Don't Make Your LLM an Evaluation Benchmark Cheater

Anonymous ACL submission

Abstract

To assess the capacity of large language models (LLMs), a typical approach is to construct evaluation benchmarks for measuring their ability level in different aspects. Although a surge of high-quality benchmarks have been released, the concerns about the appropriate use of benchmarks and the fair comparison are increasingly growing. In this paper, we discuss the potential risk and impact of inappropriately using evaluation benchmarks and misleadingly interpreting the evaluation results. Specially, we focus on a special issue that would lead to inappropriate evaluation, *i.e., benchmark leakage*, referring that the data related to evaluation sets is occasionally used for model training. This phenomenon now becomes more common since pre-training data is often prepared ahead of model test. We conduct extensive experiments to study the effect of benchmark leakage, and find that it can dramatically boost the evaluation results, which would finally lead to an unreliable assessment of model performance. We hope this work can draw attention to appropriate training and evaluation of LLMs.

1 Introduction

Recently, a surge of high-quality evaluation benchmarks (Chang et al., 2023) have been proposed to provide a comprehensive capability evaluation of large language models (LLMs) (Brown et al., 2020; OpenAI, 2023; Zhao et al., 2023), for better understanding how LLMs evolve in model capacity. Typical benchmarks include MMLU (Hendrycks et al., 2021) (for measuring multitask language understanding ability) and Big-Bench (Srivastava et al., 2022) (for quantifying and extrapolating the capabilities of LLMs). Based on these benchmarks, one can conveniently examine the effect of new training strategies or monitor the training status of LLMs (either pre-training or supervised fine-tuning). It has become common to report the results on benchmarks for demonstrating the effectiveness of newly released LLMs (Touvron et al., 2023b; Anil et al., 2023). Furthermore, to compare the performance of different LLMs, various leaderboards have been also created to rank LLMs according to their performance on existing or new evaluation benchmarks, such as OpenCompass (Contributors, 2023) and C-Eval (Huang et al., 2023).

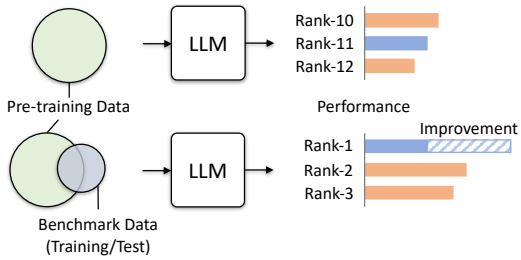

Figure 1: Illustration of the potential risk about data leakage. Once the pre-training data with overlap to the benchmark data is used for training LLM, its benchmark performance would be greatly increased.

Despite the wide use of these benchmarks and leaderboards, increasing concerns (Aiyappa et al., 2023; Li, 2023) are growing about the fairness and reliability in evaluating existing LLMs. A major issue is that the data contamination or leakage is likely to occur for large-scale benchmark evaluation, which means that LLMs are trained with relevant or exactly the same data for test. Such an issue could be unconsciously triggered, since we might be unaware of the future evaluation datasets when preparing the pre-training corpus. For example, GPT-3 has found that Children's Book Test dataset (Hill et al., 2016) was included in the pre-training corpus, and LLaMA-2 has mentioned that the contexts in BoolQ dataset (Clark et al., 2019) are extracted verbatim from the webpages, which may be included in the publicly available corpus.

Indeed, when conducting evaluation with existing benchmarks, the results of evaluated LLMs are mostly obtained by running them on local servers or via API calls. During this process, there is no strict

checking on any potentially inappropriate ways (*e.g.,* data contamination) that would cause an unnormal improvement of evaluation performance. To make matters worse, the detailed composition (*e.g.,* data sources) of the training corpus is often regarded as the core "secret" of existing LLMs. Therefore, it becomes difficult to directly examine the contamination issues when performing the evaluation for benchmark maintainers.

