Priority on High-Quality: Selecting Instruction Data via Consistency Verification of Noise Injection

Hong Zhang, Feng Zhao*, Ruilin Zhao, Cheng Yan, Kangzheng Liu
Natural Language Processing and Knowledge Graph Lab,
School of Computer Science and Technology,
Huazhong University of Science and Technology, Wuhan, China
{zhang_hong,zhaof,ruilinzhao,frankluis}@hust.edu.cn,
yan_cheng2@ctg.com.cn

Abstract

Large Language Models (LLMs) have demonstrated a remarkable understanding of language nuances through instruction tuning, enabling them to effectively tackle various natural language processing tasks. Recent research has focused on the quality of instruction data rather than the quantity of instructions. However, existing high-quality instruction selection methods rely on external models or rules, overlooking the intrinsic association between pre-trained model and instruction data, making it difficult to select data that align with the preferences of pre-trained model. To address this challenge, we propose a strategy that utilizes noise injection to identify the quality of instruction data, without relying on external model. We also implement the strategy of combining inter-class diversity and intra-class diversity to improve model performance. The experimental results demonstrate that our method significantly outperforms the model trained on the entire dataset and established baselines. Our study provides a new perspective on noise injection in the field of instruction tuning, and also illustrates that the pre-trained model itself should be considered in defining high-quality. Additionally, we publish our selected high-quality instruction data at https://github.com/HUSTNLP-codes/Alpaca-selectd.

1 Introduction

Large Language Models (LLMs) have the ability to carry out intricate natural language processing tasks in various situations and fields through instruction tuning (Iyer et al., 2022; Ouyang et al., 2022; OpenAI, 2023; Chen et al., 2023; Sun et al., 2023; Caruccio et al., 2024). In the realm of instruction tuning, previous research has primarily concentrated on how the quantity of instruction data impacts training results (Wei et al., 2022; Chung et al., 2022; Longpre et al., 2023). Consequently, some researches focus on researching methods to automatically generate instruction data (Wang et al., 2023; Taori et al., 2023; Xu et al., 2023b), thus promoting the continuous expansion of the scale of instruction data. Training models on constantly expanding datasets is not practical because of the significant costs involved.

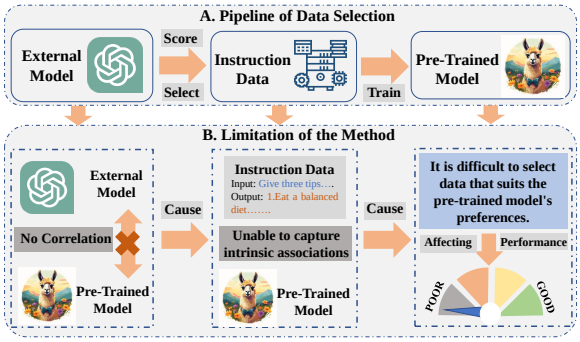

Figure 1: The figure illustrates the pipeline of filtering data by using external models, as well as the limitations of using external models.

Therefore, current researches are investing in research on the quality of instruction data (Zhou et al., 2023a; Köpf et al., 2023; Li et al., 2023a). Specifically, LIMA (Zhou et al., 2023a) demonstrates the importance of data quality over data quantity, while also raising the question of how to evaluate the quality of each instruction. Consequently, some research aims to use external models to develop a scoring mechanism for individual instructions, thereby achieving the identification of high-quality instructions (Li et al., 2024a,b; Chen et al., 2024). Additionally, research has indicated that using the length of instruction outputs as a criterion for data filtering can also achieve significant optimization results (Zhao et al., 2024). As shown in Figure 1, relying on external models or explicit rules for instruction filtering overlooks the relationship between pre-trained models and instruction data, potentially limiting fine-tuning performance.

*Corresponding author

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 20776–20788
November 4-9, 2025

It struggles to identify data that genuinely corresponds to the model's own preferences. This limitation raises a critical challenge:

> *How to select data that aligns with the model's own preferences to achieve optimized instruction tuning?*

To address this challenge, we devote to developing a novel high-quality instruction filtering technique that delves into the intrinsic association between the implicit knowledge learned by pretrained model and instruction data. Inspired by previous research on noise utilization (Namysl et al., 2020; Hua et al., 2022; Jain et al., 2023), we propose to define the quality of each data by introducing noise. Specifically, we inject noise into the input part of the instruction, then analyze the changes in the output probability distribution of the pre-trained model for the entire instruction, and select those data with high probability distribution consistency as high-quality data. Moreover, we combine the strategies of inter-class diversity and intra-class diversity to improve the coverage of the selected data and reduce the redundancy in the datasets. We have conducted extensive experiments on different models and various instruction datasets. The experimental results show that our method not only outperforms the results of full-data training but also surpasses existing baselines.

