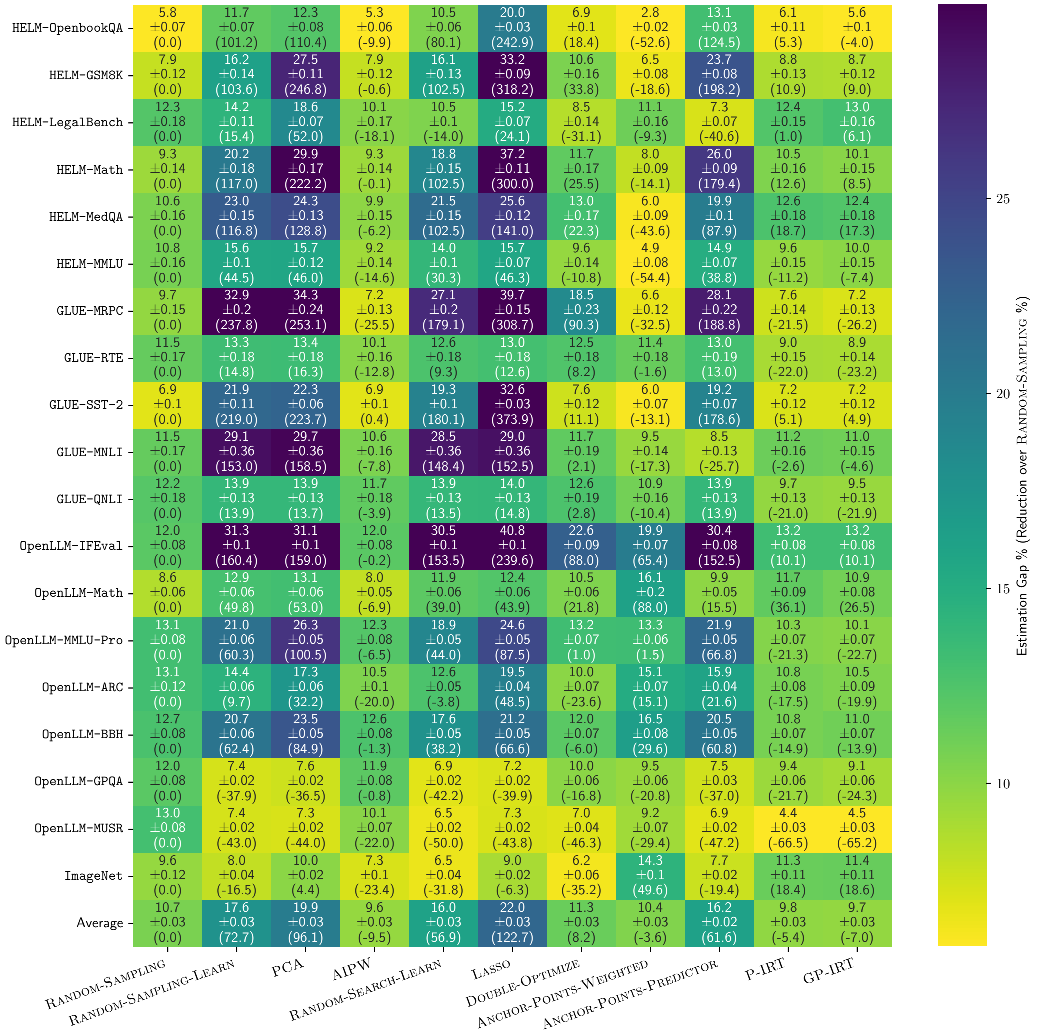
	RANDOM-SAMPLING	RANDOM-SAMPLING-LEARN	PCA	AIPW	RANDOM-SEARCH-LEARN	LASSO	DOUBLE-OPTIMIZE	ANCHOR-POINTS-WEIGHTED	ANCHOR-POINTS-PREDICTOR	P-IRT	GP-IRT
HELM-OpenbookQA	2.4 ±0.04 (0.0)	2.3 ±0.05 (-4.0)	2.0 ±0.04 (-16.5)	2.1 ±0.04 (-13.2)	2.2 ±0.05 (-8.8)	2.5 ±0.08 (3.7)	2.0 ±0.04 (-14.7)	2.6 ±0.06 (9.3)	4.6 ±0.1 (92.9)	1.9 ±0.04 (-21.8)	1.8 ±0.04 (-23.2)
HELM-GSM8K	3.0 ±0.05 (0.0)	2.7 ±0.05 (-11.3)	2.5 ±0.04 (-16.4)	2.6 ±0.05 (-14.2)	2.7 ±0.05 (-10.4)	4.7 ±0.09 (56.0)	2.7 ±0.05 (-10.5)	4.2 ±0.07 (39.0)	5.4 ±0.09 (80.0)	2.7 ±0.05 (-10.3)	2.7 ±0.05 (-9.1)
HELM-LegalBench	3.8 ±0.07 (0.0)	2.8 ±0.05 (-27.1)	3.8 ±0.09 (-1.8)	2.6 ±0.05 (-31.0)	2.6 ±0.05 (-32.2)	3.4 ±0.07 (-11.4)	2.6 ±0.05 (-33.2)	4.2 ±0.06 (10.3)	2.7 ±0.06 (-28.8)	2.3 ±0.04 (-40.7)	2.2 ±0.04 (-42.0)
HELM-Math	2.9 ±0.05 (0.0)	2.7 ±0.05 (-7.1)	2.4 ±0.04 (-17.9)	2.7 ±0.05 (-7.4)	2.7 ±0.05 (-8.3)	4.2 ±0.09 (44.1)	2.7 ±0.05 (-9.1)	4.3 ±0.07 (47.3)	6.5 ±0.11 (124.0)	2.5 ±0.05 (-12.8)	2.7 ±0.05 (-7.5)
HELM-MedQA	3.3 ±0.06 (0.0)	3.0 ±0.06 (-10.2)	2.9 ±0.05 (-13.2)	2.8 ±0.05 (-17.3)	3.0 ±0.06 (-9.3)	3.6 ±0.09 (7.7)	2.9 ±0.05 (-12.2)	3.6 ±0.06 (6.9)	4.9 ±0.09 (47.5)	2.7 ±0.05 (-18.6)	2.7 ±0.05 (-18.1)
HELM-MMLU	3.2 ±0.06 (0.0)	2.6 ±0.05 (-17.3)	2.5 ±0.04 (-20.8)	2.4 ±0.04 (-24.1)	2.6 ±0.05 (-18.9)	2.7 ±0.07 (-15.2)	2.5 ±0.04 (-22.5)	3.0 ±0.05 (-4.3)	4.1 ±0.08 (30.6)	2.2 ±0.04 (-29.2)	2.3 ±0.04 (-27.7)
GLUE-MRPC	3.3 ±0.05 (0.0)	2.0 ±0.04 (-38.9)	1.9 ±0.04 (-41.3)	2.1 ±0.04 (-34.9)	1.8 ±0.04 (-45.0)	2.4 ±0.05 (-28.0)	2.4 ±0.04 (-26.7)	2.7 ±0.06 (-18.6)	3.5 ±0.06 (5.8)	1.9 ±0.04 (-42.9)	1.8 ±0.04 (-45.0)
GLUE-RTE	3.3 ±0.06 (0.0)	1.6 ±0.05 (-50.3)	1.7 ±0.05 (-47.8)	2.3 ±0.04 (-28.3)	1.5 ±0.04 (-54.4)	1.7 ±0.05 (-48.8)	1.9 ±0.04 (-40.4)	1.8 ±0.05 (-46.0)	2.7 ±0.06 (-17.9)	1.4 ±0.04 (-56.7)	1.4 ±0.04 (-58.4)
GLUE-SST-2	3.0 ±0.05 (0.0)	1.8 ±0.03 (-39.7)	1.6 ±0.03 (-44.8)	1.8 ±0.04 (-40.2)	1.7 ±0.04 (-44.1)	2.9 ±0.05 (-3.3)	2.1 ±0.04 (-30.5)	2.5 ±0.06 (-15.4)	2.4 ±0.05 (-19.4)	1.8 ±0.04 (-40.7)	1.7 ±0.04 (-43.8)
GLUE-MNLI	3.8 ±0.06 (0.0)	1.9 ±0.04 (-50.0)	7.6 ±0.2 (101.3)	1.9 ±0.04 (-49.7)	1.9 ±0.04 (-50.6)	4.1 ±0.07 (7.6)	2.5 ±0.05 (-34.1)	2.1 ±0.05 (-44.1)	1.8 ±0.04 (-52.0)	2.2 ±0.05 (-41.7)	2.2 ±0.05 (-42.7)
GLUE-QNLI	3.8 ±0.06 (0.0)	1.9 ±0.05 (-50.3)	3.4 ±0.09 (-11.5)	2.0 ±0.05 (-47.6)	1.8 ±0.05 (-51.5)	2.8 ±0.08 (-25.5)	2.4 ±0.05 (-36.7)	1.8 ±0.05 (-52.0)	1.9 ±0.05 (-51.0)	2.0 ±0.05 (-48.5)	1.9 ±0.05 (-49.1)
OpenLLM-IFEval	2.9 ±0.02 (0.0)	2.3 ±0.02 (-22.2)	2.5 ±0.02 (-13.6)	2.5 ±0.02 (-14.6)	2.2 ±0.02 (-25.3)	3.0 ±0.03 (2.7)	2.3 ±0.02 (-20.7)	6.0 ±0.04 (105.1)	5.2 ±0.04 (76.9)	2.3 ±0.02 (-23.3)	2.4 ±0.02 (-18.7)
OpenLLM-Math	1.4 ±0.02 (0.0)	1.3 ±0.01 (-9.0)	1.4 ±0.01 (-4.6)	1.3 ±0.01 (-9.2)	1.2 ±0.01 (-18.7)	1.0 ±0.01 (-33.0)	1.1 ±0.02 (-21.4)	2.5 ±0.04 (75.5)	2.5 ±0.02 (75.4)	1.4 ±0.01 (-3.1)	1.2 ±0.01 (-15.7)
OpenLLM-MMLU-Pro	3.5 ±0.03 (0.0)	2.1 ±0.02 (-40.6)	4.8 ±0.03 (38.1)	2.5 ±0.02 (-28.5)	1.9 ±0.01 (-46.5)	2.8 ±0.03 (-20.5)	2.1 ±0.02 (-39.7)	5.1 ±0.04 (44.9)	3.5 ±0.03 (-0.7)	2.6 ±0.02 (-24.3)	2.6 ±0.02 (-25.3)
OpenLLM-ARC	3.8 ±0.04 (0.0)	2.2 ±0.02 (-42.4)	2.4 ±0.02 (-37.1)	2.3 ±0.03 (-39.1)	2.0 ±0.02 (-47.0)	1.9 ±0.02 (-50.6)	2.1 ±0.02 (-45.1)	5.9 ±0.05 (54.7)	3.7 ±0.04 (-3.6)	2.2 ±0.02 (-42.8)	2.1 ±0.02 (-43.6)
OpenLLM-BBH	4.0 ±0.03 (0.0)	2.2 ±0.02 (-44.9)	2.8 ±0.02 (-29.8)	2.9 ±0.02 (-27.3)	2.0 ±0.02 (-48.9)	2.2 ±0.02 (-45.1)	2.4 ±0.02 (-40.3)	5.2 ±0.04 (31.4)	3.0 ±0.02 (-23.5)	2.4 ±0.02 (-39.7)	2.3 ±0.02 (-40.9)
OpenLLM-GPQA	3.4 ±0.02 (0.0)	1.5 ±0.01 (-56.3)	1.7 ±0.01 (-49.6)	3.0 ±0.02 (-11.1)	1.3 ±0.01 (-60.5)	1.0 ±0.01 (-69.2)	3.0 ±0.02 (-11.0)	3.2 ±0.02 (-7.1)	2.1 ±0.02 (-39.6)	2.5 ±0.02 (-25.9)	2.5 ±0.02 (-27.7)
OpenLLM-MUSR	3.7 ±0.03 (0.0)	1.6 ±0.01 (-57.5)	2.0 ±0.01 (-46.1)	2.3 ±0.02 (-37.3)	1.5 ±0.01 (-60.0)	1.3 ±0.01 (-66.0)	2.1 ±0.02 (-42.9)	3.9 ±0.03 (4.0)	2.2 ±0.02 (-39.9)	1.8 ±0.01 (-51.6)	1.9 ±0.01 (-48.8)
ImageNet	3.3 ±0.05 (0.0)	1.7 ±0.03 (-46.9)	3.8 ±0.07 (15.3)	1.7 ±0.03 (-47.8)	1.6 ±0.03 (-52.3)	1.5 ±0.04 (-55.6)	1.6 ±0.02 (-52.6)	11.2 ±0.22 (241.8)	1.6 ±0.03 (-49.8)	1.6 ±0.02 (-51.2)	1.6 ±0.02 (-51.1)
Average	3.3 ±0.01 (0.0)	2.1 ±0.01 (-32.9)	2.8 ±0.01 (-13.6)	2.3 ±0.01 (-27.5)	2.0 ±0.01 (-36.4)	2.6 ±0.01 (-18.4)	2.3 ±0.01 (-28.6)	4.0 ±0.02 (25.4)	3.4 ±0.01 (10.9)	2.1 ±0.01 (-32.9)	2.1 ±0.01 (-33.6)

Figure 9: The estimation gaps (↓) for target models (calculated as equation 1) under interpolation model split, where source models are identically distributed with target models. Each target model can only be evaluated on $n = 100$ data points. We also report ± the standard error of the mean and the estimation gap reduction (↓) over RANDOM-SAMPLING in parentheses. A negative reduction implies that the method achieves a lower estimation gap than RANDOM-SAMPLING. % is omitted. Best viewed in color.



