Beyond the Singular: Revealing the Value of Multiple Generations in Benchmark Evaluation

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Large language models (LLMs) have demonstrated significant utility in real-world applications, exhibiting impressive capabilities in natural language processing and understanding. Benchmark evaluations are crucial for assessing the capabilities of LLMs as they can provide a comprehensive assessment of their strengths and weaknesses. However, current evaluation methods often overlook the inherent randomness of LLMs by employing deterministic generation strategies or relying on a single random sample, resulting in unaccounted sampling variance and unreliable benchmark score estimates. In this paper, we propose a hierarchical statistical model that provides a more comprehensive representation of the benchmarking process by incorporating both benchmark characteristics and LLM randomness. We show that leveraging multiple generations improves the accuracy of estimating the benchmark score and reduces variance. Multiple generations also allow us to define $\mathbb{P}(\text{correct})$, a prompt-level difficulty score based on correct ratios, providing fine-grained insights into individual prompts. Additionally, we create a data map that visualizes difficulty and semantics of prompts, enabling error detection and quality control in benchmark construction.

1 Introduction

In recent years, advanced large language models have demonstrated remarkable versatility across a wide range of tasks and domains, with their development continuing to accelerate. To effectively track their progress, numerous generative benchmark datasets have been curated to assess both their general and specialized capabilities.

There are two primary ways for generating responses from large language models (LLMs): greedy decoding and random sampling [10]. Greedy decoding selects the next token with the highest probability, resulting in a deterministic output. In contrast, random sampling, such as nucleus sampling [10], incorporates randomness during decoding by sampling a token at each step based on a probability distribution. This approach leads to non-deterministic output. Current LLM benchmarks typically employ one of these methods; for instance, LiveBench [30] WildBench [15] and OpenLLM leaderboard [1] use greedy decoding, while TrustLLM [11], MT Bench [33] and Alpaca Eval [13] employ a non-deterministic sampling configuration. During evaluations, LLMs generate a single response for each prompt in the benchmark, and the correctness of these responses is determined by comparing them to the ground truth answers. The final benchmark score is then calculated as the average of these individual scores.

However, this presents challenges within the current generative-evaluation paradigm. Firstly, deterministic generation does not align with the real-world application of LLMs, where randomness is inherent. This misalignment can lead to biased estimations of LLM performance. Even with random generation, relying on a single generation can result in significant variance in benchmark scores,

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

particularly when the sample size is small. Furthermore, a single generation is not sufficiently informative for individual prompts, as it cannot address prompt-level questions such as, "Which question is more challenging?" This limitation creates obstacles to understanding the overall composition of the benchmark data.

In this paper, we regard the benchmark as an estimation problem characterized by a statistical model and highlight the significance of incorporating multiple random generations in a principled way. We theoretically demonstrate that increasing the number of generations decreases the variance in benchmark score estimation. Moreover, by leveraging multiple samples, we introduce a fine-grained difficulty metric, $\mathbb{P}\left(\text{correct}\right)$, derived from the inherent latent parameters of our statistical model, to quantify the difficulty of individual prompts. This enables comparisons across different prompts. Additionally, we demonstrate that mislabeled or ambiguous prompts can be effectively detected using multiple generations, highlighting its potential as a tool in benchmark construction.

2 Benchmarking Procedure is a Hierarchical Model

In this section, we show that the benchmark is an estimation problem. Without loss of generality, we consider random sampling as the generation strategy where each token is randomly sampled from a token distribution conditional on previously generated tokens. We also assume the correctness of generations can be obtained using a judgment function, which can be accomplished either by comparing the response with ground truth or by determining whether it passes unit tests.

Given an LLM parameterized by parameters θ, including both model parameters and sampling parameters, for example temperature T and top P, etc.), and a benchmark dataset $\mathcal{D} = \{x_i\}_{i=1}^{n}$, we can define difficulty of the i-th prompt with respect to the LLM as a random variable drawn from the unknown benchmark difficulty distribution $\mathbb{P}(\mu, \sigma; \theta)$, with mean μ and standard deviation σ. Without loss of generality, with k generations per prompt, we can then regard the benchmarking procedure as a hierarchical model as follows:

$$\begin{aligned} p_i &\sim \mathbb{P}(\mu, \sigma; \theta) \quad \text{for } i = 1, \cdots, n, \\ y_{i,j} &\sim \text{Bernoulli}(p_i) \quad \text{for } j = 1, \cdots, k, \end{aligned} \tag{1}$$

where prompt difficulty p_i is sampled from $\mathbb{P}(\mu, \sigma; \theta)$ and p_i represents the probability that the LLM can correctly answer the i-th prompt., i.e., $\mathbb{P}\left(\text{A generated answer to } i\text{-th prompt is correct}\right) = p_i$. This represents a latent difficulty of prompts, We denote the he k-th generation of the $i-$th prompt as $z_{i,j}$ and then $y_{i,j}$ is the correctness indicator for it, where $y_{i,j} = 1$ if it's correct otherwise $y_{i,k} = 0$.

