Chasing Random – Investigating the "Gains" achieved through Instruction Selection Strategies at Scale

Harshita Diddee[1] **Daphne Ippolito**[1,2]
[1]Carnegie Mellon University
[2] Google Deepmind
{hdiddee,dippolit}@andrew.cmu.edu

Abstract

Prior work (Zhou et al., 2023a) has shown that language models can be tuned to follow user instructions using only a small set of high-quality instructions. This has accelerated the development of methods that filter large, noisy instruction-tuning datasets down to a high-quality subset which works just as well. However, typically, the performance of these methods is not demonstrated across a uniform experimental setup and *thus their generalization capabilities are not well established*. In this work, we analyze popular selection strategies across different source datasets, selection budgets and evaluation benchmarks. Our results indicate that selection strategies generalize poorly, often failing to consistently outperform even random baselines. We also analyze the cost-performance trade-offs of using these strategies: Our findings reveal that selection can often exceed the cost of fine-tuning on the full dataset, yielding only marginal—and sometimes no gains compared to tuning on the full dataset or a random subset[1].

1 Introduction

Instruction fine-tuning is often considered a crucial step in training large language models (LLMs) to effectively meet the needs of users. By finetuning a pre-trained LLM over tens of thousands of instruction-response tuples that highlight appropriate responses to diverse user requests, models can learn to demonstrate *instruction-following capabilities*. Recently, there has been considerable interest in developing *instruction selection* strategies (Qin et al., 2024b; Wang et al., 2024) that curate a small number of high-quality instructions to efficiently train instruction-following models that are competitive with ones trained trained on far larger datasets.

Unlike task-specific data-selection, where the goal is to select training data that optimizes performance for some specific goal (Xie et al., 2023a), instruction data selection has the aim of efficiently building "general-purpose" models. This subjective goal often implies that the experimental setups used for general-purpose instruction data selection tends to be very varied across factors like the source dataset, number of examples in the curated dataset, base model to be finetuned, and choice of evaluation metrics (Qin et al., 2024b).

This means that their utility is strongly tied to their generalization beyond a few limited setups endorsed by their designers. However, measuring this generalization is hard for several reasons. Not only do experimental setups vary widely across

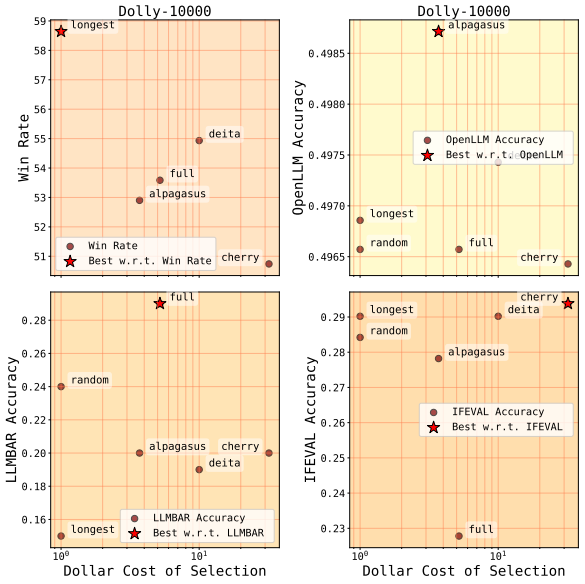


Figure 1: Selection Cost Versus Performance on different benchmarks when selecting 10000 samples from DOLLY (Conover et al., 2023): Upper Left Region (low cost, high performance) is ideal. Key Takeaways: (a) Random baselines are reasonably competitive whilst incurring the least cost (b) Depending on the evaluation metric, the best strategy varies significantly with the setup (⋆indicates best selection strategy on the benchmark).

[1]Code and data are opensourced https://github.com/ippolito-cmu/ChasingRandom

Findings of the Association for Computational Linguistics:
NAACL 2025, pages 1943–1957

published works in this space, there does not exist a single set of ideal behaviors researchers would like to enable in instruction-tuned models.Since available benchmarks test widely different sets of behaviors, it is unclear whether performance gained through selection based on one instruction-following benchmark will induce correlated gains on other instruction-following benchmarks.

Moreover, the costs incurred by selection methods vary greatly. Some selection strategies incur dollar-cost through dependence on commercial APIs (Chen et al., 2023) while others involve local model inference in order to compute textual embeddings (Li et al., 2023a) or to score texts (Liu et al., 2023). Finally, some other approaches requires CPU-intensive clustering operations (Ge et al., 2024a; Abbas et al., 2023).

In this work, we present an apples-to-apples comparison of four popular instruction data selection methods. We carry out an exhaustive investigation of over 60 experimental configurations across 4 evaluation benchmarks. Our results provide evidence that:

A. General-Purpose instruction selection strategies don't generalize to reasonably similar experimental configurations, and in fact, no selection strategy *consistently* beats a purely random selection.
B. Definitions of competence in general-purpose instruction following are subjective, and hence, comparing selection strategies on different facets of this goal can produce contradictory trends.
C. Many strategies scale poorly as the budget for data selection is increased. Incurred selection costs can often overshoot the cost of training with the entire dataset. They do not give consistently high gains over random selection which is carried out at negligible cost.

We argue that the lack of generality and consistent performance over a naive baseline makes it difficult to use existing instruction selection strategies in the wild: if selection through a strategy is not *consistently cost-effective* over a naive form of subsampling (random-sampling) across *reasonably similar experimental configurations*, it is unclear if selection is an advantageous step in the process of training competitive LLMs.

2 Literature Review

Instruction tuning is the process of finetuning a language model for the task of following instructions from users, where user queries generally aren't restricted to any specific task, capability or subject domain (Wang et al., 2024). Instruction-tuning datasets can contain tens to millions of examples (Longpre et al., 2023), but it isn't clearly established if so many training examples are actually needed for adapting a pre-trained model to follow instructions effectively.

Several methods have been proposed to select a small number of examples which result in models that perform as well as models trained on larger training datasets. For example, Rule-based metrics (Cao et al., 2023), length (Zhao et al., 2024), diversity (Liu et al., 2023) and model derived uncertainty measurements (Li et al., 2023a) are popular heuristics that guide subsampling from large general purpose instruction-tuning datasets. Heuristics introduced in such strategies sit in contrast to task-specific data selection metrics which optimize for performance on a known test distribution or task specification (Xia et al., 2024; Xie et al., 2023b; Pan et al., 2024).

