Take the essence and discard the dross: A Rethinking on Data Selection for Fine-Tuning Large Language Models

Anonymous ACL submission

Abstract

Data selection for fine-tuning large language models (LLMs) aims to choose a high-quality subset from existing datasets, allowing the trained model to outperform baselines trained on the full dataset. However, the growing body of works lacks a clear framework as guidance, and the variation in experimental settings hinders systematic comparisons. While most existing surveys provide a comprehensive overview of the various stages and methods involved in data selection, they often overlook providing a detailed introduction to the fine-tuning stage. In response, we conduct a fine-grained survey of data selection techniques for fine-tuning LLMs, covering a dozen key studies. Specifically, we first propose a three-stage framework for data selection (feature extraction, criteria design, and selector evaluation) to categorize existing methods accordingly. We then design a unified comparison approach with ratio-based efficiency and ranking-based feasibility indicators to comprehensively assess methods, overcoming the inconsistency in experimental settings. We provide a final ranking based on it and find that the data selection methods with more targeted quality measurement achieve higher efficiency while at the expense of feasibility. Finally, we discuss trends and highlight four key challenges in measuring data quality, offering potential directions for future research.

1 Introduction

Supervised fine-tuning (SFT) manages to use a handful of instruction-pair data to both unleash large language models' instruction-following capabilities and improve generalization ability across numerous tasks (Wei et al., 2022; Albalak et al., 2024; Zhang et al., 2024; Singh et al., 2023; Radford et al.). Recent research has found that the quality of data is more important than quantity (Nakkiran et al., 2021; Shumailov et al., 2024; Zhou et al., 2024; Jindal et al., 2024) for fine-tuning models.

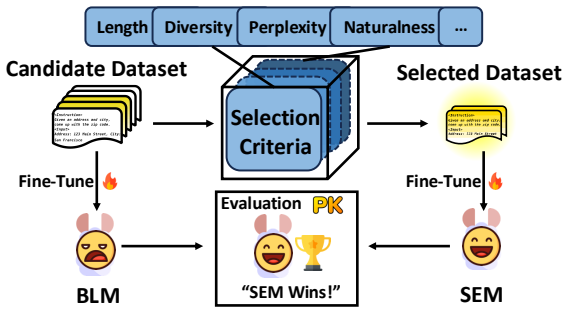

Figure 1: An illustration of data selection for fine-tuning LLMs. Fine-tuning a backbone model on the complete candidate dataset results in a BaseLine Model (BLM), while the model trained on the selected dataset is called Selective-Enhanced Model (SEM), which should outperform BLM.

Thus, various data curation methods are proposed to improve the training data quality, including data selection (Chen et al., 2024; Li et al., 2024b), data evolution (Xu et al., 2023; Wang et al., 2023a), and data reflection (Yin et al., 2023; Mukherjee et al., 2023). As one of the promising directions, data selection chooses a better subset from a candidate dataset guided by selection criteria to improve the performance of the fine-tuned model, as shown in Figure 1. It selects higher-quality samples instead of polishing or augmentation, which improves the training dataset's quality like other data curation methods and enhances training efficiency with fewer samples.

However, the rapidly growing data selection methods currently lack a unified framework as the guidance and cannot be compared with each other directly due to various settings. Although the existing surveys (Bommasani et al., 2021; Wang et al., 2024; Albalak et al., 2024) cover the data curation methods for pre-training, fine-tuning, and reinforcement learning stages, they usually only give the overview from a macro level and lack detailed and targeted introduction to the fine-tuning stage.

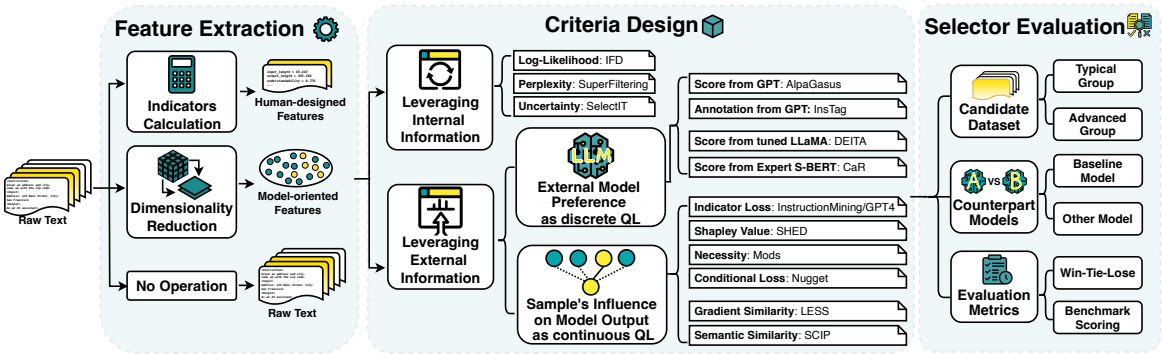

Figure 2: The Three-stage Scheme of Data Selection for Fine-tuning LLMs.

It brings challenges for followers to understand its context and conduct sustained research. Therefore, our survey focuses on a fine-grained review of data selection methods for instruction fine-tuning LLMs by rethinking their recipes, proposing a unified comparison method to compare existing methods directly, and discussing main trends and challenges for future research.

To this end, we first survey the existing dozen data selection works and propose a three-stage scheme to cover them according to the three key elements (data, criteria, and evaluation) in the data selection pipeline, as shown in Figure 2. In the feature extraction stage (Section 2), we divide the existing methods into three types (human-designed features, model-oriented features, and raw text) based on the candidate data forms. In the criterion design stage (Section 3), we divide the methods into utilizing internal and external information of the data according to the source of the sample quality label, and the latter can be further divided into two categories: from model preference or from sample influence. In the selector evaluation stage (Section 4), we conclude that existing data selection work mostly considers three aspects for evaluation, including candidate datasets, compared models, and evaluation metrics.

Then, we design a unified comparison method for existing works, which takes into account ratio-based efficiency indicators and ranking-based feasibility indicators that are overlooked in previous works (Section 5). It can address the inconsistency of existing works in evaluation settings. Initially, we construct a quantitative comparison of two-dimensional efficiency based on the performance improvement ratio and dataset fraction, aligning them through the efficiency curve assumption. Consequently, we consider the feasibility of the method from the perspectives of flexibility indicators and simplicity indicators. It qualitatively ranks existing models by manually considering algorithm complexity and reproducibility (the number of training models, algorithm steps, and whether it is open source), as well as its transferability and scalability (the degree of dependence on data and models).

Finally, we discuss the main trends and challenges faced for data selection. We first sort out the existing works chronologically from three aspects (Candidate Dataset, Quality Measurement, and Selected Feature) to grasp the current research focus (Section 6). We then point out the most important open questions (How can we design effective sample measurement for the data selection?) with four challenges: balance the efficiency and feasibility; ensure the measurement objectivity; improve specific tasks/domains performance without compromising others and satisfy multiple goals.

2 Feature Extraction

For feature extraction, some works preserve the original texts and believe that they contain the most information (Li et al., 2024b; Chen et al., 2024), while others transform raw texts into representative features (such as human-designed features and model-oriented features) for selection. The former complies with human instinct, such as the linguistic indicators, and the latter is directly extracted from the model itself, such as model gradients.

Human-designed Features. To guide data selection with respect to human preference, Instruction-Mining (Cao et al., 2023) uses explainable human-designed features with linguistic information. They convert samples into vectors to select them, which consist of several NLP metrics, including coherence, naturalness, understandability, etc.

Model-oriented Features. For more direct and

targeted selection, other works use model-oriented features extracted from the model as the representations of data. For example, LESS (Xia et al., 2024) creates a datastore of effective and reusable low-dimensional gradient features from the LLM instead of relying on surface features to directly minimize loss on a target task.

