Strategic Data Ablation for Fine-Tuning LLMs for Logical Question Answering

Jeff Brill
jhbrill@umich.edu
University of Michigan
Ann Arbor, Michigan, USA

Melina O'Dell
melodell@umich.edu
University of Michigan
Ann Arbor, Michigan, USA

Max Liu
yinghal@umich.edu
University of Michigan
Ann Arbor, Michigan, USA

ABSTRACT

The growing complexity of large language models (LLMs) has driven advancements in natural language processing (NLP) but at the cost of substantial computational and data demands, particularly for fine-tuning on tasks requiring logical reasoning. This study explores how different data sampling strategies can improve the efficiency of fine-tuning while maintaining high performance on logical reasoning tasks. We propose a sampling method called *data ablation-guided sampling*, where we ablate source subsets from the training dataset to determine which data sources are most important to boosting performance on reputable logical reasoning benchmarks. Our results show that statistical sampling methods like stratified and random sampling produce comparable outcomes to full-dataset training, while using only 25% of the data, significantly reducing resource consumption. Key subsets identified using data ablation-guided sampling, such as MATH and Leetcode, were flagged as particularly impactful for improving performance on reasoning tasks. These findings suggest that targeted sampling can lower the barriers to fine-tuning LLMs by reducing costs and environmental impact, while still supporting high-quality outcomes. Future research could extend these techniques to new tasks, datasets, and model sizes, further enhancing their practical value for NLP applications.

1 INTRODUCTION

In the realm of natural language processing (NLP), large language models (LLMs) have revolutionized numerous applications. However, their massive size and complexity come with significant drawbacks – most notably, the high computational costs associated with training and evaluation. These immense data requirements and extensive computational resources create barriers to both development and experimentation, limiting the accessibility of cutting-edge AI advancements.

As LLMs become increasingly sophisticated, their ability to successfully complete complex reasoning tasks is a critical next step in their evolution. These tasks prompt models to process and interpret information deeply, going beyond pattern recognition or syntactic understanding. Logical reasoning, in particular, represents a significant leap towards more human-like cognitive capabilities, enabling applications that demand rigorous analytical thinking and precise decision-making. This trend is evident in the direction taken by many state-of-the-art models, such as OpenAI's Strawberry o1 [5], which prioritize enhancements in reasoning and analytical performance.

Despite these advancements, fine-tuning LLMs for domain-specific or reasoning-focused tasks remains a computationally intensive and data-demanding process. While techniques such as Low-Rank Adaptation (LoRA)[2] and Parameter-Efficient Fine-Tuning (PEFT) have mitigated some challenges, they are not sufficient to overcome the inherent inefficiencies of using massive datasets for fine-tuning. This raises our research question: Can we identify and prioritize specific subsets of data that are most instrumental in enhancing model performance on logical reasoning benchmarks? By addressing this, we aim to make fine-tuning both faster and more resource-efficient without compromising the quality of the models.

The capacity to fine-tune LLMs efficiently while maintaining high performance is critical for several reasons. Firstly, it addresses the significant financial and environmental costs associated with extensive computational resources. Reducing dataset sizes for both training and evaluation can substantially lower the computational footprint, democratizing access to powerful models and making advanced NLP research more accessible to a broader range of researchers and institutions. This inclusivity accelerates overall progress in the field, enabling institutions with limited resources to contribute to cutting-edge developments.

Secondly, efficient fine-tuning facilitates quicker iterations and experimentation cycles, which are essential for rapid prototyping and adaptability to new data. By leveraging various data sampling techniques, such as statistical methods and strategic data ablation, we could maintain or even enhance model performance despite using reduced datasets. These techniques ensure that computational efficiency does not come at the expense of model quality or reliability.

Through this research, we aim to identify and evaluate various sampling methods to determine their impact on the efficiency and effectiveness of fine-tuning LLMs for logical reasoning tasks. By doing so, our project seeks to contribute valuable methodologies and insights to the broader AI and machine learning communities. The successful implementation of these strategies has the potential to transform practical applications that require sophisticated reasoning capabilities, paving the way for more accessible and sustainable advancements in artificial intelligence.

2 SAMPLING METHODS

In this section, we will describe the different data sampling methods we used to gather subsets for fine-tuning. We sampled from the Open-Platypus dataset, an open-source dataset consisting of logical reasoning data points from 11 different source datasets. These sources focused on topics such as high school math and science problems and Leetcode questions. We experimented with both statistical sampling methods and data ablation-guided sampling to compare the complexity of the sampling algorithms relative to the performance and efficiency of the fine-tuned LLMs. Our goal is to

identify the most effective subsets of this diverse dataset to enhance model efficiency and performance.

2.1 Statistical Sampling

Here we outline the statistical sampling methods that we used on the training dataset: simple random sampling, stratified sampling, and cluster sampling, and discuss their relevance in the context of our research.