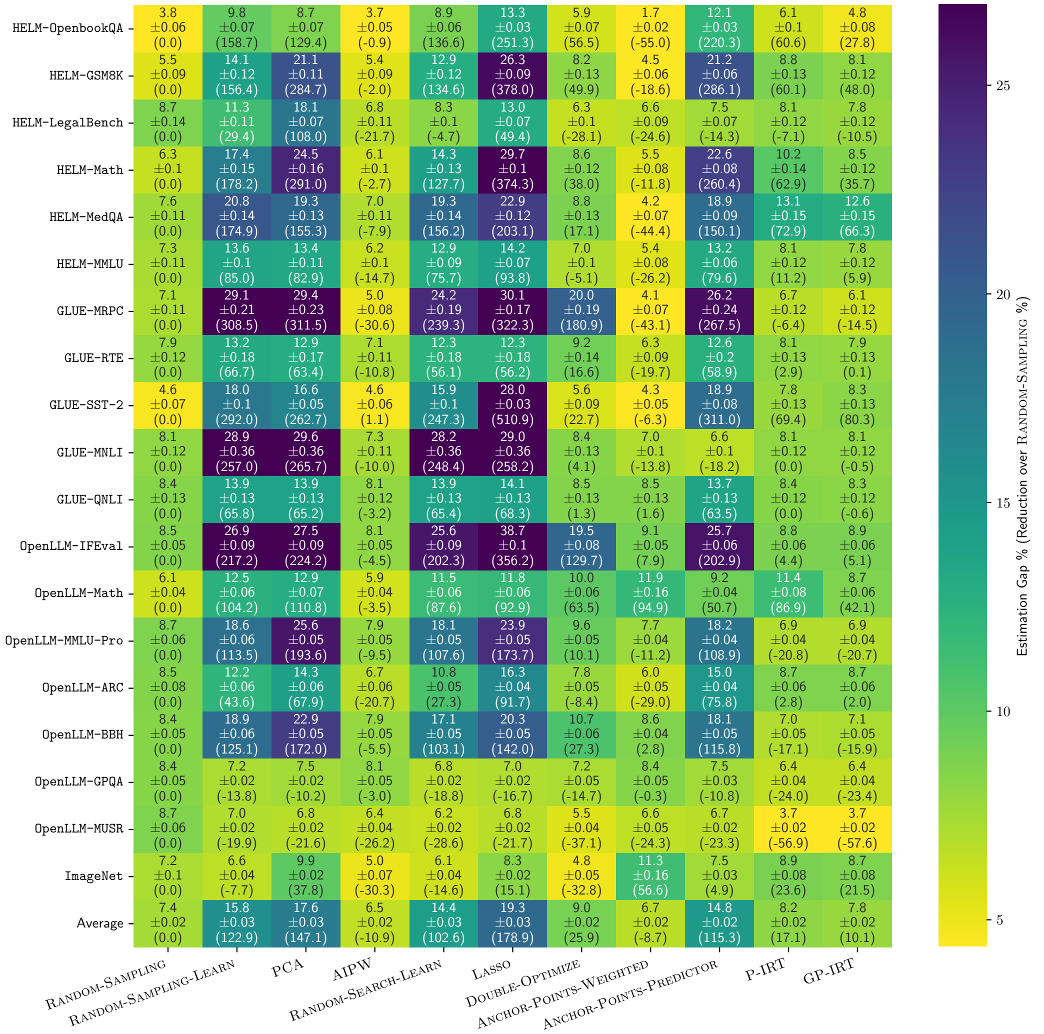
	RANDOM-SAMPLING	RANDOM-SAMPLING-LEARN	PCA	AIPW	RANDOM-SEARCH-LEARN	LASSO	DOUBLE-OPTIMIZE	ANCHOR-POINTS-WEIGHTED	ANCHOR-POINTS-PREDICTOR	P-IRT	GP-IRT
HELM-OpenbookQA	1.4 ±0.03 (0.0)	1.5 ±0.04 (8.9)	1.2 ±0.02 (-15.1)	1.4 ±0.03 (-2.1)	1.5 ±0.04 (11.1)	2.5 ±0.08 (78.2)	1.6 ±0.03 (14.4)	1.6 ±0.05 (12.7)	7.9 ±0.12 (472.3)	1.1 ±0.03 (-22.3)	1.1 ±0.03 (-18.6)
HELM-GSM8K	2.0 ±0.04 (0.0)	1.9 ±0.03 (-9.0)	1.8 ±0.03 (-12.3)	1.8 ±0.03 (-10.5)	1.8 ±0.03 (-11.4)	4.1 ±0.08 (100.0)	2.1 ±0.04 (2.4)	3.0 ±0.05 (48.3)	5.1 ±0.09 (147.4)	1.7 ±0.03 (-16.8)	1.9 ±0.04 (-6.4)
HELM-LegalBench	2.6 ±0.04 (0.0)	2.0 ±0.04 (-22.7)	3.3 ±0.09 (25.7)	1.8 ±0.03 (-30.4)	1.9 ±0.04 (-25.9)	3.4 ±0.07 (29.1)	2.2 ±0.04 (-17.3)	3.4 ±0.05 (31.1)	2.3 ±0.05 (-12.0)	1.6 ±0.03 (-39.2)	1.6 ±0.03 (-40.4)
HELM-Math	1.7 ±0.03 (0.0)	1.6 ±0.03 (-4.9)	1.5 ±0.03 (-13.4)	1.8 ±0.03 (7.5)	1.6 ±0.03 (-5.6)	3.9 ±0.09 (127.6)	1.9 ±0.03 (9.6)	2.5 ±0.05 (48.6)	7.3 ±0.11 (325.6)	1.4 ±0.03 (-17.1)	1.7 ±0.03 (1.7)
HELM-MedQA	2.3 ±0.04 (0.0)	2.1 ±0.04 (-9.2)	2.1 ±0.04 (-8.5)	1.9 ±0.03 (-15.2)	2.1 ±0.04 (-9.6)	3.6 ±0.09 (56.4)	2.3 ±0.04 (-0.3)	2.9 ±0.05 (25.7)	4.9 ±0.08 (114.5)	1.8 ±0.03 (-23.1)	1.9 ±0.03 (-16.5)
HELM-MMLU	2.2 ±0.04 (0.0)	1.8 ±0.04 (-19.1)	1.7 ±0.03 (-21.8)	1.8 ±0.03 (-16.7)	1.8 ±0.04 (-19.6)	2.7 ±0.07 (22.5)	2.1 ±0.04 (-6.6)	2.2 ±0.04 (-0.9)	8.5 ±0.09 (286.0)	1.5 ±0.03 (-32.4)	1.6 ±0.03 (-27.8)
GLUE-MRPC	2.0 ±0.03 (0.0)	1.2 ±0.02 (-39.1)	1.2 ±0.02 (-40.3)	2.0 ±0.03 (0.5)	1.1 ±0.02 (-42.2)	2.4 ±0.05 (19.1)	1.6 ±0.03 (-20.0)	1.7 ±0.04 (-13.4)	8.2 ±0.11 (311.3)	1.0 ±0.02 (-48.0)	1.1 ±0.03 (-44.0)
GLUE-RTE	1.4 ±0.03 (0.0)	1.1 ±0.04 (-25.5)	0.8 ±0.02 (-42.6)	3.2 ±0.06 (124.5)	1.0 ±0.04 (-29.6)	1.5 ±0.05 (8.6)	1.6 ±0.03 (15.0)	0.9 ±0.03 (-35.6)	4.5 ±0.08 (215.1)	0.6 ±0.02 (-55.5)	0.7 ±0.02 (-53.2)
GLUE-SST-2	1.9 ±0.03 (0.0)	1.1 ±0.02 (-39.6)	1.1 ±0.02 (-39.9)	1.3 ±0.03 (-33.6)	1.1 ±0.02 (-42.5)	2.4 ±0.06 (27.8)	1.7 ±0.03 (-10.4)	1.8 ±0.04 (-3.0)	3.3 ±0.05 (74.0)	1.1 ±0.03 (-42.9)	1.2 ±0.03 (-38.2)
GLUE-MNLI	2.6 ±0.04 (0.0)	1.3 ±0.03 (-49.5)	6.5 ±0.19 (153.9)	1.3 ±0.03 (-48.6)	1.3 ±0.03 (-49.8)	3.3 ±0.06 (28.0)	2.3 ±0.04 (-10.8)	1.7 ±0.04 (-34.5)	1.5 ±0.03 (-43.2)	1.4 ±0.03 (-45.3)	1.4 ±0.03 (-46.1)
GLUE-QNLI	2.6 ±0.04 (0.0)	1.4 ±0.04 (-48.6)	3.2 ±0.09 (22.8)	1.4 ±0.03 (-48.8)	1.3 ±0.04 (-50.9)	2.7 ±0.07 (1.2)	2.4 ±0.04 (-8.8)	1.3 ±0.03 (-49.0)	1.7 ±0.05 (-36.6)	1.3 ±0.03 (-49.9)	1.3 ±0.03 (-50.8)
OpenLLM-IFEval	1.9 ±0.01 (0.0)	1.5 ±0.01 (-20.4)	1.7 ±0.01 (-10.2)	2.0 ±0.01 (2.6)	1.5 ±0.01 (-22.6)	1.6 ±0.01 (-14.3)	2.0 ±0.02 (4.2)	4.0 ±0.03 (109.9)	4.4 ±0.03 (126.8)	1.4 ±0.01 (-26.1)	1.7 ±0.01 (-13.7)
OpenLLM-Math	1.0 ±0.01 (0.0)	0.9 ±0.01 (-6.1)	0.9 ±0.01 (-6.1)	0.9 ±0.01 (-6.8)	0.9 ±0.01 (-10.5)	0.7 ±0.01 (-26.1)	1.0 ±0.01 (6.6)	2.2 ±0.04 (125.8)	2.3 ±0.02 (139.6)	0.7 ±0.01 (-31.4)	0.8 ±0.01 (-17.5)
OpenLLM-MMLU-Pro	2.3 ±0.02 (0.0)	1.5 ±0.01 (-35.3)	2.5 ±0.02 (7.7)	1.8 ±0.01 (-24.0)	1.5 ±0.01 (-37.9)	1.5 ±0.01 (-37.9)	1.8 ±0.01 (-21.2)	4.3 ±0.03 (84.7)	3.3 ±0.02 (41.6)	1.8 ±0.01 (-24.6)	1.7 ±0.01 (-25.6)
OpenLLM-ARC	2.5 ±0.03 (0.0)	1.5 ±0.02 (-39.3)	1.7 ±0.02 (-32.2)	1.7 ±0.02 (-33.7)	1.5 ±0.02 (-40.5)	1.4 ±0.02 (-46.1)	1.8 ±0.02 (-29.4)	4.7 ±0.04 (88.6)	3.7 ±0.04 (47.8)	1.4 ±0.02 (-42.4)	1.5 ±0.02 (-40.9)
OpenLLM-BBH	2.7 ±0.02 (0.0)	1.6 ±0.01 (-40.4)	2.0 ±0.02 (-27.0)	2.0 ±0.02 (-26.2)	1.5 ±0.01 (-43.9)	1.4 ±0.01 (-49.7)	2.3 ±0.02 (-15.5)	3.9 ±0.03 (44.4)	2.8 ±0.02 (2.8)	1.7 ±0.01 (-37.2)	1.6 ±0.01 (-40.2)
OpenLLM-GPQA	2.3 ±0.02 (0.0)	1.2 ±0.01 (-46.2)	1.4 ±0.01 (-39.6)	2.2 ±0.02 (-3.7)	1.1 ±0.01 (-50.1)	0.9 ±0.01 (-61.8)	2.7 ±0.02 (19.0)	2.2 ±0.02 (-2.9)	1.9 ±0.01 (-16.0)	1.7 ±0.01 (-26.6)	1.7 ±0.01 (-25.6)
OpenLLM-MUSR	2.4 ±0.02 (0.0)	1.2 ±0.01 (-48.7)	1.6 ±0.01 (-31.6)	1.8 ±0.01 (-24.4)	1.2 ±0.01 (-51.5)	0.9 ±0.01 (-63.6)	2.0 ±0.02 (-17.1)	3.1 ±0.02 (28.7)	2.2 ±0.02 (-9.8)	1.3 ±0.01 (-47.2)	1.4 ±0.01 (-41.3)
ImageNet	2.3 ±0.03 (0.0)	1.3 ±0.03 (-40.6)	3.1 ±0.06 (37.6)	1.2 ±0.02 (-45.1)	1.3 ±0.02 (-44.0)	1.1 ±0.03 (-50.0)	1.3 ±0.02 (-42.8)	13.6 ±0.25 (501.0)	1.5 ±0.03 (-34.5)	1.0 ±0.02 (-53.8)	1.0 ±0.02 (-54.1)
Average	2.1 ±0.01 (0.0)	1.5 ±0.01 (-28.2)	2.1 ±0.01 (-4.9)	1.8 ±0.01 (-12.4)	1.4 ±0.01 (-30.4)	2.2 ±0.01 (7.8)	1.9 ±0.01 (-6.8)	3.2 ±0.02 (53.2)	4.1 ±0.02 (113.3)	1.3 ±0.01 (-35.9)	1.4 ±0.01 (-31.5)

Figure 10: The estimation gaps (↓) for target models (calculated as equation 1) under interpolation model split, where source models are identically distributed with target models. Each target model can only be evaluated on $n = 200$ data points. We also report ± the standard error of the mean and the estimation gap reduction (↓) over RANDOM-SAMPLING in parentheses. A negative reduction implies that the method achieves a lower estimation gap than RANDOM-SAMPLING. % is omitted. Best viewed in color.



