CPQS-Tuning: Model Perception-Based Fast and Efficient Instruction Fine-Tuning

Anonymous ACL submission

Abstract

Instruction fine-tuning is a key technique for improving the performance of large language models (LLMs), but it is often limited by low-quality and redundant data. Recent studies suggest that a small amount of high-quality data can yield significant performance gains. However, existing data selection methods may not be suitable for rapidly evolving LLMs, and each model may require distinct types of high-quality data. To address these issues, we propose a novel perspective: the hidden states of LLMs implicitly assess the quality of the training data. Building on this insight, we introduce a new instruction-tuning data evaluation metric, the Contrastive Perception Quality Score (CPQS), and a dataset filtering approach based on this metric. Experimental results demonstrate that, when trained on less than 2% of the dataset (1,000 samples) from the general-purpose Alpaca and Alpaca_GPT4 datasets, our method outperforms models trained on the full dataset. Furthermore, our approach surpasses current state-of-the-art data selection methods in terms of performance. In downstream tasks, our method achieves an average performance improvement of over 3% compared to models trained with leading data selection algorithms, across multiple benchmark tests, including GSM8K, HumanEval, and HumanEval-Plus.

1 Introduction

Large Language Models (LLMs) (Brown et al., 2020; Chiang et al., 2023; Yang et al., 2024a; Zeng et al., 2024), such as ChatGPT (Ouyang et al., 2022a; OpenAI, 2023) have led to a groundbreaking shift in the realm of artificial intelligence in recent years. These models excel in understanding and handling a wide array of complex language tasks. A critical factor behind their success is instruction tuning. Instruction tuning (Ouyang et al., 2022b; Yu et al., 2023; Sun et al., 2023; Ding et al., 2023) enables models to accurately follow user instructions and exhibit outstanding performance on multiple downstream tasks (Wang et al., 2023a; Zhou et al., 2024; Guo et al., 2024; Ren et al., 2024; Sun et al., 2025).

Early studies focused on building high-quality datasets, relying on expert-designed responses (Khashabi et al., 2020; Ye et al., 2021; Wang et al., 2022), but these efforts faced limitations in terms of labor and cost. More recent research has employed a powerful teacher LLM to generate data (Wang et al., 2023b; Li et al., 2024a; Lee et al., 2024), though the quality of such data depends heavily on model performance and is challenging to control. Zhou et al. introduced the LIMA model (Zhou et al., 2023), demonstrating that as few as 1,000 high-quality instructions can significantly improve model performance. This suggests that pretraining has already endowed models with substantial instruction-following capability, making only a small amount of high-quality data sufficient for fine-tuning.

Building on this idea, many subsequent studies (Cao et al., 2023b; Chiang et al., 2023; Liu et al., 2024c; Chen et al., 2023a) have explored how to select a small, high-quality subset from large-scale datasets generated by teacher models. For example, some researchers have attempted to use advanced LLMs (Chen et al., 2024; Lu et al., 2024) or reward models (Bukharin et al., 2024) to score data and filter it based on these scores. Other studies have analyzed data quality from multiple dimensions (Du et al., 2023; Wu et al., 2023; Li et al., 2024c; Yu et al., 2024; Li et al., 2024d) and selected data according to defined quality metrics. Additionally, some research (Li et al., 2024b) has explored using smaller, weaker models to filter datasets for fine-tuning larger, more powerful models. However, these existing methods typically rely on external evaluation models or manually designed evaluation metrics, often lacking in-depth consideration of

the internal architecture and intrinsic features of large language models. This oversight may lead to inaccurate data quality assessment and inefficient filtering processes due to differences in training tasks.

In contrast to these approaches, this paper proposes the idea that the hidden states of Large Language Models (LLMs) contain an implicit evaluation of the quality of the training data, based on an analysis of the internal features of LLMs. Since each LLM requires different high-quality data depending on its architecture and training corpus, analyzing the hidden states of such models can provide a more accurate assessment of the training data quality. To this end, we propose a new instruction-tuning data evaluation metric, **C**ontrastive **P**erception **Q**uality **S**core (**CPQS**), and introduce a method for dataset filtering based on this metric. Our approach consists of the following four steps. ① We first construct an instruction fine-tuning dataset generated by LLMs with varying performance, containing both positive and negative samples. ② We then extract the hidden states of the target fine-tuning model for each instruction. ③ Based on these hidden states, we train a Convolutional Neural Network (CNN) model to identify the current testing sample is effective (of high quality) or not. ④ During the prediction phase, the CNN model analyzes the hidden states perceived by the LLM for each instruction, generating a prediction probability and classification result. The prediction probability classified as effective is referred to as CPQS, which serves as the criterion for dataset filtering.

Through extensive experiments, we validate that our algorithm performs excellently in data selection for general and downstream domain tasks. In the general domain, we tested using the Alpaca and Alpaca_GPT4 datasets (Taori et al., 2023a). The selected data amount was less than 2% of the original dataset (1,000 entries), yet it outperformed models trained on the entire dataset. Our algorithm was proven to surpass various state-of-the-art algorithms on multiple LLMs. In downstream task domains, such as mathematical problems and programming tasks, our algorithm outperformed existing state-of-the-art algorithms by an average of 3 percentage points on benchmark tests like GSM8K, HumanEval (Chen et al., 2021), and HumanEval-Plus (Liu et al., 2023, 2024b) with the same data scale.

The main contributions of this paper can be summarized as follows:

- We revealed that the hidden states of LLMs contain an implicit evaluation of the quality of training data and introduced a new instruction-tuning data evaluation metric, the Contrastive Perception Quality Score (CPQS).
- We proposed an efficient and accurate data selection method based on the LLM's own contrastive perception quality score, significantly enhancing instruction-tuning performance.
- This paper presents extensive empirical studies using two general fine-tuning datasets and two task-specific datasets. The results show that the proposed data selection method achieves optimal performance in both general tasks and specific areas such as mathematics and programming.

2 Proposed Methods

This section provides a detailed description of our method. The core of our approach is to train an external CNN model by extracting the hidden states perceived by a large language model on high-quality and low-quality general fine-tuning datasets for each data point. This model will be used to analyze the implicit evaluation of the training data within the hidden states of the large language model. The method consists of four key steps, as shown in Figure 1. We will discuss each step in depth and analyze them accordingly.

2.1 Construction of Positive and Negative Sample Datasets

We first randomly selected 5,000 data points from the Alpaca_GPT4 dataset to form the positive sample dataset. The Alpaca_GPT4 dataset is a high-quality, general-domain fine-tuning dataset containing 52,000 entries. Each consisting of a triplet: $(Instruction, [Input], Response)$, where Instruction describes the task to be performed, Input provides additional context for task execution, and Response is the response generated by GPT-4 to the given instruction.