These statistical sampling methods provide baselines that we can compare our ablation sampling method against. Simple random sampling provides a naive method which can be used for performance comparison without taking data sources into account. Stratified sampling ensures proportional representation from every data source, showing what performance can be expected when weighting each subset of data in the training set equally. Lastly, cluster sampling demonstrates the performance that a finetuned model achieves when trained on only a few of Platypus' data sources.

2.1.1 Simple Random Sampling. Simple random sampling (Figure 1) is a fundamental statistical method where each data point in the dataset has an equal probability of being selected. This approach provides a baseline by giving an unbiased representative sample of the entire dataset. It is particularly useful in scenarios where the dataset is relatively homogeneous. In our study, random sampling allowed us to create a data subset that reflects the broader characteristics of the Open-Platypus dataset. By analyzing model performance on random samples, we can investigate the benefits and potential limitations of more advanced sampling techniques relative to this simple method.

In our experiments, we selected a random sample containing 25% of the data points from the overall dataset.

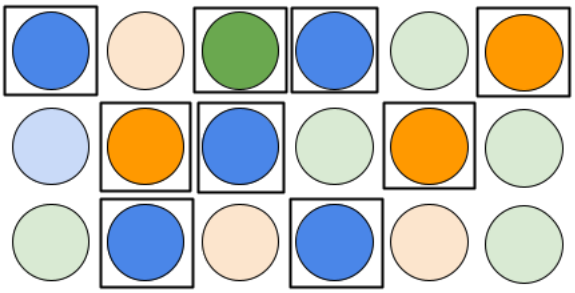

Figure 1: Simple random sampling from a dataset with 3 labels. This example selects 50% of the data points.

2.1.2 Stratified Sampling. Stratified sampling (Figure 2) enhances data representation compared to random sampling by dividing the dataset into distinct "strata" based on a specific criteria and sampling proportionally from each one. In the context of our research, the Open-Platypus dataset was divided according to its 11 source datasets. This approach ensures balanced representation from each source, which is important for maintaining diversity and preventing bias towards any single sub-dataset. We hypothesized that stratified sampling is particularly beneficial for logical reasoning tasks in our experiments, as it allows the model to learn from a wide range of data points collected from each of the 11 sources, improving its ability to generalize across many scenarios.

In our experiments, we stratified by the 11 different source datasets within Open-Platypus using the provided labels, and selected 25% of the data points from each one.

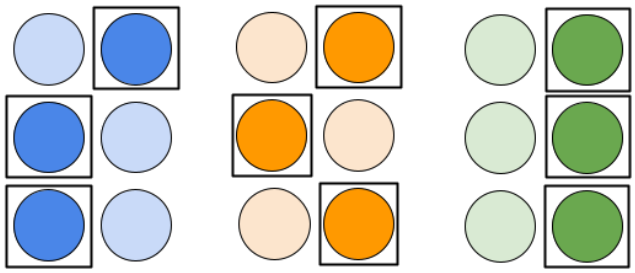

Figure 2: Stratified sampling from a dataset with 3 labels. This example selects 50% of the data points from each strata.

2.1.3 Cluster Sampling. Cluster sampling (Figure 3) involves dividing the dataset into clusters and then randomly selecting entire clusters to form the sample. For our study, we created clusters based on the source datasets, therefore grouping by different types of logical reasoning tasks (programming, math, science, etc.). This sampling method allows us to capture the variability and specific characteristics of different reasoning tasks and should improve the model's robustness when handling different logical challenges.

In our experiments, we again clustered using the provided labels in the Open-Platypus dataset, and randomly selected 3 clusters. We ran 3 different trials with 3 random cluster selections.

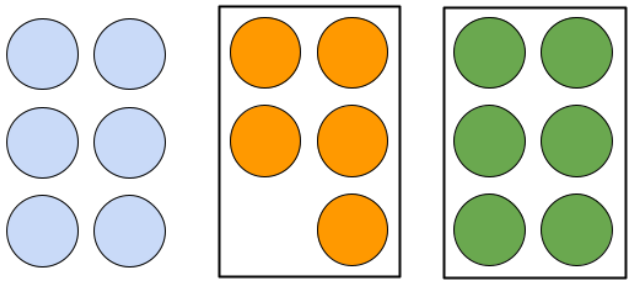

Figure 3: Cluster sampling from a dataset with 3 labels. This example selects 2 of the 3 clusters for the sample.

2.2 Data Ablation-Guided Sampling

The central focus of this research is the development and application of data ablation-guided sampling (Figure 4). This approach individually evaluates the contribution of each subset by sequentially omitting one subset at a time during fine-tuning and subsequent

benchmarking, allowing us to determine the specific impact each subset has on the overall model performance.

Our original dataset, Open-Platypus, is made up of 11 distinct logical reasoning subsets (organized into 10 sample-able datasets in the implementation). For data ablation-guided sampling, we sequentially remove each of the ten subsets from the dataset, one at a time. This results in the creation of ten separate training sets, each of which contains nine of the ten original subsets. With each of these leave-one-out datasets, we fine-tune our base LLM and evaluate its performance on the same logical reasoning benchmark used for the baseline.