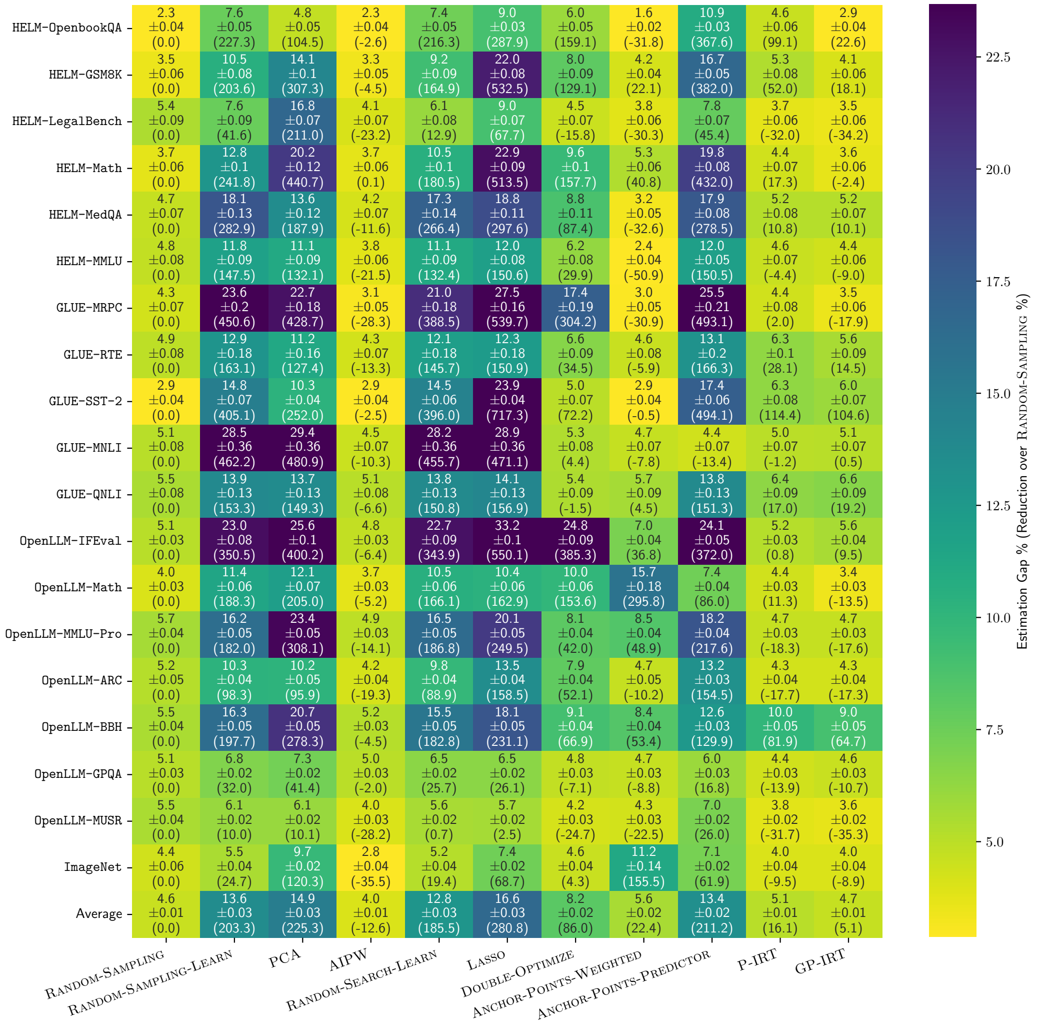
	Random-Sampling	Random-Sampling-Learn	PCA	AIPW	Random-Search-Learn	Lasso	Double-Optimize	Anchor-Points-Weighted	Anchor-Points-Predictor	P-IRT	GP-IRT
HELM-OpenbookQA	5.8 ±0.07 (0.0)	11.7 ±0.07 (101.2)	12.3 ±0.08 (110.4)	5.3 ±0.06 (-9.9)	10.5 ±0.06 (80.1)	20.0 ±0.03 (242.9)	6.9 ±0.1 (18.4)	2.8 ±0.02 (-52.6)	13.1 ±0.03 (124.5)	6.1 ±0.11 (5.3)	5.6 ±0.1 (-4.0)
HELM-GSM8K	7.9 ±0.12 (0.0)	16.2 ±0.14 (103.6)	27.5 ±0.11 (246.8)	7.9 ±0.12 (-0.6)	16.1 ±0.13 (102.5)	33.2 ±0.09 (318.2)	10.6 ±0.16 (33.8)	6.5 ±0.08 (-18.6)	23.7 ±0.08 (198.2)	8.8 ±0.13 (10.9)	8.7 ±0.12 (9.0)
HELM-LegalBench	12.3 ±0.18 (0.0)	14.2 ±0.11 (15.4)	18.6 ±0.07 (52.0)	10.1 ±0.17 (-18.1)	10.5 ±0.1 (-14.0)	15.2 ±0.07 (24.1)	8.5 ±0.14 (-31.1)	11.1 ±0.16 (-9.3)	7.3 ±0.07 (-40.6)	12.4 ±0.15 (1.0)	13.0 ±0.16 (6.1)
HELM-Math	9.3 ±0.14 (0.0)	20.2 ±0.18 (117.0)	29.9 ±0.17 (222.2)	9.3 ±0.14 (-0.1)	18.8 ±0.15 (102.5)	37.2 ±0.11 (300.0)	11.7 ±0.17 (25.5)	8.0 ±0.09 (-14.1)	26.0 ±0.09 (179.4)	10.5 ±0.16 (12.6)	10.1 ±0.15 (8.5)
HELM-MedQA	10.6 ±0.16 (0.0)	23.0 ±0.15 (116.8)	24.3 ±0.13 (128.8)	9.9 ±0.15 (-6.2)	21.5 ±0.15 (102.5)	25.6 ±0.12 (141.0)	13.0 ±0.17 (22.3)	6.0 ±0.09 (-43.6)	19.9 ±0.1 (87.9)	12.6 ±0.18 (18.7)	12.4 ±0.18 (17.3)
HELM-MMLU	10.8 ±0.16 (0.0)	15.6 ±0.1 (44.5)	15.7 ±0.12 (46.0)	9.2 ±0.14 (-14.6)	14.0 ±0.1 (30.3)	15.7 ±0.07 (46.3)	9.6 ±0.14 (-10.8)	4.9 ±0.08 (-54.4)	14.9 ±0.07 (38.8)	9.6 ±0.15 (-11.2)	10.0 ±0.15 (-7.4)
GLUE-MRPC	9.7 ±0.15 (0.0)	32.9 ±0.2 (237.8)	34.3 ±0.24 (253.1)	7.2 ±0.13 (-25.5)	27.1 ±0.2 (179.1)	39.7 ±0.15 (308.7)	18.5 ±0.23 (90.3)	6.6 ±0.12 (-32.5)	28.1 ±0.22 (188.8)	7.6 ±0.14 (-21.5)	7.2 ±0.13 (-26.2)
GLUE-RTE	11.5 ±0.17 (0.0)	13.3 ±0.18 (14.8)	13.4 ±0.18 (16.3)	10.1 ±0.16 (-12.8)	12.6 ±0.18 (9.3)	13.0 ±0.18 (12.6)	12.5 ±0.18 (8.2)	11.4 ±0.18 (-1.6)	13.0 ±0.19 (13.0)	9.0 ±0.15 (-22.0)	8.9 ±0.14 (-23.2)
GLUE-SST-2	6.9 ±0.1 (0.0)	21.9 ±0.11 (219.0)	22.3 ±0.06 (223.7)	6.9 ±0.1 (0.4)	19.3 ±0.1 (180.1)	32.6 ±0.03 (373.9)	7.6 ±0.12 (11.1)	6.0 ±0.07 (-13.1)	19.2 ±0.07 (178.6)	7.2 ±0.12 (5.1)	7.2 ±0.12 (4.9)
GLUE-MNLI	11.5 ±0.17 (0.0)	29.1 ±0.36 (153.0)	29.7 ±0.36 (158.5)	10.6 ±0.16 (-7.8)	28.5 ±0.36 (148.4)	29.0 ±0.36 (152.5)	11.7 ±0.19 (2.1)	9.5 ±0.14 (-17.3)	8.5 ±0.13 (-25.7)	11.2 ±0.16 (-2.6)	11.0 ±0.15 (-4.6)
GLUE-QNLI	12.2 ±0.18 (0.0)	13.9 ±0.13 (13.9)	13.9 ±0.13 (13.7)	11.7 ±0.18 (-3.9)	13.9 ±0.13 (13.5)	14.0 ±0.13 (14.8)	12.6 ±0.19 (2.8)	10.9 ±0.16 (-10.4)	13.9 ±0.13 (13.9)	9.7 ±0.13 (-21.0)	9.5 ±0.13 (-21.9)
OpenLLM-IFEval	12.0 ±0.08 (0.0)	31.3 ±0.1 (160.4)	31.1 ±0.1 (159.0)	12.0 ±0.08 (-0.2)	30.5 ±0.1 (153.5)	40.8 ±0.1 (239.6)	22.6 ±0.09 (88.0)	19.9 ±0.07 (65.4)	30.4 ±0.08 (152.5)	13.2 ±0.08 (10.1)	13.2 ±0.08 (10.1)
OpenLLM-Math	8.6 ±0.06 (0.0)	12.9 ±0.06 (49.8)	13.1 ±0.06 (53.0)	8.0 ±0.05 (-6.9)	11.9 ±0.06 (39.0)	12.4 ±0.06 (43.9)	10.5 ±0.06 (21.8)	16.1 ±0.2 (88.0)	9.9 ±0.05 (15.5)	11.7 ±0.09 (36.1)	10.9 ±0.08 (26.5)
OpenLLM-MMLU-Pro	13.1 ±0.08 (0.0)	21.0 ±0.06 (60.3)	26.3 ±0.05 (100.5)	12.3 ±0.08 (-6.5)	18.9 ±0.05 (44.0)	24.6 ±0.05 (87.5)	13.2 ±0.07 (1.0)	13.3 ±0.06 (1.5)	21.9 ±0.05 (66.8)	10.3 ±0.07 (-21.3)	10.1 ±0.07 (-22.7)
OpenLLM-ARC	13.1 ±0.12 (0.0)	14.4 ±0.06 (9.7)	17.3 ±0.06 (32.2)	10.5 ±0.1 (-20.0)	12.6 ±0.05 (-3.8)	19.5 ±0.04 (48.5)	10.0 ±0.07 (-23.6)	15.1 ±0.07 (15.1)	15.9 ±0.04 (21.6)	10.8 ±0.08 (-17.5)	10.5 ±0.09 (-19.9)
OpenLLM-BBH	12.7 ±0.08 (0.0)	20.7 ±0.06 (62.4)	23.5 ±0.05 (84.9)	12.6 ±0.08 (-1.3)	17.6 ±0.05 (38.2)	21.2 ±0.05 (66.6)	12.0 ±0.07 (-6.0)	16.5 ±0.08 (29.6)	20.5 ±0.05 (60.8)	10.8 ±0.07 (-14.9)	11.0 ±0.07 (-13.9)
OpenLLM-GPQA	12.0 ±0.08 (0.0)	7.4 ±0.02 (-37.9)	7.6 ±0.02 (-36.5)	11.9 ±0.08 (-0.8)	6.9 ±0.02 (-42.2)	7.2 ±0.02 (-39.9)	10.0 ±0.06 (-16.8)	9.5 ±0.06 (-20.8)	7.5 ±0.03 (-37.0)	9.4 ±0.06 (-21.7)	9.1 ±0.06 (-24.3)
OpenLLM-MUSR	13.0 ±0.08 (0.0)	7.4 ±0.02 (-43.0)	7.3 ±0.02 (-44.0)	10.1 ±0.07 (-22.0)	6.5 ±0.02 (-50.0)	7.3 ±0.02 (-43.8)	7.0 ±0.04 (-46.3)	9.2 ±0.07 (-29.4)	6.9 ±0.02 (-47.2)	4.4 ±0.03 (-66.5)	4.5 ±0.03 (-65.2)
ImageNet	9.6 ±0.12 (0.0)	8.0 ±0.04 (-16.5)	10.0 ±0.02 (4.4)	7.3 ±0.1 (-23.4)	6.5 ±0.04 (-31.8)	9.0 ±0.02 (-6.3)	6.2 ±0.06 (-35.2)	14.3 ±0.1 (49.6)	7.7 ±0.02 (-19.4)	11.3 ±0.11 (18.4)	11.4 ±0.11 (18.6)
Average	10.7 ±0.03 (0.0)	17.6 ±0.03 (72.7)	19.9 ±0.03 (96.1)	9.6 ±0.03 (-9.5)	16.0 ±0.03 (56.9)	22.0 ±0.03 (122.7)	11.3 ±0.03 (8.2)	10.4 ±0.03 (-3.6)	16.2 ±0.02 (61.6)	9.8 ±0.03 (-5.4)	9.7 ±0.03 (-7.0)

Figure 11: The estimation gaps (↓) for target models (calculated as equation 1) under extrapolation model split, where source models are the lowest-performing 50%, and target models are the top 30% based on average performance over the full benchmark. Each target model can only be evaluated on $n = 10$ data points. We also report ± the standard error of the mean and the estimation gap reduction (↓) over Random-Sampling in parentheses. A negative reduction implies that the method achieves a lower estimation gap than Random-Sampling. % is omitted. Best viewed in color.