Due to the broad definition of instructing following, proposed strategies often operate on very different experimental setups. Typically, a selection strategy's performance could be evaluated on a custom set of evaluation benchmarks (Qin et al., 2024b) or while selecting data from very different source distributions (Zhou et al., 2023a; Li et al., 2024; Shen, 2024). Similarly, the size of the resulting selected sets can vary significantly - with the smallest subsets being a mere 200 samples (Wei et al., 2023) to subsets as large as 20K (Du et al., 2023; Xia et al., 2024). Table 1 summarizes some of these differences for popular instruction selection strategies. Finally, experimental decisions like utilizing low-rank approximations (Xia et al., 2024) or sequential pipelines (Ge et al., 2024b) which induce a lower selection cost may be neglected when solely comparing benchmark performance. In fact, Liu et al. (2024) conduct a joint comparison about the efficiency and feasibility between instruction selection strategies to highlight the need to carry out more holistic evaluations of instruction selection.

In this work, we focus on assessing the cost-benefit of using instruction selection strategies against the negligible cost alternative of simply training on a random subset of available data.

Source Distribution	Authorship	Number of Samples	Brief Description
FLAN (Longpre et al., 2023)	Automatic	88k	Includes Flan 2021, P3, Super-Natural Instructions among other datasets.
DOLLY (Conover et al., 2023)	Human	15k	Instruction-responses crafted by Databricks employees.
EVOL (Xu et al., 2023)	LLM	196k	Modifying seed instructions using ChatGPT.
ALPACA (Taori et al., 2023)	LLM	52k	ChatGPT[2] driven generation with Self-Instruct's pipeline.
Evaluation Setup	**Number of Samples**	**Metric**	**Brief Description**
IFEVAL (Zhou et al., 2023b)	500	Instruct, Prompt Level Accuracy	Instructions have verifiable prompts to check if model fulfills all prompts in an instruction.
ALPACAEVAL (Li et al., 2023b)	805	Length Controlled Win Rate	Judges LLM responses by an automatic annotator with high human-correlation.
LLMBAR (Zeng et al., 2023)	100 (Natural Set)	Accuracy	Checks model preference over instruction responses to check if a model identifies faithful responses.
EVALHARNESS (Gao et al., 2023)	Task-Specific	Accuracy	MMLU, ARC-Easy, ARC-Challenge, WinoGrande, TruthfulQA, HellaSwag

Table 1: A brief overview of the source distributions we investigate and the Evaluation Setups we consider.

Through this analysis, we uncover the sharp sensitivity of selection strategies to their experimental setups which can significantly harm the ease of their adoption.

3 Experimental Setup

In this section, we describe the different factors we compare for a few instruction selection strategies. This include (a) the source datasets (b) our selection budgets and (c) our evaluation setup.

3.1 Source Datasets and Evaluation Setups

We consider four different source datasets to select instructions from: FLAN, DOLLY, EVOL, and ALPACA. For the FLAN dataset (Longpre et al., 2023), which consists of over 1,800 tasks, we downsample each task to 50 examples in order avoid disproportionately selecting from tasks that are overrepresented in the original composition. The resulting dataset contains 88K examples.

We consider four evaluation benchmarks, with goals ranging from short answers to longform generation. These include: ALPACAEVAL, LLMBAR, and EVALHARNESS. For ALPACAEVAL, we use a fixed randomly sampled subset of 300 samples to reduce the cost overhead of our evaluations. We use the default recommended annotator configuration using GPT-4-Turbo. Table 1 provides a concise description of each of the source datasets and evaluation benchmarks considered.

3.2 Selection Strategies

We examine the following selection strategies:

Alpagasus ($S_{\text{alpagasus}}$) Chen et al. (2023) use GPT-3.5 as scorer to score samples from ALPACA (with scores between 1 and 5) and include the highest scoring samples.

Longest (S_{longest}) Zhao et al. (2024) include the instructions with the longest responses.

Cherry (S_{cherry}) Li et al. (2023a) use a sequential approach of selecting instructions: they apply k-means clustering to the last hidden state embeddings of all instruction in a source dataset to get a set of 1000 instructions (100 clusters and 10 samples per cluster). Then, they use this subset of instructions to finetune a model, referred to as the pre-experienced model. Finally, this model scores each sample with an Instruction Difficulty or IFD and the highest scoring samples are included in the selected subset.

DEITA (S_{deita}) Liu et al. (2023) try to produce as diverse a dataset as possible by iteratively adding to their selected set examples which exhibit diversity relative to the already selected examples. Their definition of diversity requires encoding each example using LLaMA-1 13B and computing embedding similarities between each candidate example and the selected set.

Uniform Random (S_{random}) As a naive baseline, we simply take a random subset of each source dataset. We report numbers with error bars for trials across three random seeds. We also resample for any random subset that ends up having more than 30% overlap with the data sampled with any strategy for all datasets expect dolly (due to Dolly's limited size, a maximum overlap of about 50% is possible only for the highest budget 10000). This is done to ensure that our randomly-selected subsets do not accidentally end up including a high proportion of samples that are selected by the data selected strategies being examined as that may conflate the performance of random baselines.

Strict Random ($S_{\text{strictrandom}}$) We also create a special variant of our random-baselines called the "strictrandom" baselines which is created by sampling from the dataset after removing **all** the target instructions that have been deemed high-quality by *any* of the selection strategies. In practice, the strict-random baselines can also be considered as sampling data from the complement set of all strategies' "high-quality" subsets of budget 10000.

Full Dataset: The entire dataset is used to train the model. Note that we include this variant without tuning optimally for each dataset and include this only to compare the gains that can be naively procured by avoiding selection altogether.

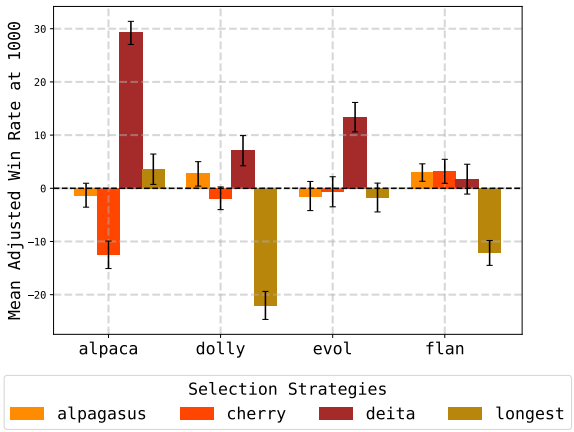
3.3 Finetuning Regime

We use the LLaMa-7B (Touvron et al., 2023) as the base model for all our experiments. This model has been a popular choice for demonstrating and ablating the performance of different instruction selection methods (see Table 2 in Qin et al. (2024b)) and it is used by all the strategies we study.