To construct the negative sample dataset, we chose two additional models: Llama-3.2-1B-Instruct (Dubey et al., 2024) and Qwen2.5-1.5B-Instruct (Yang et al., 2024b), which are among the smallest language models currently available. The process for constructing the negative sample dataset is as follows:

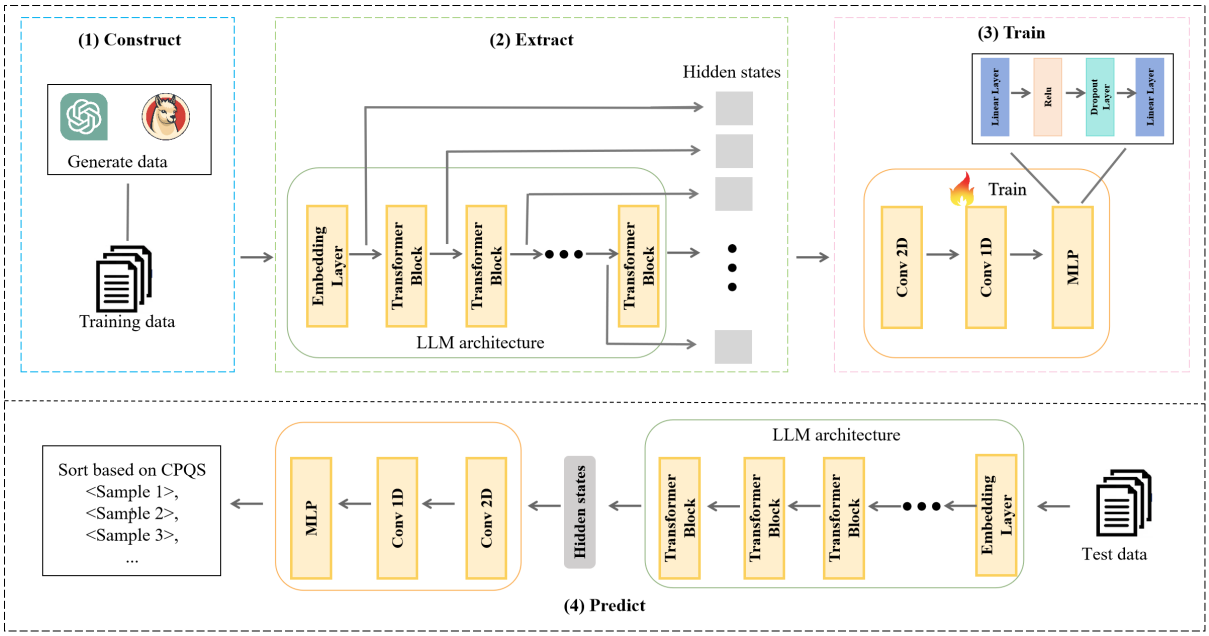

Figure 1: Overall algorithm architecture diagram

1. **Extract Instructions and Inputs**: Extract the $(Instruction, [Input])$ part from each entry in the Alpaca_GPT4 dataset.

2. **Generate Responses**: Use Llama-3.2-1B-Instruct and Qwen2.5-1.5B-Instruct to generate corresponding Response values, forming new triplets: $(Instruction, [Input], Response)$.

3. **Sample Extraction**: Extract 5,000 entries from the generated responses of each model, totaling 10,000 data points as the negative sample dataset.

Finally, we mix the positive and negative sample datasets to create a complete training dataset.

2.2 Extraction of Hidden States

For the collected training dataset, we concatenate the Instruction and Input parts of each entry and use this concatenated value as the "user" input to the model, while the Response part is used as the "assistant" input. This combined entry is then passed to the model to obtain the hidden states across all layers of the model. We retain only the hidden state corresponding to the Response part of each entry. This choice is made because the evaluation of the fine-tuning dataset's quality primarily depends on the quality of the Response.

2.3 Training of the External CNN Model

The core idea of our algorithm is that the hidden states generated by the LLM contain an implicit evaluation of the quality of the training data. To analyze the LLM's quality assessment of the training data, we propose using an external model to interpret the hidden state vectors produced by the LLM. The external model is based on a simple Convolutional Neural Network (CNN) architecture, which can effectively capture the detailed information within the hidden state vectors.

During the training process, we use the hidden states of each sample obtained in the previous section, along with their corresponding positive and negative labels, for training. The ratio of positive to negative samples is 1:2, totaling 15,000 samples. Drawing from the concept of contrastive learning, the goal is to train the CNN model as a binary classification task. This enables the model to distinguish between the different information perceived by the hidden states of the LLM for good and bad samples. In doing so, the CNN model can more accurately reflect the LLM's evaluation of the training data quality.

2.4 Prediction of Contrastive Perception Quality Score (CPQS)

In the prediction phase, we introduce the Contrastive Perception Quality Score (CPQS), which serves as an indicator to evaluate the training quality of each instruction-following sample. A higher

$\mathcal{CPQS}$ indicates that the LLM places more importance on the data, suggesting that the training effect of the data is better and its quality is higher.

The process for calculating the $\mathcal{CPQS}$ is as follows:

For each instruction-following sample to be filtered, we concatenate the Instruction and Input parts as the "user" input to the model, while the Response part serves as the "assistant" input. The entire entry is then passed through the LLM to extract the hidden states for the Response part. These hidden states are then fed into the pre-trained CNN model, which outputs the predicted class and corresponding probability values. We focus primarily on the probability that the predicted class is 1, which reflects how likely it is that the LLM interprets the data entry as a high-quality sample. In other words, the higher this probability, the greater the assessed data quality. This value serves as the quality evaluation metric for the data.

Based on these evaluation metrics, we rank the entire dataset and select the top-ranked data points for further processing. The calculation formula for $\mathcal{CPQS}$ is as follows:

$$\mathcal{CPQS}_i = p_i = f\left(\mathbf{x}_i\right)$$

where p_i is the predicted probability for the i-th sample, representing the likelihood that the sample belongs to the positive class, and $f\left(\mathbf{x}_i\right)$ is the output of the LLM's hidden state vector for the i-th entry, processed by the CNN model to produce the probability that the sample belongs to class 1.

After calculating the $\mathcal{CPQS}$ for all samples, we sort the entire dataset based on these probabilities in descending order and select the top K samples for further processing. The specific selection process is represented as:

$$\mathcal{D}_{\text{selected}} = \mathbf{top}_K\left(\{\mathcal{CPQS}_i\}_{i=1}^{N}\right)$$

where $\mathcal{D}_{\text{selected}}$ is the set of the top K selected samples from the dataset based on their $\mathcal{CPQS}$ values, and $\mathbf{top}_K$ denotes selecting the top K samples after sorting.

3 Experimental Setup

3.1 Datasets and Models

To assess our method's effectiveness, we utilized four datasets and three open-source LLMs. The datasets cover both general and specific task domains, while the models include foundational and dialogue-optimized variants.

For the general domain, we selected the Alpaca data set (Taori et al., 2023b) and the Alpaca_GPT4 data set. The Alpaca Dataset (Taori et al., 2023a) contains 52,000 instruction-response pairs generated via the self-instruct method, categorized as a medium-quality classic dataset. The Alpaca_GPT4 Dataset builds on this by using GPT-4 to regenerate responses, significantly improving data quality and classifying it as a high-quality classic dataset.