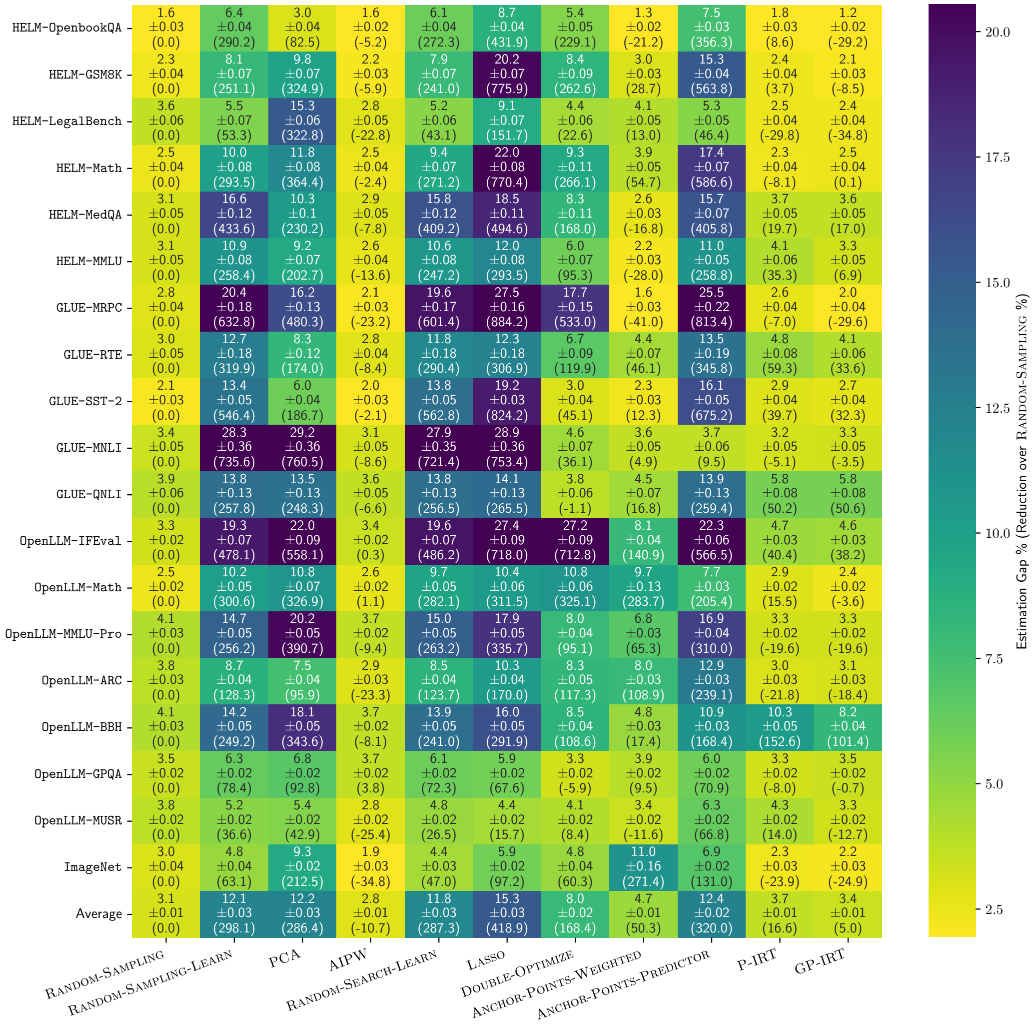
	Random-Sampling	Random-Sampling-Learn	PCA	AIPW	Random-Search-Learn	Lasso	Double-Optimize	Anchor-Points-Weighted	Anchor-Points-Predictor	P-IRT	GP-IRT
HELM-OpenbookQA	3.8 ±0.06 (0.0)	9.8 ±0.07 (158.7)	8.7 ±0.07 (129.4)	3.7 ±0.05 (-0.9)	8.9 ±0.06 (136.6)	13.3 ±0.03 (251.3)	5.9 ±0.07 (56.5)	1.7 ±0.02 (-55.0)	12.1 ±0.03 (220.3)	6.1 ±0.1 (60.6)	4.8 ±0.08 (27.8)
HELM-GSM8K	5.5 ±0.09 (0.0)	14.1 ±0.12 (156.4)	21.1 ±0.11 (284.7)	5.4 ±0.09 (-2.0)	12.9 ±0.12 (134.6)	26.3 ±0.09 (378.0)	8.2 ±0.13 (49.9)	4.5 ±0.06 (-18.6)	21.2 ±0.06 (286.1)	8.8 ±0.13 (60.1)	8.1 ±0.12 (48.0)
HELM-LegalBench	8.7 ±0.14 (0.0)	11.3 ±0.11 (29.4)	18.1 ±0.07 (108.0)	6.8 ±0.11 (-21.7)	8.3 ±0.1 (-4.7)	13.0 ±0.07 (49.4)	6.3 ±0.1 (-28.1)	6.6 ±0.09 (-24.6)	7.5 ±0.07 (-14.3)	8.1 ±0.12 (-7.1)	7.8 ±0.12 (-10.5)
HELM-Math	6.3 ±0.1 (0.0)	17.4 ±0.15 (178.2)	24.5 ±0.16 (291.0)	6.1 ±0.1 (-2.7)	14.3 ±0.13 (127.7)	29.7 ±0.1 (374.3)	8.6 ±0.12 (38.0)	5.5 ±0.08 (-11.8)	22.6 ±0.08 (260.4)	10.2 ±0.14 (62.9)	8.5 ±0.12 (35.7)
HELM-MedQA	7.6 ±0.11 (0.0)	20.8 ±0.14 (174.9)	19.3 ±0.13 (155.3)	7.0 ±0.11 (-7.9)	19.3 ±0.14 (156.2)	22.9 ±0.12 (203.1)	8.8 ±0.13 (17.1)	4.2 ±0.07 (-44.4)	18.9 ±0.09 (150.1)	13.1 ±0.15 (72.9)	12.6 ±0.15 (66.3)
HELM-MMLU	7.3 ±0.11 (0.0)	13.6 ±0.1 (85.0)	13.4 ±0.11 (82.9)	6.2 ±0.1 (-14.7)	12.9 ±0.09 (75.7)	14.2 ±0.07 (93.8)	7.0 ±0.1 (-5.1)	5.4 ±0.08 (-26.2)	13.2 ±0.06 (79.6)	8.1 ±0.12 (11.2)	7.8 ±0.12 (5.9)
GLUE-MRPC	7.1 ±0.11 (0.0)	29.1 ±0.21 (308.5)	29.4 ±0.23 (311.5)	5.0 ±0.08 (-30.6)	24.2 ±0.19 (239.3)	30.1 ±0.17 (322.3)	20.0 ±0.19 (180.9)	4.1 ±0.07 (-43.1)	26.2 ±0.24 (267.5)	6.7 ±0.12 (-6.4)	6.1 ±0.12 (-14.5)
GLUE-RTE	7.9 ±0.12 (0.0)	13.2 ±0.18 (66.7)	12.9 ±0.17 (63.4)	7.1 ±0.11 (-10.8)	12.3 ±0.18 (56.1)	12.3 ±0.18 (56.2)	9.2 ±0.14 (16.6)	6.3 ±0.09 (-19.7)	12.6 ±0.2 (58.9)	8.1 ±0.13 (2.9)	7.9 ±0.13 (0.1)
GLUE-SST-2	4.6 ±0.07 (0.0)	18.0 ±0.1 (292.0)	16.6 ±0.05 (262.7)	4.6 ±0.06 (1.1)	15.9 ±0.1 (247.3)	28.0 ±0.03 (510.9)	5.6 ±0.09 (22.7)	4.3 ±0.05 (-6.3)	18.9 ±0.08 (311.0)	7.8 ±0.13 (69.4)	8.3 ±0.13 (80.3)
GLUE-MNLI	8.1 ±0.12 (0.0)	28.9 ±0.36 (257.0)	29.6 ±0.36 (265.7)	7.3 ±0.11 (-10.0)	28.2 ±0.36 (248.4)	29.0 ±0.36 (258.2)	8.4 ±0.13 (4.1)	7.0 ±0.1 (-13.8)	6.6 ±0.1 (-18.2)	8.1 ±0.12 (0.0)	8.1 ±0.12 (-0.5)
GLUE-QNLI	8.4 ±0.13 (0.0)	13.9 ±0.13 (65.8)	13.9 ±0.13 (65.2)	8.1 ±0.12 (-3.2)	13.9 ±0.13 (65.4)	14.1 ±0.13 (68.3)	8.5 ±0.13 (1.3)	8.5 ±0.13 (1.6)	13.7 ±0.13 (63.5)	8.4 ±0.12 (0.0)	8.3 ±0.12 (-0.6)
OpenLLM-IFEval	8.5 ±0.05 (0.0)	26.9 ±0.09 (217.2)	27.5 ±0.09 (224.2)	8.1 ±0.05 (-4.5)	25.6 ±0.09 (202.3)	38.7 ±0.1 (356.2)	19.5 ±0.08 (129.7)	9.1 ±0.05 (7.9)	25.7 ±0.06 (202.9)	8.8 ±0.06 (4.4)	8.9 ±0.06 (5.1)
OpenLLM-Math	6.1 ±0.04 (0.0)	12.5 ±0.06 (104.2)	12.9 ±0.07 (110.8)	5.9 ±0.04 (-3.5)	11.5 ±0.06 (87.6)	11.8 ±0.06 (92.9)	10.0 ±0.06 (63.5)	11.9 ±0.16 (94.9)	9.2 ±0.04 (50.7)	11.4 ±0.08 (86.9)	8.7 ±0.06 (42.1)
OpenLLM-MMLU-Pro	8.7 ±0.06 (0.0)	18.6 ±0.06 (113.5)	25.6 ±0.05 (193.6)	7.9 ±0.05 (-9.5)	18.1 ±0.05 (107.6)	23.9 ±0.05 (173.7)	9.6 ±0.05 (10.1)	7.7 ±0.04 (-11.2)	18.2 ±0.04 (108.9)	6.9 ±0.04 (-20.8)	6.9 ±0.04 (-20.7)
OpenLLM-ARC	8.5 ±0.08 (0.0)	12.2 ±0.06 (43.6)	14.3 ±0.06 (67.9)	6.7 ±0.06 (-20.7)	10.8 ±0.05 (27.3)	16.3 ±0.04 (91.7)	7.8 ±0.05 (-8.4)	6.0 ±0.05 (-29.0)	15.0 ±0.04 (75.8)	8.7 ±0.06 (2.8)	8.7 ±0.06 (2.0)
OpenLLM-BBH	8.4 ±0.05 (0.0)	18.9 ±0.06 (125.1)	22.9 ±0.05 (172.0)	7.9 ±0.05 (-5.5)	17.1 ±0.05 (103.1)	20.3 ±0.05 (142.0)	10.7 ±0.06 (27.3)	8.6 ±0.04 (2.8)	18.1 ±0.05 (115.8)	7.0 ±0.05 (-17.1)	7.1 ±0.05 (-15.9)
OpenLLM-GPQA	8.4 ±0.05 (0.0)	7.2 ±0.02 (-13.8)	7.5 ±0.02 (-10.2)	8.1 ±0.05 (-3.0)	6.8 ±0.02 (-18.8)	7.0 ±0.02 (-16.7)	7.2 ±0.05 (-14.7)	8.4 ±0.05 (-0.3)	7.5 ±0.03 (-10.8)	6.4 ±0.04 (-24.0)	6.4 ±0.04 (-23.4)
OpenLLM-MUSR	8.7 ±0.06 (0.0)	7.0 ±0.02 (-19.9)	6.8 ±0.02 (-21.6)	6.4 ±0.04 (-26.2)	6.2 ±0.02 (-28.6)	6.8 ±0.02 (-21.7)	5.5 ±0.04 (-37.1)	6.6 ±0.05 (-24.3)	6.7 ±0.02 (-23.3)	3.7 ±0.02 (-56.9)	3.7 ±0.02 (-57.6)
ImageNet	7.2 ±0.1 (0.0)	6.6 ±0.04 (-7.7)	9.9 ±0.02 (37.8)	5.0 ±0.07 (-30.3)	6.1 ±0.04 (-14.6)	8.3 ±0.02 (15.1)	4.8 ±0.05 (-32.8)	11.3 ±0.16 (56.6)	7.5 ±0.03 (4.9)	8.9 ±0.08 (23.6)	8.7 ±0.08 (21.5)
Average	7.4 ±0.02 (0.0)	15.8 ±0.03 (122.9)	17.6 ±0.03 (147.1)	6.5 ±0.02 (-10.9)	14.4 ±0.03 (102.6)	19.3 ±0.03 (178.9)	9.0 ±0.02 (25.9)	6.7 ±0.02 (-8.7)	14.8 ±0.02 (115.3)	8.2 ±0.02 (17.1)	7.8 ±0.02 (10.1)

Figure 12: The estimation gaps (↓) for target models (calculated as equation 1) under extrapolation model split, where source models are the lowest-performing 50%, and target models are the top 30% based on average performance over the full benchmark. Each target model can only be evaluated on $n = 20$ data points. We also report ± the standard error of the mean and the estimation gap reduction (↓) over Random-Sampling in parentheses. A negative reduction implies that the method achieves a lower estimation gap than Random-Sampling. % is omitted. Best viewed in color.