Here both benchmark distribution $\mathbb{P}(\mu, \sigma; \mathcal{D})$ and p_i are unknown and needs to be estimated.

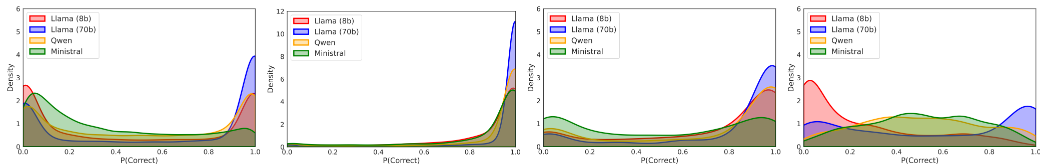
To estimate p_i and μ, we can use a straight forward method of moment estimators $\hat{p}_i = \frac{\sum_{j=1}^{k} y_{i,j}}{k}$, $\hat{\mu} = \frac{\sum_{i=1}^{n} \hat{p}_i}{n} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{k} y_{i,j}}{nk}$. We observe that a widely used item response theory [21, 16, 5], employed to model the difficulty of prompts, represents a specific parametrization of $\mathbb{P}(\mu, \sigma; \mathcal{D})$. Further elaboration on this can be found in Appendix C.

Note that, when $k = 1$, the benchmark score computed based on a single random generation is an estimation of μ, which only utilizes a single generation which leads to a large variance. We can show this by explicitly calculating the variance of our estimators.

Lemma 2.1. *Given the hierarchical model in (1) and the moment estimators $\hat{\mu} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{k} y_{i,j}}{nk}$. Then $\hat{\mu}$ is an unbiased estimator for μ and its variance equals:*

$$\mathit{Var}(\hat{\mu}) = \underbrace{\frac{1}{nk}\left(\mu - \mu^2 - \sigma^2\right)}_{\textit{Withth-prompt Variance}} + \underbrace{\frac{1}{n}\sigma^2}_{\textit{Between-prompt Variance}}. \tag{2}$$

Here, $\text{Var}(\hat{\mu})$ can be decomposed into within-prompt variance and between-prompt variance. Both terms decrease as the number of benchmark data n increases. However, since benchmark data is typically fixed, we analyze the influence of sampling in terms of k. Within-prompt variance captures the randomness in sampling y_{ij} conditional on the $i-$th prompt, and it can be effectively reduced by increasing the number of samples k, converging to 0 as $k \to \infty$. The between-prompt variance term, on the other hand, captures the variability of prompt difficulty p_i across groups, reflecting the randomness of difficulty distribution $\mathbb{P}(\mu, \sigma; \theta)$, and thus remains unaffected by k.



(a) MMLU-Pro (b) GSM8K (c) IFEval (d) MuSR

Figure 1: Distribution of $\mathbb{P}\left(\text{correct}\right)$ of 4 benchmarks.

We can further plug in sample variance $\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(\hat{p}_i - \frac{\sum_{i=1}^{n}\hat{p}_i}{n})^2$ and $\hat{\mu}$ into (2) to get $\widehat{\text{Var}(\hat{\mu})}$.

Finally, based on the central limit theorem, a 95% confidence interval is: $\hat{\mu} \pm 1.96\sqrt{\widehat{\text{Var}(\hat{\mu})}}$.

2.1 Prompt Level Difficulty: $\mathbb{P}\left(\textbf{correct}\right)$

Our goal is to develop a granular, quantifiable measure of prompt difficulty, enabling us to gain a deeper understanding of their relative complexities. By quantifying prompt difficulty at the individual level, we can address fundamental questions such as: 'Which prompts are most challenging?' and 'How do different prompts compare in terms of difficulty?' A fine-grained understanding of prompt difficulty will provide valuable insights into the strengths and weaknesses of language models, as well as the composition of benchmark datasets, ultimately informing the development of more effective models and evaluation frameworks.

We refer to $\mathbb{P}\left(\text{correct}\right) = p_i$ in (1) and its estimation $\widehat{\mathbb{P}}\left(\text{correct}\right) = \hat{p}_i = \frac{\sum_{j=1}^{k} y_{i,j}}{k}$. When the number of generations k increases, it will converge to the true $\mathbb{P}\left(\text{correct}\right)$ and therefore more fine-grained. The probability of correctness p_i can be interpreted as a difficulty score at the prompt level: the higher the p_i, the easier the prompt since the language model has a higher probability of generating a correct response. We demonstrate the use of difficulty scores in the analysis section.

3 Experiments

Benchmark. We choose multiple benchmarks: MMLU-Pro [29], GSM8K [3], MuSR [24], IFEval [34]. For MMLU-Pro, GSM8K, and MUSR, we use accuracy as the metric, while for IFEval, we utilize instance-level strict accuracy. More details of benchmarks are in Appendix D.