3 Criteria Design

The source of the quality label is the most important factor in the design of the selection criteria, which can be divided into internal and external information. The former refers to data quality information inherent in the candidate dataset, while the latter assesses quality through external measures, such as the preferences of external LLMs or the sample influence on models.

3.1 Leveraging Internal Information

To mine the internal information within the given candidate dataset, the pioneering work (Li et al., 2024b) proposes Instruction Following Difficulty (IFD) as the quality label, which measures the contribution of the instruction to the generation of the corresponding output. This study first trains an LLaMA-2-7B as the pre-experienced model on a subset of the target dataset. The IFD score is subsequently determined by measuring the change in the likelihood that this pre-experienced model generates a specific response when given an instruction compared to when no instruction is provided. Inspired by the IFD work, SuperFiltering (Li et al., 2024a) adopts a smaller model (GPT-2) as the pre-experienced model for data selection and harnessing the consistency in IFD scores and perplexity between small and large models to facilitate fine-tuning from weaker to stronger models. Moreover, SelectIT (Liu et al., 2024a) uses a three-level reflection strategy (token-level, sentence-level, and model-level) to measure a backbone model's uncertainty about a candidate sample as a quality label to reduce bias and improve reliability.

3.2 Leveraging External Information

Based on external information sources, existing works can be divided into external model preferences and sample influence on the model output.

3.2.1 External Model Preference as Discrete Quality Labels

To reduce the high-cost and time-consuming human annotations of sample quality, some works use exclusive LLMs (such as ChatGPT), community LLMs (such as LLaMA), or even expert small models to annotate quality automatically. Such a quality label for the sample is usually discrete and given by the external models.

Exclusive LLM Preference. AlpaGasus (Chen et al., 2024) is a notable example, which generates quality labels by prompting ChatGPT to directly score each sample and selects samples based on these scores, mimicking human evaluation. The prompt uses a template with designed evaluation criteria, such as helpfulness and accuracy, applicable across various datasets and backbones. Unlike AlpaGasus assigning a single score, Instag (Lu et al., 2023) produces fine-grained quality labels (tags for instruction intention) annotated by ChatGPT, assessing sample quality based on diversity and complexity. They then introduce a complexity-first diversity sampling algorithm to incorporate both dimensions in data selection.

Community Model Preference. DEITA (Liu et al., 2024b) employs the Evol-Instruct method (Xu et al., 2023) to create samples with varying complexity and quality for training community models (LLaMA) instead of ChatGPT as the scorer. The scorer assesses instruction complexity and response quality scores separately for each candidate sample. Then they design a score-first, diversity-aware algorithm to select samples with a combined ranking of both scores. To optimize the selection cost, CaR (Ge et al., 2024) introduces an expert-aligned and diversity-preserving approach by training a Sentence-BERT (135M) as the quality scorer, followed by clustering and ranking to sample top-K candidates while maintaining diversity.

3.2.2 Sample's Influence on Model Output as Continuous Quality Labels

Other methods calculate the sample's influence on the model's final performance as a continuous quality label for data selection.

Most of them leverage the model to predict inference loss, shapely value, performance changes, or necessity score to estimate the sample's influence on the final performance. Instruction-Mining (Cao et al., 2023) and InstructionGPT-4 (Wei et al., 2023) construct a predictor that uses various feature representations of the samples to estimate the inference loss of a model fine-tuned on them, obtaining quality labels. Then, they utilize BLEND-SEARCH (Wang et al., 2021), combining global

Method	Candidate Datasets	Evaluating SEMs	Counterpart Models		Evaluation Metrics	
			BLM	Others	Wins-ties-losses	Benchmark Scoring
AlpaGasus	Alpaca	LLaMA-2 7B	✔	✔	Vicuna, Koala, WizardLM, Self-Instruct	InstructEval
Instruction-Mining	OpenOrca & DOLLY	LLaMA-1 7B	✔	✔	MT-Bench	OPENLLM, MT-Bench
CaR	Alpaca	LLaMA-2 7B/13B/30B	✔	✔	Vicuna, PandaLM, CoachLM, Self-instruct	-
SelectIT	Alpaca-GPT4	LLaMA-2 7B/13B/70B	✔	✔	-	MMLU, TYDIQA, BBH, GSM, HumanEval, AlpacaEval
InstructionGPT-4	MiniGPT-4	LLaMA-2	✔	✘	LLaVA-Bench	MME, VQA, MT-Bench
IFD	Alpaca & WizardLM	LLaMA-2 7B	✔	✘	Vicuna, Koala, WizardLM, Self-Instruct, LIMA	OPENLLM
Superfiltering	Alpaca & Alpaca-GPT4	LLaMA-2 7B/13B	✔	✘	WizardLM	OPENLLM, AlpacaEval
Nuggets	Alpaca	LLaMA-2 7B	✔	✘	-	MT-Bench, AlpacaEval
LESS	FLAN V2 & CoT & DOLLY & Oasst	LLaMA-2 13B; Mistral 7B	✔	✘	-	MMLU, TYDIQA, BBH
SHED	MMLU & WizardLM	LLaMA-2 7B/13B, Vicuna 7B, GPT-2	✔	✘	-	MT-Bench, AlpacaEval
MoDS	Alpaca & HC3 & Alpaca-Evol-Instruct & DOLLY & InstructWild & LIMA	LLaMA-2 7B	✔	✘	Koala, WizardLM, Self-instruct, Vicuna, LIMA	-
InsTag	WizardLM & UltraChat & ShareGPT	LLaMA-1/-2	✘	✔	-	MT-Bench
DEITA	Alpaca & DOLLY & Oasst & FLAN 2022 & WizardLM & UltraChat & ShareGPT	LLaMA-1/-2 13B; Mistral 7B	✘	✔	-	OPENLLM, MT-Bench

Table 1: The candidate dataset, SEMs, counterpart models, and evaluation metrics used in each method. The "✔" under BLM means the work uses the same BLM as the SEM; under Others, the "✔" means the work uses models other than BLM, including oracle LLMs and other fine-tuned SEMs.

and local optimizations to determine the optimal data size. To more precisely quantify the sample influence at a lower cost, SHED (He et al., 2024) first employs a model-agnostic clustering algorithm (K-means based on Sentence-BERT) to identify representative proxy data, then calculate their Shapley values as quality scores for each cluster to guide data selection. Nuggets (Li et al., 2024c) directly selects the sample by calculating conditional loss, which reflects the sample's impact on the model: comparing the model's performance difference on a predefined task with and without the sample in the context (one-shot score - zero-shot score). MoDS (Du et al., 2023) uses the high-quality data subset identified through external model scoring to train an initial LLM on the selected seed data. This initial LLM is then evaluated within the high-quality subset by the necessity scores of the samples to perform the final data selection. It obtains better samples through twice measurements, achieving significant improvements of the fine-tuned model.

Different from the above work relying on the model prediction, LESS (Xia et al., 2024) assesses a sample's influence on the model's performance by calculating its similarity to known high-quality target samples based on gradient information. It measures the gradient similarity between candidate samples and existing task datasets. They begin with training a LoRA model to compute gradient features for the candidate data and randomly project them for storage. Samples from the validation task dataset in the few-shot setting are also projected into this space to calculate gradient similarity and obtain the data quality labels for selection.

4 Selector Evaluation

To evaluate the effectiveness of the selectors, the existing works usually take the selectively enhanced model (SEM) that is trained on a subset selected from a candidate dataset as the comparison objective. Then, they compared the performance of the SEM with the same model fine-tuned on the full dataset (Baseline model, BLM) or other popular LLMs on win-tie-lose or benchmark scores. Table 1 details the evaluation setup, including the choice of candidate datasets, comparison models, and evaluation metrics used to measure performance.