In summary, our contributions are as follows:

- We propose a method for selecting high-quality instruction data without using additional models and taking into account an effective combination of quality and diversity.
- Our method creatively applies noise injection to measure the quality of each instruction data, providing a new application perspective for noise in the field of instruction tuning.
- The overall performance of our method surpasses that of full-data training when selecting only 5%-15% of the entire dataset, which not only reduces the training cost, but also improves the performance of the model.
- We publish the high-quality instruction dataset filtered from Alpaca by our method.

2 Method

2.1 Preliminaries

LIMA (Zhou et al., 2023a) indicates that the pretraining phase is where large models accumulate most of their knowledge. In contrast, the goal of instruction tuning is to steer the model towards a particular interaction style or format, effectively demonstrating its built-in knowledge and abilities.

Therefore, we hypothesize that instruction data with high semantic relevance to pretrained knowledge are more effectively utilized by the model, thereby facilitating the release of its latent capabilities. To validate this hypothesis, we propose a method based on perturbation consistency that introduces noise into the low-dimensional embedding space of the instructions and assesses the stability of model outputs to identify high-quality instruction samples closely aligned with pre-trained knowledge. To ensure data diversity and reduce redundancy, we incorporate k-means clustering combined with cosine similarity-based filtering. The overall framework of our method is illustrated in Figure 2. Through extensive experiments, we demonstrate the effectiveness of our core hypothesis in activating the latent knowledge of the model (Section 4.1 & 4.3), as well as the superiority of our method in capturing the intrinsic semantic relationships between pre-trained knowledge and instruction data (Section 4.5).

2.2 Consistency Selection

Adding interference directly to a high-dimensional space such as the original text can easily cause semantic changes. Therefore we perform noise injection on the embedding of the input part of the text. And we use Gaussian noise which is widely used in image processing. In particular, we introduced β to change the mean and variance to control the size of the noise. For each instruction d_i in the initial dataset D_0, where d_i is represented as (X, Y). The embedding for each d_i instruction is expressed as $(\mathbf{e^x_{1,i}} \cdots \mathbf{e^x_{n,i}}, \mathbf{e^y_{1,i}} \cdots \mathbf{e^y_{m,i}})$. We introduce a specific level of noise to the embedding of the input section of the instructions, as per the following formulas:

$$\mathbf{n_{k,i}} = \beta(\mu^{\mathbf{i}}_{\mathbf{x}} + \sigma^{\mathbf{i}}_{\mathbf{x}}\epsilon_{\mathbf{i}}), \epsilon_{\mathbf{i}} \sim \mathcal{N}(\mathbf{0}, \mathbf{1}), \quad (1)$$

$$\tilde{\mathbf{e}}^{\mathbf{x}}_{\mathbf{k,i}} = \mathbf{e}^{\mathbf{x}}_{\mathbf{k,i}} + \mathbf{n_{k,i}}, \quad (2)$$

where β represents the scaling factor of noise magnitude, σ^i_xdenotes the standard deviation of input part X in the i^{th} instruction, and μ^i_x stands for the mean of input part X in the i^{th} instruction, $\mathbf{e^x_{k,i}}$ represents the embedding of the $\mathbf{k^{th}}$ token in the i^{th} data, $\tilde{\mathbf{e}}^{\mathbf{x}}_{\mathbf{k,i}}$ represents the embedding $\mathbf{e^x_{k,i}}$ after adding noise.