In the downstream task domain, we chose the GSM8K Dataset (Cobbe et al., 2021) and the Magicoder-Evol-Instruct-110K Dataset (Wei et al., 2024). The GSM8K Dataset (Cobbe et al., 2021), released by OpenAI, comprises approximately 8,500 elementary math problems designed to evaluate and enhance models' multi-step reasoning abilities, with 7,500 for training and 1,000 for testing. The Magicoder-Evol-Instruct-110K Dataset (Wei et al., 2024) focuses on programming tasks, including code generation and problem-solving, featuring around 110,000 entries across various programming languages and real-world scenarios to improve models' programming comprehension and generation skills.

For model selection, we used three open-source LLMs: Llama 2-7B (Touvron et al., 2023), Llama 2-7B-Chat (Touvron et al., 2023), and Qwen2.5-7B-Instruct (Yang et al., 2024b). Llama 2-7B (Touvron et al., 2023) by Meta is a foundational model with 7 billion parameters, suitable for diverse language tasks with stable performance. Llama 2-7B-Chat (Touvron et al., 2023), also with 7 billion parameters, is optimized for dialogue, providing more fluent and natural interactions. Qwen2.5-7B-Instruct (Yang et al., 2024b), developed by the Qwen team, is a versatile model focused on text and code generation, mathematical computations, and instruction following, excelling in various benchmarks and ranking among the most advanced open-source models available.

3.2 Comparison Algorithms

To validate the effectiveness of our algorithm, we selected three state-of-the-art data selection methods as comparison algorithms:

ALPAGASUS (Chen et al., 2024): Chen et al. leverage LLMs, such as ChatGPT, to automatically identify and filter low-quality data.

MoDS (Du et al., 2023): Du et al. propose a data selection strategy based on quality, coverage,

and necessity criteria.

Superfiltering (Li et al., 2024b): Li et al. introduce a method that uses a smaller model to filter data based on instruction-following difficulty before fine-tuning a larger model.

3.3 Implementation Details

We conducted our experiments on a platform equipped with two NVIDIA RTX 4090 GPUs. We leveraged the LoRA-based fine-tuning method within the Llama-Factory framework (Zheng et al., 2024) developed by Zheng et al. During the supervised fine-tuning (SFT) phase, we set the training precision to bf16, the number of training epochs to three, the learning rate to 5e-5, a batch size of 16, and a maximum sequence length of 2048.

For the deployment and inference of our model, we utilized vLLM (Kwon et al., 2023). During inference, we configured the temperature to 0, maintained the precision at bf16, and set the maximum sequence length to 2048.

3.4 Evaluation Metrics

3.4.1 General Domain Evaluation Standards

We evaluate our method using four test sets from Chen et al.: Koala (Vu et al., 2023), WizardLM (Xu et al., 2024), Self-instruct (Wang et al., 2023b), and Vicuna (Chiang et al., 2023), containing 180, 218, 252, and 80 instructions, respectively, covering diverse domains such as mathematics, programming, and writing. Models generate responses based on these instructions. Responses are evaluated by GPT-4o, scoring from 1 to 10 based on relevance and accuracy. Two rounds of evaluation are conducted for responses from each model pair, using varied prompt orders to reduce position bias. Model performance is compared as follows:

- **Win**: Better in both evaluations, or better in one and tied in the other.
- **Tie**: Tied in both evaluations or mixed results.
- **Loss**: Inferior in both evaluations, or worse in one and tied in the other.

Additionally, we used two datasets to assess general domain performance:

MMLU (Hendrycks et al., 2021): A benchmark with 14,000 questions across 57 domains widely used to track LLM capabilities.

MMLU-Pro (Wang et al., 2024): An enhanced version featuring over 12,000 challenging questions in 14 domains, including biology, business, chemistry, and more.

3.4.2 Evaluation Criteria for Downstream Task Domains

For downstream task evaluations, we mainly focus on two domains: mathematical reasoning and code generation, using different standards and datasets to assess the models' actual capabilities. We use GSM8K Datetset (Cobbe et al., 2021) to evaluate mathematical abilities. The test set consists of 1,000 arithmetic, algebra, and geometry problems at middle and high school levels, designed to assess the model's mathematical reasoning and problem-solving skills. For code generation, we employ the HumanEval (Chen et al., 2021) and HumanEval-Plus (Liu et al., 2023, 2024b) datasets. HumanEval contains 164 programming problems, testing the model's basic code generation and algorithm understanding capabilities. HumanEval-Plus, as a more challenging extended version, includes more complex tasks, evaluating the model's code generation accuracy, reasoning abilities, and adaptability. It also tests the model's stability and performance under diverse input conditions through a variety of data points.

4 Experimental Result

4.1 General Domain Evaluation

In this section, we present the comparison results of our method against three other state-of-the-art data selection methods across different models and datasets. We selected the Alpaca_dataset and Alpaca_GPT4_dataset as test datasets and filtered out the top 1000 highest-quality data points from each dataset. For each selected dataset, we trained a model and evaluated it on the test set using GPT-4o as a judge, comparing the performance of our algorithm with the other three methods. It is worth noting that the ALPAGASUS algorithm could only filter out 9K high-quality data points, so we randomly selected 1000 data points from this set for comparison.

As shown in Figure 2, our method consistently outperforms the other algorithms. The model trained with our selected 1000 data points also outperforms the one trained with the full 52K dataset.

To assess the generalization ability, we evaluated each model using the MMLU and MMLU-Pro

Figure 2: Performance Comparison of Data Selection Methods on the Llama2-7B Model Using the Alpaca Dataset.

	Self	MoDs	Alp.	Sup.	Full
M	**41.35**	35.91	39.97	38.37	40.61
M-Pro	12.08	**12.92**	12.75	12.20	12.57
Avg.	**26.72**	24.42	26.36	25.29	26.59

Table 1: Performance of Models Trained with Different Algorithms on MMLU and MMLU-Pro Datasets. (*Note:* "M" is short for MMLU, "M-Pro" is short for MMLU-Pro, "Alp." is short for Alpagaus, "Sup." is short for Superfiltering, and "Full" refers to the full dataset.)

datasets. As shown in Table 1, the model trained with our algorithm achieved a score of 41.35 on MMLU, 3.27 points higher than the other algorithms and 0.74 points higher than the full dataset model. In MMLU-Pro, while the MoDs algorithm performed the best with a score of 12.92, our model achieved an average score of 26.715 across both datasets, 1.36 points higher than the other algorithms and 0.125 points higher than the full dataset model.

We further verified our algorithm's effectiveness on the higher-quality Alpaca_GPT4_dataset. As shown in Figure 3, our method again outperforms the other algorithms, with the model trained on 1000 selected data points outperforming the one trained on the full 52K dataset. In the MMLU evaluation (Table 2), our method achieved a score of 44.76, 3.31 points higher than the other algorithms and 2.61 points higher than the full dataset model.

While models trained on the Alpaca_GPT4_dataset perform better than those on the Alpaca_dataset in MMLU, all algorithms performed worse in MMLU-Pro, likely due to the dataset's increased difficulty. Despite this, our method outperformed the average of the other algorithms by 1.08 points and the full dataset model by 1.01 points, highlighting the significant performance boost with higher-quality data.