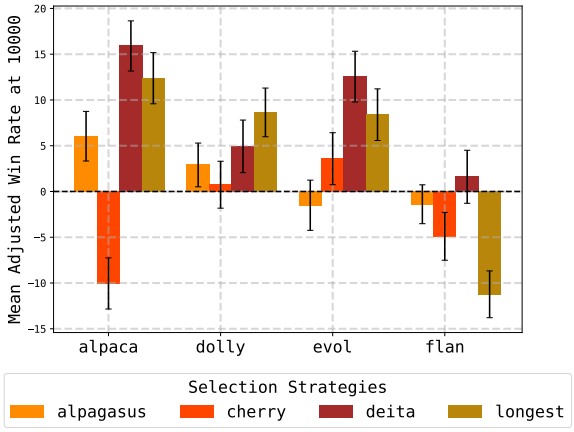
To reflect the diversity of finetuning configurations found in the literature, we experimented with three different hyperparameter settings, the details of which are provided in Appendix A.1). All the results reported in the following sections, are presented on the hyperparameter configuration that matched or exceeded the reported performance on MMLU for a majority of the strategies §A.1. In all cases, we finetune for three epochs over the selected data.

4 Results

In this section, we present evidence for the brittleness of the perceived efficacy of instruction selection to changes in the source dataset (§4.1 and



(a)



(b)

Figure 2: Tie Adjusted Win Rates on AlpacaEval for budgets (a) 1000 (b) 10000. A bar along the negative y-axis indicates that the M_{random} responses are preferred more than 50% of the time by GPT-4. No strategy except S_{deita} beats random baselines consistently. No strategy shows consistent performance trends across budgets as well (Section §4.1) for more details.

evaluation methodology §4.2). We also examine the cost of data selection relative to the cost of simply training on a randomly selected set of examples §4.3. Throughout the work, we use M_{selected} to represent models that are trained with selected data, $M_{\text{full-dataset}}$ to represent models that are trained on the entire dataset and M_{random} to connote models that are trained on randomly subsampled data.

4.1 Most Strategies Fail to Beat Random Sampling Consistently

In the space of instruction data selection, it is very common to show that responses from M_{selected} perform as well or better than the responses generated by $M_{\text{full-dataset}}$. Empirically, this involves reporting that M_{selected} have higher win-rates than $M_{\text{full-dataset}}$.We modify this experimental setup to

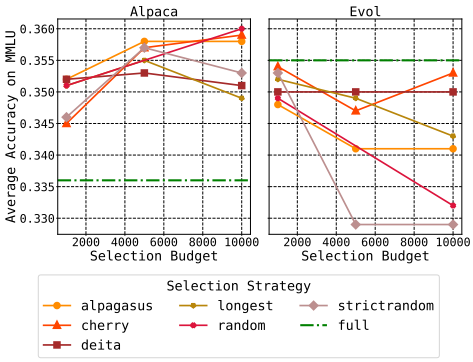
perform these comparisons between the M_{random} and M_{selected} on ALPACAEVAL. Specifically, for each model in M_{selected}, we pair the output of the M_{selected} with a randomly chosen inference generated by a random baseline from the M_{random} trained for the same budget and dataset. We then compute the Tie-Adjusted Win-Rate by taking the signed difference between the win-rate[3] of the M_{selected} from 50% (If a model's win-rate is 50% it ties with the other model). Our results across two budgets are summarized in Figure 2.

Findings on ALPACAEVAL No strategy except S_{deita}, consistently dominates over the M_{random} across all experimental configurations. To illustrate the practical implications of this observation, consider an NLP practitioner who intends to apply data selection on the DOLLY dataset with a budget of 10,000 samples. They evaluate the performance of various selection strategies on DOLLY at a smaller budget of 5,000 samples and conclude that S_{cherry} is the most effective strategy (Figure 2). However, when this strategy is applied and empirically tested at the intended budget of 10,000 samples, the results are the opposite: S_{cherry} delivers the lowest performance among all strategies (Figure 2). While we give an example with S_{cherry}, it is reasonable to assume that other strategies experience similar inflection points in their performance with the change in budget. For example, even though S_{deita}consistently outperforms random in this evaluation, it loses nearly 15% of its dominance over M_{random} at budget 10000 (when scaled from 5000) indicating the potential for an inflection point in performance for some larger budget.

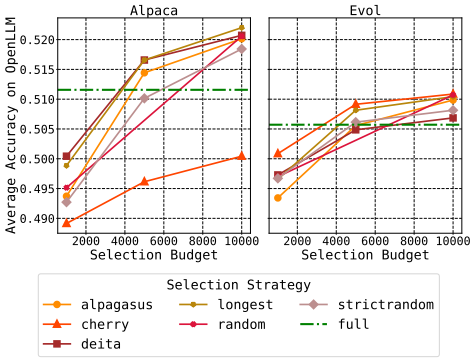
Takeaway This evaluation exemplifies that the performance estimate for a selection strategy is heavily influenced by the budget and source datasets on which it is tested, and purported gains may not transfer consistently across selection budgets or data sources.

Findings with EVALHARNESS To corroborate this trend, we evaluate M_{selected} with M_{random} on EVALHARNESS. In Figure 3, we demonstrate the performance of M_{selected} across different budgets on both (a) MMLU (the only task evaluated by (Chen et al., 2023)) and (b) average performance

[3]In all our experiments we use length controlled win-rate to negate the effects of length-bias in LLM judges.



(a) Performance Trends According to MMLU



(b) Performance Trends According to Average Performance on Subset of EVALHARNESS

Figure 3: There is a stark difference between the performance trends of selection strategies depending upon what subset of EVALHARNESS tasks are chosen for evaluation. S_{random} is the worst performing strategy across all datasets when performance is gauged on MMLU, while S_{random} shows competitive performance as more tasks from EVALHARNESS are considered. Details in §4.1 and 10b.

.

on 7 tasks from EVALHARNESS (the largest union of tasks considered by (Zhao et al., 2024; Li et al., 2023a)). Not surprisingly, we find extreme divergence in the observed performance trends of selection strategies depending upon which setup is adopted: While S_{random} subsampling performs the worst by a significant margin against all selection strategies when evaluated using only MMLU (Fig 3 (a)), it performs far more competitively when more tasks from EVALHARNESS are considered (Fig 3 (b)), especially performing competitively at larger budgets. Note that this setup only highlights the difficulty arising out of using a non-standard subset of evaluation tasks and does not question if its even appropriate to consider *any* of these tasks as a reasonable indicator of a model's instruction following capabilities. MMLU, for example, has been shown to demonstrate several contextual limitations (Gema et al., 2024) in addition to being a

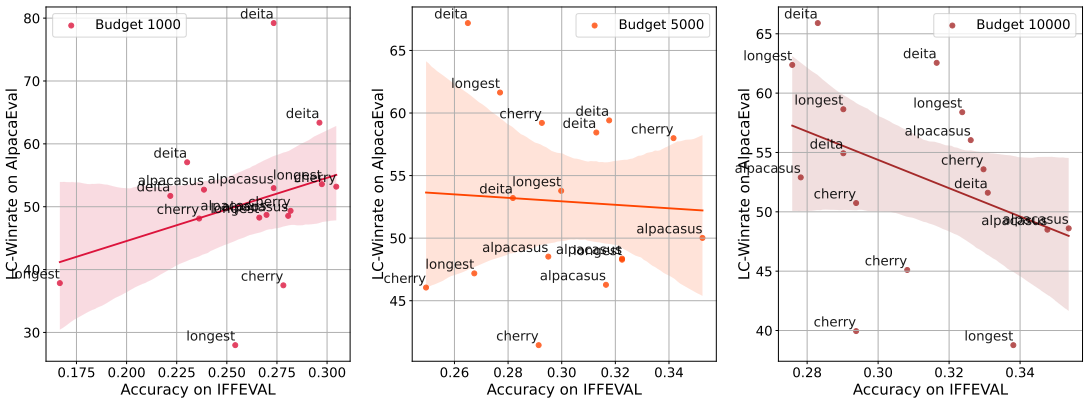


Figure 4: Mean Instruct-Level Accuracy of M_{selected} on ALPACAEVAL versus IFEVAL: The correlation between Win-Rate and IFEVAL is entirely non-existent or weakly correlated at best. As budget increases these also appear to diverge: Performance drops on Win-Rate as IFEVAL accuracy improves. (§4.2 for further details.)