Candidate Datasets. Most of the works (Li et al., 2024a,b; Liu et al., 2024b) use the popular open-sourced datasets as candidate datasets to push forward better performance of fine-tuned models by selecting higher-quality samples in them. The candidate dataset is further divided into the typical group, including Alpaca (Taori et al., 2023), Dolly (Conover et al., 2023), FLAN (Wei et al., 2022), etc., and the advanced group developed from the typical datasets to achieve higher quality, including WizardLM (Xu et al., 2023), UltraChat (Ding et al., 2023a), etc.

Counterpart Models. Most works choose BLM as the counterpart model for comparison to evaluate the performance of the SEM. They tend to use the popular LLaMA series (Chen et al., 2024; Lu et al., 2023) as well as Mistral (Liu et al., 2024b; Xia et al., 2024) models as backbones to obtain relative improvement evaluation, which directly shows the improvement effect of the selector. Other works

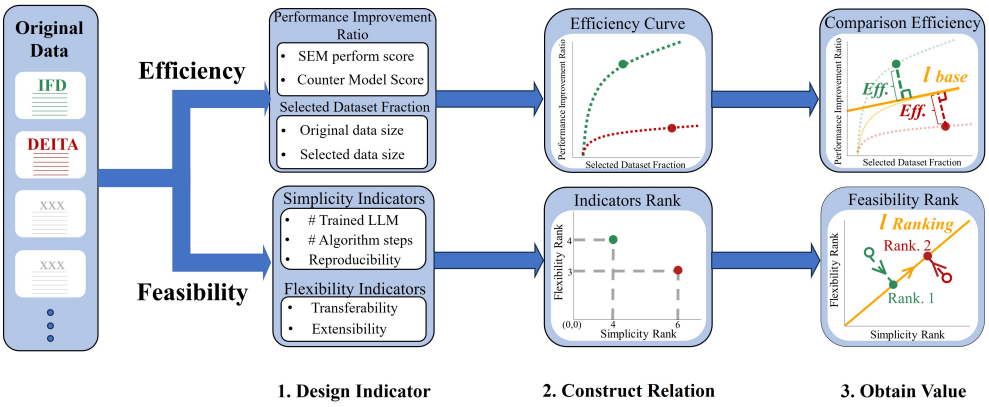

Figure 3: The illustration of the framework of our comparison method. To better illustrate the comparison process, IFD and DEITA are chosen as examples.

(Xia et al., 2024; Chen et al., 2024) compare the SEM with SOTA models (such as GPT-4, Claude, and LLaMA-Chat 7B) to obtain absolute improvement evaluation, which indicates how good SEM achieves.

Evaluation Metrics. The evaluation metric adopts both relative and absolute methods to evaluate the selector comprehensively. The absolute metric uses wins-ties-losses pairing scored by GPT-4 to indicate the direct performance difference between the SEM and counterpart model, while the absolute metric uses benchmark scoring to directly score and rank the SEM. Benchmark scoring is separated into a traditional group examining the loss of the model's response on test datasets (such as MMLU and TYDIQA) and a popular group using GPT-4 to score on various benchmarks, such as OPENLLM and MT-Bench.

5 An Unified Comparison Among Data Selection Methods

To directly compare various existing methods while addressing inconsistencies in their experimental setups, we propose a unified comparison approach that incorporates ratio-based efficiency and ranking-based feasibility. Figure 3 illustrates the three-step framework: (1) Design indicator: evaluate existing methods using designed indicators under a consistent setting (Alpaca as the candidate dataset, LLaMA-2 7B/13B as the SEM, and wins-ties-losses against the BLM as the metric); (2) Construct relationships: plot the points of various methods on a two-dimensional space representing Efficiency (Feasibility) comprehensively; (3) Obtain value: align the methods with a baseline to enable comparison by translating their relationships into accessible measurements.

5.1 Efficiency of the Selector

The efficiency of the data selector reflects its accuracy in choosing the ground-truth high-quality data. As shown in Figure 3, the process involves three steps: (1) We first define an efficiency space using two ratio-based indicators: the Performance Improvement Ratio (PIR) and the Selected Dataset Fraction (SDF), which adjust for biases in the experimental settings; (2) We then establish an efficiency curve to represent the relationship between these indicators and overall efficiency, using extrapolation to align them; (3) Finally, we set a baseline and apply a homeomorphism function to convert each method's latent efficiency into an explicitly signed distance from the baseline, enabling direct comparisons.

5.1.1 Ratio-based Efficiency Indicators

Performance Improvement Ratio (PIR). PIR is calculated as the average ratio across different evaluation metrics, where the ratio is the SEM's performance score divided by that of the counterpart model. For methods lacking evaluation data under the unified settings, we leverage the consistency of the selector across various settings to estimate the PIR. Specifically, we compute three additional average ratios under different evaluation conditions that encompass all methods and use the observed linear relationship among these ratios to estimate the PIR for the missing data (see Appendix A.2 for more details).

Selected Dataset Fraction (SDF). SDF assesses the effect of data size in a uniform way, which is determined by the ratio of the selected dataset's size to the total size of the original candidate dataset. It

ensures proportional representation across datasets and eliminates bias due to varying dataset sizes (from 3,439 samples (Wei et al., 2023) to 306,044 samples (Lu et al., 2023)).

5.1.2 Comparison Based on Efficiency Curve

After plotting existing works using PIR and SDF, efficiency remains a latent variable. Although higher PIR and lower SDF indicate greater efficiency, comparing works with differing values for both indicators remains a challenge. To address this, we draw inspiration from scaling laws (Kaplan et al., 2020; Chung et al., 2024) and LIMA (Zhou et al., 2024) and introduce the efficiency curve extrapolation assumption. We define two key properties of the unified efficiency curve: (1) The curve follows a logarithmic-like pattern, with a rapid but brief increase before approaching linearity; (2) The slope of a superior efficiency curve is always steeper than that of an inferior one (see Appendix A.3 for details). Using these properties, we transform efficiency into an explicit representation on the $SDF \times PIR$ space, as shown in Figure 3.

Furthermore, we introduce a more direct and robust comparison measure to address the limitations of the explicit efficiency curve, which only offers a rough approximation of the relationship between SDF and PIR. It applies a homeomorphism function, transforming mutual efficiency comparisons into measurable distances between different works, with each work's distance from the baseline representing its comparative efficiency, thereby making all works directly comparable, as shown in Figure 4.

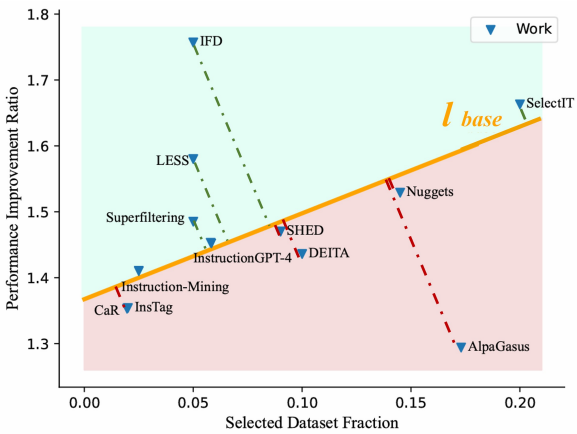

Figure 4: The comparison efficiency among popular data selection methods. The yellow line is the efficiency baseline (l_{base}). The distance between work and baseline represents their difference in efficiency, thus it represents the comparison efficiency of each work.