Considering this issue, the aim of this paper is to draw attention on appropriately using existing evaluation benchmarks and avoiding any misleading behaviors in obtaining or interpreting the evaluation results. Specifically, we mainly focus on discussing the potential effect of *benchmark leakage*, which refers to the case that test data or relevant data (*e.g.,* training set) has been included in the pre-training corpus. It would cause an unfair performance advantage when comparing different LLMs or assessing the ability level of some specific LLMs. As we discussed before, this issue tends to become increasingly more common as we try to collect more public text data for training. To investigate this issue, we set up several benchmark leakage settings *that should be totally avoided during evaluation*, including the leakage of training sets, test prompts, and test sets. Based on the three settings, we continually train four popular language models, ranging from 1.3B to 7B, and test the performance of the four models on a number of existing benchmarks. In addition, we also examine the potential risk of benchmark leakage on other abilities.

Experimental results reveal that benchmark leakage can lead to an unfair boost in the evaluation performance of LLMs. Smaller LLMs (*e.g.,* 1.3B models) can be deliberately elevated to outperform $10\times$ larger models on certain tasks. As a side effect, the performance of these specially trained LLMs on other normally tested tasks would likely be adversely affected if we fine-tune or train the model only with these leaked data. By examining the potential risks of benchmark leakage, we would like to emphasize the importance of fair and appropriate evaluation for LLMs, and propose several suggestions in Appendix B.

2 Empirical Study: Benchmark Leakage

During pre-training, the data contamination or leakage about possible evaluation benchmarks, is likely to be unconsciously triggered (Oren et al., 2023; Sainz et al., 2023). It would violate regular evaluation settings for assessing zero/few-shot generalization capability, thus affecting the capability assessment of LLMs. To better understand the potential influence of the benchmark leakage issue, we conduct an empirical study that continually trains small-sized LLMs on three settings with different levels of information leakage.

2.1 Experimental Setup

Training Settings with Benchmark Leakage. We aim to test the influence of possible benchmark leakage issues on the evaluation results of LLMs. A benchmark typically contains a set of test examples, and relies on fixed templates to prompt LLMs for evaluation. Such an evaluation process may lead to three types of benchmark leakage risks, including **test prompt**, **test set**, or **other relevant data** (*e.g.,* training set) into the pre-training corpus. Considering the above settings, we simulate three extreme leakage issues where the three types of information have been used for continually training LLMs, and design the following evaluation settings.

- *Using MMLU Training Set*: the auxiliary training set provided by the official MMLU benchmark (Hendrycks et al., 2021) is used for training.[1]
- *Using All Training Sets*: in addition to MMLU training set, the training sets of all other collected evaluation benchmarks are also used for training.
- *Using All Training Sets with Test Prompt*: all the training sets, with their corresponding test prompts, *e.g.,* task description and few-shot demonstration, are used for training.
- *Using All Training and Test Sets with Test Prompt*: all the training sets, test prompts, and test sets of all the collected benchmarks are used for training. (CAUTION: the most extreme case only for reference, where all information is leaked.)

Evaluation Benchmark and LLMs. To conduct the empirical study, we select the widely-used benchmark MMLU (Hendrycks et al., 2021) and employ seven QA, three reasoning, and five reading comprehension datasets for evaluation. To thoroughly analyze the effect of benchmark leakage on the evaluation performance, we select four models for evaluation, which have provided pre-training details or conducted careful data contamination analysis. These baseline models include GPT-Neo-1.3B (Black et al., 2021), phi-1.5 (Li et al., 2023), OpenLLaMA-3B (Geng and Liu, 2023), and

[1] https://github.com/hendrycks/test. It contains data collected from other QA datasets *e.g.,* ARC and OBQA.