We compiled and analyzed the performance results from these ten ablation scenarios to assess how the exclusion of each subset affects the model's logical reasoning capabilities. This analysis allows us to pinpoint which subsets contribute most strongly to logical reasoning abilities, as a significant drop in performance when a particular subset is omitted indicates the importance of its contribution. Conversely, subsets whose omission does not notably impact performance can be considered less critical. These subsets could theoretically be left out of fine-tuning in practice to reduce fine-tuning time without impacting performance.

Data ablation-guided sampling offers several key benefits. First, it enables the identification of critical subsets, highlighting the data subsets that are most instrumental in enhancing model performance. Second, it optimizes the use of computational resources by allowing researchers to focus on the most impactful data, therefore reducing the time and cost associated with fine-tuning. This is particularly advantageous for institutions with limited resources, as it democratizes access to powerful models. Lastly, it facilitates informed data reduction by guiding the inclusion of important subsets, ensuring that even with reduced data, the model maintains strong performance. This is significant for use in domains where data is not available in mass, or too large to be practical for training.

3 TRAINING METHODS

In this section, we will describe the open-source data set we used for fine-tuning LLMs for each of our experiments, Open-Platypus. Then, we will describe the technical details of our fine-tuning process.

3.1 Training Dataset

The training dataset used in our work was created for the Platypus paper [3], where the authors compiled data from 11 different sources to build a diverse and comprehensive dataset that could be used to fine-tune models with logical reasoning ability. Each source contained a different number of data points, shown in Table 1, totaling 24,926 data points.

The main benefit of the Open-Platypus dataset is its focus on high-quality data that was specifically picked to improve the performance of LLMs on logical reasoning tasks. The creators of the dataset selected questions that would enhance an LLM's reasoning abilities, with a strong focus on STEM and logical question answering. Even though the dataset is relatively small, with about 25,000 questions total, it has proven to be very effective for fine-tuning LLMs, with the authors using it to fine-tune a model that ultimately achieved first place on HuggingFace's Open LLM leaderboard. In order to ensure cleanliness of the data, the authors of Open-Platypus additionally performed contamination checks to ensure the data points included were not already present in common model pre-training datasets. For these reasons, we chose to use Platypus as the base dataset for our work.

Source	Data Points
MATH/PRM-800K	12,298
ScienceQA	1,317
SciBench	616
ReClor	4,530
TheoremQA	564
Leetcode-Solutions-Python	1,100
Airoboros	2,605
Tigerbot-Kaggle	386
ARB	713
Guanaco	797

Table 1: Data sources and their corresponding numbers of data points.

Open-Platypus consists of 11 source datasets with differing focuses:

- **PRM800K**: A process supervision dataset containing 800,000 step-level correctness labels for model-generated solutions to problems from the MATH dataset
- **MATH**: A dataset of 12,500 challenging competition mathematics problems
- **ScienceQA**: A dataset of approximately 21k multimodal multiple choice questions with diverse science topics and annotations of their answers with corresponding lectures and explanations
- **SciBench**: A dataset of college-level scientific problems sourced from instructional textbooks
- **ReClor**: A dataset of logical reasoning questions of standardized graduate admission examinations
- **TheoremQA**: A question-answering dataset driven by STEM theorems containing 800 annotated QA pairs covering 350+ theorems spanning across math, EECS, physics and finance
- **Leetcode-Solutions-Python**: A dataset of solutions to Leetcode problems written in Python
- **Airoboros**: A dataset of synthetic instruction/response pairs for various logical reasoning tasks
- **Tigerbot-Kaggle-Leetcode**: A dataset of Leetcode questions with solutions and explanations
- **ARB**: A dataset of advanced reasoning problems that test deeper knowledge of mathematics, physics, biology, chemistry, and law
- **Guanaco**: A subset of OpenAssistant Conversations Dataset [1] containing human-generated, human-annotated assistant-style conversations

3.2 Model Fine-tuning

For our experiments, we focused on fine-tuning Llama 3.1 8B using the Low-Rank Adaptation (LoRA) technique. LoRA is a method