Specifically, we take the yellow line connecting Instruction-Mining and Instruction-GPT4 as the baseline, approximating their shared efficiency curve due to their similar selection methods. The signed distance from this baseline indicates whether a method's efficiency is superior (green dashed line) or inferior (red dashed line) than the baseline, with greater distances reflecting larger efficiency differences (see Appendix A.4 for details).

As shown in Figure 4, it can be intuitively seen that IFD is the most efficient, while AlpaGasus is the least efficient. This is because IFD is more targeted: The Instruction Following Difficulty score is calculated not only based on information from within the candidate dataset but also on the feature extracted by the Backbone model. Moreover, its quality labels come from the loss of the model, which is more direct and does not introduce external information interference. On the other hand, AlpaGasus only utilized external ChatGPT scoring without considering the impact of specific data quality distribution, characteristics of the Backbones, or optimization objectives on the performance of the model trained after data selection.

Additionally, we can observe the relative rankings of other methods that were previously difficult to compare directly due to differing experimental setups. For example, LESS outperforms SuperFiltering, SuperFiltering surpasses Nuggets, Nuggets outperform Instag, and Instag is better than DEITA.

5.2 Feasibility of the Selector

Unlike previous studies, which primarily focus on efficiency, our survey also considers the practical ease of using each method, a factor often overlooked. To assess this, we follow the three-step process shown in Figure 3 for manual evaluation, as relevant metrics are lacking. First, we identify five key factors that compose two ranking-based feasibility indicators influencing the simplicity and flexibility of each method, and we manually annotate them. Then, we derive simplicity and flexibility ranks and position the works on a two-dimensional feasibility measurement plane for visual representation. Finally, we integrate both ranks into an overall feasibility score by mapping each work.

5.2.1 Ranking-based Feasibility Indicators

Simplicity. This metric assesses the complexity of the selection process and the reproducibility (Rep.) of the method. For complexity, we examine the implementation cost of the method, which

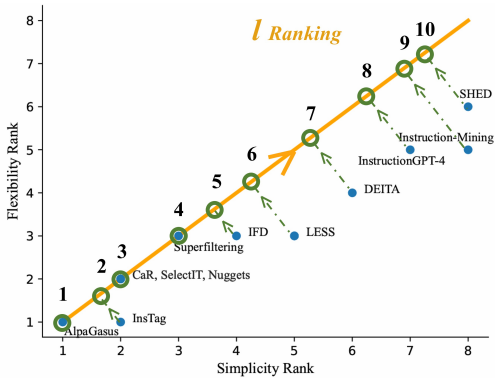

Figure 5: The feasibility rank among popular data selection methods. The yellow line is the ranking line ($l_{ranking}$). The sequence of the projection points alone the line represents the ranking of works.

includes the number of models to be trained and the complexity of the algorithms to be executed. We also verify the reproducibility manually, that is, whether it has been open-source implemented, which is encouraged and valued by the research community. Further details are provided in Appendix A.5.1.

Flexibility. Additionally, we evaluate the model's flexibility in terms of transferability and extensibility, both crucial for a method's generalization in practical applications. For transferability, a model-free method can choose whatever models to use in the selection process without retraining the selector model, meaning the model does not indispensably in obtaining quality labels; while a dataset-free selector can apply to any given candidate dataset without re-training, emphasizing its independence to the specific information contained in the candidate dataset for selection.

For extensibility, we consider whether the algorithm contains fixed components that limit its adaptability to other algorithms. For instance, methods like AlpaGasus and InsTag rely heavily on third-party tools such as ChatGPT, which hinders their ability to extend to other frameworks due to the immutability of ChatGPT.

5.2.2 Comparison with Simplicity and Flexibility

Based on the published papers for each work, we annotate their simplicity and flexibility, then plot them according to the final rankings, as shown in Figure 5. We assign equal weight to simplicity and flexibility and draw the $l_ranking$ line to provide a comprehensive ranking based on both factors. Specifically, to determine the feasibility, we project each work perpendicularly onto the ranking line (which passes through the origin), representing the average of the two ranks. The sequence of the projected points corresponds to the feasibility ranking of the works.

As shown in Figure 5, AlpaGasus demonstrates the highest flexibility, despite its less-than-satisfactory performance. Its simplicity, stemming from the lack of a need for LLM training and fewer steps in the data selection process, makes it easy for subsequent works to reproduce its results—even though no official publicly available source code exists. Moreover, it can be more easily transferred to other scenarios (both model-free and dataset-free) as it relies solely on ChatGPT without additional information. Although Instruction-Mining and InstructionGPT-4 outperform AlpaGasus, they compromise feasibility due to their heavy reliance on fine-tuning multiple LLMs and using complex quality indicators derived from both datasets and models.

5.3 Overall Consideration of the Selector

In summary, a clear observation from our comparison of efficiency and feasibility is that existing methods struggle to achieve both high performance and ease of use. Additionally, we find that the more tailored a data selection method is, the better the performance of the SEM tends to be. For instance, DEITA is more complex than Alpaca because it trains the LLM based on a backbone model and accounts for data diversity, leading to improved performance. However, more complex processes and algorithms may introduce additional external factors rather than directly optimizing the target, making them harder to transfer. For example, LESS underperforms compared to IFD, despite being more complex due to the inclusion of external datasets.

We also evaluate MoDS, which demonstrates high comparative efficiency (PIR = 3.34, SDF = 0.02) but has the lowest feasibility (details in Appendix more details in Appendix A.4). Its use of two coupled phases of LLM training, akin to applying the selection method twice, results in its exclusion from comparisons with other works.

6 Discussions

6.1 Trends

Figure 6 shows the current research trends, where we have sorted out the existing work from three aspects (Candidate Dataset, Quality Measure-

ment, and Selected Feature) in chronological order. There are three main trends we can conclude:

(1) In terms of candidate datasets, existing work has gradually shifted from not relying on general datasets building the selector (Cao et al., 2023) to the specific datasets participating in the construction process of selectors (Li et al., 2024b).

(2) Quality Measurement has also shifted from considering the quality and diversity of samples from an external scorers (Chen et al., 2024; Liu et al., 2024b) to a more direct measuring of the improving model performance by the sample (Xia et al., 2024).

(3) The indicators of the selection are from using concrete indicators (single quality score (Chen et al., 2024) and multi-dimension indicators (Lu et al., 2023)) to abstract indicators (the training loss (Li et al., 2024a) and gradient similarity (Xia et al., 2024)).

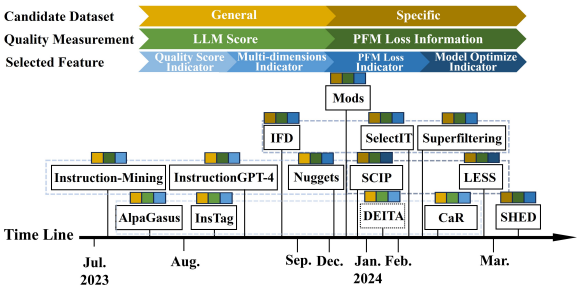

Figure 6: The timeline of the data selection methods.

6.2 Challenges

Despite substantial progress in data selection for fine-tuning large language models (LLMs), several critical challenges remain, particularly in addressing the open question: **How can we design effective sample measurements for data selection?**

(1) Balancing the efficiency and feasibility.As discussed in the Comparing Data Selection Methods section, targeted approaches that directly measure the impact on model performance often demonstrate higher efficiency. However, these approaches can compromise feasibility and applicability in diverse settings, a limitation overlooked in previous methods (Cao et al., 2023; Wei et al., 2023; Xia et al., 2024). Developing a measurement strategy and data selection algorithm that balances efficiency and feasibility remains an open research challenge.