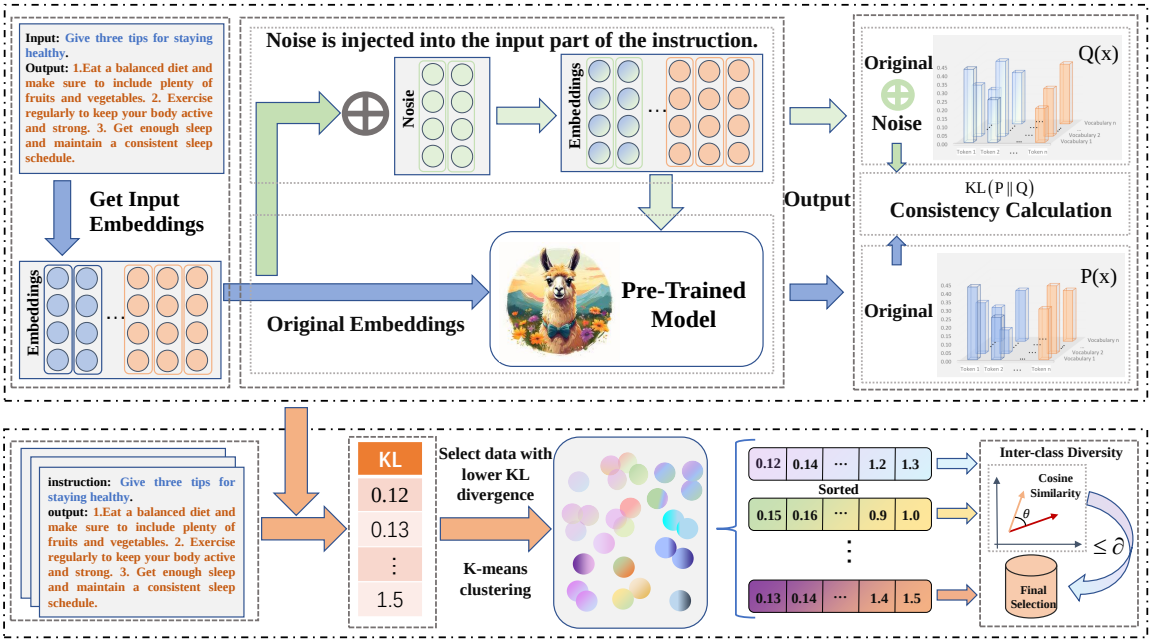

Figure 2: The overall framework. The top portion of the figure illustrates the method for determining the quality of each data, whereas the bottom part depicts the procedure for integrating quality with diversity selection strategies.

In order to assess the consistency of the model in predicting word-level granularity before and after introducing noise, we collected the probability distribution predictions of the model at each vocabulary position after adding noise. Subsequently, we compared the consistency of model prediction probabilities between the original instructions and the instructions after noise is added. A higher level of consistency indicates better data quality. The formula for calculating the consistency of probabilities is as follows:

$$D_{\mathrm{KL}}(P||Q) = \frac{1}{n}\sum_{i} P(i)\log\left(\frac{P(i)}{Q(i)}\right), \quad (3)$$

where n represents the token length of an instruction, including the input x and the output y. P_i represents the probability output of the i_{th}instruction after passing through the model, while Q_i denotes the probability output of the i_{th} instruction after adding noise to the input portion and passing through the model.

A lower KL divergence value suggests a greater consistency in the probability distribution, thereby indirectly indicating the quality of the data. when perturbations are introduced, there will be a certain degree of randomness in the actual noise generation. Therefore, in the actual experimental operation, we took three independent sampling processes and calculated the corresponding KL divergence values, and finally took the average of the three as our consistency evaluation result.

2.3 Diversity Selection

Relying solely on consistency calculations for sorting and selection may result in the selected data set having only a few categories, resulting in reduced model performance. In order to improve the category diversity of the selected data set, we adopt the inter-class diversity selection and intra-class diversity selection strategies.

In the inter-class diversity selection strategy, our core goal is to expand the coverage of the selected data while ensuring the quality of each piece of data. To this end, we prioritize data that ranks higher in the initial ranking, while implementing inter-class diversity selection to ensure that the selected data set is broadly representative at the class level. We calculate the overall semantic embedding of each data point using the following formula. We then utilize the K-means (Lloyd, 1982) clustering algorithm for inter-class diversity filtering to optimize the quality of the dataset and the generalization performance of the model. The relevant calculation formulas are as follows:

$$[\mathbf{h_1^i}\cdots\mathbf{h_n^i}] = PLM(\mathbf{e_1^i}\cdots\mathbf{e_n^i}), \quad (4)$$

$$\mathbf{H_i} = \frac{\sum_{k=1}^{n}\mathbf{h_k^i}}{n}, \quad (5)$$

$$(\mathrm{cluster}_1\cdots\mathrm{cluser}_k) = \text{K-means}(\mathbf{H_1}\cdots\mathbf{H_i}), \quad (6)$$

where PLM denotes a pre-trained model, while $\mathbf{h_n^i}$ represents the ultimate hidden states of the i^{th} instructions. $\mathbf{H_i}$ is the vector representation of the entire statement.

After confirming data coverage using the inter-class diversity selection strategy, we observed that data points within the same class might exhibit significant similarities, leading to redundant data. To diminish redundancy and enhance dataset diversity, we implemented an intra-class diversity selection strategy. More precisely, we assess the quality of data within each category and then calculate the cosine similarity between instructions by utilizing sentence embedding. The diversity of the dataset is improved by choosing instructions that have similarities under a set limit and adding these less similar data points to the filtered subset.