	Random-Sampling	Random-Sampling-Learn	PCA	AIPW	Random-Search-Learn	Lasso	Double-Optimize	Anchor-Points-Weighted	Anchor-Points-Predictor	P-IRT	GP-IRT
HELM-OpenbookQA	2.3 ±0.04 (0.0)	7.6 ±0.05 (227.3)	4.8 ±0.05 (104.5)	2.3 ±0.04 (-2.6)	7.4 ±0.05 (216.3)	9.0 ±0.03 (287.9)	6.0 ±0.05 (159.1)	1.6 ±0.02 (-31.8)	10.9 ±0.03 (367.6)	4.6 ±0.06 (99.1)	2.9 ±0.04 (22.6)
HELM-GSM8K	3.5 ±0.06 (0.0)	10.5 ±0.08 (203.6)	14.1 ±0.1 (307.3)	3.3 ±0.05 (-4.5)	9.2 ±0.09 (164.9)	22.0 ±0.08 (532.5)	8.0 ±0.09 (129.1)	4.2 ±0.04 (22.1)	16.7 ±0.05 (382.0)	5.3 ±0.08 (52.0)	4.1 ±0.06 (18.1)
HELM-LegalBench	5.4 ±0.09 (0.0)	7.6 ±0.09 (41.6)	16.8 ±0.07 (211.0)	4.1 ±0.07 (-23.2)	6.1 ±0.08 (12.9)	9.0 ±0.07 (67.7)	4.5 ±0.07 (-15.8)	3.8 ±0.06 (-30.3)	7.8 ±0.07 (45.4)	3.7 ±0.06 (-32.0)	3.5 ±0.06 (-34.2)
HELM-Math	3.7 ±0.06 (0.0)	12.8 ±0.1 (241.8)	20.2 ±0.12 (440.7)	3.7 ±0.06 (0.1)	10.5 ±0.1 (180.5)	22.9 ±0.09 (513.5)	9.6 ±0.1 (157.7)	5.3 ±0.06 (40.8)	19.8 ±0.08 (432.0)	4.4 ±0.07 (17.3)	3.6 ±0.06 (-2.4)
HELM-MedQA	4.7 ±0.07 (0.0)	18.1 ±0.13 (282.9)	13.6 ±0.12 (187.9)	4.2 ±0.07 (-11.6)	17.3 ±0.14 (266.4)	18.8 ±0.11 (297.6)	8.8 ±0.11 (87.4)	3.2 ±0.05 (-32.6)	17.9 ±0.08 (278.5)	5.2 ±0.08 (10.8)	5.2 ±0.07 (10.1)
HELM-MMLU	4.8 ±0.08 (0.0)	11.8 ±0.09 (147.5)	11.1 ±0.09 (132.1)	3.8 ±0.06 (-21.5)	11.1 ±0.09 (132.4)	12.0 ±0.08 (150.6)	6.2 ±0.08 (29.9)	2.4 ±0.04 (-50.9)	12.0 ±0.05 (150.5)	4.6 ±0.07 (-4.4)	4.4 ±0.06 (-9.0)
GLUE-MRPC	4.3 ±0.07 (0.0)	23.6 ±0.2 (450.6)	22.7 ±0.18 (428.7)	3.1 ±0.05 (-28.3)	21.0 ±0.18 (388.5)	27.5 ±0.16 (539.7)	17.4 ±0.19 (304.2)	3.0 ±0.05 (-30.9)	25.5 ±0.21 (493.1)	4.4 ±0.08 (2.0)	3.5 ±0.06 (-17.9)
GLUE-RTE	4.9 ±0.08 (0.0)	12.9 ±0.18 (163.1)	11.2 ±0.16 (127.4)	4.3 ±0.07 (-13.3)	12.1 ±0.18 (145.7)	12.3 ±0.18 (150.9)	6.6 ±0.09 (34.5)	4.6 ±0.08 (-5.9)	13.1 ±0.2 (166.3)	6.3 ±0.1 (28.1)	5.6 ±0.09 (14.5)
GLUE-SST-2	2.9 ±0.04 (0.0)	14.8 ±0.07 (405.1)	10.3 ±0.04 (252.0)	2.9 ±0.04 (-2.5)	14.5 ±0.06 (396.0)	23.9 ±0.04 (717.3)	5.0 ±0.07 (72.2)	2.9 ±0.04 (-0.5)	17.4 ±0.06 (494.1)	6.3 ±0.08 (114.4)	6.0 ±0.07 (104.6)
GLUE-MNLI	5.1 ±0.08 (0.0)	28.5 ±0.36 (462.2)	29.4 ±0.36 (480.9)	4.5 ±0.07 (-10.3)	28.2 ±0.36 (455.7)	28.9 ±0.36 (471.1)	5.3 ±0.08 (4.4)	4.7 ±0.07 (-7.8)	4.4 ±0.07 (-13.4)	5.0 ±0.07 (-1.2)	5.1 ±0.07 (0.5)
GLUE-QNLI	5.5 ±0.08 (0.0)	13.9 ±0.13 (153.3)	13.7 ±0.13 (149.3)	5.1 ±0.08 (-6.6)	13.8 ±0.13 (150.8)	14.1 ±0.13 (156.9)	5.4 ±0.09 (-1.5)	5.7 ±0.09 (4.5)	13.8 ±0.13 (151.3)	6.4 ±0.09 (17.0)	6.6 ±0.09 (19.2)
OpenLLM-IFEval	5.1 ±0.03 (0.0)	23.0 ±0.08 (350.5)	25.6 ±0.1 (400.2)	4.8 ±0.03 (-6.4)	22.7 ±0.09 (343.9)	33.2 ±0.1 (550.1)	24.8 ±0.09 (385.3)	7.0 ±0.04 (36.8)	24.1 ±0.05 (372.0)	5.2 ±0.03 (0.8)	5.6 ±0.04 (9.5)
OpenLLM-Math	4.0 ±0.03 (0.0)	11.4 ±0.06 (188.3)	12.1 ±0.07 (205.0)	3.7 ±0.03 (-5.2)	10.5 ±0.06 (166.1)	10.4 ±0.06 (162.9)	10.0 ±0.06 (153.6)	15.7 ±0.18 (295.8)	7.4 ±0.04 (86.0)	4.4 ±0.03 (11.3)	3.4 ±0.03 (-13.5)
OpenLLM-MMLU-Pro	5.7 ±0.04 (0.0)	16.2 ±0.05 (182.0)	23.4 ±0.05 (308.1)	4.9 ±0.03 (-14.1)	16.5 ±0.05 (186.8)	20.1 ±0.05 (249.5)	8.1 ±0.04 (42.0)	8.5 ±0.04 (48.9)	18.2 ±0.04 (217.6)	4.7 ±0.03 (-18.3)	4.7 ±0.03 (-17.6)
OpenLLM-ARC	5.2 ±0.05 (0.0)	10.3 ±0.04 (98.3)	10.2 ±0.05 (95.9)	4.2 ±0.04 (-19.3)	9.8 ±0.04 (88.9)	13.5 ±0.04 (158.5)	7.9 ±0.04 (52.1)	4.7 ±0.05 (-10.2)	13.2 ±0.03 (154.5)	4.3 ±0.04 (-17.7)	4.3 ±0.04 (-17.3)
OpenLLM-BBH	5.5 ±0.04 (0.0)	16.3 ±0.05 (197.7)	20.7 ±0.05 (278.3)	5.2 ±0.03 (-4.5)	15.5 ±0.05 (182.8)	18.1 ±0.05 (231.1)	9.1 ±0.04 (66.9)	8.4 ±0.04 (53.4)	12.6 ±0.03 (129.9)	10.0 ±0.05 (81.9)	9.0 ±0.05 (64.7)
OpenLLM-GPQA	5.1 ±0.03 (0.0)	6.8 ±0.02 (32.0)	7.3 ±0.02 (41.4)	5.0 ±0.03 (-2.0)	6.5 ±0.02 (25.7)	6.5 ±0.02 (26.1)	4.8 ±0.03 (-7.1)	4.7 ±0.03 (-8.8)	6.0 ±0.03 (16.8)	4.4 ±0.03 (-13.9)	4.6 ±0.03 (-10.7)
OpenLLM-MUSR	5.5 ±0.04 (0.0)	6.1 ±0.02 (10.0)	6.1 ±0.02 (10.1)	4.0 ±0.03 (-28.2)	5.6 ±0.02 (0.7)	5.7 ±0.02 (2.5)	4.2 ±0.03 (-24.7)	4.3 ±0.03 (-22.5)	7.0 ±0.02 (26.0)	3.8 ±0.02 (-31.7)	3.6 ±0.02 (-35.3)
ImageNet	4.4 ±0.06 (0.0)	5.5 ±0.04 (24.7)	9.7 ±0.02 (120.3)	2.8 ±0.04 (-35.5)	5.2 ±0.04 (19.4)	7.4 ±0.02 (68.7)	4.6 ±0.04 (4.3)	11.2 ±0.14 (155.5)	7.1 ±0.02 (61.9)	4.0 ±0.04 (-9.5)	4.0 ±0.04 (-8.9)
Average	4.6 ±0.01 (0.0)	13.6 ±0.03 (203.3)	14.9 ±0.03 (225.3)	4.0 ±0.01 (-12.6)	12.8 ±0.03 (185.5)	16.6 ±0.03 (280.8)	8.2 ±0.02 (86.0)	5.6 ±0.02 (22.4)	13.4 ±0.02 (211.2)	5.1 ±0.01 (16.1)	4.7 ±0.01 (5.1)

Figure 13: The estimation gaps (↓) for target models (calculated as equation 1) under extrapolation model split, where source models are the lowest-performing 50%, and target models are the top 30% based on average performance over the full benchmark. Each target model can only be evaluated on $n = 50$ data points. We also report ± the standard error of the mean and the estimation gap reduction (↓) over RANDOM-SAMPLING in parentheses. A negative reduction implies that the method achieves a lower estimation gap than RANDOM-SAMPLING. % is omitted. Best viewed in color.



	Random-Sampling	Random-Sampling-Learn	PCA	AIPW	Random-Search-Learn	Lasso	Double-Optimize	Anchor-Points-Weighted	Anchor-Points-Predictor	P-IRT	GP-IRT
HELM-OpenbookQA	1.6 ±0.03 (0.0)	6.4 ±0.04 (290.2)	3.0 ±0.04 (82.5)	1.6 ±0.02 (-5.2)	6.1 ±0.04 (272.3)	8.7 ±0.04 (431.9)	5.4 ±0.05 (229.1)	1.3 ±0.02 (-21.2)	7.5 ±0.03 (356.3)	1.8 ±0.03 (8.6)	1.2 ±0.02 (-29.2)
HELM-GSM8K	2.3 ±0.04 (0.0)	8.1 ±0.07 (251.1)	9.8 ±0.07 (324.9)	2.2 ±0.03 (-5.9)	7.9 ±0.07 (241.0)	20.2 ±0.07 (775.9)	8.4 ±0.09 (262.6)	3.0 ±0.03 (28.7)	15.3 ±0.04 (563.8)	2.4 ±0.04 (3.7)	2.1 ±0.03 (-8.5)
HELM-LegalBench	3.6 ±0.06 (0.0)	5.5 ±0.07 (53.3)	15.3 ±0.06 (322.8)	2.8 ±0.05 (-22.8)	5.2 ±0.06 (43.1)	9.1 ±0.07 (151.7)	4.4 ±0.06 (22.6)	4.1 ±0.05 (13.0)	5.3 ±0.05 (46.4)	2.5 ±0.04 (-29.8)	2.4 ±0.04 (-34.8)
HELM-Math	2.5 ±0.04 (0.0)	10.0 ±0.08 (293.5)	11.8 ±0.08 (364.4)	2.5 ±0.04 (-2.4)	9.4 ±0.07 (271.2)	22.0 ±0.08 (770.4)	9.3 ±0.11 (266.1)	3.9 ±0.05 (54.7)	17.4 ±0.07 (586.6)	2.3 ±0.04 (-8.1)	2.5 ±0.04 (0.1)
HELM-MedQA	3.1 ±0.05 (0.0)	16.6 ±0.12 (433.6)	10.3 ±0.1 (230.2)	2.9 ±0.05 (-7.8)	15.8 ±0.12 (409.2)	18.5 ±0.11 (494.6)	8.3 ±0.11 (168.0)	2.6 ±0.03 (-16.8)	15.7 ±0.07 (405.8)	3.7 ±0.05 (19.7)	3.6 ±0.05 (17.0)
HELM-MMLU	3.1 ±0.05 (0.0)	10.9 ±0.08 (258.4)	9.2 ±0.07 (202.7)	2.6 ±0.04 (-13.6)	10.6 ±0.08 (247.2)	12.0 ±0.08 (293.5)	6.0 ±0.07 (95.3)	2.2 ±0.03 (-28.0)	11.0 ±0.05 (258.8)	4.1 ±0.06 (35.3)	3.3 ±0.05 (6.9)
GLUE-MRPC	2.8 ±0.04 (0.0)	20.4 ±0.18 (632.8)	16.2 ±0.13 (480.3)	2.1 ±0.03 (-23.2)	19.6 ±0.17 (601.4)	27.5 ±0.16 (884.2)	17.7 ±0.15 (533.0)	1.6 ±0.03 (-41.0)	25.5 ±0.22 (813.4)	2.6 ±0.04 (-7.0)	2.0 ±0.04 (-29.6)
GLUE-RTE	3.0 ±0.05 (0.0)	12.7 ±0.18 (319.9)	8.3 ±0.12 (174.0)	2.8 ±0.04 (-8.4)	11.8 ±0.18 (290.4)	12.3 ±0.18 (306.9)	6.7 ±0.09 (119.9)	4.4 ±0.07 (46.1)	13.5 ±0.19 (345.8)	4.8 ±0.08 (59.3)	4.1 ±0.06 (33.6)
GLUE-SST-2	2.1 ±0.03 (0.0)	13.4 ±0.05 (546.4)	6.0 ±0.04 (186.7)	2.0 ±0.03 (-2.1)	13.8 ±0.05 (562.8)	19.2 ±0.03 (824.2)	3.0 ±0.04 (45.1)	2.3 ±0.03 (12.3)	16.1 ±0.05 (675.2)	2.9 ±0.04 (39.7)	2.7 ±0.04 (32.3)
GLUE-MNLI	3.4 ±0.05 (0.0)	28.3 ±0.36 (735.6)	29.2 ±0.36 (760.5)	3.1 ±0.05 (-8.6)	27.9 ±0.35 (721.4)	28.9 ±0.36 (753.4)	4.6 ±0.07 (36.1)	3.6 ±0.05 (4.9)	3.7 ±0.06 (9.5)	3.2 ±0.05 (-5.1)	3.3 ±0.05 (-3.5)
GLUE-QNLI	3.9 ±0.06 (0.0)	13.8 ±0.13 (257.8)	13.5 ±0.13 (248.3)	3.6 ±0.05 (-6.6)	13.8 ±0.13 (256.5)	14.1 ±0.13 (265.5)	3.8 ±0.06 (-1.1)	4.5 ±0.07 (16.8)	13.9 ±0.13 (259.4)	5.8 ±0.08 (50.2)	5.8 ±0.08 (50.6)
OpenLLM-IFEval	3.3 ±0.02 (0.0)	19.3 ±0.07 (478.1)	22.0 ±0.09 (558.1)	3.4 ±0.02 (0.3)	19.6 ±0.07 (486.2)	27.4 ±0.09 (718.0)	27.2 ±0.09 (712.8)	8.1 ±0.04 (140.9)	22.3 ±0.06 (566.5)	4.7 ±0.03 (40.4)	4.6 ±0.03 (38.2)
OpenLLM-Math	2.5 ±0.02 (0.0)	10.2 ±0.05 (300.6)	10.8 ±0.07 (326.9)	2.6 ±0.02 (1.1)	9.7 ±0.05 (282.1)	10.4 ±0.06 (311.5)	10.8 ±0.06 (325.1)	9.7 ±0.13 (283.7)	7.7 ±0.03 (205.4)	2.9 ±0.02 (15.5)	2.4 ±0.02 (-3.6)
OpenLLM-MMLU-Pro	4.1 ±0.03 (0.0)	14.7 ±0.05 (256.2)	20.2 ±0.05 (390.7)	3.7 ±0.02 (-9.4)	15.0 ±0.05 (263.2)	17.9 ±0.05 (335.7)	8.0 ±0.04 (95.1)	6.8 ±0.03 (65.3)	16.9 ±0.04 (310.0)	3.3 ±0.02 (-19.6)	3.3 ±0.02 (-19.6)
OpenLLM-ARC	3.8 ±0.03 (0.0)	8.7 ±0.04 (128.3)	7.5 ±0.04 (95.9)	2.9 ±0.03 (-23.3)	8.5 ±0.04 (123.7)	10.3 ±0.04 (170.0)	8.3 ±0.05 (117.3)	8.0 ±0.03 (108.9)	12.9 ±0.03 (239.1)	3.0 ±0.03 (-21.8)	3.1 ±0.03 (-18.4)
OpenLLM-BBH	4.1 ±0.03 (0.0)	14.2 ±0.05 (249.2)	18.1 ±0.05 (343.6)	3.7 ±0.02 (-8.1)	13.9 ±0.05 (241.0)	16.0 ±0.05 (291.9)	8.5 ±0.04 (108.6)	4.8 ±0.03 (17.4)	10.9 ±0.03 (168.4)	10.3 ±0.05 (152.6)	8.2 ±0.04 (101.4)
OpenLLM-GPQA	3.5 ±0.02 (0.0)	6.3 ±0.02 (78.4)	6.8 ±0.02 (92.8)	3.7 ±0.02 (3.8)	6.1 ±0.02 (72.3)	5.9 ±0.02 (67.6)	3.3 ±0.02 (-5.9)	3.9 ±0.02 (9.5)	6.0 ±0.02 (70.9)	3.3 ±0.02 (-8.0)	3.5 ±0.02 (-0.7)
OpenLLM-MUSR	3.8 ±0.02 (0.0)	5.2 ±0.02 (36.6)	5.4 ±0.02 (42.9)	2.8 ±0.02 (-25.4)	4.8 ±0.02 (26.5)	4.4 ±0.02 (15.7)	4.1 ±0.02 (8.4)	3.4 ±0.02 (-11.6)	6.3 ±0.02 (66.8)	4.3 ±0.02 (14.0)	3.3 ±0.02 (-12.7)
ImageNet	3.0 ±0.04 (0.0)	4.8 ±0.04 (63.1)	9.3 ±0.02 (212.5)	1.9 ±0.03 (-34.8)	4.4 ±0.03 (47.0)	5.9 ±0.02 (97.2)	4.8 ±0.04 (60.3)	11.0 ±0.16 (271.4)	6.9 ±0.02 (131.0)	2.3 ±0.03 (-23.9)	2.2 ±0.03 (-24.9)
Average	3.1 ±0.01 (0.0)	12.1 ±0.03 (298.1)	12.2 ±0.03 (286.4)	2.8 ±0.01 (-10.7)	11.8 ±0.03 (287.3)	15.3 ±0.03 (418.9)	8.0 ±0.02 (168.4)	4.7 ±0.01 (50.3)	12.4 ±0.02 (320.0)	3.7 ±0.01 (16.6)	3.4 ±0.01 (5.0)