We also tested our algorithm on the Llama2-7B-Chat and Qwen 2.5-7B-Instruct models. As shown in Appendix A, our method consistently outperforms the other algorithms, with models trained on 1000 selected data points outperforming those trained on the full dataset. Notably, in the Qwen 2.5-7B-Instruct model, trained with the full Alpaca dataset (Taori et al., 2023a), the model achieved an average score of 49.385, while our model trained with 1000 data points scored 62.815, surpassing the full dataset model by 13.43 points. *This highlights the significant impact of data quality on model performance, particularly with more advanced models. When combined with advanced models, our algorithm delivers even more outstanding results.*

We also examined the iterative nature of our algorithm and the factors that affect its performance. Detailed experiments can be found in Appendix B and Appendix C.

4.2 Downstream Task Evaluation

We evaluated the performance of our algorithm for downstream task dataset selection, focusing on Llama 2-7B-Chat and Qwen 2.5-7B-Instruct models. Common methods for downstream task

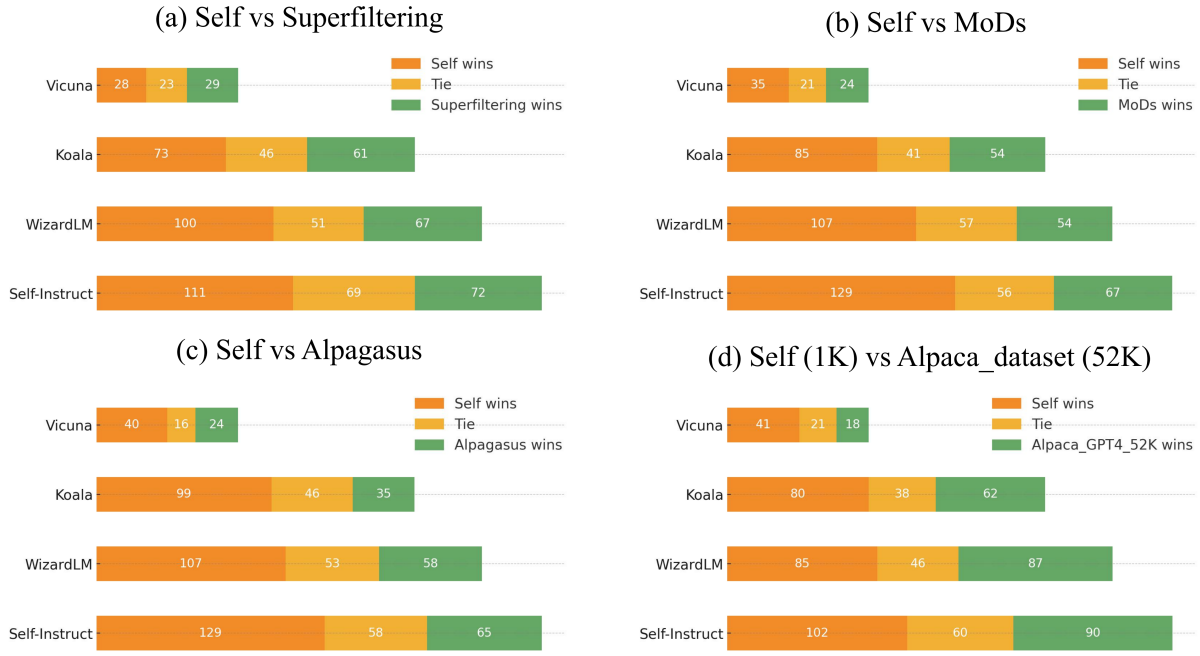

Figure 3: Performance Comparison of Data Selection Methods on the Llama2-7B Model Using the Alpaca_GPT4_Dataset.

	Self	MoDs	Alp.	Sup.	Full
M	**44.76**	41.57	41.25	41.54	42.15
M-Pro	10.00	11.62	**12.35**	10.48	10.60
Avg.	**27.38**	26.60	26.30	26.01	26.38

Table 2: Performance of Models Trained with Different Algorithms on MMLU and MMLU-Pro Datasets. (*Note:* "M" is short for MMLU, "M-Pro" is short for MMLU-Pro, "Alp." is short for Alpagaus, "Sup." is short for Superfiltering, and "Full" refers to the full dataset.)

training involve fine-tuning pre-optimized models. To validate our approach, we used the GSM8K (Cobbe et al., 2021) and Magicoder-Evol-Instruct-110K (Wei et al., 2024) datasets for filtering, reducing GSM8K to 500 data points and Magicoder-Evol-Instruct-110K to 1000 data points. We assessed performance on the GSM8K test set, Humaneval, and HumanEval-Plus datasets. For the ALPAGASUS algorithm, we used the GPT-4o-mini model for data selection, as it outperforms GPT-4 in both effectiveness and cost.

Table 3 presents results on the GSM8K Dataset, where our algorithm outperforms the others by 4.17 points on Llama 2-7B-Chat and 2.9 points on Qwen 2.5-7B-Instruct. Notably, the model trained on the full GSM8K dataset underperforms on Qwen 2.5-7B-Instruct (score 76.04), while the model trained on 500 selected data points achieves 84.91, surpassing the full dataset by 8.87 points. *This highlights that more advanced models require higher-quality datasets for optimal downstream task adaptation.*

Tables 4 and 5 show results with the Magicoder-Evol-Instruct-110K dataset. On Llama 2-7B-Chat, our algorithm outperforms others by 4.3 points on average, and models trained on 1000 data points from other algorithms performed worse than the original. In contrast, the model trained on 1000 points selected by our algorithm improved by 2.45 points. On Qwen 2.5-7B-Instruct, the model trained with our selected data outperformed others by 1.47 points on average. However, all models trained on the filtered Magicoder-Evol-Instruct-110K dataset showed performance degradation, likely due to its lower quality for this model. *These results further confirm that our algorithm excels in dataset selection, especially when high-quality datasets are available, leading to substantial improvements in model performance.*

5 Related Work

5.1 Data Selection Strategies During Fine-Tuning

The goal of instruction tuning (Wei et al., 2022; Sanh et al., 2022; Longpre et al., 2023; Liu et al., 2024a) is to help large language models better understand human task requirements. Early research

Model	Original	Self	MoDs	Alpagasus	Superfiltering	Full
Llama 2-7B-Chat	24.56	25.25	23.05	23.12	17.06	**35.56**
Qwen2.5-7B-Instruct	79.76	**84.91**	80.74	81.27	84.00	76.04

Table 3: Performance Evaluation of Models Trained with Different data selection Methods on the GSM8K Dataset in the Mathematics Domain.

	Ori.	Self	MoDs	Alp.	Sup.
H	13.4	**16.5**	12.2	12.2	10.0
H-Plus	11.6	**13.4**	9.8	10.4	9.1
Avg.	12.5	**14.95**	11	11.3	9.55

Table 4: Performance Evaluation of Models Trained with Different Algorithms on Llama2-7B-Chat in the Code Domain (pass@1). (*Note:* "H" is short for Humaneval, "Ori." is short for Original, "Alp." is short for Alpagaus, "Sup." is short for Superfiltering.)