Backbone	Training Setting	MMLU	BoolQ	PIQA	HSwag	WG	ARC-E	ARC-C	OBQA
LLaMA-13B	(None)	46.90	76.70	79.70	60.00	73.00	79.00	49.40	34.60
LLaMA-30B	(None)	57.80	83.39	80.63	63.39	76.08	80.55	51.62	36.40
LLaMA-65B	(None)	**64.50**	**85.40**	**81.70**	**64.90**	**77.20**	**80.80**	**52.30**	**38.40**
GPT-Neo (1.3B)	(None)	24.04	62.57	70.57	38.65	55.72	55.98	23.29	21.40
	+MMLU Train S	35.84	57.89	68.39	37.27	52.17	50.93	27.39	20.40
	+All Train S	35.10	**78.32**	68.61	42.46	61.72	63.68	33.36	29.40
	+All Train S+Test P	**36.15**	76.91	**73.72**	**42.75**	**64.25**	**64.39**	**34.13**	**31.80**
	+All Train S+Test P&S	52.25	87.25	85.96	62.98	80.66	88.17	70.31	63.20
phi-1.5 (1.3B)	(None)	42.87	74.34	76.50	47.99	**73.56**	75.84	44.97	38.40
	+MMLU Train S	46.08	74.37	76.50	47.80	73.09	**75.93**	**48.63**	40.00
	+All Train S	45.20	82.35	**74.37**	**54.64**	69.46	75.00	47.87	**42.40**
	+All Train S+Test P	**46.80**	**82.72**	74.27	54.55	70.56	75.00	47.18	39.80
	+All Train S+Test P&S	75.05	92.60	97.55	77.88	96.05	97.47	92.92	94.20
OpenLLaMA (3B)	(None)	26.49	66.51	74.81	49.42	60.85	69.57	33.87	26.60
	+MMLU Train S	43.12	74.10	71.22	47.28	62.43	58.92	35.41	32.00
	+All Train S	44.86	85.41	**76.82**	**54.42**	71.11	**72.26**	41.55	**42.00**
	+All Train S+Test P	**48.31**	**85.57**	76.50	54.34	**72.30**	71.80	**41.64**	40.80
	+All Train S+Test P&S	87.31	97.55	98.26	97.61	96.37	99.16	97.87	96.20
LLaMA-2 (7B)	(None)	42.95	71.68	70.78	55.34	67.96	72.52	41.30	32.20
	+MMLU Train S	51.61	81.96	69.64	49.46	70.64	61.87	36.52	36.80
	+All Train S	52.15	**88.72**	79.05	61.08	**79.95**	76.60	49.49	**48.00**
	+All Train S+Test P	**56.04**	87.86	**79.11**	**61.19**	76.56	**76.64**	**50.26**	45.00
	+All Train S+Test P&S	96.34	99.08	99.62	99.47	97.47	99.54	99.23	99.40

Table 1: The comparison among benchmark leakage settings and the original LLMs on MMLU and QA tasks. *Train S*, *Test P* and *Test P&S* denote the data leakage scenarios that use the training set, test prompt, and both test set and test prompt during training, respectively. The task abbreviations are as follows: HSwag (Hellaswag), WG (WinoGrande), ARC-E (ARC-Easy), ARC-C (ARC-Challenge), and OBQA (OpenBookQA). The results in gray are the worst leakage setting using all the test sets. The best results in each group are in **bold** except for the worst case.

LLaMA-2-7B (Touvron et al., 2023b). We provide more detailed experimental settings in Appendix A.

2.2 Results and Analysis

We report the results of LLMs after training with the benchmark leakage settings in Table 1 and 4 (in Appendix). We have the following observations.

First, using MMLU training set can greatly boost the evaluation results on the MMLU benchmark. However, this improvement comes with the cost of performance decrease on tasks unrelated to MMLU, (*e.g.,* HellaSwag and GSM8k), suggesting that overemphasizing a specific task may lower the model generalization capability. Besides, when incorporating all the training sets of the evaluated benchmarks, there is a notable performance increase across almost all the evaluated tasks. Incorporating training data converts the original zero/few-shot evaluation into an in-domain test task, making it easier for LLMs to achieve higher results.

Second, when the test prompts were leaked, smaller LLMs can even surpass much larger LLMs, *e.g.,* phi-1.5-1.3B outperforms LLaMA-65B on RACE-M and RACE-H. This highlights the significance of the test prompt as valuable information from the evaluation benchmark, since it contains the detailed input format during test. Furthermore, this observation raises concerns about using fixed test prompts in the evaluation benchmark, as it may not be resilient to the aforementioned leakage risk.

Finally, as the results in grey font, test data leakage significantly inflates benchmark performance, leading 1.3B LLMs to outperform 65B LLMs across most tasks. Evidently, this increase does not imply any improvement in capacity, but rather benchmark cheating.

Overall, benchmark leakage directly leads to an unfair advantage in evaluation results of the involved models, which should be strictly avoided when conducting any evaluation.