Figure 14: The estimation gaps (↓) for target models (calculated as equation 1) under extrapolation model split, where source models are the lowest-performing 50%, and target models are the top 30% based on average performance over the full benchmark. Each target model can only be evaluated on $n = 100$ data points. We also report ± the standard error of the mean and the estimation gap reduction (↓) over Random-Sampling in parentheses. A negative reduction implies that the method achieves a lower estimation gap than Random-Sampling. % is omitted. Best viewed in color.

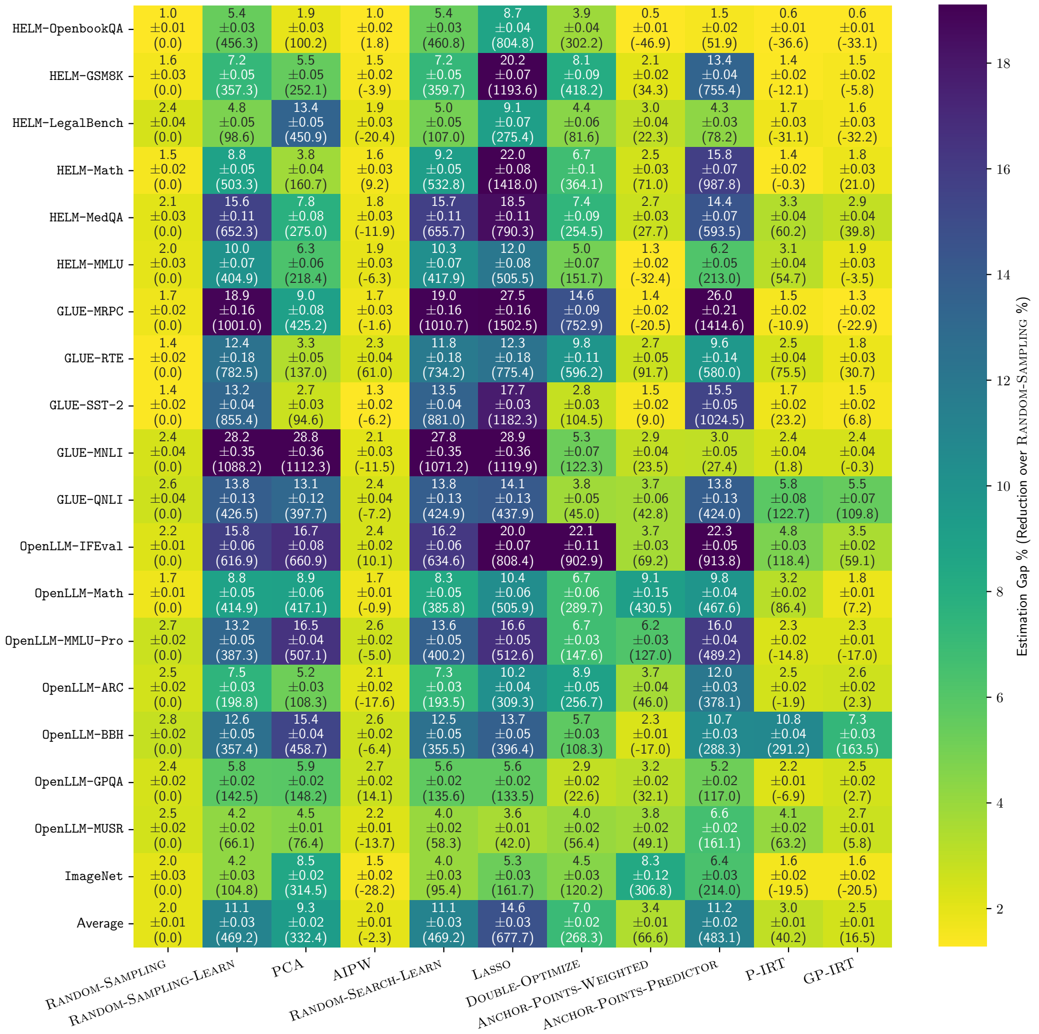


Figure 15: The estimation gaps (↓) for target models (calculated as equation 1) under extrapolation model split, where source models are the lowest-performing 50%, and target models are the top 30% based on average performance over the full benchmark. Each target model can only be evaluated on $n = 200$ data points. We also report ± the standard error of the mean and the estimation gap reduction (↓) over RANDOM-SAMPLING in parentheses. A negative reduction implies that the method achieves a lower estimation gap than RANDOM-SAMPLING. % is omitted. Best viewed in color.

Table 2: Training and inference time of each method on `ImageNet` with $N = 50000$ data points and $|\mathcal{F}| = 110$ models. Training is based on 83 source models, and inference is on 27 target models.

	Training Time (s)	Inference Time (s)
Random-Sampling	0.00	0.00
Random-Sampling-Learn	0.02	0.00
PCA	0.59	19.20
AIPW	0.00	0.27
Random-Search-Learn	81.02	0.00
Lasso	105.58	0.01
Double-Optimize	4.88	0.00
Anchor-Points-Weighted	84.26	0.00
Anchor-Points-Predictor	197.71	0.26
P-IRT	585.72	0.90
GP-IRT	1750.20	0.89

Table 3: Average estimation gap between the predicted rankings based on the coreset and the actual rankings based on the full benchmark, measured by Kendall's τ (↑). The results are averaged over all benchmarks.

	Interpolation					Extrapolation				
	$n = 10$	$n = 20$	$n = 50$	$n = 100$	$n = 200$	$n = 10$	$n = 20$	$n = 50$	$n = 100$	$n = 200$
Random-Sampling	0.52	0.61	0.70	0.78	0.84	0.36	0.43	0.53	0.63	0.73
Random-Sampling-Learn	0.57	0.66	0.75	0.81	0.86	0.07	0.12	0.18	0.27	0.36
PCA	0.55	0.63	0.72	0.78	0.83	0.04	0.10	0.21	0.40	0.57
AIPW	0.52	0.62	0.72	0.79	0.84	0.33	0.40	0.51	0.61	0.70
Random-Search-Learn	0.66	0.70	0.76	0.82	0.86	0.13	0.13	0.20	0.29	0.38
Lasso	0.68	0.71	0.77	0.81	0.82	0.05	0.06	0.12	0.19	0.22
Double-Optimize	0.58	0.66	0.76	0.81	0.84	0.31	0.36	0.44	0.50	0.58
Anchor-Points-Weighted	0.65	0.70	0.76	0.81	0.85	0.37	0.43	0.50	0.60	0.69
Anchor-Points-Predictor	0.67	0.72	0.77	0.80	0.80	0.21	0.25	0.32	0.38	0.44
P-IRT	0.52	0.58	0.71	0.80	0.87	0.28	0.31	0.42	0.56	0.69
GP-IRT	0.53	0.59	0.72	0.80	0.86	0.28	0.33	0.45	0.59	0.71

D.3 Running time

While some of the benchmark prediction methods could potentially benefit from the use of GPUs, we opted to run all methods without them, as they are sufficiently fast on standard hardware. Table 2 presents the training and inference times for each method on `ImageNet`. Among the models, GP-IRT is the slowest during training because it involves fitting a large Item Response Theory (IRT) model. During inference, PCA is the slowest, as it requires multiple imputations of the entire matrix. Although AIPW needs training a separate regressor for each target model during inference, the regressor is small, making the inference process remain efficient.

D.4 Ranking Preservation

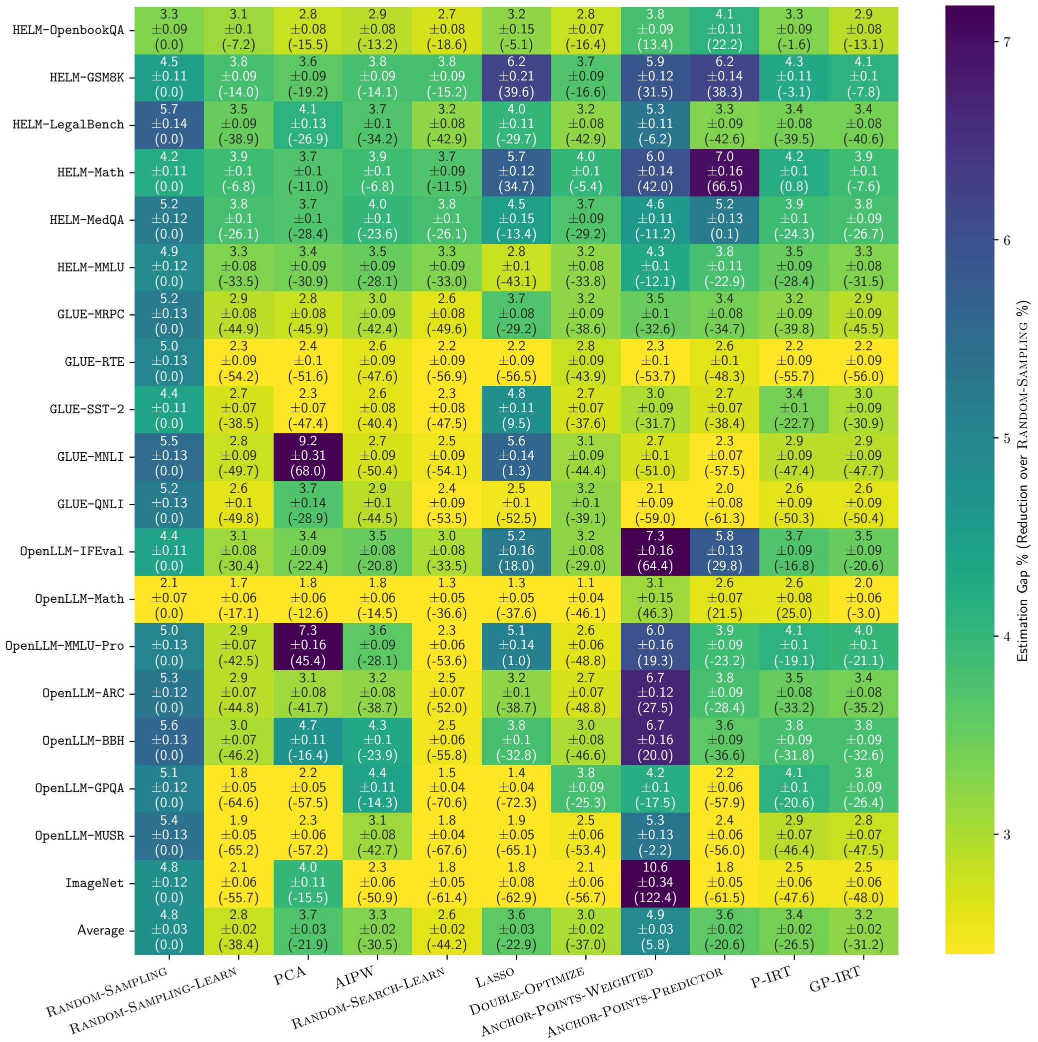
We further compare the predicted rankings of target models with the actual rankings based on the full benchmark using Kendall's τ. Specifically, we calculate Kendall's τ for each random trial and average the results over 100 trials. Our conclusions mostly remain unchanged, with almost all benchmark prediction methods outperforming Random Sampling under interpolation, while none can surpass Random-Sampling under extrapolation.