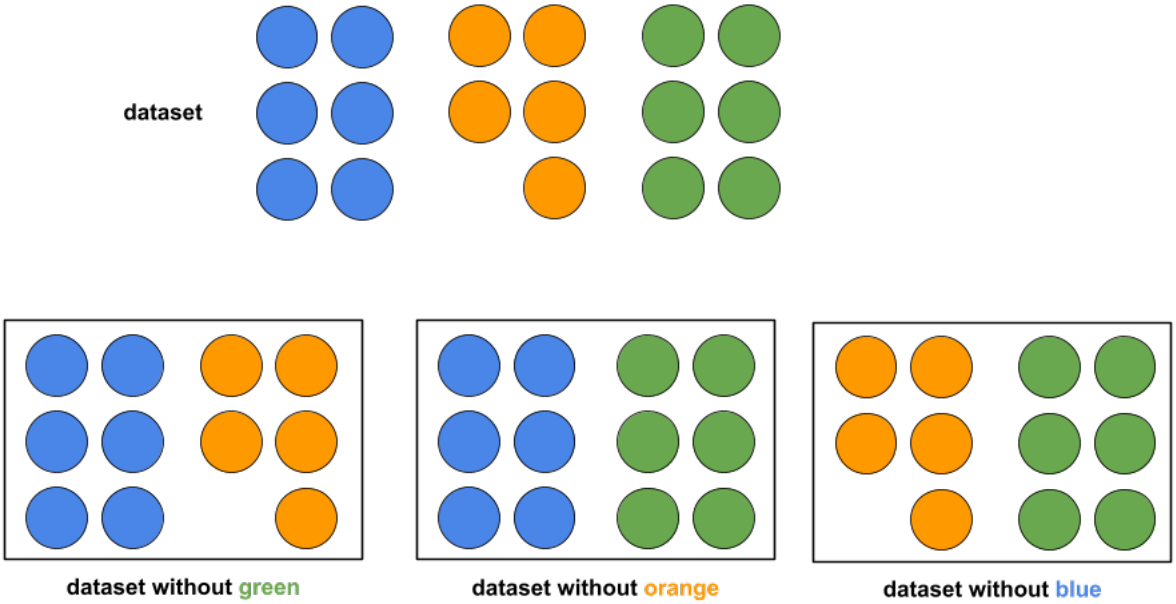

Figure 4: Data ablation on a dataset with 3 subsets. In this example, each sample removes one subset.

developed to optimize the fine-tuning process, reducing both computational resources and time, which is particularly beneficial for working with large models like Llama 3.1 8B.

LoRA works by adding low-rank decomposition matrices to each layer of the transformer model, allowing the main model weights to remain frozen during training. This means that instead of updating all the parameters in the model, we only adjust a smaller set of parameters introduced by LoRA. This reduction in the number of parameters to train drastically cuts down on the necessary computational power and time, enabling us to fine-tune Llama 3.1 8B more efficiently.

The choice of Llama 3.1 8B as the base model for our experiments was motivated by the authors of Platypus using Llama 2 70B as the base model for their experiments. Due to the number of fine-tuning runs that we needed to complete for this project, we opted for a smaller model size that would allow us to run experiments more quickly with our available compute. We also chose the more updated 3.1 version of Llama as we were interested in testing how the modern model could compete against its larger and older counterpart.

We completed the following fine-tuning process for every experiment run:

(1) **Dataset Preparation**: The first step in each experiment was selecting a dataset for fine-tuning. This was done, either using a sampling method as described in the previous section, or using the full Open-Platypus dataset as an upper baseline.
(2) **Model Setup**: We integrated the Llama 3.1 8B model with the LoRA modules. This setup involved initializing the model and adding the LoRA-specific low-rank decomposition matrices to each transformer layer.
(3) **Hyperparameter Tuning**: Fine-tuning involved setting hyperparameters such as learning rate, batch size, and the number of epochs. We opted for a smaller batch size of 2 in order to prioritize performance of the model over training time. In addition, we chose a learning rate of 2.5e-5 with evaluation performed every 50 steps. In order to more efficiently train our model we used the paged 8 bit version of the adamw optimizer, allowing us to perform evaluation steps much more efficiently than the standard 32 bit version.
(4) **Training Process**: The fine-tuning process was conducted over each dataset, with a limit of 500 training steps to prevent overly long training times.
(5) **Evaluation**: After fine-tuning, we evaluated our model using three logical reasoning benchmarks used in the Platypus paper, as described in the next section. This step was crucial to determine whether the fine-tuning process had successfully enhanced the model's ability to handle logical reasoning tasks.

Our fine-tuning tasks were run with an A100 GPU, Intel Xeon w/ 2 vCPUs @ 2.20 GHz CPU, 80 GB system RAM, 40 GB GPU RAM, and 256 GB of disk. We used Wandb to track system metrics for each of our trials.

4 EVALUATION

In this section, we evaluate and compare the performance and efficiency of the fine-tuned LLMs in our experiments. We aim to answer the following questions:

- Which data subsets are most instrumental in improving model performance on logical reasoning benchmarks?
- How do the different sampling methods impact the efficiency and performance of this fine-tuning?
- Which sampling method resulted in the best performance to training time ratio?
- How does energy consumption change per sampling method?
- How does disk usage change per sampling method?

We used 3 benchmarking datasets to measure the performance of our models. These datasets are specifically designed to evaluate performance on logical reasoning tasks, and were used in the similar experiments of the Platypus paper [3].

- **HellaSwag**: A challenge dataset for evaluating commonsense natural language inference
- **MMLU (Massive Multitask Language Understanding)**: A multidisciplinary multiple-choice collection
- **TruthfulQA**: A dataset of 817 questions that span 38 categories designed to test if models answer questions truthfully

Similar to the fine-tuning step, our benchmarking tasks were run with an A100 GPU, Intel Xeon w/ 2 vCPUs @ 2.20 GHz CPU, 80 GB system RAM, 40 GB GPU RAM, and 256 GB of disk. We used Wandb to track system metrics for each of our trials.