3 Potential Risk of Benchmark Leakage

In addition to the influence on the reliability of capability estimation, we also investigate whether benchmark leakage would lead to potential risks in model capacity. Limited by the training compute, we only continually pre-train the LLMs on the training sets of all the selected evaluation benchmarks as in Section 2. Such a way is the most direct way for benchmark cheating (should be avoided). We

Backbone	Training	LAMB	XSum	HEval
GPT-Neo (1.3B)	(None)	**46.10**	**7.54**	2.44
	+Leak	46.00	6.84	**3.05**
OpenLLaMA (3B)	(None)	**56.50**	**8.31**	**4.27**
	+Leak	53.20	0.19	1.83
LLaMA-2 (7B)	(None)	**68.20**	**8.67**	**26.83**
	+Leak	61.00	0.25	8.54

Table 2: The comparison among LLMs on two text generation and a code synthesis tasks. "*Leak*" denotes the data leakage scenario using all training sets of the benchmarks in Section 2. LAMB and HEval refer to the LAMBADA and HumanEval datasets, respectively.

Backbone	Training	LAMB	XSum	HEval
GPT-Neo (1.3B)	+IT	**45.40**	**8.34**	**14.24**
	+Leak+IT	43.50	8.25	12.20
OpenLLaMA (3B)	+IT	**54.00**	**3.50**	**9.15**
	+Leak+IT	46.20	2.61	6.71
LLaMA-2 (7B)	+IT	**60.30**	**8.64**	**28.66**
	+Leak+IT	53.60	8.55	20.73

Table 3: The comparison among LLMs after instruction tuning. "*Leak*" denotes the data leakage using all training sets of the benchmarks in Section 2. "*IT*" denotes the instruction tuning using Alpaca and CodeAlpaca for text generation and code synthesis tasks, respectively.

speculate that it is likely to affect the capacities of LLMs on normally tested tasks (without data leakage), due to "catastrophe forgetting" (Luo et al., 2023; Goodfellow et al., 2013).

3.1 Effect on the Performance of Other Tasks

Experimental Setup. After training on the leaked benchmark data, it would potentially mislead LLMs to overemphasize the specific knowledge and output style of the benchmark data, thereby affecting their performance on other tasks. In this part, we conduct experiments to validate the effect. We select three tasks that are not involved in the leaked training data, consisting of two text generation tasks, *i.e.,* LAMBADA (Paperno et al., 2016) and XSum (Narayan et al., 2018), and a code synthesis task HumanEval (Chen et al., 2021) to evaluate LLMs in the zero-shot setting.

Results Analysis. We show the results of LLMs *with* and *without* benchmark leakage in Table 2. First, we can observe that after training on the leaked data, the performance of all LLMs degrades on the two text generation and the code synthesis tasks. Specifically, the text summarization ability of OpenLLaMA-3B and LLaMA-2-7B, seems to be weakened a lot after training on the leaked data (*e.g.,* 0.19 and 0.25 Rouge-L in XSum). This demonstrates that benchmark leakage may have a negative impact on the performance of these normally tested tasks (without data leakage).

3.2 Effect on Model Adaptation

Experimental Setup. After training on the leaked data, LLMs would be specially fit for the benchmark data. However, LLMs might need to be further fine-tuned for attaining some specific goals (*e.g.,* solving new tasks or serving emergent applications). In this part, we investigate the influence of data leakage on LLMs' adaptation capability. We select two instruction datasets to fine-tune LLMs with or without training on the leaked data, *i.e.,* Alpaca (Taori et al., 2023) and CodeAlpaca (Chaudhary, 2023), which are synthetic natural language and code generation instructions, respectively. Then, we evaluate their performance on the text generation and code synthesis tasks.

Results Analysis. In Table 3, by comparing the performance of the instruction-tuned LLMs (+Alpaca or +CodeAlpaca) *with* and *without* training on the leaked data, we can see that the LLMs with benchmark leakage still underperform their non-leaked counterparts. For the HumanEval dataset, the performance improvements of instruction tuning for LLMs trained with leaked data only reach approximately 80% of those achieved by models that are not trained on leaked data. This indicates that benchmark leakage may lead to a decline in the adaptation ability, constraining the improvement of LLMs through subsequent fine-tuning processes.