LLM and Setup. We utilize four widely-used open-source LLMs: Llama 3.1 (8B and 70B Instruct) [6], Qwen 2.5 (7B Instruct) [31], and Ministral (8B Instruct) [12]. We evaluate both greedy decoding and random sampling on these models, with the latter using a temperature of 0.7 and top-p of 1.0. For each prompt across all benchmarks, we generate 50 samples ($k = 50$) using a 0-shot chain-of-thought prompting strategy.

3.1 Main Results

Results are shown in Figures 1 and Table 1, with the full table available in Appendix Table 2. Key takeaways are summarized below.

Distribution of $\mathbb{P}\left(\textbf{correct}\right)$ show diffuse density in challenging tasks, behaving like random samplers. For the distribution of $\mathbb{P}\left(\text{correct}\right)$, we define stable behavior as a density distribution with high concentrations near 0 and 1, and lower density in between. Conversely, a distribution with a high density between 0 and 1 indicates high randomness. As shown in Figure 1, when confronted with benchmarks that require strong reasoning skills (MMLU-Pro, IFEval, and MuSR), all models display a diffuse density distribution over the support $[0, 1]$. This suggests that LLMs resemble random samplers when handling prompts requiring strong reasoning, underscoring the complexity and sensitivity of their reasoning processes. In contrast, the simpler task GSM8K display densities with more pronounced tails and reduced uncertainty. A plausible explanation is that GSM8K is easier and involves shorter reasoning lengths, which in turn decreases the likelihood of diverse reasoning paths emerging. Additionally, the Llama 70B model shows the most stable performance across benchmarks, indicating that larger models yield more consistent reasoning.

Table 1: Results on four benchmark datasets with four open source LLMs. "n" is the number of prompts, "Greedy" denotes greedy decoding, "Sample (k=50)" is the random sample with 50 generations and "Δ ($k = 1$)" denotes the performance gap between the best and worst run with 1 generation. We include both benchmark score and SE.

Benchmark	**n**	**Llama 3.1 8b Instruct**			**Llama3.1 70b Instruct**		
		Greedy	**Sample** ($k = 50$)	$\Delta(k = 1)$	**Greedy**	**Sample** ($k = 50$)	$\Delta(k = 1)$
MMLU-Pro	$12,187$	46.2 (0.45)	46.1 (0.39)	10.0	63.8 (0.44)	63.4 (0.40)	3.9
GSM8K	$1,319$	86.1 (0.95)	85.6 (0.68)	18.6	95.6 (0.56)	95.3 (0.45)	4.8
IFEval	541	74.5 (1.87)	71.1 (1.51)	8.3	82.6 (1.64)	80.2 (1.42)	5.9
MuSR	756	24.8 (1.65)	29.0 (1.00)	8.2	56.3 (1.80)	57.9 (1.40)	5.4

Estimation differs noticeably between greedy decoding and random sampling, with a single random generation being unstable. Table 1 presents the benchmark scores, highlighting the performance differences between greedy decoding and random sampling. Notably, for GSM8K and MuSR, the absolute differences in benchmark score between these two methods for Llama3 8B are 3.4 and 4.2, respectively, indicating a relatively large performance gap. This discrepancy can also be observed in other models and datasets. Furthermore, we observe considerable variability with one generation, characterized by large values of $\Delta(k = 1)$. This suggests that random sampling with limited generations is ineffective for benchmark evaluation, particularly for small datasets, aligning with our Lemma 2.1. We also investigate how sampling parameters influence the $\mathbb{P}\,(\text{correct})$ distribution, and results are in Appendix E.2.

Multiple generations can help detect labeling errors: a case study on GSM8K. Benchmark construction can involve label errors or ambiguous prompts, such as the approximately 5% error rate in GSM8K. Manually cleaning large datasets is costly, but we found that using multiple generations from advanced LLMs can help identify mislabeled or ambiguous prompts. Based on multiple generations, we can create a data map to visualize $\mathbb{P}(\text{correct})$ against $\mathbb{S}(\text{consistency})$, which measures the semantic consistency of generations. Given a set of k generations and clustering them into C semantic sets, $\mathbb{S}\,(\text{consistency})$ is defined as: $\mathbb{S}\,(\text{consistency}) = \sum_{c=1}^{C} \text{Prop}_c \log \text{Prop}_c$, where Prop_c measures the proportion of generations in group c and its empirical estimator $\widehat{\text{Prop}}_c = \frac{\text{\# generations in set c}}{k}$. This can be seen as negative semantic set entropy; the larger, the more consistent. Semantic clusters in GSM8K can be derived from final answers and can be extended to more open-ended QA by embeddings or LLMs as judges. We hypothesize that prompts with low $\mathbb{P}(\text{correct})$ and high $\mathbb{S}(\text{consistency})$ may be mislabeled or ambiguous due to contradicting with the self-consistency [28]. Self-consistency [28, 18] leverages the intuition that a challenging reasoning problem typically admits multiple reasoning paths leading to its unique correct answer. To verify our hypothesis, we utilize the data map of Llama3 70B for GSM8K and selected prompts with $\mathbb{P}(\text{correct}) \leq 0.1$ and $\mathbb{S}(\text{consistency}) \geq -0.8$, totaling 18 prompts. After manually reviewing the selected prompts, we found that 44.4% prompts were either mislabeled or ambiguous (having multiple valid interpretations of a question). Examples are shown in the Appendix Figure 5. Our results demonstrate the potential of data maps for dataset cleaning, extending prior work [26] from classification to generative models. Notably, our approach only utilizes a single LLM and a simple semantic metric, underscoring future research opportunities to enhance accuracy through multiple models and improved semantic metrics.