	Ori.	Self	MoDs	Alp.	Sup.
H	**82.9**	80.0	78.7	78.0	79.3
H-Plus	**75.6**	74.4	72.6	72.6	73.2
Avg.	**79.25**	77.2	75.65	75.3	76.25

Table 5: Performance Evaluation of Models Trained with Different Algorithms on Qwen 2.5-7B-Instruct in the Code Domain. (*Note:* "H" is short for Humaneval, "Ori." is short for Original, "Alp." is short for Alpagasus, "Sup." is short for Superfiltering.)

primarily focused on building large-scale instruction datasets, but studies like LIMA have shown that only a small amount of high-quality data is needed during instruction fine-tuning to achieve good results. Existing data selection methods can be classified into four categories: indicator-based methods, trainable LLM-based methods, powerful LLM-based methods, and small-model-based methods.

Indicator-based methods use a metric system to identify multiple evaluation indicators to comprehensively assess data quality (Cao et al., 2023a; Wei et al., 2023). Trainable LLM-based methods treat the large language model as a trainable data selector, processing and assigning scores to each instruction fine-tuning data for selection (Li et al., 2024c; Chen et al., 2023b). Powerful LLM-based approaches, such as those using models like ChatGPT, typically design prompt templates and leverage the model's capabilities to quantitatively evaluate the quality of data samples (Chen et al., 2024; Liu et al., 2024c). Finally, small-model-based methods involve using external small models to score the data or projecting the data samples into vectors with a small model for further processing and selection (Chen et al., 2023a; Li et al., 2024b).

5.2 Performance Evaluation of LLMs

The evaluation of LLMs is typically done through automatic evaluation, human evaluation, and using LLMs as evaluators. Automatic evaluation relies on predefined criteria and quantitative assessment (Hendrycks et al., 2021; Wang et al., 2024; Chen et al., 2021). Human evaluation focuses on qualitative aspects like clarity, consistency, and factual accuracy, and is essential for quality assessment (van der Lee et al., 2021; Zheng et al., 2023). However, due to its time and labor demands, using powerful LLMs (Chen et al., 2024; Huang et al., 2024) to evaluate other LLMs has become a popular approach in recent years.

6 Conclusion

This paper addresses the issue of low-quality and redundant data in LLM instruction fine-tuning. We propose a data selection method using the Contrastive Perception Quality Score, a metric that evaluates data quality by analyzing LLM hidden states. Our method filters high-quality data subsets, enhancing fine-tuning performance.

Experimental results show that our approach achieves superior performance with less than 2% of the original data (1,000 samples) compared to models trained on the full dataset. It also outperforms existing data selection techniques at equal data scales. In downstream tasks, such as mathematical problem solving (GSM8K) and programming (HumanEval, HumanEval+), our method provides a 3% average performance improvement over current state-of-the-art data selection algorithms.

7 Limitations

This paper presents a data selection method using the Contrastive Perception Quality Score to filter high-quality data. However, there are still areas for improvement: (1) The method focuses on data quality but does not consider data diversity. (2) While effective during fine-tuning, the method needs fur-

ther exploration for incremental pre-training and reinforcement learning stages of LLMs.

References

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

Alexander Bukharin, Shiyang Li, Zhengyang Wang, Jingfeng Yang, Bing Yin, Xian Li, Chao Zhang, Tuo Zhao, and Haoming Jiang. 2024. Data diversity matters for robust instruction tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 3411–3425. Association for Computational Linguistics.

Yihan Cao, Yanbin Kang, and Lichao Sun. 2023a. Instruction mining: High-quality instruction data selection for large language models. *CoRR*, abs/2307.06290.

Yihan Cao, Yanbin Kang, Chi Wang, and Lichao Sun. 2023b. Instruction mining: Instruction data selection for tuning large language models. *arXiv preprint arXiv:2307.06290*.

Hao Chen, Yiming Zhang, Qi Zhang, Hantao Yang, Xiaomeng Hu, Xuetao Ma, Yifan Yanggong, and Junbo Zhao. 2023a. Maybe only 0.5% data is needed: A preliminary exploration of low training data instruction tuning. *CoRR*, abs/2305.09246.

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. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *CoRR*, abs/2107.03374.

Yongrui Chen, Haiyun Jiang, Xinting Huang, Shuming Shi, and Guilin Qi. 2023b. Tegit: Generating high-quality instruction-tuning data with text-grounded task design. *CoRR*, abs/2309.05447.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6.

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

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 3029–3051. Association for Computational Linguistics.

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

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon,

Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y. Wu, Y. K. Li, Fuli Luo, Yingfei Xiong, and Wenfeng Liang. 2024. Deepseek-coder: When the large language model meets programming - the rise of code intelligence. *CoRR*, abs/2401.14196.

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

Hui Huang, Yingqi Qu, Jing Liu, Muyun Yang, and Tiejun Zhao. 2024. An empirical study of llm-as-a-judge for LLM evaluation: Fine-tuned judge models are task-specific classifiers. *CoRR*, abs/2403.02839.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 1896–1907. Association for Computational Linguistics.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP 2023, Koblenz, Germany, October 23-26, 2023*, pages 611–626. ACM.

Nicholas Lee, Thanakul Wattanawong, Sehoon Kim, Karttikeya Mangalam, Sheng Shen, Gopala Anumanchipalli, Michael W. Mahoney, Kurt Keutzer, and Amir Gholami. 2024. LLM2LLM: boosting llms with novel iterative data enhancement. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 6498–6526. Association for Computational Linguistics.

Ming Li, Lichang Chen, Jiuhai Chen, Shwai He, Jiuxiang Gu, and Tianyi Zhou. 2024a. Selective reflection-tuning: Student-selected data recycling for LLM instruction-tuning. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 16189–16211. Association for Computational Linguistics.

Ming Li, Yong Zhang, Shwai He, Zhitao Li, Hongyu Zhao, Jianzong Wang, Ning Cheng, and Tianyi Zhou. 2024b. Superfiltering: Weak-to-strong data filtering for fast instruction-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 14255–14273. Association for Computational Linguistics.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2024c. From quantity to quality: Boosting LLM performance with self-guided data selection for instruction tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 7602–7635. Association for Computational Linguistics.

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. 2024d. One-shot learning as instruction data prospector for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 4586–4601. Association for Computational Linguistics.

Fuxiao Liu, Xiaoyang Wang, Wenlin Yao, Jianshu Chen, Kaiqiang Song, Sangwoo Cho, Yaser Yacoob, and Dong Yu. 2024a. MMC: advancing multimodal chart understanding with large-scale instruction tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 1287–1310. Association for Computational Linguistics.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2023. Is your code generated by chatGPT really correct? rigorous evaluation of large language models for code generation. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Jiawei Liu, Songrun Xie, Junhao Wang, Yuxiang Wei, Yifeng Ding, and Lingming Zhang. 2024b. Evaluating language models for efficient code generation. In *First Conference on Language Modeling*.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2024c. What makes good data for alignment? A comprehensive study of automatic data

selection in instruction tuning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Shayne 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, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 22631–22648. PMLR.