D.5 Case Studies

We further investigate two additional experimental settings that deviate from the primary setting in the main paper.

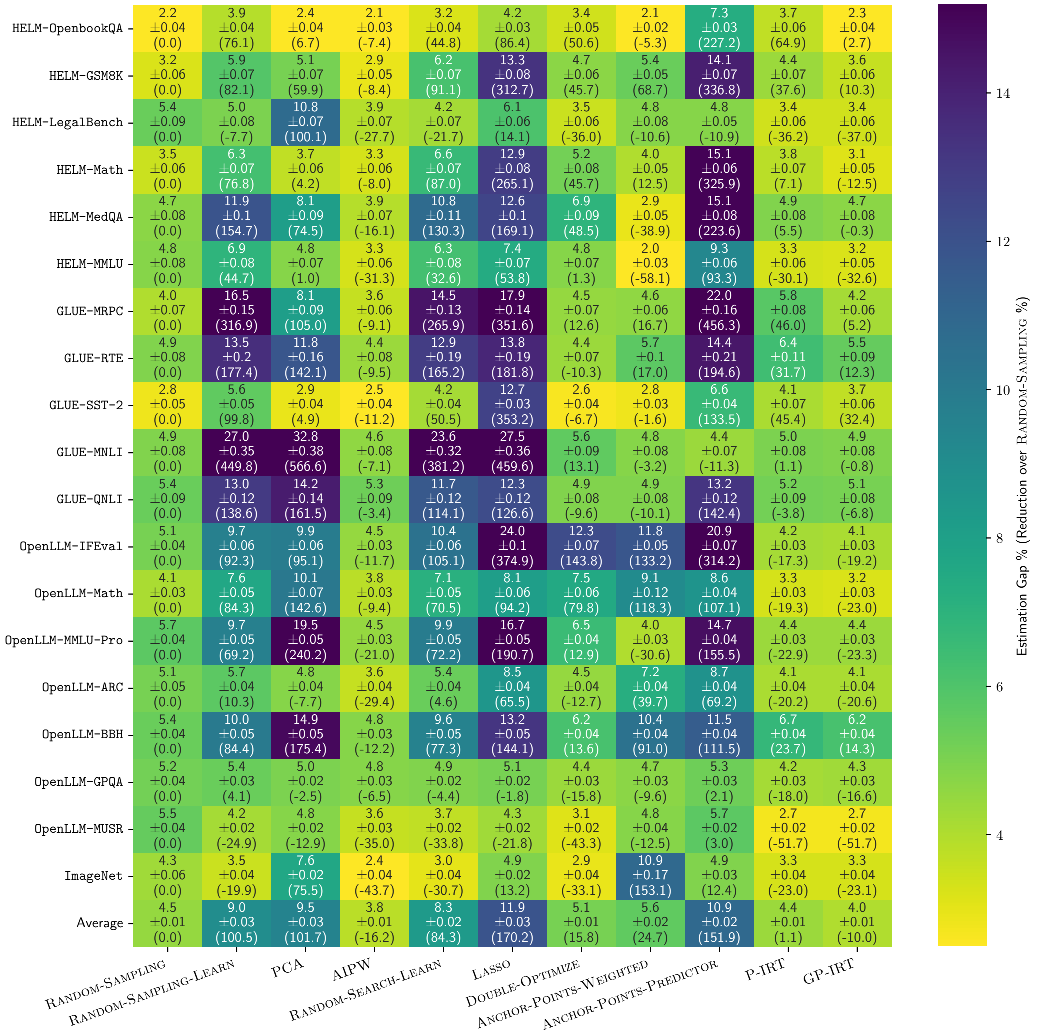
Fewer source models under interpolation. Different from the previous interpolation setting that utilized 75% of models as source models, we now use only 10 models as source models for each benchmark and use the rest as target models. All other settings remain unchanged. This setting allows us to assess the effectiveness of benchmark prediction when "training data" from source models is more limited. Results are shown in Figure 16. Consistent with the findings in the paper, most methods still outperform RANDOM-SAMPLING, while RANDOM-SEARCH-LEARN and RANDOM-SAMPLING-LEARN remain to be the best-performing methods.

Near extrapolation. We modify the previous extrapolation setting, which used the lowest-performing 50% of models as source models and the top 30% as target models. In this new setting, we designate the top 25% of models as target models and utilize all remaining models as source models. All other settings remain unchanged. This setup enables us to examine whether benchmark prediction methods demonstrate improved performance when the distribution gap between source and target models is reduced. Results are shown in Figure 17. Consistent with the findings in the paper, most methods fail to consistently outperform RANDOM-SAMPLING, except for AIPW.



	Random-Sampling	Random-Sampling-Learn	PCA	AIPW	Random-Search-Learn	Lasso	Double-Optimize	Anchor-Points-Weighted	Anchor-Points-Predictor	P-IRT	GP-IRT
HELM-OpenbookQA	3.3 ±0.09 (0.0)	3.1 ±0.1 (-7.2)	2.8 ±0.08 (-15.5)	2.9 ±0.08 (-13.2)	2.7 ±0.08 (-18.6)	3.2 ±0.15 (-5.1)	2.8 ±0.07 (-16.4)	3.8 ±0.09 (13.4)	4.1 ±0.11 (22.2)	3.3 ±0.09 (-1.6)	2.9 ±0.08 (-13.1)
HELM-GSM8K	4.5 ±0.11 (0.0)	3.8 ±0.09 (-14.0)	3.6 ±0.09 (-19.2)	3.8 ±0.09 (-14.1)	3.8 ±0.09 (-15.2)	6.2 ±0.21 (39.6)	3.7 ±0.09 (-16.6)	5.9 ±0.12 (31.5)	6.2 ±0.14 (38.3)	4.3 ±0.11 (-3.1)	4.1 ±0.1 (-7.8)
HELM-LegalBench	5.7 ±0.14 (0.0)	3.5 ±0.09 (-38.9)	4.1 ±0.13 (-26.9)	3.7 ±0.1 (-34.2)	3.2 ±0.08 (-42.9)	4.0 ±0.11 (-29.7)	3.2 ±0.08 (-42.9)	5.3 ±0.11 (-6.2)	3.3 ±0.09 (-42.6)	3.4 ±0.08 (-39.5)	3.4 ±0.08 (-40.6)
HELM-Math	4.2 ±0.11 (0.0)	3.9 ±0.1 (-6.8)	3.7 ±0.1 (-11.0)	3.9 ±0.1 (-6.8)	3.7 ±0.09 (-11.5)	5.7 ±0.12 (34.7)	4.0 ±0.1 (-5.4)	6.0 ±0.14 (42.0)	7.0 ±0.16 (66.5)	4.2 ±0.1 (0.8)	3.9 ±0.1 (-7.6)
HELM-MedQA	5.2 ±0.12 (0.0)	3.8 ±0.1 (-26.1)	3.7 ±0.1 (-28.4)	4.0 ±0.1 (-23.6)	3.8 ±0.1 (-26.1)	4.5 ±0.15 (-13.4)	3.7 ±0.09 (-29.2)	4.6 ±0.11 (-11.2)	5.2 ±0.13 (0.1)	3.9 ±0.1 (-24.3)	3.8 ±0.09 (-26.7)
HELM-MMLU	4.9 ±0.12 (0.0)	3.3 ±0.08 (-33.5)	3.4 ±0.09 (-30.9)	3.5 ±0.09 (-28.1)	3.3 ±0.09 (-33.0)	2.8 ±0.1 (-43.1)	3.2 ±0.08 (-33.8)	4.3 ±0.1 (-12.1)	3.8 ±0.11 (-22.9)	3.5 ±0.09 (-28.4)	3.3 ±0.08 (-31.5)
GLUE-MRPC	5.2 ±0.13 (0.0)	2.9 ±0.08 (-44.9)	2.8 ±0.08 (-45.9)	3.0 ±0.09 (-42.4)	2.6 ±0.08 (-49.6)	3.7 ±0.08 (-29.2)	3.2 ±0.09 (-38.6)	3.5 ±0.1 (-32.6)	3.4 ±0.08 (-34.7)	3.2 ±0.09 (-39.8)	2.9 ±0.09 (-45.5)
GLUE-RTE	5.0 ±0.13 (0.0)	2.3 ±0.09 (-54.2)	2.4 ±0.1 (-51.6)	2.6 ±0.09 (-47.6)	2.2 ±0.09 (-56.9)	2.2 ±0.09 (-56.5)	2.8 ±0.09 (-43.9)	2.3 ±0.1 (-53.7)	2.6 ±0.1 (-48.3)	2.2 ±0.09 (-55.7)	2.2 ±0.09 (-56.0)
GLUE-SST-2	4.4 ±0.11 (0.0)	2.7 ±0.07 (-38.5)	2.3 ±0.07 (-47.4)	2.6 ±0.08 (-40.4)	2.3 ±0.08 (-47.5)	4.8 ±0.11 (9.5)	2.7 ±0.07 (-37.6)	3.0 ±0.09 (-31.7)	2.7 ±0.07 (-38.4)	3.4 ±0.1 (-22.7)	3.0 ±0.09 (-30.9)
GLUE-MNLI	5.5 ±0.13 (0.0)	2.8 ±0.09 (-49.7)	9.2 ±0.31 (68.0)	2.7 ±0.09 (-50.4)	2.5 ±0.09 (-54.1)	5.6 ±0.14 (1.3)	3.1 ±0.09 (-44.4)	2.7 ±0.1 (-51.0)	2.3 ±0.07 (-57.5)	2.9 ±0.09 (-47.4)	2.9 ±0.09 (-47.7)
GLUE-QNLI	5.2 ±0.13 (0.0)	2.6 ±0.1 (-49.8)	3.7 ±0.14 (-28.9)	2.9 ±0.1 (-44.5)	2.4 ±0.09 (-53.5)	2.5 ±0.1 (-52.5)	3.2 ±0.1 (-39.1)	2.1 ±0.09 (-59.0)	2.0 ±0.08 (-61.3)	2.6 ±0.09 (-50.3)	2.6 ±0.09 (-50.4)
OpenLLM-IFEval	4.4 ±0.11 (0.0)	3.1 ±0.08 (-30.4)	3.4 ±0.09 (-22.4)	3.5 ±0.08 (-20.8)	3.0 ±0.08 (-33.5)	5.2 ±0.16 (18.0)	3.2 ±0.08 (-29.0)	7.3 ±0.16 (64.4)	5.8 ±0.13 (29.8)	3.7 ±0.09 (-16.8)	3.5 ±0.09 (-20.6)
OpenLLM-Math	2.1 ±0.07 (0.0)	1.7 ±0.06 (-17.1)	1.8 ±0.06 (-12.6)	1.8 ±0.06 (-14.5)	1.3 ±0.05 (-36.6)	1.3 ±0.05 (-37.6)	1.1 ±0.04 (-46.1)	3.1 ±0.15 (46.3)	2.6 ±0.07 (21.5)	2.6 ±0.08 (25.0)	2.0 ±0.06 (-3.0)
OpenLLM-MMLU-Pro	5.0 ±0.13 (0.0)	2.9 ±0.07 (-42.5)	7.3 ±0.16 (45.4)	3.6 ±0.09 (-28.1)	2.3 ±0.06 (-53.6)	5.1 ±0.14 (1.0)	2.6 ±0.06 (-48.8)	6.0 ±0.16 (19.3)	3.9 ±0.09 (-23.2)	4.1 ±0.1 (-19.1)	4.0 ±0.1 (-21.1)
OpenLLM-ARC	5.3 ±0.12 (0.0)	2.9 ±0.07 (-44.8)	3.1 ±0.08 (-41.7)	3.2 ±0.08 (-38.7)	2.5 ±0.07 (-52.0)	3.2 ±0.1 (-38.7)	2.7 ±0.07 (-48.8)	6.7 ±0.12 (27.5)	3.8 ±0.09 (-28.4)	3.5 ±0.08 (-33.2)	3.4 ±0.08 (-35.2)
OpenLLM-BBH	5.6 ±0.13 (0.0)	3.0 ±0.07 (-46.2)	4.7 ±0.11 (-16.4)	4.3 ±0.1 (-23.9)	2.5 ±0.06 (-55.8)	3.8 ±0.1 (-32.8)	3.0 ±0.08 (-46.6)	6.7 ±0.16 (20.0)	3.6 ±0.09 (-36.6)	3.8 ±0.09 (-31.8)	3.8 ±0.09 (-32.6)
OpenLLM-GPQA	5.1 ±0.12 (0.0)	1.8 ±0.05 (-64.6)	2.2 ±0.05 (-57.5)	4.4 ±0.11 (-14.3)	1.5 ±0.04 (-70.6)	1.4 ±0.04 (-72.3)	3.8 ±0.09 (-25.3)	4.2 ±0.1 (-17.5)	2.2 ±0.06 (-57.9)	4.1 ±0.1 (-20.6)	3.8 ±0.09 (-26.4)
OpenLLM-MUSR	5.4 ±0.13 (0.0)	1.9 ±0.05 (-65.2)	2.3 ±0.06 (-57.2)	3.1 ±0.08 (-42.7)	1.8 ±0.04 (-67.6)	1.9 ±0.05 (-65.1)	2.5 ±0.06 (-53.4)	5.3 ±0.13 (-2.2)	2.4 ±0.06 (-56.0)	2.9 ±0.07 (-46.4)	2.8 ±0.07 (-47.5)
ImageNet	4.8 ±0.12 (0.0)	2.1 ±0.06 (-55.7)	4.0 ±0.11 (-15.5)	2.3 ±0.06 (-50.9)	1.8 ±0.05 (-61.4)	1.8 ±0.08 (-62.9)	2.1 ±0.06 (-56.7)	10.6 ±0.34 (122.4)	1.8 ±0.05 (-61.5)	2.5 ±0.06 (-47.6)	2.5 ±0.06 (-48.0)
Average	4.8 ±0.03 (0.0)	2.8 ±0.02 (-38.4)	3.7 ±0.03 (-21.9)	3.3 ±0.02 (-30.5)	2.6 ±0.02 (-44.2)	3.6 ±0.03 (-22.9)	3.0 ±0.02 (-37.0)	4.9 ±0.03 (5.8)	3.6 ±0.02 (-20.6)	3.4 ±0.02 (-26.5)	3.2 ±0.02 (-31.2)