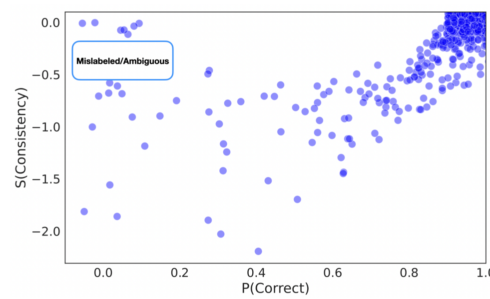


Figure 2: Data map for GSM8K with Llama 70b.

4 Conclusion

In this paper, we investigate the value of multiple generations in LLM benchmark evaluation. By leveraging a hierarchical model, we show that multiple generations help quantify prompt difficulty, reduce variance, and detect labeling errors, making evaluations more robust and informative. Future research could explore the minimal number of generations required for robust evaluation or consider incorporating the covariance structure into the estimation process.

References

[1] Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. Open llm leaderboard (2023-2024), 2023.

[2] Li Cai, Kilchan Choi, Mark Hansen, and Lauren Harrell. Item response theory. *Annual Review of Statistics and Its Application*, 3(1):297–321, 2016.

[3] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[4] Kobe Desender, Filip Van Opstal, and Eva Van den Bussche. Subjective experience of difficulty depends on multiple cues. *Scientific reports*, 7(1):44222, 2017.

[5] Mucong Ding, Chenghao Deng, Jocelyn Choo, Zichu Wu, Aakriti Agrawal, Avi Schwarzschild, Tianyi Zhou, Tom Goldstein, John Langford, Anima Anandkumar, et al. Easy2hard-bench: Standardized difficulty labels for profiling llm performance and generalization. *arXiv preprint arXiv:2409.18433*, 2024.

[6] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[7] Mark E Glickman. Example of the glicko-2 system. *Boston University*, 28, 2012.

[8] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

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

[10] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

[11] Yue Huang, Lichao Sun, Haoran Wang, Siyuan Wu, Qihui Zhang, Yuan Li, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, et al. Trustllm: Trustworthiness in large language models. *arXiv preprint arXiv:2401.05561*, 2024.

[12] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[13] Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacaeval: An automatic evaluator of instruction-following models. `https://github.com/tatsu-lab/alpaca_eval`, 5 2023.

[14] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*, 2022.

[15] Bill Yuchen Lin, Yuntian Deng, Khyathi Chandu, Faeze Brahman, Abhilasha Ravichander, Valentina Pyatkin, Nouha Dziri, Ronan Le Bras, and Yejin Choi. Wildbench: Benchmarking llms with challenging tasks from real users in the wild. *arXiv preprint arXiv:2406.04770*, 2024.

[16] Lovish Madaan, Aaditya K Singh, Rylan Schaeffer, Andrew Poulton, Sanmi Koyejo, Pontus Stenetorp, Sharan Narang, and Dieuwke Hupkes. Quantifying variance in evaluation benchmarks. *arXiv preprint arXiv:2406.10229*, 2024.

[17] Evan Miller. Adding error bars to evals: A statistical approach to language model evaluations. *arXiv preprint arXiv:2411.00640*, 2024.

[18] Eric Mitchell, Joseph Noh, Siyan Li, Will Armstrong, Ananth Agarwal, Patrick Liu, Chelsea Finn, and Christopher D Manning. Enhancing self-consistency and performance of pre-trained language models through natural language inference. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 1754–1768, 2022.

[19] Prathiba Natesan, Ratna Nandakumar, Tom Minka, and Jonathan D Rubright. Bayesian prior choice in irt estimation using mcmc and variational bayes. *Frontiers in psychology*, 7:1422, 2016.

[20] Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. tinybenchmarks: evaluating llms with fewer examples. In *Forty-first International Conference on Machine Learning*.

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

[22] 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 preprint arXiv:2311.12022*, 2023.