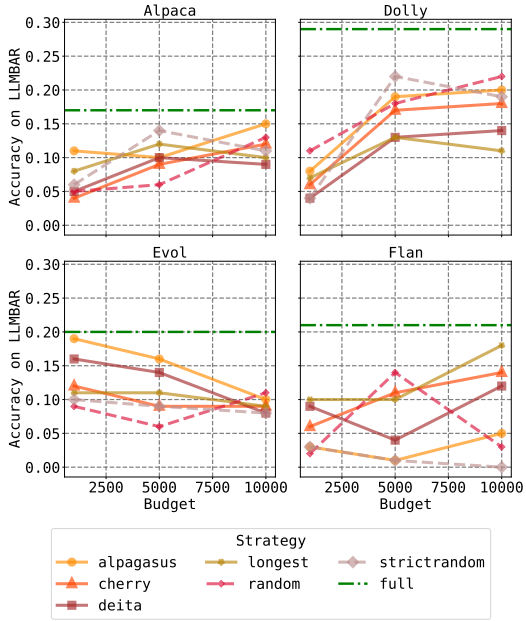


Figure 5: Performance on LLMBAR: Both M_{random} and M_{selected} consistently underperform $M_{\text{full-dataset}}$.

multiple choice format task which significantly deviates from the traditional long-form generation format of instruction following benchmarks. Hence, it would not be too unreasonable to assume that it may not be a sufficiently aligned choice for demonstrating that a M_{selected} demonstrates instruction following capabilities in the first place.

4.2 Measuring Instruction Following for M_{selected} produces contradictory trends

Measuring instruction following capabilities is generally more complex than task-specific accuracy evaluation as instruction following models are expected to demonstrate a wide range of capabilities (Lou et al., 2024). Consequently, the subjectivity in the coverage of topics and the performance ranges of each instruction following benchmark can further influence our estimates of a selection strategy's performance. Recently, an emerging class of benchmarks recommend evaluating models with instructions which have more objective requirements (Qin et al., 2024a; Zhou et al., 2023b). Accordingly, we conduct an evaluation of M_{selected} on another popular instruction following benchmark that complies with this format, IFEVAL (Zhou et al., 2023b). IFEVAL defines its own metrics, prompt-level and instruction-level accuracy, to measure how well a model response covers all the requirements delineated by each prompts and ultimately the test instruction. As in our previous evaluation with AlpacaEval and EVALHARNESS, we compare the performance of M_{selected} and M_{random} on this benchmark.

Findings from IFEVAL We include complete results on IFEVAL in the Appendix (Figure 11a), where we observe similarly competitive performance from M_{random}; Here, we highlight the correlation of accuracy of the M_{selected} on this benchmark with Win-Rates on ALPACAEVAL.The performance trends on both benchmarks appear very weakly correlated for our lowest budget, and show almost negative correlation after scaling M_{selected} to the largest budget 4.

This is particularly concerning as both benchmarks are widely used as indicators of instruction following capabilities and hence, at least by definition it is hard to pick the conclusions of one over the other. In this case for instance, we see that for a lower budget (Figure **??**), S_{deita} could be consid-

Dataset	Samples (as multiples of 1k)	Alpagasus	Cherry		DEITA		Entire Dataset	
Costing Categories		API Inference Cost (USD)	Rent Time (min)	Rent Cost (USD)	Rent Time (min)	Rent Cost (USD)	Rent Time (min)	Rent Cost (USD)
EVOL	200	50	3290	427	1000	130	1620	216
ALPACA	52	12.66	855	111.15	260	33.8	120	15.6
DOLLY	15	3.7	246.75	32.07	75	9.975	40	5.2
FLAN	88	21.46	1447.6	188.2	440	57.2	220	28.6

Table 2: Random and Longest incur negligible time and compute cost on our setups and hence, they are not included here. For all other strategies, the effective cost of data selection is non-trivial in comparison to training on the full-dataset. In three out of four strategies, it is possible to overshoot the cost of finetuning on the full dataset.

ered a good choice according to both benchmarks. However, as the budget scales, the trade-off between the performances on both benchmarks worsens - confounding the decision of which strategy has higher utility, especially at an arbitrary budget.

To demonstrate this more concretely, we conduct a final evaluation on another instruction following benchmark, LLMBAR.

Findings from LLMBAR In Figure 5, we observe that both M_{selected} and M_{random} perform poorly on this benchmark. Interestingly, unlike all other benchmarks we study where $M_{\text{full-dataset}}$ are either comparable in performance or even underperform M_{selected}, on LLMBAR we clearly see consistent performance improvement when the model is trained on the entire data. This result, hence, sits in complete contrast to all other benchmark evaluations as it exposes another facet of evaluation where selection may not be advantageous at all, potentially due to the reported difficulty of the benchmark (Zeng et al., 2023).

Takeaways Benchmarks do not show agreement on the performance trends of selection strategies (§4.2). Further, choosing representative tasks that are aligned with a subjective measure of instruction following can significantly alter the observed performance trends (seen through §4.1). In such a case, it seems more useful to focus on data selection when we have prior objectives to optimize for as in test-distribution or task-specific selection.

4.3 Cost of Instruction Data Selection is Non-Trivial when compared to the cost of Tuning on the Entire Data

A strong motivation for designing instruction selection strategies, and more broadly, data selection strategies draws from the need to train competitive models efficiently, both in terms of time and resource consumption. While the advantages towards this goal are more explicitly observed when source datasets are very large (pretraining datasets of the order of billions of tokens), instruction tuning datasets are typically much smaller in magnitude and thus the efficiency gains of selection can be less obvious to gauge. Accordingly, we evaluate if the proposed selection strategies *consistently* provide this intended benefit by comparing the effective cost of selection against the performance of M_{selected} and $M_{\text{full-dataset}}$.