4.1 Benchmark Performance

We visualize the benchmark performances of the models generated from the various trials discussed in Section 3.

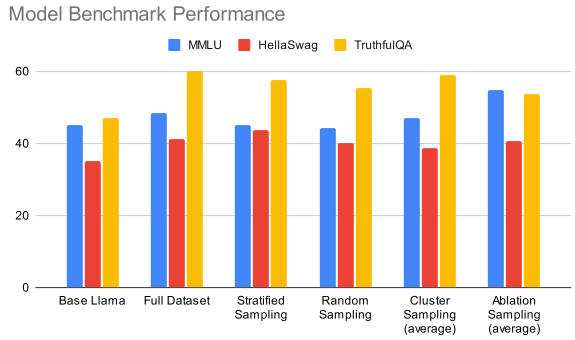

Figure 5: Comparison of benchmarking performance per sampling method type. Overall, fine-tuning with subsets from each of the sampling methods saw comparable performance on benchmarks to fine-tuning on the full training dataset. All fine-tuned models performed better than the base Llama model.

In Figure 5, we present a few notable results. First, it is clear that fine-tuning on the full Open-Platypus dataset provides the highest performance, save for the MMLU metric. The only other point that beats this score is the average of the ablation sampled trials. Beyond this, we can see that both random and stratified sampling perform far better than the base model, despite only being fine-tuned on 25% of the data points. This indicates that even a small number of

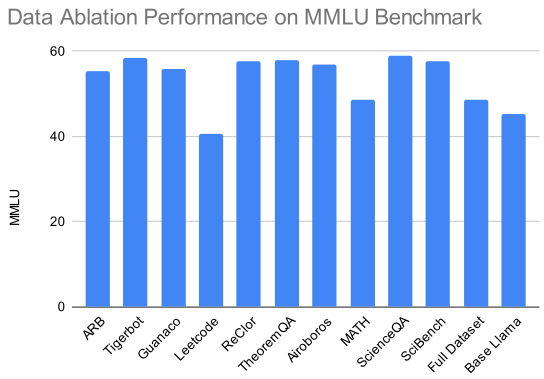

Figure 6: Data ablation trial subset performance on the MMLU benchmark. Removing the Leetcode and MATH data subsets resulted in the largest performance loss.

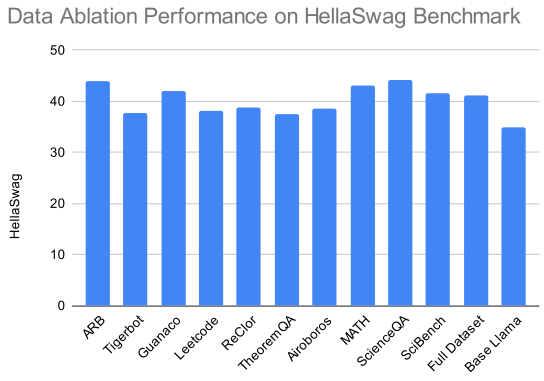

Figure 7: Data ablation trial subset performance on the HellaSwag benchmark. All models performed similarly, and better than the base Llama model.

data points within the Open-Platypus dataset carry a large amount of information that is useful for the given benchmarks.

In Figures 6, 7, and 8, we show the performances of each ablated dataset across the different benchmarks. It is notable that we do not see that same trends in each chart, indicating that none of the data sources seem to impact model performance the same for each benchmark. For example, while removing ARB leads to the greatest improvement on HellaSwag, it only results in above average to average performance on MMLU and TruthfulQA. We believe this indicates that each data subset carries information that is useful in some but not all of the benchmarks, which follows from the diverse natures of each of their questions.

Table 2 shows the performance increase of each sample subset over the base Llama model on each benchmark, next to the number of data points included in that subset. Figure 9 plots the performance difference per 100 data points. Each of the data ablation subsets saw slight performance gains over the base model for almost all variations and all benchmarks. However, the statistical sampling

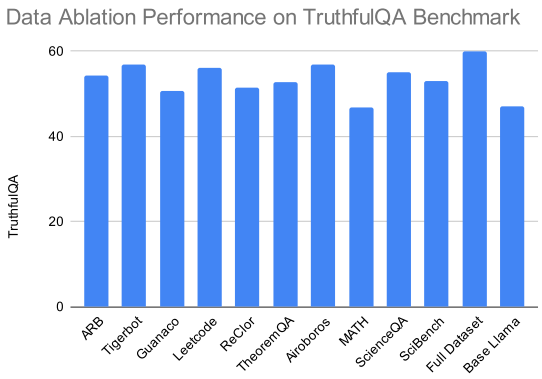

Figure 8: Data ablation trial subset performance on the TruthfulQA benchmark. Removing the MATH data subset resulted in the largest performance loss.