[23] Yifan Song, Guoyin Wang, Sujian Li, and Bill Yuchen Lin. The good, the bad, and the greedy: Evaluation of llms should not ignore non-determinism. *arXiv preprint arXiv:2407.10457*, 2024.

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

[25] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

[26] Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. Dataset cartography: Mapping and diagnosing datasets with training dynamics. *arXiv preprint arXiv:2009.10795*, 2020.

[27] Boxin Wang, Weixin Chen, Hengzhi Pei, Chulin Xie, Mintong Kang, Chenhui Zhang, Chejian Xu, Zidi Xiong, Ritik Dutta, Rylan Schaeffer, et al. Decodingtrust: A comprehensive assessment of trustworthiness in gpt models. In *NeurIPS*, 2023.

[28] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

[29] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*, 2024.

[30] Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Ben Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Siddartha Naidu, et al. Livebench: A challenging, contamination-free llm benchmark. *arXiv preprint arXiv:2406.19314*, 2024.

[31] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

[32] Wenbo Zhang, Zihang Xu, and Hengrui Cai. Defining boundaries: A spectrum of task feasibility for large language models. *arXiv preprint arXiv:2408.05873*, 2024.

[33] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023.

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

A Related Work

A.1 LLM Benchmark Evaluation

Recent benchmark evaluations have significantly enhanced our understanding of Large Language Models (LLMs) and have driven further advancements in the field. Notable benchmarks like MMLU [9], HELM [14], and BIG-bench [25] have expanded assessments to include language generation, general knowledge understanding, and complex reasoning. Several other benchmarks assess the trustworthiness of large language models (LLMs) [27, 11, 32] in terms of safety, bias, privacy, and hallucination, etc. Leaderboards like the OpenLLM Leaderboard [1] facilitate performance comparisons across LLMs by evaluating a range of tasks, each targeting different capabilities, to provide a comprehensive assessment of LLMs. However, most benchmark evaluations, even on leaderboards, rely on a single output per example, either greedy decoding or random sampling. Song et al. [23] also examines the performance gap between the two types of generation strategies and highlights the importance of randomness. There is also concurrent work by Miller [17] that mentions using multiple generations to reduce variance, but their contribution is primarily conceptual. In contrast, we provide both theoretical support and empirical results. Additionally, we propose several benefits of using multiple generations, such as difficulty quantification and mislabeled prompt detection, which distinguish our work from theirs.

A.2 Prompt Difficulty in Benchmark

Understanding prompt-level difficulty is crucial for analyzing benchmark composition and some benchmark datasets include difficulty scores for each prompt provided by humans. For example, the MATH dataset [8] offers a variety of high-school-level problems with a broad five-level difficulty rating. Similarly, the GPQA dataset [22] contains graduate-level multiple-choice questions rated on a 4-point scale by two experts. Recent studies [5, 20] also attempted to estimate difficulty scores of individual prompts using item response theory [2, 19] or Glicko-2 [7], based on offline evaluation results from a pool of large language models (LLMs) or human participants. This approach seeks to provide an objective difficulty score by encompassing a diverse range of testers, including both humans and LLMs. However, this can lead to misalignment when focusing solely on a target LLM. A question that is easy for one model might be difficult for others, highlighting the inherently subjective nature of difficulty [4]. Therefore, it is more relevant to consider the subjective difficulty specific to the target LLM.

B Limitation

While using multiple generations in benchmark evaluation is promising, it demands more computational resources during inference time. Future research could explore the minimal number of generations required for robust evaluation, potentially reducing within-prompt variance. Additionally, our statistical model assumes that all prompts are independently sampled from the benchmark difficulty distribution, which may not be accurate in practice, as prompts can originate from the same subjects or resources. Future work should consider incorporating the covariance structure into the estimation process. Another drawback is the detection of mislabeled prompts. Although our method efficiently reduces the effort needed to filter samples, the true positive rate is not high (around 50%). Potential research could leverage more sophisticated semantic metrics and model ensembles to better detect mislabeled or ambiguous prompts.

C IRT is a special parametrization of $\mathbb{P}\left(\text{correct}\right)$

$\mathbb{P}\left(\text{correct}\right)$ is closely connected to item response theory. Many studies [21, 16, 5] utilize IRT to quantify the difficulty of prompts using multiple LLMs. One variation of the IRT model is the one-parameter logistic (1PL) model as defined below:

$$\mathbb{P}\left(y_{li}=1 \mid \theta_l, b_i\right)=\frac{1}{1+\exp^{-\left(-\theta_l-b_i\right)}}, \tag{3}$$

where $\mathbb{P}\left(y_{li}=1 \mid \theta_l, b_i\right)$ is the probability that LLM l can answer the j-th prompt correctly. θ_l represents the latent ability of LLM l, b_i is the difficulty parameter of the j-th prompt.