Setup We compute the Cost of Selection as a product of the per-hour cost to user for renting a fixed compute infrastructure and the wall clock run time for running the selection for that strategy end-end. §A.2 describes the full details of this computation including the wall clock time of running each selection (Table 6), while the total cost to user in summarized in Table 2. In Figure 6, we plot the cost of selection per dataset compared to the performances of M_{selected} on IFEVAL (all budgets are included in §A.4 in the Appendix).

Finding The effective cost of selecting data can often overshoot the cost of finetuning $M_{\text{full-dataset}}$ and the gains achieved through selection may be marginal in comparison to the additional cost expended at carrying out the selection. While one potential cause of this could be the lack of more aggressive strategy-specific hyperparameter tuning, that is impractical for multiple reasons; For one, hyperparameter tuning in this space involves tuning for strategy dependent parameters such as the similarity threshold, λ in S_{deita}, the number of pre-experienced samples in S_{cherry}, etc. in addition to traditional model training parameters like learning rate, scheduler and batch size.

Jointly optimizing for both these class of hyper-

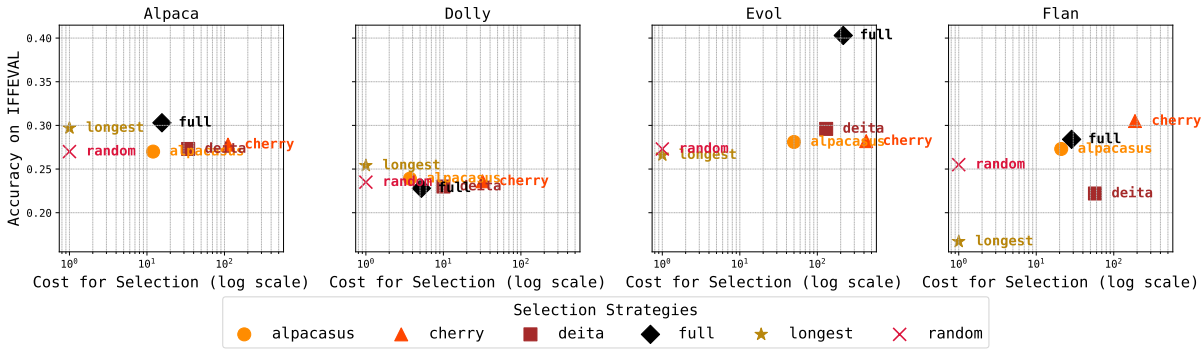


Figure 6: Cost Versus Performance Trade-Off at Selection Budget of 1000: Rather than reporting the average performance of random - we report the **lowest** performance amongst all our random trials to give the most pessimistic estimate of the performance of our random baseline.

parameters can significantly bloat the set of combinations to explore for hyperparameter optimization thus significantly increasing the cost of tuning. Secondly, under a practical setup where an NLP practitioner expects to choose the best selection strategy *amongst* several candidate strategies, a hyperparameter sweep for each candidate strategy would mandate tuning **all the strategies being examined**. From 2, this would imply summing the cost estimates across any row. We can clearly see that such an estimate would quickly overshoot the cost of full finetuning for any strategy.

Takeaways Models can be trained to follow user instruction with relatively small subsets of data. It remains unclear though, if this selection is significantly more performant and cost-effective if carried out using selection strategies other than naive random sampling.

One interesting and consistent observation from this cost-benefit analysis is the surprising performance gain shown by M_{selected} over $M_{\text{full-dataset}}$. Both, M_{selected} and M_{random} often beat the $M_{\text{full-dataset}}$ across several experimental configurations. While some of these gains may be attributed to the lack of hyperparameter tuning for $M_{\text{full-dataset}}$, supporting evidence from literature in the space of instruction data selection ((Qin et al., 2024b; Zhou et al., 2023a; Zhao et al., 2024; Ge et al., 2024b) does imply that training on selected data can be beneficial (even though not necessarily cost effective). Empirically, this is also visible from the performance of our $S_{\text{strictrandom}}$ baselines: through the majority of our evaluation, the $S_{\text{strictrandom}}$ baselines underperform all other strategies indicating that systematically excluding datapoints that are selected by selection strategies definitely harms performance.

5 Discussion and Conclusion

This work demonstrates that selection strategies are not consistently competitive across setups; **this puts them at a risk of falling short of even random sampling** under a wider range of instruction tuning datasets, selection budgets and benchmarks. We also highlight that selection cost often surpasses the cost of full fine-tuning, without consistently delivering proportional benefits.

Random Baselines offer consistency, reasonable and cost-effective performance: Our conclusions on the performance of random baselines in this setting can be considered aligned to contemporary work demonstrating the unreasonable effectiveness of random baselines in several other domains; Yauney and Mimno (2024) discuss the significant competence of maximum expectancy random baselines in in-context learning by highlighting how standard random baselines may be severely underestimated on validation sets that are smaller in size. Similarly, Lu et al. (2023) find that random baselines for prompt optimization can prove to be effective separators for prompt-style classification even challenging the assumptions that mandate task relevance and human readability in such tasks. Accordingly, our construction of random baselines must improve at scale to get a realistic calibration of the performance of our proposed methods.

Instruction selection performance claims do not stand agnostic to the adopted experimental configurations This dependence significantly harms

their ease of adoption. Conversely, proposed instruction selection strategies may be more usable to NLP practioners if the efficacy of methods are tested across a wider range of experimental parameters (more budgets, datasets of differing distributions, etc.).

The Limits of *Selective* Training in *General*-Purpose Instruction Following General purpose instruction following is an unbounded recall problem as it can involve a fairly vast set of capabilities depending upon the context. There isn't a clear consensus on what are the sufficient conditions for claiming competence in general purpose instruction following: Models trained on selected data may show performance improvement against few limited facets but degrade it on unseen ones. Even using automatic metrics that act as proxies for human judgement seems unreliable as these metrics are also fallible (Zheng et al., 2024) and susceptible to bias (Panickssery et al., 2024). Finally, since instruction following has evolving expectations, standardizing the choice of evaluation through human corroboration may only provide a stopgap solution (van der Meer et al., 2024; Shen et al., 2023). As the complexity of such evaluation can be simplified for known test distributions, selection design effort may be more reliable and consistent in such fields.

Limitations

Since our work's goal is study the competency of models on a highly subjective goal, general purpose instruction following, conducting a comprehensive human evaluation to support our conclusions was not feasible. Our work also does not address other attributes that selection strategies differ by: including but not limited to the use of different base models, their impact on tuning strategies (like preference optimization) and alignment objectives.

Ethics Statement

This work highlights the potential of availing negative utility in the field of instruction data selection. Through the evidence in this work, we encourage a more conscious allocation of compute and dollar cost to reduce unnecessary computational overheads. Our code base and training logs (to validate wall clock times) will be released under the MIT License.

Acknowledgments

We are grateful to Sumanth Doddapaneni,Yiming Zhang, Xinyue Liu, Vinay Samuel and Barry Wang for providing feedback throughout this project. This work was supported by a CMU Cylab Seed Grant.