Keming Lu, Hongyi Yuan, Zheng Yuan, Runji Lin, Junyang Lin, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2024. #instag: Instruction tagging for analyzing supervised fine-tuning of large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

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

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022a. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022b. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Yi Ren, Tianyi Zhang, Xurong Dong, Weibin Li, Zhiyang Wang, Jie He, Hanzhi Zhang, and Licheng Jiao. 2024. Watergpt: Training a large language model to become a hydrology expert. *Water*, 16(21):3075.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Lianshan Sun, Diandong Liu, Maoxue Wang, Yongyi Han, Yanqing Zhang, Biwei Zhou, Yi Ren, et al. 2025. Taming unleashed large language models with blockchain for massive personalized reliable healthcare. *IEEE Journal of Biomedical and Health Informatics*.

Zhiqing Sun, Yikang Shen, Qinhong Zhou, Hongxin Zhang, Zhenfang Chen, David D. Cox, Yiming Yang, and Chuang Gan. 2023. Principle-driven self-alignment of language models from scratch with minimal human supervision. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

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

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023b. Stanford alpaca: An instruction-following llama model.

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

Chris van der Lee, Albert Gatt, Emiel van Miltenburg, and Emiel Krahmer. 2021. Human evaluation of automatically generated text: Current trends and best practice guidelines. *Comput. Speech Lang.*, 67:101151.

Thuy-Trang Vu, Xuanli He, Gholamreza Haffari, and Ehsan Shareghi. 2023. Koala: An index for quantifying overlaps with pre-training corpora. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023 - System Demonstrations, Singapore, December 6-10, 2023*, pages 90–98. Association for Computational Linguistics.

Haochun Wang, Chi Liu, Nuwa Xi, Zewen Qiang, Sendong Zhao, Bing Qin, and Ting Liu. 2023a. Huatuo: Tuning llama model with chinese medical knowledge. *CoRR*, abs/2304.06975.

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

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. 2022. Super-naturalinstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 5085–5109. Association for Computational Linguistics.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *CoRR*, abs/2406.01574.

Jason Wei, Maarten Bosma, Vincent Y. 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 *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

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

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. 2024. Magicoder: Empowering code generation with oss-instruct. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Shengguang Wu, Keming Lu, Benfeng Xu, Junyang Lin, Qi Su, and Chang Zhou. 2023. Self-evolved diverse data sampling for efficient instruction tuning. *CoRR*, abs/2311.08182.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2024. Wizardlm: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

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

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024b. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. 2021. Crossfit: A few-shot learning challenge for cross-task generalization in NLP. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 7163–7189. Association for Computational Linguistics.

Yue Yu, Yuchen Zhuang, Jieyu Zhang, Yu Meng, Alexander J. Ratner, Ranjay Krishna, Jiaming Shen, and Chao Zhang. 2023. Large language model as attributed training data generator: A tale of diversity and bias. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Zhaojian Yu, Xin Zhang, Ning Shang, Yangyu Huang, Can Xu, Yishujie Zhao, Wenxiang Hu, and Qiufeng Yin. 2024. Wavecoder: Widespread and versatile enhancement for code large language models by instruction tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*

(Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024, pages 5140–5153. Association for Computational Linguistics.

Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Juanzi Li, Lei Zhao, Lindong Wu, Lucen Zhong, Mingdao Liu, Minlie Huang, Peng Zhang, Qinkai Zheng, Rui Lu, Shuaiqi Duan, Shudan Zhang, Shulin Cao, Shuxun Yang, Weng Lam Tam, Wenyi Zhao, Xiao Liu, Xiao Xia, Xiaohan Zhang, Xiaotao Gu, Xin Lv, Xinghan Liu, Xinyi Liu, Xinyue Yang, Xixuan Song, Xunkai Zhang, Yifan An, Yifan Xu, Yilin Niu, Yuantao Yang, Yueyan Li, Yushi Bai, Yuxiao Dong, Zehan Qi, Zhaoyu Wang, Zhen Yang, Zhengxiao Du, Zhenyu Hou, and Zihan Wang. 2024. Chatglm: A family of large language models from GLM-130B to GLM-4 all tools. *CoRR*, abs/2406.12793.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. LIMA: less is more for alignment. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Zhi Zhou, Jiang-Xin Shi, Peng-Xiao Song, Xiaowen Yang, Yi-Xuan Jin, Lan-Zhe Guo, and Yufeng Li. 2024. Lawgpt: A chinese legal knowledge-enhanced large language model. *CoRR*, abs/2406.04614.

A Performance Evaluation of Our Algorithm on Multiple Models and Datasets

Tables 6 and 7 along with Figures 4 and 5 display the performance of our algorithm on the Alpaca_GPT4 Dataset and the Alpaca Dataset (Taori et al., 2023a) using the Qwen2.5-7B-Instruct model. Similarly, Tables 8 and 9 along with Figures 6 and 7 show the performance of our algorithm on the Alpaca_GPT4 Dataset and the Alpaca Dataset using the Llama 2-7B-Chat model.

B Experiment on Iterative Model Training and Data Selection

We conducted experiments to assess the iterativeness of our algorithm. First, we trained a CNN model using our algorithm on the Qwen2.5-7B-Instruct dataset. This trained model was then used to predict and rank the Alpaca_GPT4 dataset. Based on the ranking, we extracted the top 5,000 and the last 10,000 data samples. We then retrained another CNN model using this subset to further filter 1,000 samples from the Alpaca_GPT4 dataset for additional training. As shown in Figure 8, the performance of the newly trained model demonstrated a significant improvement over the previous version.

C Impact of Hidden Layer Selection and Dataset Preferences

C.1 The Impact of Different Hidden Layers on Model Performance

In this section, we investigate the impact of different hidden layers of the model on data selection performance. To this end, we chose the Qwen 2.5-7B-Instruct model for experimentation and focused on analyzing the contribution of each layer's hidden states to the selection performance. Specifically, we used the hidden states from the first 9 layers, the middle 9 layers, the last 11 layers, and the final layer to train separate CNN models, and validated them on the GSM8K dataset and Magicoder-Evol-Instruct-110K Dataset (Wei et al., 2024).

As shown in Table 10, on the GSM8K Dataset (Cobbe et al., 2021), the model trained using the hidden states from the first 9 layers performed the best, with a score of 84.23. However, its performance was still not as good as the model trained with hidden states from all layers. On the Magicoder-Evol-Instruct-110K Dataset (Wei et al., 2024), the model trained with the hidden states from the last 11 layers performed the best, with an average score of 76.55, although it still lagged behind the performance of the model trained using all layers.

	Self	Mods	ALPAGASUS	Superfiltering	Original model	ALpaca(52k)
MMLU	73.32	60.52	70.66	67.17	**75.60**	61.89
MMLU-Pro	**52.31**	47.32	51.92	51.57	49.65	36.88
Average	**62.82**	53.92	61.29	59.36	62.63	49.39

Table 6: Performance of Models Trained with Different Algorithms on MMLU and MMLU-Pro using the ALpaca Dataset with Qwen2.5-7B-Instruct.

Figure 4: Performance Comparison of Data Selection Methods on the Qwen2.5-7B-Instruct Model Using the Alpaca Dataset.