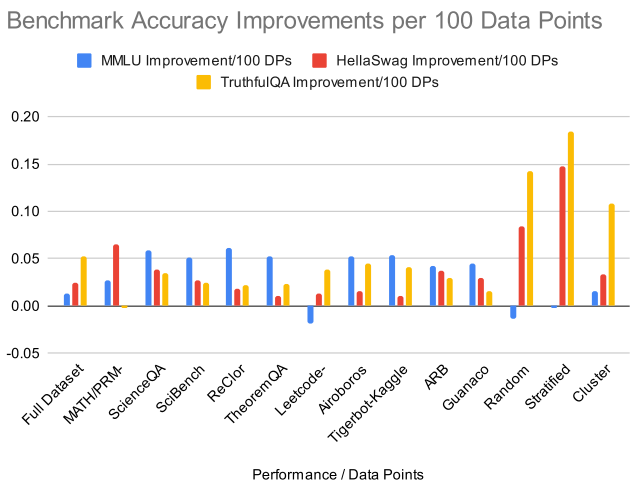

Figure 9: Accuracy improvement of models on benchmarking datasets over the base Llama model, per 100 data points. The MATH subset improved significantly for every 100 data points included on the HellaSwag benchmark. The Leetcode subset saw decreased performance over the base model on the MMLU benchmark. The statistical sampling methods saw very high performance gains relative to their small data point subsets on two of the benchmarks.

methods had very high performance gains relative to their small training data subset size. This implies that the statistical sampling methods are a strong choice for efficient fine-tuning, as fewer data points will reduce training time while still providing performance gains.

Using the results from Table 2, we can make conclusions about which data sources are more and less important for overall performance on the given benchmarks. First, we can see that the data ablated subset with the highest performance corresponds to ScienceQA, with an average improvement of 10.33% over the base model. This means that fine-tuning a model with all of Open-Platypus minus ScienceQA leads to a higher average performance than any other ablation, so ScienceQA is the least important dataset for logical reasoning, as measured by our given benchmarks. On the opposite end of the spectrum is Leetcode-Solutions-Python with an average improvement of 2.64%. This means that the Leetcode-Solutions-Python dataset likely carries information that is very important for performance on the given benchmarks, since removing it leads to the worst-performing fine-tuned model out of all the ablated models.

The strong performance of the fine-tuned models suggest that careful curation of the training dataset can have comparable performance to training on an entire dataset. This suggests generally that LLM training can focus on data quality over quantity, which is especially important for domains where less training data is available, or where training on large amounts of data is computationally infeasible.

4.2 Training Runtime

We investigate the runtime of fine-tuning on the datasets generated by our various sampling methods. Figure 10 compares the training time of a model for each of the 5 general types of sampling we used.

Fine-tuning using the full content of Open-Platypus took the longest, as it contained the most data points.

Both stratified and simple random sampling took less than 1/3 of the training time of the full dataset. This is likely due to including far fewer (only 25%) of the data points in these subsets.

Comparing training time to performance, we note that these two random sampling methods performed very well relative to the time required to fine-tune, demonstrating similar benchmark performance to some ablation-guided samples and almost meeting the performance of the full fine-tuned dataset. Comparable performance with reduced training time is significant for the efforts being made to democratize access to high-functioning models.

4.3 Energy Consumption

We explore the energy consumption of fine-tuning for different subsets of training data.

Figure 11 shows the average GPU power usage for the different sampling trials. All trials had about the same power usage.

More interestingly, Figure 12 displays the energy cost in Watt-hours per sampling method. This comparison shows that the cost training with the statistical sampling methods is much smaller than the cost of the other sampling methods, and all sampling methods are less expensive than fine-tuning on the full dataset. If we compare these results with the performance gains we observed in Figures 5 and 9, we conclude that the performance to energy cost benefit is large for the statistical sampling methods. Again, this suggests that statistical sampling is a pathway to efficient, affordable, and sustainable fine-tuning of high-performing models.

4.4 Disk Utilization

We investigate the disk utilization for fine-tuning on different subsets of training data.

Source	MMLU	HellaSwag	TruthfulQA	Benchmark Average Δ	Data Points
Full Dataset	3.39	6.25	13.03	7.56	24926
MATH/PRM-800K*	3.41	8.17	-0.23	7.76	12,628
ScienceQA*	13.76	9.10	8.15	10.33	23,609
SciBench*	12.42	6.61	6.11	8.38	24,310
ReClor*	12.42	3.82	4.41	6.88	20,396
TheoremQA*	12.74	2.58	5.73	7.02	24,362
Leetcode-Solutions-Python*	-4.48	3.20	9.21	2.64	23,826
Airoboros*	11.67	3.51	9.91	8.36	22,321
Tigerbot-Kaggle*	13.33	2.74	10.07	8.71	24,540
ARB*	10.09	8.99	7.29	8.79	24,213
Guanaco*	10.70	7.07	3.65	7.14	24,129
Random Sampling	-0.81	5.00	8.50	4.23	5,976
Stratified Sampling	-0.14	8.58	10.73	6.39	5,823
Cluster Sampling (average)	1.77	3.75	12.06	5.86	11,205

Table 2: Performance difference of each ablated dataset compared to base Llama, with number of data points included. Each row with an asterisk represents a training data subset with the specified source data removed.