References

Amro Abbas, Kushal Tirumala, Dániel Simig, Surya Ganguli, and Ari S Morcos. 2023. Semdedup: Data-efficient learning at web-scale through semantic deduplication. *arXiv preprint arXiv:2303.09540*.

Yihan Cao, Yanbin Kang, Chi Wang, and Lichao Sun. 2023. Instruction mining: When data mining meets large language model finetuning.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, et al. 2023. Alpagasus: Training a better alpaca with fewer data. *arXiv preprint arXiv:2307.08701*.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Qianlong Du, Chengqing Zong, and Jiajun Zhang. 2023. Mods: Model-oriented data selection for instruction tuning. *ArXiv*, abs/2311.15653.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.

Yuan Ge, Yilun Liu, Chi Hu, Weibin Meng, Shimin Tao, Xiaofeng Zhao, Hongxia Ma, Li Zhang, Boxing Chen, Hao Yang, et al. 2024a. Clustering and ranking: Diversity-preserved instruction selection through expert-aligned quality estimation. *arXiv preprint arXiv:2402.18191*.

Yuan Ge, Yilun Liu, Chi Hu, Weibin Meng, Shimin Tao, Xiaofeng Zhao, Hongxia Ma, Li Zhang, Hao Yang, and Tong Xiao. 2024b. Clustering and ranking: Diversity-preserved instruction selection through expert-aligned quality estimation. *Preprint*, arXiv:2402.18191.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, Claire Barale, Robert McHardy, Joshua Harris, Jean Kaddour,

Emile van Krieken, and Pasquale Minervini. 2024. Are we done with mmlu? *ArXiv*, abs/2406.04127.

J. Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *ArXiv*, abs/2106.09685.

Ming Li, Yong Zhang, Shwai He, Zhitao Li, Hongyu Zhao, Jianzong Wang, Ning Cheng, and Tianyi Zhou. 2024. Superfiltering: Weak-to-strong data filtering for fast instruction-tuning. *Preprint*, arXiv:2402.00530.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2023a. From quantity to quality: Boosting llm performance with self-guided data selection for instruction tuning. *ArXiv*, abs/2308.12032.

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

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2023. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. *ArXiv*, abs/2312.15685.

Ziche Liu, Rui Ke, Feng Jiang, and Haizhou Li. 2024. Take the essence and discard the dross: A rethinking on data selection for fine-tuning large language models. *ArXiv*, abs/2406.14115.

S. Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V. Le, Barret Zoph, Jason Wei, and Adam Roberts. 2023. The flan collection: Designing data and methods for effective instruction tuning. In *International Conference on Machine Learning*.

Renze Lou, Kai Zhang, and Wenpeng Yin. 2024. Large language model instruction following: A survey of progresses and challenges. *Preprint*, arXiv:2303.10475.

Yao Lu, Jiayi Wang, Sebastian Riedel, and Pontus Stenetorp. 2023. Strings from the library of babel: Random sampling as a strong baseline for prompt optimisation. In *North American Chapter of the Association for Computational Linguistics*.

Xingyuan Pan, Luyang Huang, Liyan Kang, Zhicheng Liu, Yu Lu, and Shanbo Cheng. 2024. G-DIG: Towards gradient-based DIverse and hiGh-quality instruction data selection for machine translation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15395–15406, Bangkok, Thailand. Association for Computational Linguistics.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. 2024. Llm evaluators recognize and favor their own generations. *Preprint*, arXiv:2404.13076.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. 2024a. Infobench: Evaluating instruction following ability in large language models. *Preprint*, arXiv:2401.03601.

Yulei Qin, Yuncheng Yang, Pengcheng Guo, Gang Li, Hang Shao, Yuchen Shi, Zihan Xu, Yun Gu, Ke Li, and Xing Sun. 2024b. Unleashing the power of data tsunami: A comprehensive survey on data assessment and selection for instruction tuning of language models. *Preprint*, arXiv:2408.02085.

Ming Shen. 2024. Rethinking data selection for supervised fine-tuning. *Preprint*, arXiv:2402.06094.

Tianhao Shen, Renren Jin, Yufei Huang, Chuang Liu, Weilong Dong, Zishan Guo, Xinwei Wu, Yan Liu, and Deyi Xiong. 2023. Large language model alignment: A survey. *ArXiv*, abs/2309.15025.

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, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971.

Michiel van der Meer, Neele Falk, Pradeep K. Murukannaiah, and Enrico Liscio. 2024. Annotator-centric active learning for subjective nlp tasks. *Preprint*, arXiv:2404.15720.

Jiahao Wang, Bolin Zhang, Qianlong Du, Jiajun Zhang, and Dianhui Chu. 2024. A survey on data selection for llm instruction tuning. *Preprint*, arXiv:2402.05123.

Lai Wei, Zihao Jiang, Weiran Huang, and Lichao Sun. 2023. Instructiongpt-4: A 200-instruction paradigm for fine-tuning minigpt-4. *Preprint*, arXiv:2308.12067.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. Less: Selecting influential data for targeted instruction tuning. *ArXiv*, abs/2402.04333.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy Liang. 2023a. Data selection for language models via importance resampling. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy Liang. 2023b. Data selection for language models via importance resampling. *ArXiv*, abs/2302.03169.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *ArXiv*, abs/2304.12244.

Gregory Yauney and David M. Mimno. 2024. Stronger random baselines for in-context learning. *ArXiv*, abs/2404.13020.

Zhiyuan Zeng, Jiatong Yu, Tianyu Gao, Yu Meng, Tanya Goyal, and Danqi Chen. 2023. Evaluating large language models at evaluating instruction following. *ArXiv*, abs/2310.07641.

Hao Zhao, Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. 2024. Long is more for alignment: A simple but tough-to-beat baseline for instruction fine-tuning.

Xiaosen Zheng, Tianyu Pang, Chao Du, Qian Liu, Jing Jiang, and Min Lin. 2024. Cheating automatic llm benchmarks: Null models achieve high win rates.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, L. Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023a. Lima: Less is more for alignment. *ArXiv*, abs/2305.11206.

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

A Appendix

A.1 Hyperparameter Configurations

We do our evaluations across 3 setups, trying to maximize the coverage of training setups that have been adopted by the strategies we reproduce. Additionally, we carried out one evaluation with LORA (Hu et al., 2021) to test if some correlation between the performance trends of selection strategies could be gleaned from low-rank finetuned models. The results for that evaluation are discussed in more detail in a later section. The standard deviation with reported numbers along with confidence values for our hyperparameter runs across MMLU are provided in Table 5. Since the work we study did not report IFEVAL, LLMBAR or ALPACAEVAL length-controlled win-rates - we were only able to utilize MMLU numbers (reported by all) as our sanity check for replication. All the results reported in the 4, are presented on the hyperparameter configuration that matched the reported performance on MMLU.