C.2 Preferences of Different Models for High-Quality Datasets

In this section, we explore whether different LLMs have distinct preferences for high-quality datasets. To this end, we trained the models on high-quality datasets selected from the GSM8K dataset and Magicoder-Evol-Instruct-110K Dataset (Wei et al., 2024) using Llama 2-7B-Chat and Qwen 2.5-7B-Instruct. We then compared the performance of these models when exchanging datasets. Specifically, we trained Qwen 2.5-7B-Instruct and Llama 2-7B-Chat on each other's selected datasets and compared their performance with training on their own selected datasets.

As shown in Table 11 and Table 12, whether in the mathematical or coding domains, the models trained after swapping datasets did not perform as well as those trained on their original datasets. For both Llama 2-7B-Chat and Qwen 2.5-7B-Instruct, the high-quality dataset considered by one model did not yield the same results when used by the other model. Therefore, our experiment shows that different models exhibit significant differences in selecting high-quality datasets, with each model having its own definition of what constitutes a "high-quality dataset."

This finding is crucial for data selection and model optimization, as it suggests that the structure and characteristics of a model may significantly influence its evaluation of data quality when selecting datasets.

	Self	MoDs	Alpagasus	Superfiltering	ALpaca_GPT4 (52k)
MMLU	75.55	73.03	71.34	75.27	**75.60**
MMLU-Pro	51.66	50.73	**51.99**	50.65	49.65
Average	**63.61**	61.88	61.67	62.96	62.63

Table 7: Performance of Models Trained with Different Algorithms on MMLU and MMLU-Pro using the ALpaca_GPT4 Dataset with Qwen2.5-7B-Instruct.

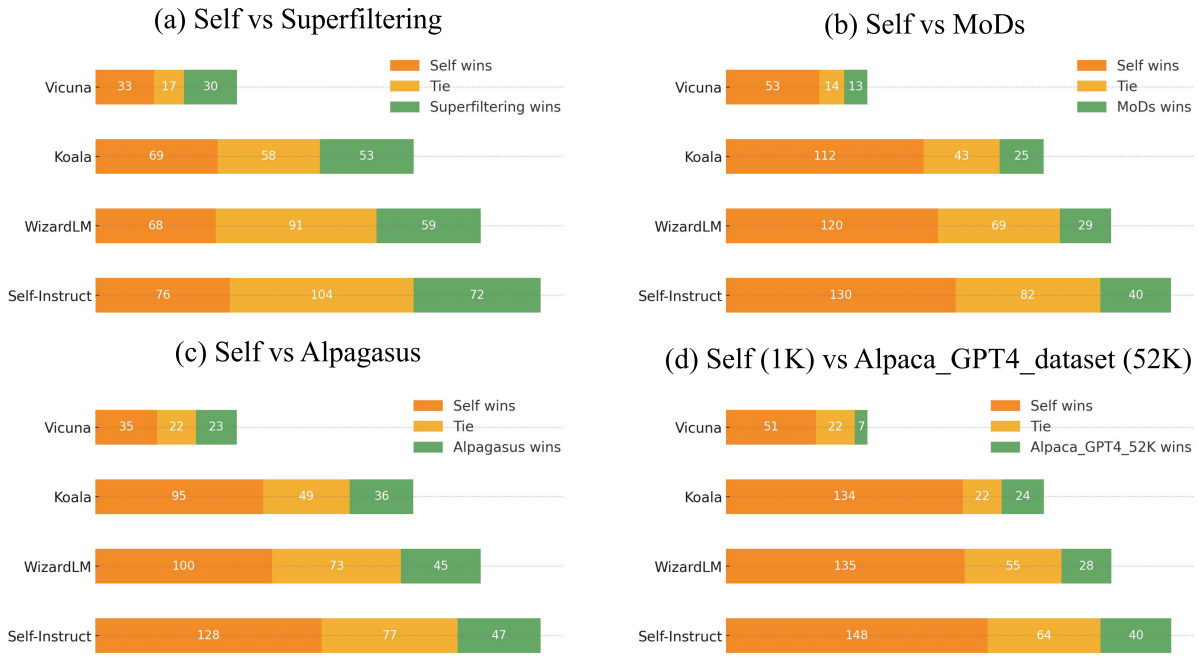

Figure 5: Performance Comparison of Data Selection Methods on the Qwen2.5-7B-Instruct Model Using the Alpaca_GPT4 Dataset.

	Self	MoDs	ALPAGASUS	Superfiltering	Original model	ALpaca_52k
MMLU(14k)	45.25	45.82	45.62	**46.34**	46.02	42.08
MMLU-Pro(12k)	17.49	16.72	16.53	16.03	**18.92**	17.18
Average	31.37	31.27	31.08	31.19	**32.47**	29.63

Table 8: Performance of Llama 2-7B-Chat Trained with Different Algorithms on MMLU and MMLU-Pro on the Alpaca Dataset.

	Self	MoDs	ALPAGASUS	Superfiltering	Llama 2-7B-Chat	ALpaca_GPT4(52k)
MMLU	46.09	45.82	45.44	46.16	46.02	**46.43**
MMLU-Pro	18.92	17.96	18.18	**19.09**	18.92	17.59
Average	32.51	31.89	31.81	**32.63**	32.47	32.01

Table 9: Performance of Llama 2-7B-Chat trained with different algorithms on MMLU and MMLU-Pro on the Alpaca_GPT4_Dataset.

	Full	Early (9)	Middle (9)	last (11)	final (1)
GSM8K	**84.91**	84.23	83.85	83.70	83.70
HumEval(pass@1)	**80.0**	75.0	77.4	79.3	78.7
HumanEval-Plus(pass@1)	**74.4**	70.1	71.3	73.8	72.6

Table 10: Comparison of CNN Model Performance Trained on Different Hidden Layer Parts of Qwen2.5-7B-Instruct Model.