(2) Ensuring the measurement objectivity. For methods based on existing LLM preferences (Chen et al., 2024; Lu et al., 2023; Liu et al., 2024b), designing appropriate scoring prompts to obtain reliable scores is crucial, as biases persist even in GPT-4. For methods based on sample's influence (Li et al., 2024b; Xia et al., 2024; Li et al., 2024c), designing metric functions that account for both sample characteristics and model specifics is essential. Achieving objective and consistent measurements across different tasks and models remains a significant challenge.

(3) Improving specific tasks/domains performance without compromising others. Research shows that performance gains from selected samples vary across tasks (Jha et al., 2023). While notable improvements are seen in writing and role-playing, gains in mathematics and reasoning are often marginal (Lu et al., 2023; Chen et al., 2024; Li et al., 2024c). Thus, improving task-specific performance without compromising others is a worthwhile direction for future research.

(4) Satisfying multiple goals in data selection. Most existing works focus on merging aspects such as data quality (Chen et al., 2024; Xia et al., 2024) and diversity (Lu et al., 2023; Liu et al., 2024b). However, data selection methods for multi-turn and multi-model scenarios are still limited, and developing these approaches is crucial for expanding the applicability of data selection techniques.

7 Conclusion

In this paper, we present a fine-grained survey of a dozen data selection methods for fine-tuning large-scale language models. We establish a three-stage data selection framework to unify the process. To address inconsistencies across different experimental setups, we also introduce a unified comparison approach using ratio-based and ranking-based indicators to quantify efficiency and assess feasibility, which has been overlooked in previous works. Based on our comprehensive ranking considering these two aspects, our findings indicate that the more target-specific designs achieve higher efficiency while at the expense of feasibility. In addition to outlining the timeline of existing work, we highlight four key challenges (balancing efficiency and flexibility, ensuring measurement objectivity, improving specific tasks/domain performance without compromising others, and satisfying multiple goals in data selection) related to improving measurement methods for sample quality, offering directions for future research.

Limitation

We mainly research data selection for instruction fine-tuning LLMs instead of data rewriting or augmentation. Although we have already comprehensively examined the existing works, we acknowledge that there may still be some works we neglected, especially the very recent work that was published on the preprint platforms.

Besides, we focus on outlining the scheme of existing work on data selection and propose an analytical method for comparing various works directly. Therefore, the descriptions of each work could be limited to key points relevant to our study rather than providing a comprehensive overview due to limited space.

References

Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, Colin Raffel, Shiyu Chang, Tatsunori Hashimoto, and William Yang Wang. 2024. A survey on data selection for language models.

Guangji Bai, Zheng Chai, Chen Ling, Shiyu Wang, Jiaying Lu, Nan Zhang, Tingwei Shi, Ziyang Yu, Mengdan Zhu, Yifei Zhang, Carl Yang, Yue Cheng, and Liang Zhao. 2024. Beyond efficiency: A systematic survey of resource-efficient large language models.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

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, and Hongxia Jin. 2024. Alpagasus: Training a better alpaca with fewer data.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

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. *Company Blog of Databricks*.

Jiaxi Cui, Wentao Zhang, Jing Tang, Xudong Tong, Zhenwei Zhang, Amie, Jing Wen, Rongsheng Wang, and Pengfei Wu. 2024. Anytasktune: Advanced domain-specific solutions through task-fine-tuning.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023a. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. 2023b. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235.

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

Isabel O Gallegos, Ryan A Rossi, Joe Barrow, Md Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K Ahmed. 2024. Bias and fairness in large language models: A survey. *Computational Linguistics*, pages 1–79.

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

Prakhar Gupta, Cathy Jiao, Yi-Ting Yeh, Shikib Mehri, Maxine Eskenazi, and Jeffrey P Bigham. 2022. Instructdial: Improving zero and few-shot generalization in dialogue through instruction tuning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 505–525.

Yexiao He, Ziyao Wang, Zheyu Shen, Guoheng Sun, Yucong Dai, Yongkai Wu, Hongyi Wang, and Ang Li. 2024. Shed: Shapley-based automated dataset refinement for instruction fine-tuning. *arXiv preprint arXiv:2405.00705*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Aditi Jha, Sam Havens, Jeremy Dohmann, Alex Trott, and Jacob Portes. 2023. Limit: Less is more for instruction tuning across evaluation paradigms.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, Toronto, Canada. Association for Computational Linguistics.

Ashvini Kumar Jindal, Pawan Kumar Rajpoot, and Ankur Parikh. 2024. Birbal: An efficient 7b instruct-model fine-tuned with curated datasets. *arXiv preprint arXiv:2403.02247*.

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

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

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2024b. From quantity to quality: Boosting llm performance with self-guided data selection for instruction tuning.

Yunshui Li, Binyuan Hui, Xiaobo Xia, Jiaxi Yang, Min Yang, Lei Zhang, Shuzheng Si, Ling-Hao Chen, Junhao Liu, Tongliang Liu, Fei Huang, and Yongbin Li. 2024c. One-shot learning as instruction data prospector for large language models.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 1950–1965. Curran Associates, Inc.

Liangxin Liu, Xuebo Liu, Derek F Wong, Dongfang Li, Ziyi Wang, Baotian Hu, and Min Zhang. 2024a. Selectit: Selective instruction tuning for large language models via uncertainty-aware self-reflection. *arXiv preprint arXiv:2402.16705*.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2024b. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning.

Keming Lu, Hongyi Yuan, Zheng Yuan, Runji Lin, Junyang Lin, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2023. instag: Instruction tagging for analyzing supervised fine-tuning of large language models.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

Elizaveta Moskovskaya, Olesya Chebotareva, Valeria Efimova, and Sergey Muravyov. 2023. Predicting dataset size for neural network fine-tuning with a given quality in object detection task. *Procedia Computer Science*, 229:158–167.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv preprint arXiv:2306.02707*.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. 2021. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2023. A comprehensive overview of large language models.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners.

Matthias C Rillig, Marlene Ågerstrand, Mohan Bi, Kenneth A Gould, and Uli Sauerland. 2023. Risks and benefits of large language models for the environment. *Environmental Science & Technology*, 57(9):3464–3466.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. 2024. Ai models collapse when trained on recursively generated data. *Nature*, 631(8022):755–759.

Avi Singh, John D Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Peter J Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron Parisi, et al. 2023. Beyond human data: Scaling self-training for problem-solving with language models. *arXiv preprint arXiv:2312.06585*.

Feifan Song, Bowen Yu, Hao Lang, Haiyang Yu, Fei Huang, Houfeng Wang, and Yongbin Li. 2024. Scaling data diversity for fine-tuning language models in human alignment.

Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. 2017. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Chi Wang, Qingyun Wu, Silu Huang, and Amin Saied. 2021. Economic hyperparameter optimization with blended search strategy. In *International Conference on Learning Representations*.

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

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023a. Self-instruct: Aligning language models with self-generated instructions.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, et al. 2022. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109.

Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. 2023b. Aligning large language models with human: A survey.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

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

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

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.

Da Yin, Xiao Liu, Fan Yin, Ming Zhong, Hritik Bansal, Jiawei Han, and Kai-Wei Chang. 2023. Dynosaur: A dynamic growth paradigm for instruction-tuning data curation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4031–4047, Singapore. Association for Computational Linguistics.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2023. Instruction tuning for large language models: A survey.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2024. Instruction tuning for large language models: A survey.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Ming Zhong, Yang Liu, Da Yin, Yuning Mao, Yizhu Jiao, Pengfei Liu, Chenguang Zhu, Heng Ji, and Jiawei Han. 2022. Towards a unified multi-dimensional evaluator for text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2023–2038, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2024. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36.