4 Conclusion

In this paper, we conducted empirical studies to investigate the potential risk and impact of *benchmark leakage* on LLM evaluation, to draw the attention to the appropriate use of existing evaluation benchmarks for LLMs. We found that data leakage can largely boost the benchmark results of LLMs (even small models), making the evaluation unfair and untrustworthy. Besides, benchmark leakage may also have negative impacts on the performance of other tasks and the adaptation capability of LLMs. These findings suggest that such attempts should be strictly avoided for fairly assessing the model performance on evaluation benchmarks.

Limitation

In this work, we conducted preliminary experiments to emphasize the potential risks associated with benchmark leakage in training LLMs. However, there are still several limitations in our study.

First, our experiments involved continually training existing pre-trained LLMs with leaked data. We do not have sufficient computational resources to investigate the impact when directly incorporating benchmark leakage during the pre-training process. Given that the pre-training dataset is significantly larger than the benchmark data, introducing data leakage during pre-training might yield different findings. Nonetheless, we strongly recommend avoiding this situation as it would breaks the nature of zero-shot/few-shot evaluation.

Second, we did not explore more fine-grained data leakage scenarios in this study, such as only leaking training examples without labels and varying the proportion of the leaked dataset. We encourage more research efforts into this issue with more systematic studies.

Third, we did not calculate the degree of contamination between the mainstream benchmarks and commonly-used pre-training datasets, which could serve as an important reference for alerting LLM developers to adjust their evaluation settings. While we suggest that developers and benchmark maintainers report contamination analyses, accurately and efficiently estimating the contamination risk of each example in the benchmark is also a challenging task. For example, the suggested n-gram hash algorithm may not detect semantic-level knowledge leakage risks.

References

Rachith Aiyappa, Jisun An, Haewoon Kwak, and Yong-Yeol Ahn. 2023. Can we trust the evaluation on chatgpt? *CoRR*, abs/2303.12767.

Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernández Ábrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan A. Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vladimir Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, and et al. 2023. Palm 2 technical report. *CoRR*, abs/2305.10403.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7432–7439. AAAI Press.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow. If you use this software, please cite it using these metadata.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Kaijie Zhu, Hao Chen, Linyi Yang, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. 2023. A survey on evaluation of large language models. *CoRR*, abs/2307.03109.

Sahil Chaudhary. 2023. Code alpaca: An instruction-following llama model for code generation. https://github.com/sahil280114/codealpaca.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harrison Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N.

Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *CoRR*, abs/2107.03374.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2924–2936. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the AI2 reasoning challenge. *CoRR*, abs/1803.05457.

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

Together Computer. 2023. Redpajama-data: An open source recipe to reproduce llama training dataset.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass.

Yiming Cui, Ting Liu, Wanxiang Che, Li Xiao, Zhipeng Chen, Wentao Ma, Shijin Wang, and Guoping Hu. 2019. A span-extraction dataset for chinese machine reading comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5882–5888. Association for Computational Linguistics.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2021. The pile: An 800gb dataset of diverse text for language modeling. *CoRR*, abs/2101.00027.

Xinyang Geng and Hao Liu. 2023. Openllama: An open reproduction of llama.

Shahriar Golchin and Mihai Surdeanu. 2023. Time travel in llms: Tracing data contamination in large language models. *CoRR*, abs/2308.08493.

Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. 2013. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *CoRR*, abs/1312.6211.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2016. The goldilocks principle: Reading children's books with explicit memory representations. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, Yao Fu, Maosong Sun, and Junxian He. 2023. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. *CoRR*, abs/2305.08322.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard H. Hovy. 2017. RACE: large-scale reading comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 785–794. Association for Computational Linguistics.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023. Textbooks are all you need II: phi-1.5 technical report. *CoRR*, abs/2309.05463.

Yucheng Li. 2023. An open source data contamination report for llama series models. *CoRR*, abs/2307.03109.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 158–167. Association for Computational Linguistics.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2023. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *CoRR*, abs/2308.08747.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2381–2391. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1797–1807. Association for Computational Linguistics.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Yonatan Oren, Nicole Meister, Niladri Chatterji, Faisal Ladhak, and Tatsunori B. Hashimoto. 2023. Proving test set contamination in black box language models. *CoRR*, abs/2307.03109.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. Coqa: A conversational question answering challenge. *Trans. Assoc. Comput. Linguistics*, 7:249–266.