Setup	LR	Optimizer	BS	MSL	Epochs	Warmup Ratio	LR Scheduler
Set 1	2e-5	Adam	128	512	3	0.03	Linear
Set 2	2e-5	Adam	128	512	3	0.03	Cosine
Set 3	1e-5	Adam	128	512	3	0.03	Linear
Set 4 [LORA]	2e-5	Adam	128	1024	5	0.3	Linear

Table 3: Hyperparameter Configurations for our experimental setup

Paper	Reported Value	Our Value (Budget - 1k)
Alpagasus at 9K	36.93	35.2
Cherry at 3.5K	36.51	35.2
Cherry at 7K	33.08	35.2

Table 4: Reported Performance versus replicated performance; While S_{deita}also used the same base model as us, they use a 13B parameter model and hence, we do not compare with their numbers. Set 2 was the closest in evaluation to these numbers so we chose Set 2 for reporting our results.

Dataset	Strategy	MMLU (Set 1)	MMLU (Set 2)	MMLU (Set 3)	Standard Deviation	Confidence Interval
alpaca	alpacasus	0.351	0.352	0.345	0.004	0.012
evol	cherry	0.354	0.354	0.348	0.003	0.011
evol	longest	0.352	0.352	0.349	0.002	0.005
flan	alpacasus	0.351	0.351	0.347	0.002	0.007
dolly	alpacasus	0.352	0.352	0.347	0.003	0.009
alpaca	longest	0.353	0.351	0.345	0.004	0.013
evol	alpacasus	0.348	0.348	0.349	0.001	0.002
flan	cherry	0.344	0.343	0.344	0.001	0.002
dolly	longest	0.348	0.347	0.345	0.002	0.005
dolly	cherry	0.348	0.349	0.345	0.002	0.006
alpaca	cherry	0.346	0.345	0.344	0.001	0.003
flan	longest	0.345	0.345	0.345	0.000	0.000

Table 5: MMLU Values for Budget 1000 across all hyperparameter setups. Since we saw the highest (relative) correlation between all benchmarks at this budget, we chose the final hyperparameter set based on this budget's value.

To replicate S_{cherry}, we used the code open-sourced by the authors on Github, making minor adaptations to add support for new datasets. Following the default setup advised in (Li et al., 2023a), we train our pre-experienced model for 1000 samples using the training configurations specified by the authors. For S_{deita} also, we adopt the code opensourced by the authors on Github. We use the Mistral-7B-v0.1 for

embedding generation, along with the quality scorer. For our similarity metrics, we used the same distance metric: cosine but different thresholds as keeping the default threshold led to an underflow for few of the models. We carry out inference using VLLM to improve efficiency of our inference in S_{deita}.

A.2 Detailed Cost Estimation Across Data Selection Budgets

All our estimates are provided assuming the following infrastructure: 8 A6000s, 128 CPUs provided by Google Cloud Estimator. The Dollar Cost of renting our infrastructure per hour is about 8 USD. A detailed breakdown of the costs associated with each step of the selection is provided in Table 6. Note that the cost of selection doesn't vary significantly with the change in the selection budget as the entire dataset needs to be sorted in accordance with the strategy guided metric, irrespective of the final budget.

$S_{\textbf{strategy}}$	**Wall Clock Time on Rented Infrastructure (hr)**
$S_{\text{alpagasus}}$	0
S_{longest}	Total Time per 1000 samples: 1 minute
S_{cherry}	1.457 minutes minutes for 1000 samples embedding construction + 7 mins for training pre-experienced model on 1000 samples (One-time cost, so ignored) + 15 minutes for computing token loss over 1000 samples Total Time per 1000 samples: 16.45 minutes
S_{deita}	2 minutes for 1000 samples for embedding construction (Mistral 7B) + 120 minutes for scoring 1000 samples w/o VLLM 2 minutes for scoring 1000 samples with VLLM + 1^4 minute *at least* for filtering 1000 sample. Total Time per 1000 samples: 5 minutes

Table 6: Wall Clock Times for Each Selection Strategy: We offset the time of computatation we subsample 100K samples from EVOL and then select samples from that subset.

A.3 Estimating Performance Using Cost-Effective Proxies

While it is not possible to largely modify the cost of a selection strategy, it might be possible to offset the cost of finetuning the models on subsets generated via different selection strategies through parameter efficient techniques. If such trends are correlated with the performance of the selected data on the full variant of the model, NLP practioners can potentially design a set of relatively low-cost experiments to rapidly identify the optimal selection strategy to further carry out their selection. Recent work like (Xia et al., 2024), even leverage such correlation to achieve great efficiency in task-specific instruction selection.

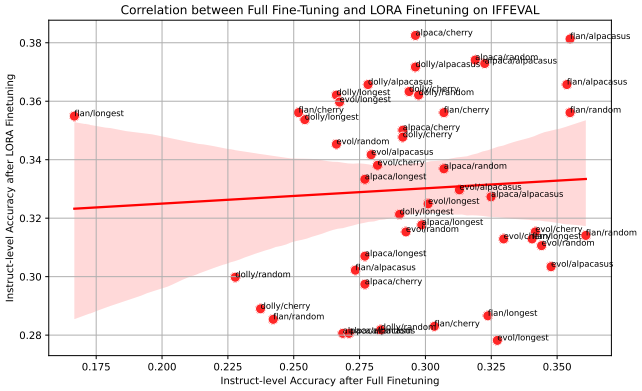


Figure 7: There isn't any observable correlation between the performance of models finetuned with and without LORA across our setups reiterating that cheaper proxies may not reliably predict the optimal selection strategy in a faster, cost-effective parameter efficient setup.

For preliminary experimentation, we rerun all our experiments with the modification of including LORA modules in our finetuning. This reduces the memory footprint of training by to only 0.0038 times of the memory footprint of full finetuning along with faster training by half of its full-finetuning counterpart. In 7 we plot the correlation between the instruct-level-accuracy on IFEVAL for models trained with and without LORA. However, we don't find any reasonable correlation between these performances highlighlting a need to identify cost-effective methods of predicting the suitability of a custom budget and source distribution to a given selection strategy.

A.4 Cost Versus Performance Trade-Offs for All Budgets

As discussed in 6, assessing the trade-off between the cost of data selection and the performance gain achieved through the selection is crucial to understand the utility of a proposed selection strategy. In 8 and 9, we report the performance of the M_{selected}on IFEVAL against their cost of selection; Note that since the entire dataset needs to be scored in accordance with a selection strategy's heuristic irrespective of the final sampling budget - the total cost of selection does not vary significantly with the change in the sampling budgets (especially in the range that we study).