A Appendix

A.1 Brief Road map to Data Selection

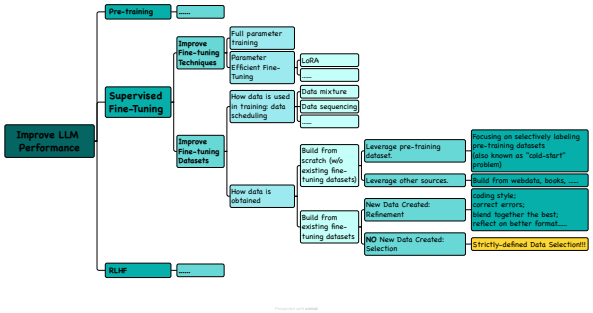

Figure 7: The above is an illustration of the brief road map from LLMs to Data Selection.

Large language models (LLMs) nowadays are powerful AI systems capable of generating authentic human languages and completing numerous classic NLP challenges (Naveed et al., 2023; Vaswani et al., 2017; Wang et al., 2022; Zhong et al., 2022). The training of these models typically involves three stages, where the supervised (instruction) fine-tuning bridges the mismatch between the knowledge-based pretraining and real-world users' needs (Zhang et al., 2024), which also endows LLMs with the most significant performance rise (Albalak et al., 2024; Zhang et al., 2024).

To optimize the fine-tuning of LLMs, researchers have focused on three areas: (1) refining the pretraining stage to equip foundational models with more comprehensive or domain-specific knowledge (Zhao et al., 2023); (2) advancing fine-tuning techniques, such as Parameter-Efficient Fine-Tuning (PEFT) (Ding et al., 2023b; Liu et al., 2022; Hu et al., 2022); and (3) curating or constructing high-quality fine-tuning datasets (Zhang et al., 2023). Such datasets can either be built from scratch or be derived from existing fine-tuning datasets. The former approach typically self-evolves new data from seeds (Xu et al., 2023; Wang et al., 2023a) or reflects/refines on extensive pretraining datasets and raw text corpora(Yin et al., 2023; Mukherjee et al., 2023). Given existing datasets, the latter approach can be further classified into two categories depending on whether new data is generated: (a) methods that involve generating new data by refining or combining existing datasets (Jiang et al., 2023), and (b) strictly defined data selection, which does not generate new data and is the focus of our study.

By selectively identifying and using high-quality data, data selection can significantly enhance fine-tuning results. Data selection improves model performance during evaluation, enhancing the model's capabilities in domain-specific tasks (Xia et al., 2024; Cui et al., 2024; Minaee et al., 2024), improving the model's generalization and instruction-following capacity (Gupta et al., 2022; Zhang et al., 2024; Wang et al., 2023b), and reducing its toxicity and bias (Song et al., 2024; Gallegos et al., 2024). Additionally, data selection boosts training efficiency by lowering computational costs and conserving energy (Rillig et al., 2023; Bai et al., 2024).

A.2 Performance Improvement Ratio

Method	SEM	Same Model		Other Models	
		Wins Rate	Bench.	Wins Rate	Bench.
AlpaGasus	LLaMA-2 7B	1.284	0.949	-	-
SuperFiltering	LLaMA-2 7B	1.475	1.010	-	-
InsTag	LLaMA 13B	1.344	-	-	0.985
DEITA	LLaMA-2 13B	1.426	-	-	1.000
InstructionGPT-4	MiniGPT-4	1.443	-	-	-
Nuggets	LLaMA-2 7B	1.519	-	-	-
IFD	LLaMA-2 7B	1.747	-	-	-
LESS	LLaMA-2 13B	(1.491)	1.015	-	-
Instruction-Mining	LLaMA-2 7B	(1.400)	-	0.212	0.991
MoDS	LLaMA-2 7B	3.337	-	-	-
SHED	LLaMA-2 7B	(1.460)	1.005	-	-
CaR	LLaMA-2 7B	1.343	-	-	-
SelectIT	LLaMA-2 7B	(1.653)	1.067	-	-

Table 2: The performance improvement under four evaluation settings. In the Same model, we compare SEM and BLM, while in other models, we compare SEM and the same-size models trained based on other backbones (such as LLaMA chat).

Since different work uses different evaluation methods, it is difficult to compare them directly. Therefore, to uniformly evaluate their performance, we first fix Alpaca as the candidate dataset as it is widely adopted and divide the various evaluation settings mentioned in all works into (1) BLM comparison with SEM on the same Backbone and (2) BLM comparison with other models (such as LLaMA chat). Then, we further divide them into wins rate and benchmark improvement (Bench.) with different kinds of evaluation metrics. In total, we have four evaluation settings, as shown in Table 2, and we take the average of each type in Eq. (1) if it uses multiple evaluation metrics or other models.

$$\frac{1}{n}\sum_{i=0}^{n}\frac{X_i}{Y_i} \tag{1}$$

where X_i and Y_i are, respectively, the performance of the SEM and the counterpart model under the

same evaluation setting i, and n is the total number of the evaluation settings using the same kind of evaluation metric (wins-ties-losses or benchmark scoring) and counterpart model. We then choose the wins rate under BLM as the ratio indicator of PIR not only because it directly reflects the improvement effect made by the selector but also because most of the works provide this value.

To fill the missing value, we leverage the consistency of model performance: the same model should perform similarly under different evaluation settings. Therefore, we obtain the bridge function by linearly regressing the other works with the wins rate under BLM as the label and other three entries in Table 2 as variables. Then, we estimate the missing value by using the bridge function to transfer the value under other entries into PIR.

For the work that does not adopt the Alpaca as a candidate dataset, we use the same regression method to scale its performance to the Alpaca dataset. Moreover, for the work evaluated under multiple SDF and thus having multiple PIR, we choose the one that has the highest PIR value to indicate the method's optimal performance.

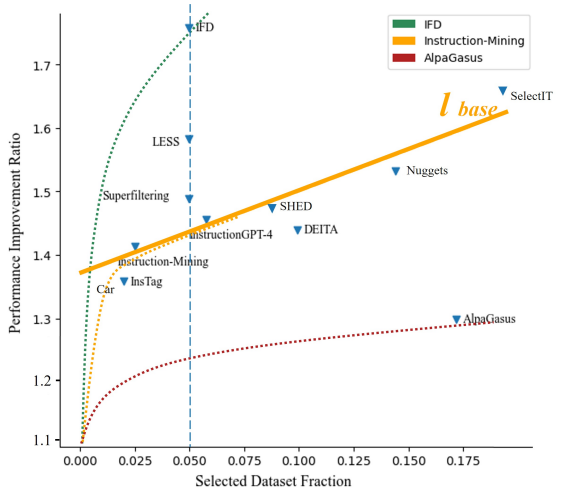

Figure 8: The illustration of the efficiency curve of three representative works (IFD, Instruction-Mining, and AlpaGasue). The yellow baseline is the line connecting Instruction-Mining and InstructionGPT-4, which is the asymptote of the efficiency curve shared by both work and works as the ground approximation of the efficiency curve.

A.3 Efficiency Curve Extrapolation

To directly compare the efficiency of work with different performance improvement ratios and selected data fractions, we establish the efficiency curve for each work using the efficiency curve Extrapolation, which provides two properties to the efficiency curve:

Property 1. According to relevant theories (such as scaling law) (Kaplan et al., 2020; Sun et al., 2017; Moskovskaya et al., 2023), which suggest that the impact of logarithmic data size on the loss is linear if the augmented dataset maintains the same quality structure. Inspired by these theories, we account for a dataset with fixed quality, its function of the performance improvement ratio, and selected dataset fraction compilation to the logarithmic-like function, which is upward, concave, and approaching linear after experiencing a rapid but short increase.