Oscar Sainz, Jon Ander Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. 2023. NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark. *CoRR*, abs/2310.18018.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8732–8740. AAAI Press.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. 2023. Detecting pretraining data from large language models. *CoRR*, abs/2310.16789.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ameet Rahane, Anantharaman S. Iyer, Anders Andreassen, Andrea Santilli, Andreas Stuhlmüller, Andrew M. Dai, Andrew La, Andrew K. Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakas, and et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *CoRR*, abs/2206.04615.

Kai Sun, Dian Yu, Dong Yu, and Claire Cardie. 2020. Investigating prior knowledge for challenging chinese machine reading comprehension. *Trans. Assoc. Comput. Linguistics*, 8:141–155.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4149–4158. Association for Computational Linguistics.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. 2021. Generalizing from a few examples:

A survey on few-shot learning. *ACM Comput. Surv.*, 53(3):63:1–63:34.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13484–13508. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*.

Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. 2019. Zero-shot learning - A comprehensive evaluation of the good, the bad and the ugly. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(9):2251–2265.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4791–4800. Association for Computational Linguistics.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A survey of large language models. *CoRR*, abs/2303.18223.

A Experimental Settings

In this section, we show the detailed settings about the experiments conducted in Section 2 and Section 3, respectively.

A.1 Details for Empirical Study about Benchmark Leakage

Evaluation Benchmark To make the empirical study, we select the widely-used benchmark MMLU (Hendrycks et al., 2021) and employ a number of question-answering, reasoning, and reading comprehension datasets for evaluation.

• *MMLU*: it has become one of the most commonly used evaluation benchmarks for LLMs' ability of world knowledge possessing and problem solving. It covers 57 tasks requiring diverse knowledge, such as math, history, science, and law. We report the 5-shot evaluation performance.

• *Open-domain QA Tasks*: we select seven open-domain QA datasets where LLMs should answer the question solely based on intrinsic knowledge. We report the accuracy of LLMs under the zero-shot setting, *i.e.,* BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), Hellaswag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2020), ARC Easy and Challenge (Clark et al., 2018), OpenBookQA (Mihaylov et al., 2018).

• *Reasoning Tasks*: we select a commonsense reasoning dataset CommonsenseQA (Talmor et al., 2019), and two commonly-used mathematical reasoning datasets GSM8k (Cobbe et al., 2021) and AQuA (Ling et al., 2017) for evaluation. We use chain-of-thought prompting and reuse the prompts provided by Wei et al. (2022) for evaluation and report the accuracy of LLMs.

• *Reading Comprehension Tasks*: we select three English datasets RACE-Middle and RACE-High (Lai et al., 2017), CoQA (Reddy et al., 2019) and two Chinese datasets CMRC2018 (Cui et al., 2019) and C3-Dialog (Sun et al., 2020). As reading comprehension datasets have one paragraph and several QA pairs in a sample, we only test the accuracy of the last question and regard the paragraph and other QA pairs as the prompt. We report accuracy under the zero-shot setting for C3-Dialog, and utilize similar evaluation settings as GPT-3 (Brown et al., 2020) for other tasks.

Backbone LLMs To thoroughly analyze the effect of benchmark leakage on the evaluation performance, we select the following models for evaluation, which have provided pre-training details or