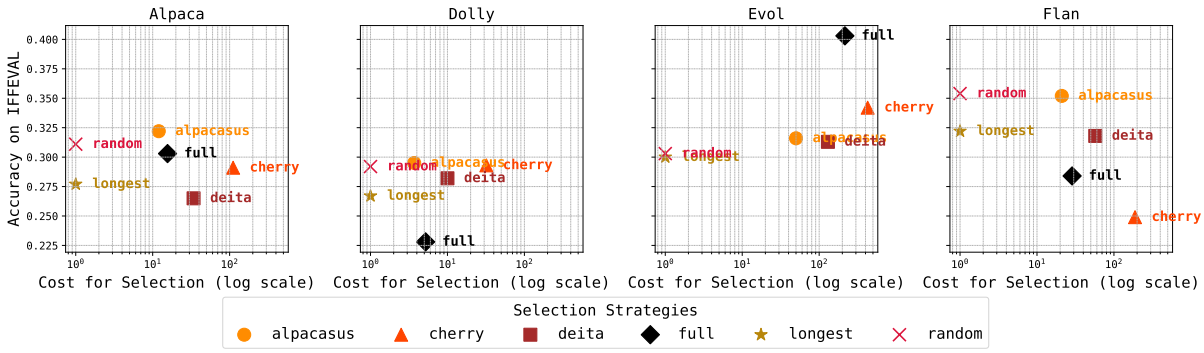


Figure 8: Cost Versus Performance Comparison at 5K budget: We highlight that random performs competitively across most setups.

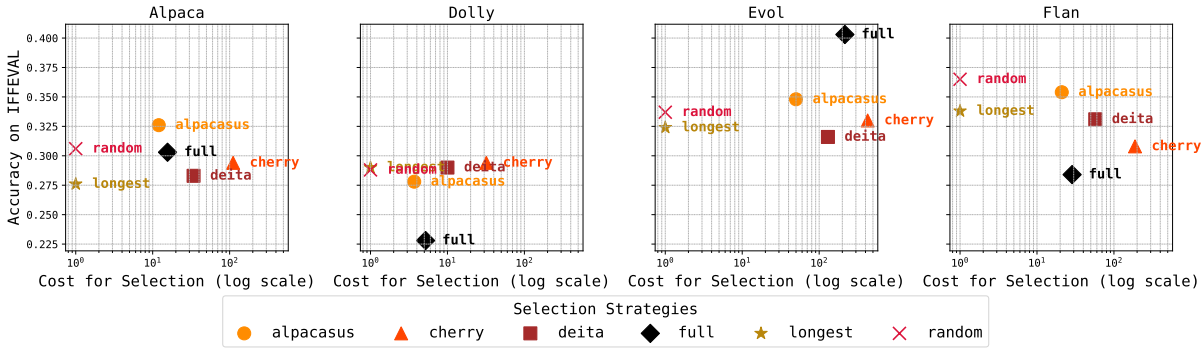
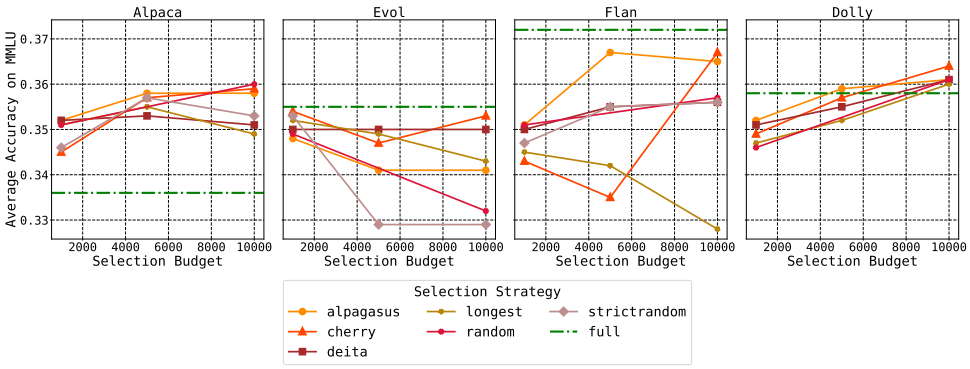


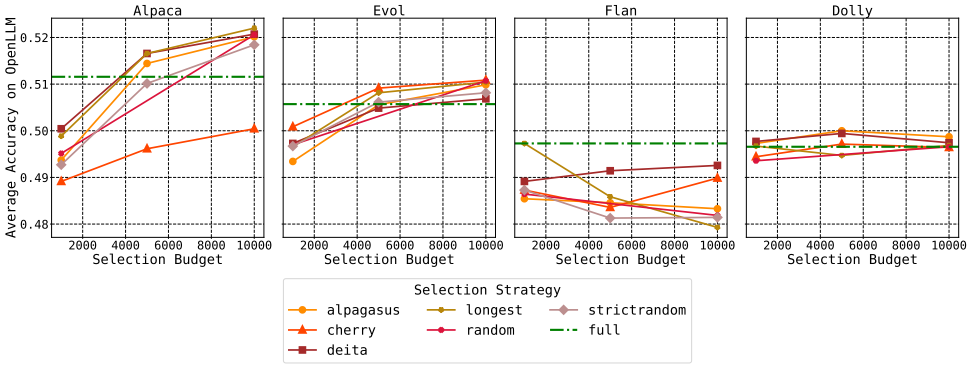
Figure 9: Cost Versus Performance Comparison at 10K:We highlight that random performs competitively across most setups.

In accordance with previous results on the $\mathcal{B}$ of 1000, we do not see consistent performance gains (at the expended cost) for most datasets. The gains from selection are definitely more pronounced at our largest budget, $\mathcal{B}$ of 10000 indicating that selection at larger budgets may provide more consistent gains.

A.5 Benchmark Evaluations for All Configurations

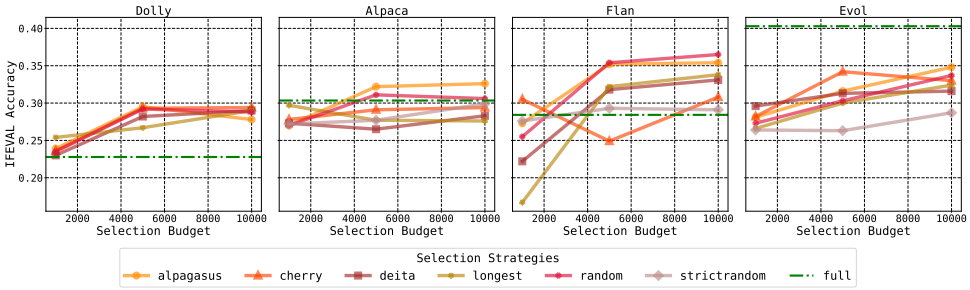


(a) M_{random} versus M_{selected} on MMLU



(b) M_{random} versus M_{selected} on EVALHARNESS

Figure 10: Comparison of M_{random} and M_{selected} on different models



(a) Performance Comparison between M_{random} and M_{selected} on IFEVAL: We report the average of all random runs (both random and strict random) for a particular configuration in any result.