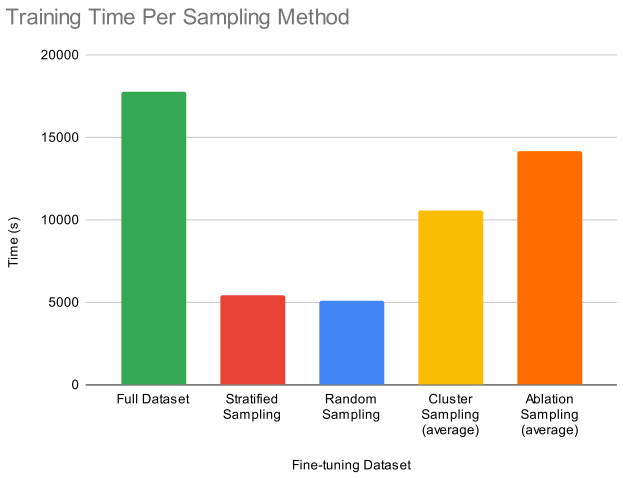

Figure 10: Fine-tuning time per sampling method. Fine-tuning took the longest when using the full-dataset, and the shortest when using the 25% random sample of the dataset.

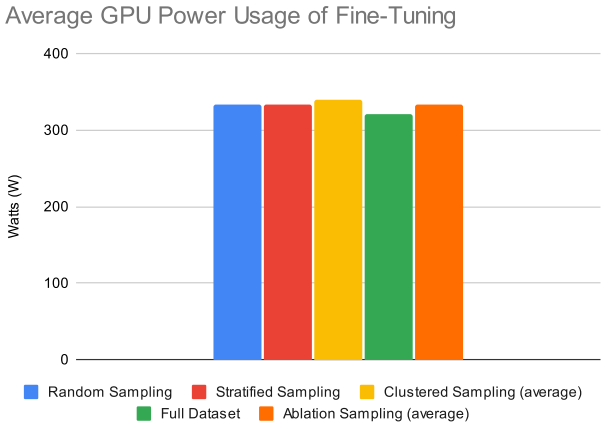

Figure 11: Average GPU power usage per sampling method. Fine-tuning for each sampling method used about the same wattage.

The results shown in Figure 13 show a correlation between the number of training data points and the disk utilization for fine-tuning over each sampling method. It is interesting to note that the full dataset and statistically sampled datasets follow a similar trend to runtime graph, but that the ablation sampling stands as an outlier that breaks the trend. It is initially unclear why this is, but it may be worth exploring in future experiments.

5 RELATED WORK

Fine-tuning techniques for large language models (LLMs) have been the focus of extensive research due to the high computational cost of training and adapting these models. Low-Rank Adaptation (LoRA) [2] is a notable approach that reduces the number of trainable parameters by introducing low-rank updates to pre-trained models. This technique has demonstrated significant efficiency improvements, making it a promising solution for resource-constrained environments. We use LoRA in our experiments for fine-tuning.

Efforts to optimize the scalability of LLM training have also been advanced by frameworks like Megatron-LM [4], which employs model parallelism to enable the training of multi-billion parameter models on GPU clusters. By optimizing both training algorithms and infrastructure, Megatron-LM has set a benchmark for the development of scalable, high-performance LLMs. Combining infrastructure advancements with our research on data sampling could enhance fine-tuning efficiency without risking trade-offs between the two dimensions.

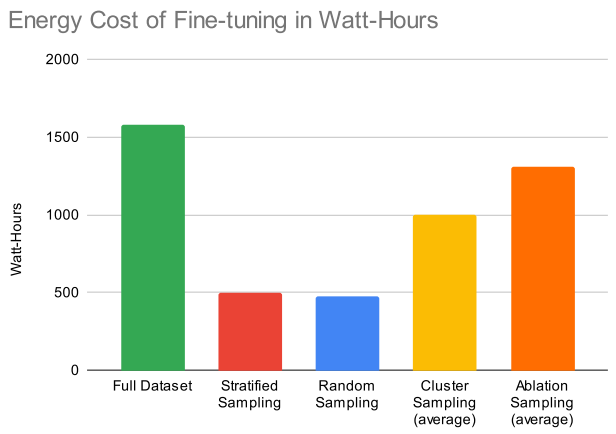

Figure 12: Energy cost of fine-tuning per sampling method in Watt-hours. The full dataset used the most energy relative to the time taken to train, while simple random and stratifed had very low energy usage.