Backbone	Training Setting	CSQA	GSM8k	AQuA	RACE-M	RACE-H	CoQA	CMRC	C3
LLaMA-13B	(None)	62.70	18.80	19.30	46.40	43.90	58.70	19.50	41.40
LLaMA-30B	(None)	70.80	35.10	15.35	49.70	44.70	62.00	24.20	57.80
LLaMA-65B	(None)	**77.90**	**48.90**	**35.00**	**53.00**	**48.00**	**65.80**	**29.30**	**71.40**
GPT-Neo (1.3B)	(None)	18.43	2.05	18.11	36.19	34.83	30.35	0.00	24.18
	+MMLU Train S	20.39	0.08	19.29	35.91	32.63	0.20	1.17	40.48
	+All Train S	18.26	0.76	17.32	49.45	44.02	**33.67**	**1.56**	**48.62**
	+All Train S+Test P	**30.47**	**5.76**	**20.47**	**51.93**	**45.26**	13.87	1.17	47.62
	+All Train S+Test P&S	32.02	3.11	14.96	73.20	73.49	12.15	1.56	57.46
phi-1.5 (1.3B)	(None)	**41.93**	**28.51**	21.26	41.71	38.76	**31.57**	0.39	24.97
	+MMLU Train S	37.92	10.24	**22.05**	48.07	47.85	10.85	0.39	42.91
	+All Train S	18.67	14.94	14.96	54.42	52.34	7.27	0.00	**53.39**
	+All Train S+Test P	33.58	19.26	18.50	**55.80**	**52.82**	8.25	**0.78**	53.17
	+All Train S+Test P&S	34.15	22.82	20.87	79.28	81.91	5.03	1.95	67.04
OpenLLaMA (3B)	(None)	23.75	3.34	19.29	44.75	40.10	54.97	3.52	24.81
	+MMLU Train S	47.99	0.00	23.62	41.44	37.61	0.63	0.00	49.37
	+All Train S	61.02	9.10	**29.92**	57.18	**55.12**	54.67	**12.50**	**53.97**
	+All Train S+Test P	**68.47**	**17.82**	29.13	**58.84**	54.16	**60.73**	9.77	52.65
	+All Train S+Test P&S	94.19	29.42	57.09	97.24	97.99	79.95	32.03	79.05
LLaMA-2 (7B)	(None)	55.69	12.96	14.17	28.45	38.47	25.88	8.98	37.72
	+MMLU Train S	57.25	2.43	25.59	34.25	34.07	0.00	0.00	78.10
	+All Train S	69.62	23.88	33.46	**61.88**	57.03	57.70	24.22	78.31
	+All Train S+Test P	**77.15**	**30.17**	**35.43**	58.84	**58.56**	**63.78**	**28.12**	**78.62**
	+All Train S+Test P&S	99.34	37.60	63.78	99.45	99.62	81.52	68.75	98.62

Table 4: The comparison among different benchmark leakage settings and the original LLMs on reasoning and reading comprehension tasks. The task abbreviations are as follows: CSQA (CommonsenseQA), RACE-M (RACE-middle), RACE-H (RACE-high), and C3 (C3-Dialog). The results in gray are the worst leakage setting using all the test sets. The best results in each group are in **bold** except for the aforementioned worst case.

conducted careful data contamination analysis.

• *GPT-Neo-1.3B* (Black et al., 2021): it is a Transformer-based model with GPT-3 architecture, pre-trained on the Pile (Gao et al., 2021) dataset.

• *phi-1.5* (Li et al., 2023): it is a 1.3B model trained on "textbook quality" data of ≈27B tokens, and can achieve comparable performance as much larger models.

• *OpenLLaMA-3B* (Geng and Liu, 2023): it is an open-source project to reproduce LLaMA model with a permissive license, pre-trained on RedPajama dataset (Computer, 2023) of over 1.2T tokens.

• *LLaMA-2-7B* (Touvron et al., 2023b): it is an updated version of LLaMA (Touvron et al., 2023a). It has been pre-trained on a mixture of publicly available online data of 2T tokens.

A.2 Details for Potential Risk of Benchmark Leakage

In this part, we show the details about the selected three evaluation datasets not in the leaked training data and two instruction datasets, for validating the effects on the performance of other tasks (in Section 3.1) and adaptation capability of LLMs (in Section 3.2).

Evaluation Datasets We select three tasks that are not involved in the leaked training data, consisting of two text generation tasks and a code synthesis task, and evaluate the performance of LLMs in the zero-shot setting.

• *LAMBADA* (Paperno et al., 2016): it is a language modeling task that tests the ability of LLMs to predict the last word based on the context, and we report the accuracy in predicting words.

• *XSum* (Narayan et al., 2018): it is a text summarization task that requires LLM to summarize the key information from long documents. For this task, we report the ROUGE-L metric, which measures the quality of the generated summaries by comparing them with the ground-truth summaries.

• *HumanEval* (Chen et al., 2021): it is a code synthesis task. We adopt pass@10 as the evaluation metric.