We observe that when we focus on a single LLM, i.e., when LLM l is fixed, $\mathbb{P}(y_{li} = 1 \mid \theta_l, b_i)$ coincides with the prompt difficulty p_i defined in (1). Consequently, the right-hand side of (3) can be viewed as a specific parametrization of the prompt difficulty using a logit link function. This implies that, theoretically, the maximum likelihood estimator of IRT and our method are equivalent via a sigmoid transformation. We use the 1PL model here for illustrative purposes, but this equivalence also holds when extended to models with more parameters.

D Benchmark Details

MMLU-Pro is a comprehensive benchmark tailored for advanced, multi-disciplinary language understanding and reasoning at the proficient level. The GSM8K dataset comprises linguistically diverse math word problems from grade school curricula, crafted by human experts. MuSR is a specialized dataset designed to assess language models' performance on multi-step soft reasoning tasks presented in natural language narratives. IFEval, meanwhile, provides verifiable instructions to test large language models' ability to follow instructions accurately.

E Additional Results

E.1 Benchmark Results

We further conduct a synthetic analysis to demonstrate the value of multiple generations. Using $k = 50$ as the oracle (i.e., the full set of generated samples), we evaluate $k = 1, 5, 10, 20$ over 1000 trials each by sampling with replacement. As shown in Fig. 3, increasing k leads to narrower 95% confidence intervals that coverage the true score. In contrast, greedy decoding exhibits a consistent performance gap, suggesting that even a modest number of sampled generations better approximates $\mathbb{P}(\text{correct})$ than greedy decoding.

Table 2: Results on four benchmark datasets with four open source LLMs. "n" is the number of prompts, "Greedy" denotes greedy decoding, "Sample (k=50)" is the random sample with 50 generations and "Δ $(k = 1)$" denotes the performance gap between the best and worst run with 1 generation. We include both benchmark score and SE.

Benchmark	**n**	**Llama 3.1 8b Instruct**			**Llama3.1 70b Instruct**		
		Greedy	**Sample** $(k = 50)$	$\Delta(k = 1)$	**Greedy**	**Sample** $(k = 50)$	$\Delta(k = 1)$
MMLU-Pro	$12,187$	46.2 (0.45)	46.1 (0.39)	10.0	63.8 (0.44)	63.4 (0.40)	3.9
GSM8K	$1,319$	86.1 (0.95)	85.6 (0.68)	18.6	95.6 (0.56)	95.3 (0.45)	4.8
IFEval	541	74.5 (1.87)	71.1 (1.51)	8.3	82.6 (1.64)	80.2 (1.42)	5.9
MuSR	756	24.8 (1.65)	29.0 (1.00)	8.2	56.3 (1.80)	57.9 (1.40)	5.4
Benchmark	**n**	**Qwen 2.5 7B Instruct**			**Ministral 8B Instruct**		
		Greedy	**Sample** $(k = 50)$	$\Delta(k = 1)$	**Greedy**	**Sample** $(k = 50)$	$\Delta(k = 1)$
MMLU-Pro	$12,187$	53.3 (0.45)	53.0 (0.36)	1.3	39.7 (0.44)	36.3 (0.29)	1.5
GSM8K	$1,319$	90.2 (0.82)	90.2 (0.65)	2.3	86.1 (0.95)	84.9 (0.73)	3.1
IFEval	541	72.6 (1.92)	71.2 (1.64)	5.9	51.4 (2.15)	49.8 (1.65)	5.6
MuSR	756	49.2 (1.82)	50.9 (0.98)	8.3	49.7 (1.82)	50.8 (0.91)	8.6

E.2 Varying Temperature T

To investigate how temperature influences the $\mathbb{P}(\text{correct})$ distribution, we vary the sampling temperatures T across 0.4, 0.7, and 1.0 for the GSM8K and MUSR datasets using the Llama 8B and 70B models. The results are in Figure E.2. We find that for the smaller 8B model, as T increases, the distribution becomes more unstable with a more diffuse density. However, for the larger model, the $\mathbb{P}(\text{correct})$ is less sensitive to changes in T.

F Semantic Consistency for Responses: $\mathbb{S}$ (consistency)

Apart from the correctness, we can also measure the difficulty of benchmark prompts by examining the semantic complexity from multiple generations. This is because analyzing the nature of errors