Figure 16: Experiment with fewer source models (randomly selected 10 models as source models) under the interpolation model split. We report the estimation gaps (↓) for target models (calculated as equation 1). We also report ± the standard error of the mean and the estimation gap reduction (↓) over RANDOM-SAMPLING in parentheses. A negative reduction implies that the method achieves a lower estimation gap than RANDOM-SAMPLING. % is omitted. Best viewed in color.



	Random-Sampling	Random-Sampling-Learn	PCA	AIPW	Random-Search-Learn	Lasso	Double-Optimize	Anchor-Points-Weighted	Anchor-Points-Predictor	P-IRT	GP-IRT
HELM-OpenbookQA	2.2 ±0.04 (0.0)	3.9 ±0.04 (76.1)	2.4 ±0.04 (6.7)	2.1 ±0.03 (-7.4)	3.2 ±0.04 (44.8)	4.2 ±0.03 (86.4)	3.4 ±0.05 (50.6)	2.1 ±0.02 (-5.3)	7.3 ±0.03 (227.2)	3.7 ±0.06 (64.9)	2.3 ±0.04 (2.7)
HELM-GSM8K	3.2 ±0.06 (0.0)	5.9 ±0.07 (82.1)	5.1 ±0.07 (59.9)	2.9 ±0.05 (-8.4)	6.2 ±0.07 (91.1)	13.3 ±0.08 (312.7)	4.7 ±0.06 (45.7)	5.4 ±0.05 (68.7)	14.1 ±0.07 (336.8)	4.4 ±0.07 (37.6)	3.6 ±0.06 (10.3)
HELM-LegalBench	5.4 ±0.09 (0.0)	5.0 ±0.08 (-7.7)	10.8 ±0.07 (100.1)	3.9 ±0.07 (-27.7)	4.2 ±0.07 (-21.7)	6.1 ±0.06 (14.1)	3.5 ±0.06 (-36.0)	4.8 ±0.08 (-10.6)	4.8 ±0.05 (-10.9)	3.4 ±0.06 (-36.2)	3.4 ±0.06 (-37.0)
HELM-Math	3.5 ±0.06 (0.0)	6.3 ±0.07 (76.8)	3.7 ±0.06 (4.2)	3.3 ±0.06 (-8.0)	6.6 ±0.07 (87.0)	12.9 ±0.08 (265.1)	5.2 ±0.08 (45.7)	4.0 ±0.05 (12.5)	15.1 ±0.06 (325.9)	3.8 ±0.07 (7.1)	3.1 ±0.05 (-12.5)
HELM-MedQA	4.7 ±0.08 (0.0)	11.9 ±0.1 (154.7)	8.1 ±0.09 (74.5)	3.9 ±0.07 (-16.1)	10.8 ±0.11 (130.3)	12.6 ±0.1 (169.1)	6.9 ±0.09 (48.5)	2.9 ±0.05 (-38.9)	15.1 ±0.08 (223.6)	4.9 ±0.08 (5.5)	4.7 ±0.08 (-0.3)
HELM-MMLU	4.8 ±0.08 (0.0)	6.9 ±0.08 (44.7)	4.8 ±0.07 (1.0)	3.3 ±0.06 (-31.3)	6.3 ±0.08 (32.6)	7.4 ±0.07 (53.8)	4.8 ±0.07 (1.3)	2.0 ±0.03 (-58.1)	9.3 ±0.06 (93.3)	3.3 ±0.06 (-30.1)	3.2 ±0.05 (-32.6)
GLUE-MRPC	4.0 ±0.07 (0.0)	16.5 ±0.15 (316.9)	8.1 ±0.09 (105.0)	3.6 ±0.06 (-9.1)	14.5 ±0.13 (265.9)	17.9 ±0.14 (351.6)	4.5 ±0.07 (12.6)	4.6 ±0.06 (16.7)	22.0 ±0.16 (456.3)	5.8 ±0.08 (46.0)	4.2 ±0.06 (5.2)
GLUE-RTE	4.9 ±0.08 (0.0)	13.5 ±0.2 (177.4)	11.8 ±0.16 (142.1)	4.4 ±0.08 (-9.5)	12.9 ±0.19 (165.2)	13.8 ±0.19 (181.8)	4.4 ±0.07 (-10.3)	5.7 ±0.1 (17.0)	14.4 ±0.21 (194.6)	6.4 ±0.11 (31.7)	5.5 ±0.09 (12.3)
GLUE-SST-2	2.8 ±0.05 (0.0)	5.6 ±0.05 (99.8)	2.9 ±0.04 (4.9)	2.5 ±0.04 (-11.2)	4.2 ±0.04 (50.5)	12.7 ±0.03 (353.2)	2.6 ±0.04 (-6.7)	2.8 ±0.03 (-1.6)	6.6 ±0.04 (133.5)	4.1 ±0.07 (45.4)	3.7 ±0.06 (32.4)
GLUE-MNLI	4.9 ±0.08 (0.0)	27.0 ±0.35 (449.8)	32.8 ±0.38 (566.6)	4.6 ±0.08 (-7.1)	23.6 ±0.32 (381.2)	27.5 ±0.36 (459.6)	5.6 ±0.09 (13.1)	4.8 ±0.08 (-3.2)	4.4 ±0.07 (-11.3)	5.0 ±0.08 (1.1)	4.9 ±0.08 (-0.8)
GLUE-QNLI	5.4 ±0.09 (0.0)	13.0 ±0.12 (138.6)	14.2 ±0.14 (161.5)	5.3 ±0.09 (-3.4)	11.7 ±0.12 (114.1)	12.3 ±0.12 (126.6)	4.9 ±0.08 (-9.6)	4.9 ±0.08 (-10.1)	13.2 ±0.12 (142.4)	5.2 ±0.09 (-3.8)	5.1 ±0.08 (-6.8)
OpenLLM-IFEval	5.1 ±0.04 (0.0)	9.7 ±0.06 (92.3)	9.9 ±0.06 (95.1)	4.5 ±0.03 (-11.7)	10.4 ±0.06 (105.1)	24.0 ±0.1 (374.9)	12.3 ±0.07 (143.8)	11.8 ±0.05 (133.2)	20.9 ±0.07 (314.2)	4.2 ±0.03 (-17.3)	4.1 ±0.03 (-19.2)
OpenLLM-Math	4.1 ±0.03 (0.0)	7.6 ±0.05 (84.3)	10.1 ±0.07 (142.6)	3.8 ±0.03 (-9.4)	7.1 ±0.05 (70.5)	8.1 ±0.06 (94.2)	7.5 ±0.06 (79.8)	9.1 ±0.12 (118.3)	8.6 ±0.04 (107.1)	3.3 ±0.03 (-19.3)	3.2 ±0.03 (-23.0)
OpenLLM-MMLU-Pro	5.7 ±0.04 (0.0)	9.7 ±0.05 (69.2)	19.5 ±0.05 (240.2)	4.5 ±0.03 (-21.0)	9.9 ±0.05 (72.2)	16.7 ±0.05 (190.7)	6.5 ±0.04 (12.9)	4.0 ±0.03 (-30.6)	14.7 ±0.04 (155.5)	4.4 ±0.03 (-22.9)	4.4 ±0.03 (-23.3)
OpenLLM-ARC	5.1 ±0.05 (0.0)	5.7 ±0.04 (10.3)	4.8 ±0.04 (-7.7)	3.6 ±0.04 (-29.4)	5.4 ±0.04 (4.6)	8.5 ±0.04 (65.5)	4.5 ±0.04 (-12.7)	7.2 ±0.04 (39.7)	8.7 ±0.04 (69.2)	4.1 ±0.04 (-20.2)	4.1 ±0.04 (-20.6)
OpenLLM-BBH	5.4 ±0.04 (0.0)	10.0 ±0.05 (84.4)	14.9 ±0.05 (175.4)	4.8 ±0.03 (-12.2)	9.6 ±0.05 (77.3)	13.2 ±0.05 (144.1)	6.2 ±0.04 (13.6)	10.4 ±0.04 (91.0)	11.5 ±0.04 (111.5)	6.7 ±0.04 (23.7)	6.2 ±0.04 (14.3)
OpenLLM-GPQA	5.2 ±0.04 (0.0)	5.4 ±0.03 (4.1)	5.0 ±0.02 (-2.5)	4.8 ±0.03 (-6.5)	4.9 ±0.02 (-4.4)	5.1 ±0.02 (-1.8)	4.4 ±0.03 (-15.8)	4.7 ±0.03 (-9.6)	5.3 ±0.03 (2.1)	4.2 ±0.03 (-18.0)	4.3 ±0.03 (-16.6)
OpenLLM-MUSR	5.5 ±0.04 (0.0)	4.2 ±0.02 (-24.9)	4.8 ±0.02 (-12.9)	3.6 ±0.03 (-35.0)	3.7 ±0.02 (-33.8)	4.3 ±0.02 (-21.8)	3.1 ±0.02 (-43.3)	4.8 ±0.04 (-12.5)	5.7 ±0.02 (3.0)	2.7 ±0.02 (-51.7)	2.7 ±0.02 (-51.7)
ImageNet	4.3 ±0.06 (0.0)	3.5 ±0.04 (-19.9)	7.6 ±0.02 (75.5)	2.4 ±0.04 (-43.7)	3.0 ±0.04 (-30.7)	4.9 ±0.02 (13.2)	2.9 ±0.04 (-33.1)	10.9 ±0.17 (153.1)	4.9 ±0.03 (12.4)	3.3 ±0.04 (-23.0)	3.3 ±0.04 (-23.1)
Average	4.5 ±0.01 (0.0)	9.0 ±0.03 (100.5)	9.5 ±0.03 (101.7)	3.8 ±0.01 (-16.2)	8.3 ±0.02 (84.3)	11.9 ±0.03 (170.2)	5.1 ±0.01 (15.8)	5.6 ±0.02 (24.7)	10.9 ±0.02 (151.9)	4.4 ±0.01 (1.1)	4.0 ±0.01 (-10.0)

Figure 17: Experiment with the near extrapolation model split by using the top 25% of available models as target models and the remaining bottom 75% models as source models. We report the estimation gaps (↓) for target models (calculated as equation 1). We also report ± the standard error of the mean and the estimation gap reduction (↓) over Random-Sampling in parentheses. A negative reduction implies that the method achieves a lower estimation gap than Random-Sampling. % is omitted. Best viewed in color.

E Detailed Conclusion

In this paper, we study the problem of benchmark prediction from fewer data and examine 11 benchmark prediction methods. Our findings call into question the necessity of meticulous core-set selection and reveal that these methods are most proficient at interpolating scores among similar models. However, except RANDOM-SAMPLING and AIPW, all methods face significant difficulties when predicting target models that differ substantially from those they have encountered before.

We caution against the indiscriminate use of benchmark prediction techniques, as their dependence on model similarity causes most of them to fail precisely when most needed: at the evaluation frontier, where the aim is to assess new models with unknown capabilities. Even in the context of interpolation, no method outperforms RANDOM-SAMPLING, when that simple baseline is given access to twice as much data. Thus, while we recommend to use AIPW as a consistent estimator with lower variance, this suggests that simply raising the sampling budget for RANDOM-SAMPLING can be competitive, especially in settings where predictions of other models for fitting AIPW are costly to obtain.

F Broarder Impacts and Limitations

This paper addresses the benchmark prediction problem in scenarios with limited data. One potential limitation of our study is the relatively small number of models examined. For both the HELM-Lite and GLUE benchmarks, we have collected full benchmark results for fewer than 100 models. Despite conducting 100 random trials for each experiment, including additional and more diverse models could further strengthen the comprehensiveness and robustness of our analysis.

We do not anticipate any direct societal impacts from this work, such as potential malicious or unintended uses, nor do we foresee any significant concerns involving fairness, privacy, or security considerations. Additionally, we have not identified potential harms resulting from the application of this technology.