Property 2. The efficiency represented by the above curve with a larger slope is superior to that represented by the below one. It is intuitively derived from property 1 with the fact that high-quality data leads to better performance of SEM (Zhou et al., 2024) Therefore, if the method is superior, which indicates its selected dataset has a good quality structure, the increases of its performance improvement must be greater than the inferior one at every point of selected dataset fraction.

Figure 8 is plotted following the efficiency curve proposition. The curve is represented with a dashed line to show that they are not ground-truth curves. The efficiency curve extrapolation just describes some basic properties of the curve without providing the exact function.

A.4 Comparison Efficiency Method

From the intuitive perspective, the comparison efficiency method virtually scales the SDF of all works into an identical value, and reflects the efficiency on the corresponding virtual PIR. Take Figure 8 as an illustration, the work can move on the curve without changing their efficiency. Therefore, every work can scale to the blue dashed line and make a comparison on the PIR.

However, in addition to the yellow baseline, the ground-truth efficiency curve of the other works is unknown. Moreover, even with the ground-truth efficiency curve, efficiency is unobtainable because the exact function of the curve is unknown. The efficiency curve extrapolation only provides some abstract properties.

To obtain the efficiency value, we use homeomorphism to transform the efficiency of work into the distance to the baseline. Specifically, let X be a set whose element is the efficiency value of

each work, and let Y be a set whose element is the work's distance to baseline in the product space of $S \times P$, where S is the set contains all SDF and P is the set contains all PIR. Then, we introduce a homeomorphism function $\varphi : X \mapsto Y$, to map the efficiency space X on to space Y, such that for any $y_1, y_2 \in Y$ and $x_1 = \varphi^{-1}(y_1)$, $x_2 = \varphi^{-1}(y_2)$, pair y_1 and y_2 preserves the exactly same relation with the pair x_1 and x_2.

Now, first, let the efficiency curve function be a homeomorphism $f : X \mapsto S \times P$, where X is a latency space. Then an efficiency curve of work k (i.e. the work with efficiency x_k) is represented by $f_S(x_k)$. According to the efficiency curve properties, it is easy to see that both f and f^{-1} are bijection and continuous. This step is equivalent to transforming the latent efficiency space into explicit efficiency curve space. In the following steps, we apply the efficiency curve properties to transform the efficiency curve space further into the distance to baseline space.

From the efficiency curve property 2, the impact of x on $f_S(x)$ is reflected on its slope at every $s \in S$, i.e. $\frac{df_s(x_1)}{ds} > \frac{df_s(x_2)}{ds}$, for $x_1, x_2 \in X$ and $x_1 \succ x_2$. Therefore, the sum of the difference of slope between x_b, x_k, should preserve their efficiency relation, which is represented as:

$$\int (\frac{df_s(x_k)}{ds} - \frac{df_s(x_b)}{ds}) ds \tag{2}$$

Formula (2) is exactly:

$$f_s(x_k) - f_s(x_b) \tag{3}$$

With Formula (3), let x_b be always the efficiency of baseline works, such that $f_S(x_b)$ is the baseline in $S \times P$. Therefore, 3 always represents the efficiency difference between the baseline and work k. Namely, Formula 3 calculates the comparison efficiency between work k and the baseline. Further more, if fix an s_i for any $s_i \in S$, $f_{s_i}(x_b)$ becomes a constant C_i. By choosing s_k be the s_i in Formula 3, where (s_k, p_k) is the sole obtainable value in $S \times P$ of the $f_S(x_k)$ of work k, We can represent the Formula (3) on work k by:

$$f_{s_k}(x_k) - C_k \tag{4}$$

To make Formula (4) more straightforward, we scale it into y_{kd}, which is the distance between the work k and the baseline, by simply multiplying a constant C, where $C = \cos(90° - \theta)$ with θ be the angle between the baseline and S-axis, then we have the equation:

$$C \cdot (f_{s_k}(x_k) - C_k) = y_k \tag{5}$$

Therefore, we obtain the comparison efficiency between work k and baseline from the function $\varphi : X \mapsto Y$ at x_k, which can be expressed by:

$$\varphi(x_k) = C \cdot (f_{s_k}(x_k) - C_k) \tag{6}$$

Finally, function $\varphi(x) = C \cdot (f_{s_k}(x) - C_k)$ is a homeomorphism because $\varphi(x)$ and φ^{-1} are both bijective and continuous, which can be deduced from the fact that $f : X \mapsto S \times P$ is a homeomorphism.

With the deduction, we show that though the efficiency of the work is a latent variable, we successfully express the mutual relation of different efficiency by using a homeomorphism function φ to transform this relation to the distance to baseline, where the baseline works as a bridge and also a comparison ground to compare different efficiency.

A.5 Feasibility

We consider simplicity and transferability as two main aspects when evaluating a selection method's feasibility. This section explains how these two aspects are qualitatively and reasonably evaluated using further refined indicators.

A.5.1 Simplicity

The simplicity of a data selection method takes into account (1) the number of LLMs trained in selector construction, (2) the number of steps in the selection algorithm given a completed selector, and (3) reproducibility, which is based on the open-source state of the code.

# of Trained LLMs. This indicator counts the number of LLMs trained during the selector construction stage. For example, AlpaGasus, InsTag, CaR, and Nuggets use purely ChatGPT (commercial LLM) or expert small model (355M in CaR) as a scorer or tagger, so the count is 0. IFD, SuperFiltering, and LESS train one warm-up model (LLaMA for IFD and LESS, GPT-2 for SuperFiltering) to obtain quality labels for candidate datasets, so the count is 1. MoDS trains one intermediate model for necessity evaluation, so the count is also 1. DEITA trains a complexity scorer and a quality scorer from ChatGPT-evolved data separately, so the count is 2. Instruction-Mining fine-tunes 129 models to obtain loss scores on 129 data

Methods	# Trained LLMs	# Algorithm Steps (# Using LLMs)	Rep.	**Simplicity**	Transferability		Extensibility	**Flexibility**	**Feasibility**
					Model Free	Dataset Free	ChatGPT/GPT-4 Free		
AlpaGasus	0	2(1)	✘	**1**	✔	✔	✘	**1**	**1**
InsTag	0	3(1)	✘	**2**	✔	✔	✘	**1**	**2**
CaR	0	4(0)	✔	**2**	✔	✔	✔	**2**	**3**
Nuggets	0	4(2)	✔	**2**	✔	✔	✔	**2**	**3**
SelectIT	0	4(3)	✔	**2**	✔	✔	✔	**2**	**3**
SuperFiltering	1*	3(1*)	✔	**3**	✘	✘	✔	**3**	**4**
IFD	1	3(1)	✔	**4**	✘	✘	✔	**3**	**5**
LESS	1	4(2)	✔	**5**	✘	✘	✔	**3**	**6**
DEITA	2	5(4)	✔	**6**	✘	✘	✘	**5**	**6**
InstructionGPT-4	30	4(1)	✔	**7**	✘	✔	✘	**5**	**7**
Instruction-Mining	129	4(0)	✘	**8**	✘	✔	✘	**5**	**8**
SHED	500	3(0)	✔	**8**	✘	✘	✔	**6**	**8**

Table 3: Feasibility rank considers both Simplicity rank and Flexibility rank. The former consists of three indicators: (1) # Trained LLMs; (2) # Algorithm Steps (# Times Using LLMs in the algorithm) and (3) Reproducibility, while the latter considers extensibility and transferability. The number in the bracket of the "# Algorithm Steps" column indicates the times of LLMs used in the selection algorithm. * indicates that SuperFiltering trains a GPT-2 instead of LLaMA.

subsets to rule-fit a linear loss score predictor, so the count is 129. The same count rule applies to InstructionGPT-4 since these two works are almost identical in method. Additionally, SHED fine-tunes 500 models during the proxy Shapley value calculating stage, so the count is 5.