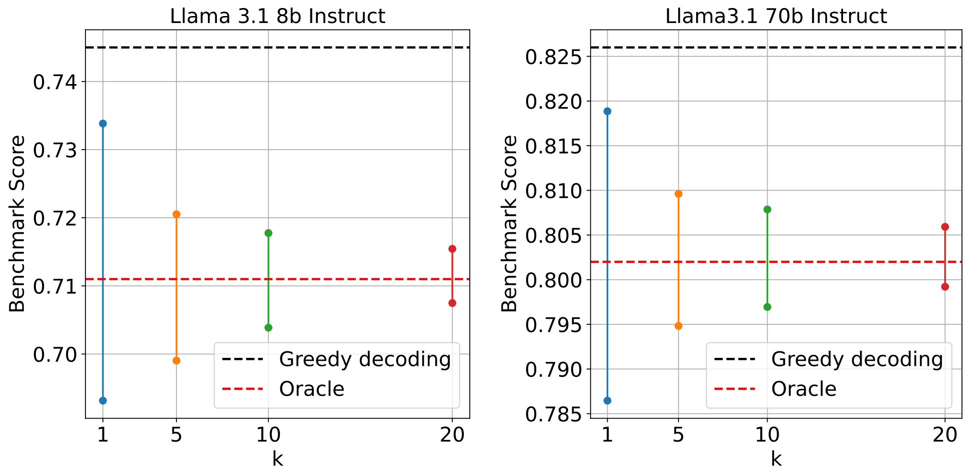
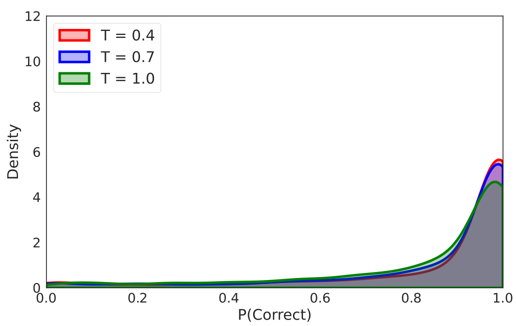
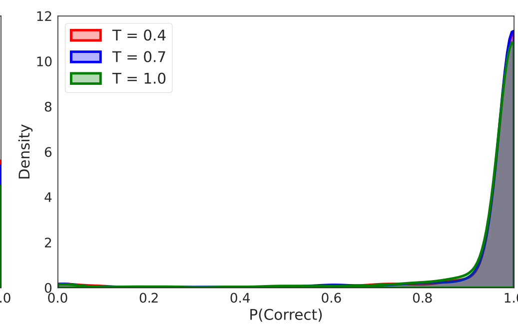


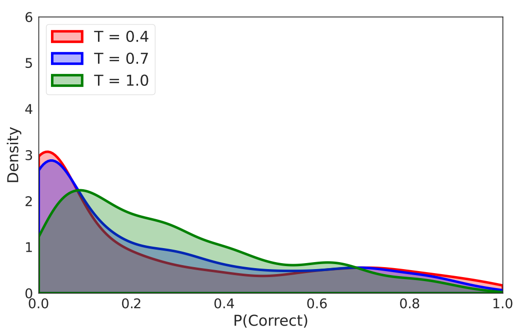
Figure 3: Benchmark score of IFEval over different k.



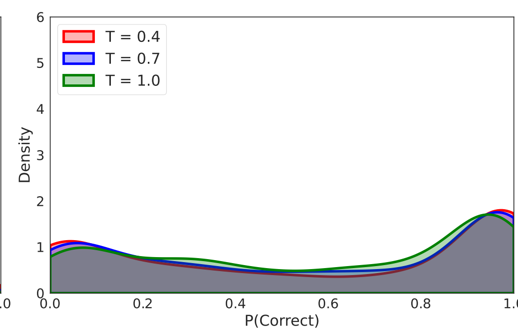
(a) GSM8K Llama 8B



(b) GSM8K Llama 70B



(c) MUSR Llama 8B



(d) MUSR Llama 70B

Figure 4: Distribution of $\mathbb{P}\left(\text{correct}\right)$ for GSM8K and MUSR when varying temperature T.

produced by LLMs can provide valuable insights into their decision-making processes. Specifically, it can help us determine whether LLMs tend to make consistent or varied mistakes, shedding light on their limitations and potential areas for improvement.

We can group responses into multiple clusters based on their semantic meaning using bidirectional entailment predictions from a Natural Language Inference (NLI) model, such as DeBERTa or a prompted large language model (LLM).

One common metric for quantifying consistency is the number of semantic sets, originally developed for uncertainty quantification in LLMs. The number of semantic sets assumes that a higher number of distinct semantic sets corresponds to lower consistency.

However, the number of semantic sets only considers the number of clusters, without taking into account the proportion of generations within each cluster. For instance, consider two scenarios with 8 generations and 2 clusters: one where 1 generation falls into the first cluster and 7 into the second, versus another where 4 generations fall into each cluster. While these scenarios clearly represent different levels of consistency, the semantic set metric fails to distinguish between them, highlighting the need for a more nuanced approach to evaluating consistency.

Here, we utilize a metric called semantic set entropy to better account for the proportions of semantic clusters. Given a set of k generations and clustering them into C semantic sets, the semantic set entropy can be represented as:

$$\mathbb{S}\,(\text{consistency}) = \sum_{c=1}^{C} \text{Prop}_c \log \text{Prop}_c,$$

where Prop_c measures the proportion of generations in group c and its empirical estimator $\widehat{\text{Prop}}_c = \frac{\text{\# generations in set c}}{m}$ with finite m samples. This can be seen as negative semantic set entropy, the larger, more consistent.