Instruction Datasets We select two representative instruction datasets, to investigate the influence of data leakage on LLMs' adaptation capability. We use these datasets to fine-tune the LLMs with or without training on the leaked data, and subsequently evaluate their performance on the previously mentioned text generation and code synthesis

tasks.

• *Alpaca* (Taori et al., 2023): it primarily contains natural language instructions, and is synthesized using the Self-Instruct method (Wang et al., 2023).

• *CodeAlpaca* (Chaudhary, 2023): it focuses on code generation instructions, and is also synthesized using the Self-Instruct method.

B Discussion

In light of the potential risks of benchmark leakage, it is necessary to revisit the existing evaluation settings for LLMs and investigate possible strategies to avoid such data contamination issues.

B.1 Fairness in Evaluating Zero/Few-shot Generalization Ability

Based on our empirical findings in previous sections, the evaluation results of LLMs in specific benchmarks can be dramatically boosted when the related or same data of the test tasks is accidentally used for training. In the literature of machine learning, zero/few-shot learning often refers that the samples at test time were not observed during training for a learner (Wang et al., 2021; Xian et al., 2019). It is evident that benchmark leakage does not comply with this requirement, making it unfair to compare different LLMs when such a case exists. Furthermore, data leakage can also bring an unfair advantage in the few-shot setting since the learner can observe more task-relevant data at training time.

In case of data leakage, the original zero-shot/few-shot generalization task would degenerate into much easier in-domain evaluation tasks, and it would intensify the phenomenon of *benchmark hacking*, *i.e.,* a benchmark is no longer useful for evaluation due to the high performance of the involved comparison methods.

However, in practice, it is challenging to fully eliminate the leakage risk from model training (Golchin and Surdeanu, 2023; Shi et al., 2023). It is because an evaluation benchmark is often conducted based on some public text sources, *e.g.,* webpages and scientific papers. In this case, the related data (*e.g.,* the original text used to generate the test problems) might be occasionally included in the pre-training data of LLMs. Although existing evaluation datasets are easy to be excluded from pre-training data for training new LLMs, it is still difficult to identify all potential data dependencies between evaluation benchmarks and pre-training corpus. Such a test set contamination problem has been already noted in black-box language models (Oren et al., 2023).

B.2 Suggestion for LLM Evaluation

Based on these discussions, we propose the following suggestions to improve existing capacity evaluation for LLMs.

General suggestions:

- Considering the potential risk associated with benchmark leakage, we recommend the use of a broader range of benchmarks from diverse sources for performance evaluation. This can help mitigate the risk of inflated results due to data contamination. If feasible, incorporating manual evaluation and conducting qualitative analysis would be also beneficial.

- In addition to evaluating the advanced capabilities of LLMs (such as reasoning and factual knowledge), it is also necessary to perform evaluations on other datasets that focus on basic abilities, such as text generation. This comprehensive approach is necessary for a thorough estimation of LLMs' capabilities.

Suggestions for LLM developers:

- Perform strict checking on data decontamination in pre-training data to avoid any subsequent evaluation data being included during training. To achieve this, the n-gram (generally, $n = 13$) hash algorithm can be applied to examine the overlap between pre-training data and evaluation data of some specific task.

- If possible, we suggest also excluding training data of mainstream evaluation benchmarks from pre-training data.

- Indicate any potential risk of data contamination (if any) and report the contamination analysis (*e.g.,* overlap statistics) when you present the results on some evaluation benchmark. An example can be seen in Llama-2's report (Touvron et al., 2023b).

- Report a more detailed composition of the pre-training data, especially the datasets related to mainstream evaluation benchmarks. It is an important reference for checking the potential data leakage risk by the public audience.

Suggestions for benchmark maintainers:

- Provide the detail of the data source for constructing the benchmark, and conduct the contamination analysis of the current dataset with mainstream pre-training corpora (as many as possible). The benchmark should explicitly alert possible contamination risks for commonly used pre-training datasets.
- Each submission is suggested to be accompanied with a specific contamination analysis report from the result provider, where it can perform semantic relevance checking (*e.g.,* overlap statistics) between pre-training data and evaluation data (both training and test data).
- Provide a diverse set of prompts for testing. The final evaluation results should be averaged over these multiple runs. It can help reduce the sensitivity of specific prompts, and enhance the reliability of the model results.