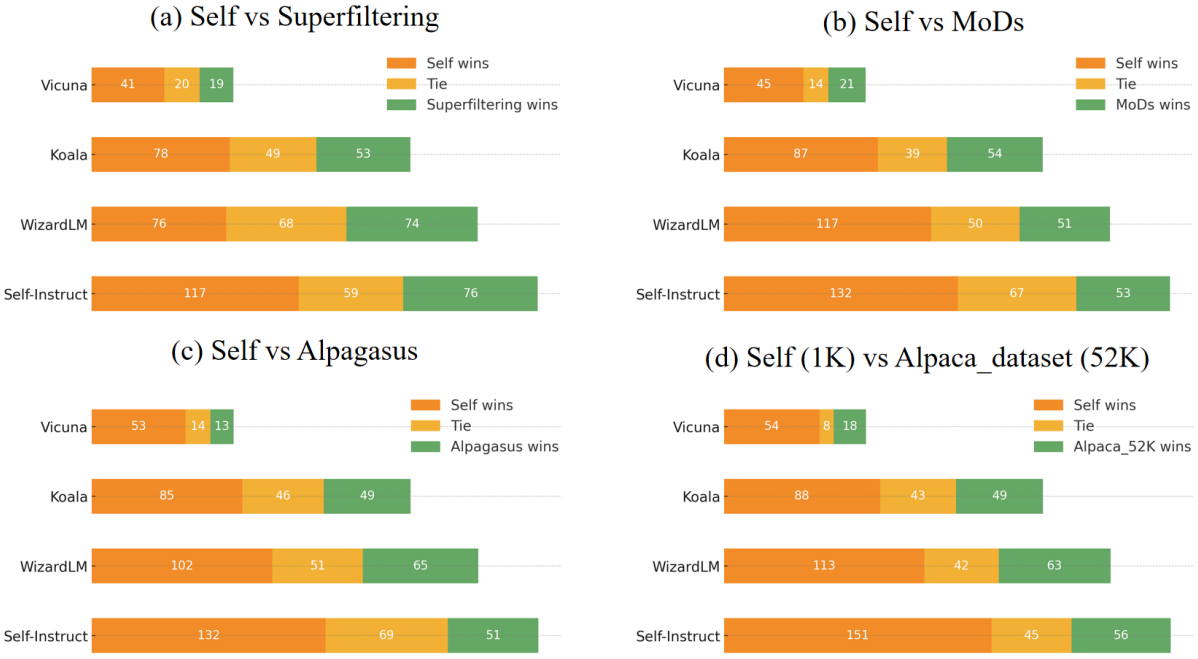

Figure 6: Performance Comparison of Data Selection Methods on the Llama 2-7B-Chat Model Using the Alpaca Dataset.

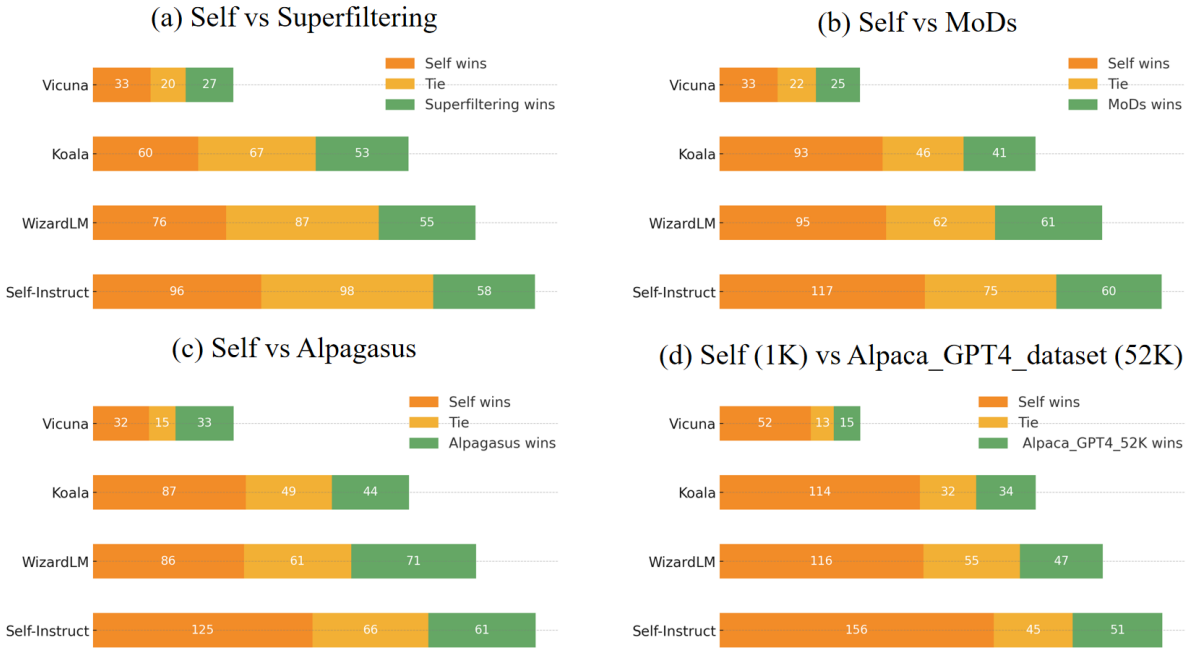

Figure 7: Performance Comparison of Data Selection Methods on the Llama 2-7B-Chat Model Using the Alpaca_GPT4 Dataset.

Training Method	Llama 2-7B-Chat	Qwen2.5-7B-Instruct
Self Training	**25.25**	**84.91**
Dataset Swapping	23.58	83.24

Table 11: Performance of Llama 2-7B-Chat and Qwen2.5-7B-Instruct Models Trained on Their Own and Swapped Datasets on the GSM8K Dataset (Cobbe et al., 2021).

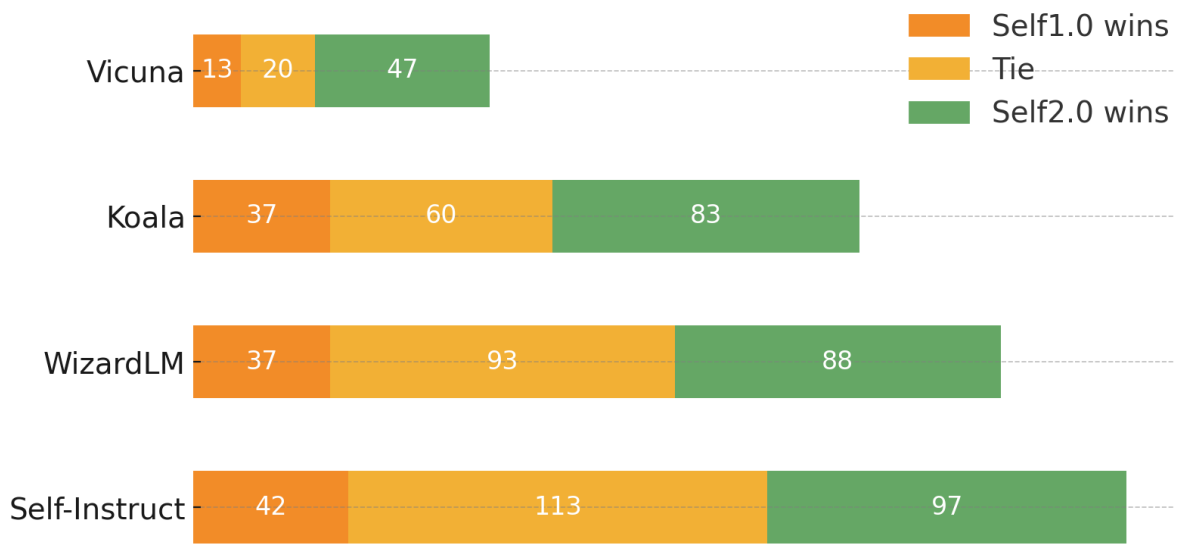

Figure 8: Comparison of Model Performance After Two-Stage Data Selection.

Training Method	HumanEval pass@1	HumanEval-Plus pass@1	Average
Llama 2-7B-Chat (Self)	**16.5**	**13.4**	**14.95**
Llama 2-7B-Chat (Swapped with Qwen2.5)	11.2	11.0	11.1
Qwen2.5-7B-Instruct (Self)	**80.0**	**74.4**	**77.2**
Qwen2.5-7B-Instruct (Swapped with Llama2)	72.0	67.1	69.55

Table 12: Performance of Llama 2-7B-Chat and Qwen2.5-7B-Instruct Models Trained on Their Own and Swapped Datasets on the HumanEval Dataset.