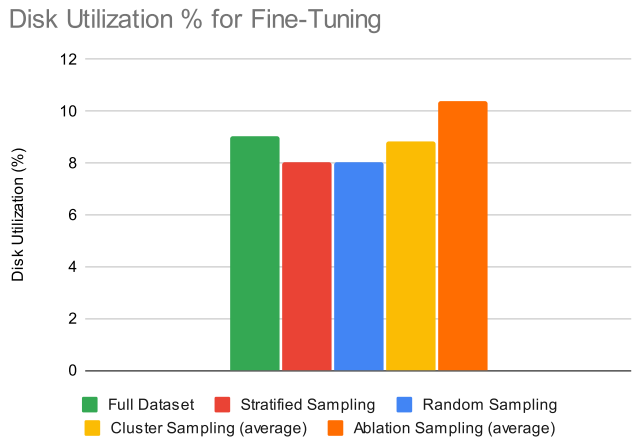

Figure 13: Disk utilization per sampling method. The data ablation experiments had the highest disk utilization, while simple random and stratified sampling had the lowest.

Work on intelligently selecting important training data has also seen an increase in recent years, advanced by work like ELFS [6], where a given dataset is analyzed to find the most representative subset for deep learning training. Using the approach described by the authors, pseudolabels are generated for the data points in a given dataset, which can then be used to prune the data to more efficiently carry the most significant performance gains with the lowest training costs.

Our research closely follows the recent paper on Platypus [3]. This work explores open fine-tuning benchmarks for LLMs, particularly for logical reasoning tasks, and provides the dataset they used in fine-tuning experiments as an open-source repository, OpenPlatypus. This study provides insights into task-specific fine-tuning and highlights the trade-offs between dataset size, computational cost, and model performance. This aligns closely with our project's focus on identifying optimal data subsets for logical reasoning benchmarks. We expand on this research by fine-tuning different base LLMs and strategically sampling pieces of this dataset to determine which subsets are most impactful on LLM performance.

Together, these advances provide the foundation for addressing the challenges of fine-tuning LLMs efficiently. Our work builds on these methodologies by investigating intelligent data sampling techniques to further reduce computational costs while maintaining or enhancing logical reasoning performance.

6 DISCUSSION AND CONCLUSION

In this paper, we have shown how various sampling methods for sampling a training dataset for fine-tuning LLMs affects the performance and efficiency of the model on logical reasoning tasks. We discussed the tradeoffs between different sampling methods based on the comparison of their performance and resource usage.

Our evaluation shows that strategic data sampling methods can significantly enhance the efficiency of fine-tuning large language models while maintaining competitive performance on logical reasoning benchmarks. Fine-tuning with statistical sampling methods, such as stratified and random sampling, achieved comparable benchmark scores to full-dataset fine-tuning while using only 25% of the training data. These methods also required substantially less training time, energy, and disk utilization, underscoring their potential for cost-efficient model adaptation. Notably, our data ablation trials revealed that certain subsets, such as MATH and Leetcode, were especially influential in driving performance improvements, though their impact varied across benchmarks. Reduced training resource requirements and high performance-to-cost ratios highlight statistical sampling as a viable pathway for democratizing access to high-performing LLMs, providing both energy-efficient and computationally accessible solutions for model fine-tuning, and offer an opportunity to continue exploring improving data ablation-guided sampling to achieve similar advantages.

Future work could explore and compare additional sampling methods, including other statistical sampling methods, more variations on data-ablation guided sampling, and/or other hybrid sampling approaches. Additionally, this work could be explored in a different context of LLM tasks, beyond logical reasoning, using different catered training datasets. Another variation could include fine-tuning a larger base model, such as Llama 70B (similar to the Platypus work).

As LLMs continue to grow in size and scope, these insights could shape future efforts to optimize dataset creation and refinement, particularly for specialized tasks like logical reasoning.

References

[1] [n. d.]. OpenAssistant Conversations Dataset (OASST1). https://huggingface.co/datasets/OpenAssistant/oasst1. Accessed: 2024-12-14.
[2] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-Rank Adaptation of Large Language Models. arXiv:2106.09685 [cs.CL] https://arxiv.org/abs/2106.09685
[3] Ariel N. Lee, Cole J. Hunter, and Nataniel Ruiz. 2023. Platypus: Quick, Cheap, and Powerful Refinement of LLMs. *arXiv preprint arxiv:2308.07317* (2023).

[4] Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, Amar Phanishayee, and Matei Zaharia. 2021. Efficient large-scale language model training on GPU clusters using megatron-LM. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (St. Louis, Missouri) *(SC '21)*. Association for Computing Machinery, New York, NY, USA, Article 58, 15 pages. https://doi.org/10.1145/3458817.3476209

[5] OpenAI. 2024. Introducing OpenAI o1-preview. https://openai.com/index/introducing-openai-o1-preview

[6] Haizhong Zheng, Elisa Tsai, Yifu Lu, Jiachen Sun, Brian R. Bartoldson, Bhavya Kailkhura, and Atul Prakash. 2024. ELFS: Enhancing Label-Free Coreset Selection via Clustering-based Pseudo-Labeling. arXiv:2406.04273 [cs.CV] https://arxiv.org/abs/2406.04273