G Proof of Lemma 2.1

Restate of Lemma 2.1:

Given the model

$$\begin{aligned} p_i &\sim \mathbb{P}(\mu, \sigma; \theta) \quad \text{for } i = 1, \cdots, n \\ y_{i,j} &\sim \text{Bernoulli}(p_i) \quad \text{for } j = 1, \cdots, k, \end{aligned} \tag{4}$$

and the moment estimator $\hat{\mu} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{k} y_{i,j}}{nk}$. Then $\hat{\mu}$ is an unbiased estimator for μ and its variance equals

$$\text{Var}(\hat{\mu}) = \underbrace{\frac{1}{nk}\left(\mu - \mu^2 - \sigma^2\right)}_{\text{Withth-prompt Variance}} + \underbrace{\frac{1}{n}\sigma^2}_{\text{Between-prompt Variance}}.$$

Proof: Firstly we show $\hat{\mu}$ is an unbiased estimation of μ, which can be directly show by the expectation:

$$\begin{aligned} \mathbb{E}\left[\hat{\mu}\right] &= \frac{\sum_{i=1}^{n}\sum_{j=1}^{k} y_{i,j}}{nk} \\ &= \frac{\sum_{i=1}^{n} \mathbb{E}\left[\sum_{j=1}^{k} y_{i,j}\right]}{nk} \\ &\overset{(3)}{=} \frac{\sum_{i=1}^{n} \mathbb{E}\left[\mathbb{E}\left[\sum_{j=1}^{k} y_{i,j} \mid p_i\right]\right]}{nk} \\ &= \frac{\sum_{i=1}^{n} k\mathbb{E}\left[p_i\right]}{nk} \\ &= \frac{\sum_{i=1}^{n} k\mu}{nk} \\ &= \mu, \end{aligned}$$

where (3) utilizes the law of total expectation. Hence $\hat{\mu}$ is unbiased estimator of μ. The variance of $\hat{\mu}$ can be further shown:

$$
\begin{aligned}
\operatorname{Var}(\hat{\mu}) &= \operatorname{Var}\left(\frac{\sum_{i=1}^{n}\sum_{j=1}^{k} y_{i,j}}{nk}\right) \\
&= \frac{1}{n^2k^2}\left(\sum_{i=1}^{n}\operatorname{Var}\left(\sum_{j=1}^{k} y_{i,j}\right)\right) \\
&\overset{(3)}{=} \frac{1}{n^2k^2}\left(\sum_{i=1}^{n}\mathbb{E}\left[\operatorname{Var}\left(\sum_{j=1}^{k} y_{ij} \mid p_i\right)\right]\right. \\
&\quad \left. + \operatorname{Var}\left(\mathbb{E}\left(\sum_{j=1}^{k} y_{ij} \mid p_i\right)\right)\right) \\
&= \frac{1}{n^2k^2}\left(\sum_{i=1}^{n}\mathbb{E}\left[kp_i\left(1-p_i\right)\right] + \operatorname{Var}\left(kp_i\right)\right) \\
&= \frac{1}{n^2k^2}\left(nk\left(\mathbb{E}\left[p_i\right] - \mathbb{E}\left[p_i^2\right]\right) + nk^2\operatorname{Var}\left(p_i\right)\right) \\
&= \underbrace{\frac{1}{nk}\left(\mu-\mu^2-\sigma^2\right)}_{\text{Withth-prompt Variance}} + \underbrace{\frac{1}{n}\sigma^2}_{\text{Between-prompt Variance}} .
\end{aligned}
$$

where (3) utilizes the low of total variance.

Mislabeled

Question: Marin and his neighbor Nancy each eat 4 apples a day. How many apples do they eat in 30 days?
Answer: In one day, Marin and Nancy eat 4 + 1 = <<4+1=5>>5 apples. In 30 days, they eat 30 * 5 = <<30*5=150>>150 apples.
Correct Answer: 30*8-240

Ambiguous

Question: Alex is getting ready to attend an event that she has hosted and wants to make sure that she has enough seats for everyone. She invites 100 people via email and each invited person says that they will also invite 2 of their friends. She then calls 10 of her friends to invite them too and 8 of them say they will be bringing their spouses. How many seats will Alex need?
Answer: Each of the people that were emailed are bringing 2 friends, which means that they will be in groups of 1 +2 = <<1+2=3>>3 people. Since 100 people were emailed, this creates a total of 3 * 100 = <<3*100=300>>300 people. Out of her friends, 8 people said that they will be bringing their spouse, so this is a total of 10 + 8 = <<10+8=18>>18 people. Including her own seat, Alex is going to need a total of 300 + 18 +1 = <<300+18+1=319>>319 seats.
The question is unclear about whether Alex should include her own seat, which creates ambiguity.

Figure 5: Examples of detected mislabeled and ambiguous prompts in GSM8K.