# of Algorithm Steps. The following pseudo algorithms help count the steps in the selecting stage, where the number in the bracket in the table is the number of LLMs used. For example, based on the Algorithm 3, AlpaGasus performs first ChatGPT scoring and then ranking to get the final selected subset, which consists of 2 steps with 1 LLM usage.

Reproducibility. "✔" means the code is open-source on GitHub, "✘" means the opposite. Additionally, AlpaGasus has been open-source by others but not by the authors, and InsTag provides a demo on ModelScope and checkpoints on HuggingFace, but no codes are open-sourced. Thus, we consider them to be close-source.

A.5.2 Flexibility

The flexibility of a selection method considers both transferability and extensibility. The former is determined by whether the method can be transferred to other methods without re-training for every new model choice or candidate dataset choice, while the latter is by whether proprietary models like ChatGPT/GPT4 are used in the selection process.

Model free means the selector model can be replaced with any other model without harming the efficiency of the data selector conceptually. For example, AlpaGasus allows users to replace the ChatGPT with other models for direct scoring, which has the downside of over-reliance on proprietary models. Similarly, InsTag prompts GPT for annotation; CaR uses an expert small model for expert-aligned scoring; and Nuggets prompts LLaMA for conditional loss, which does not need to re-train the selector model if changed. Superfiltering, IFD, and LESS need to warm up a model with a portion of the candidate dataset, and thus, changing the model requires a re-warm-up, and thus not Model-free. DEITA trains two separate scorers from LLaMA, replacing LLaMA with another model requires re-training the scorers with the same seed-evolved data. InstructionMining and InstructionGPT4 need to re-train the two sets of models for the loss estimation, therefore they are not model-free. SHED does actual fine-tuning to calculate Shapley values, while MoDS trains an intermediate model for necessity evaluation. Therefore both are not model-free.

Dataset free means the change of the candidate dataset won't require re-training the selector, which means the candidate dataset is not used as training data in the construction of the data selector. For example, from AlpaGasus to SelectIT in the table, all these works use the candidate dataset directly for score obtaining without training, so they are dataset-free. For instructionMining/GPT4, they use the mapping function to estimate loss on any candidate dataset. For work like SuperFiltering, IFD, LESS, DEITA, they all rely on the information within given candidate dataset and needs warm-up, re-training is necessary if dataset is replaced. SHED calculates Shapley by iteratively with iterative fine-tuning of the model.

A.6 Algorithms Summary

We summarize the key selection algorithm in a highly abstract way to facilitate the qualitative evaluation of **Simplicity** indicator, where algorithm

steps are counted. The following summary uses a modular way to count algorithm steps, attempting to ensure each step is of similar difficulty while capturing all the necessary steps.

Algorithm 1 IFD & SuperFiltering

1: **Init** D = Candidate Dataset, S = Backbone, U = LLM Usage
2: Use $D' \in D$ to to warm up S
–> pre-experienced S'
3: Use S' to generate IFD/Perplexity score on D (U+=1)
–> each sample with score
4: Do score ranking and pick top K
5: **Return** Selected Subset

Algorithm 2 SelecIT

1: **Init** D = Candidate Dataset, S = Backbone, U = LLM Usage, P = Prompt Templates
2: Use each prompt in P to annotate each sample
–> Prompt-attached D_p
3: Use each model in S to score each sample in D_p (U+=3)
–> D_p with score
4: Aggregate scores for each sample in D
–> each sample in D has an aggregated score
5: Do score ranking and pick top K
6: **Return** Selected Subset

Algorithm 3 AlpaGasus

1: **Init** D = Candidate Dataset, S = ChatGPT, U = LLM Usage
2: Use S to score D (U+=1)
–> sample with score
3: Do score ranking and pick top K
4: **Return** Selected Subset

Algorithm 4 InsTag

1: **Init** D = Candidate Dataset, S = ChatGPT, U = LLM Usage
2: Use S to tag D (U+=1)
–> sample with tags
3: Do tag normalization
–> sample with tag statistics
4: Do complexity-first diverse sampling
5: **Return** Selected Subset

Algorithm 5 DEITA

1: **Init** D = Candidate Dataset, S = Backbone, U = LLM Usage
2: Use evolved datasets to train two Ss (U+=1)
–> complexity scorer model S_c and quality scorer model S_q
3: Use S_c to score D (U+=1)
–> instruction with complexity score
4: Use S_q to score D (U+=1)
–> output with quality score
5: Multiply two scores and rank
–> ranked sample
6: Do score-first, diversity-aware selection (U+=1)
7: **Return** Selected Subset

Algorithm 6 CaR

1: **Init** D = Candidate Dataset, S = Expert Model(355M), U = LLM Usage=0
2: Use S to score D
–> sample with score
3: Do score ranking and pick top n1
–> selected dataset D_{n1}
4: Do clustering on D
–> K subsets D'
5: Do score ranking in each D' and pick top n2 in each D'
–> selected dataset D_{K*n2}
6: **Return** $D_{n1} + D_{K*n2}$

Algorithm 7 Instruction-Mining

Init D_c = Candidate Dataset, D_t = Training Dataset, S = Linear Selector, U = LLM Usage

Use vectorized D_t to train a linear selector
–> trained S
Do vectorization on D_c with indicators
–> vectorized D_v
Use S to predict loss on D_v
–> sample with loss score
Do score ranking and pick top K
Return Selected Subset

Algorithm 8 InstructionGPT-4

Init D_c = Candidate Dataset, D_t = Training Dataset, S = Transformer model, U = LLM Usage
Use vectorized D_t to train a self-attention NN (U+=1)
–> trained S
Do vectorization on D_c with indicators
–> vectorized D_v
Use S to predict loss on D_v
–> sample with loss score
Do score ranking and pick top K
Return Selected Subset

Algorithm 9 Nuggets

Init D = Candidate Dataset, S = Backbone, U = LLM Usage
Prompt S with zero-shot D (U+=1)
–> sample with ZeroShotScore
Prompt S with one-shot D (U+=1)
–> sample with OneShotScore
OneShotScore - ZeroShotScore
–> sample with GoldenScore
Do score ranking and pick top K
Return Selected Subset

Algorithm 10 SHED

Init D = Candidate Dataset, S = Backbone, U = LLM Usage
Do clustering on embedded D
–> Proxy dataset from each cluster D_p
Calculate Shapley Value on D_p
–> D_p with score
Do optimization-aware sampling on D according to scores
–> Selected Subset
Return Selected Subset

Algorithm 11 MoDS

Init D = Candidate Dataset, S_s = S-BERT Scorer, S_b = Backbone (LLaMA2), U = LLM Usage
Use S_s to score D, rank and pick top K on D
–> High-quality Subset D_h
Do ranking and pick top k on D_h
–> Seed Instruction Data D_s
Use D_s to warm up S_b
–> Initialized LLM S_i
Use S_i to do inference on D_h (U+=1)
–> Generated dataset D_g
Use S_s to score D_g
Compare scores of sample in D_h and D_g, pick low-score in D_g
–> Necessary dataset D_n
Return D_s + D_n

Algorithm 12 LESS

Init D_c = Candidate Dataset, D_t = Target Dataset, S = Backbone, U = LLM Usage
Use $D'_c \in D_c$ to LoRA warm up S
–> LoRA Model S'
Use S' to get gradients of D_c (U+=1)
–> gradient store of D_c
Use S' to get gradients of D_t (U+=1)
–> gradient store of D_t
Do gradient-similarity-based selection
Return Selected Subset