3 Experimental Setup

Datasets We use the Alpaca (Taori et al., 2023) dataset created by Stanford University, which contains 52K instruction data. This dataset has been widely adopted and applied in the research field of instruction filtering. To thoroughly assess the model's performance, we utilize a range of datasets for conducting specific capability tests. We use the MMLU (Hendrycks et al., 2021) dataset to measure the model's ability to handle interdisciplinary knowledge in a multilingual environment. By employing the Humaneval (Chen et al., 2021), we evaluate the model's proficiency in comprehending and producing code. The GSM-8K (Cobbe et al., 2021) is utilized to assess the model's aptitude in resolving mathematical problems. In addition, we use the CommonsenseQA (Talmor et al., 2019) to examine the model's mastery of common sense knowledge in daily life. Finally, through the NaturalQuestions (Kwiatkowski et al., 2019), we evaluate the model's performance in understanding and answering questions involving world knowledge.

Baselines We compare various state-of-the-art baseline methods. LIMA (Zhou et al., 2023a) is trained on 1k high-quality instruction-following data meticulously handcrafted. AlpaGasus (Chen et al., 2024) uses ChatGPT to score each piece of data and select the high-scoring data for training. Q2Q (Li et al., 2024b) trains a model initially with a few instructions, and subsequently assess the data quality using two distinct loss values within the model. Furthermore, Superfiltering (Li et al., 2024a) refines the process by replacing the required instruction fine-tuning model with a smaller external model for computation. Additionally we use the length of the instruction's output as a strong baseline (Zhao et al., 2024). Moreover, we add some additional methods. Alpaca-all (Taori et al., 2023) is directly trained on the complete Alpaca dataset. Random is selected from the source data set through random sampling. The details of the experimental implementation can be found in Appendix A

4 Results and Analysis

We are focusing on the instruction tuning of six key Research Questions (RQ) with a series of corresponding experiments and in-depth analysis.

4.1 Main Results

RQ1: How does our selection method compare to the SOTA method in performance? We compare our method with established baselines in different pre-trained models. The experimental results are shown in Table 1. Our method exhibits outstanding performance across various scales and structures of models, significantly enhancing the model's comprehensive knowledge capabilities. It is noteworthy that relying solely on external models for selection while ignoring the model's own characteristic biases, may limit its performance in specific domains. Specifically, in the Llama2 experiments, AlphaGasus exhibits markedly lower mathematical performance, reaching only about half the level of the other baseline models. We observe that the performance improvement of our method is particularly pronounced on smaller models compared to the full dataset. We hypothesize that this might be due to smaller models being more sensitive to data quality.

4.2 Generalization of Method

RQ2: Can our method adapt to different styles of instruction datasets? Our method demonstrates outstanding performance on the Alpaca dataset, which is created from powerful LLMs. To assess whether our method retains its efficacy across different instruction dataset types, we broaden our experimental scope. We choose two different types of instruction datasets for testing: the manually crafted instruction dataset Dolly (Dolly, 2023) and the conventional NLP-related

Models	Methods	External Model	MMLU	Math	Code	COM	NQ	Average	Δ
Qwen2-0.5B	Alpaca-All	—	35.83	14.56	20.73	52.01	7.59	26.14	—
	AlpaGasus(2024)	✓	36.23	27.22	23.17	51.92	6.54	29.02	+2.88
	Ours	✗	36.68	34.85	26.83	53.32	7.01	31.74	**+5.60**
Qwen2-1.5B	Alpaca-All	—	50.47	39.73	33.54	69.94	13.77	41.49	—
	AlpaGasus(2024)	✓	35.59	53.98	36.59	71.25	13.77	42.24	+0.75
	Ours	✗	45.10	57.54	40.24	71.25	14.16	45.66	**+4.17**
Llama2-7B	Alpaca-All	—	47.93	13.12	13.41	55.04	20.83	30.07	—
	Random	✗	45.97	10.99	11.59	52.66	29.14	30.07	0
	LIMA(2023a)	✗	40.76	19.33	15.24	44.72	11.83	26.38	-3.69
	Superfiltering(2024a)	✓	41.03	7.73	11.59	49.63	19.14	25.82	-4.25
	Q2Q(2024b)	✓	44.69	13.50	15.85	47.75	28.84	30.13	+0.05
	AlpaGasus(2024)	✓	46.51	7.73	14.63	54.05	29.75	30.53	+0.46
	Lenth(2024)	✗	45.87	16.07	14.02	50.07	30.66	31.34	+1.27
	Ours	✗	47.12	15.69	15.85	56.51	29.83	33.00	**+2.93**

Table 1: "Math" means GSM-8K,"Code" means Humaneval, "COM" means CommonsenseQA, "NQ" means NaturalQuestions.

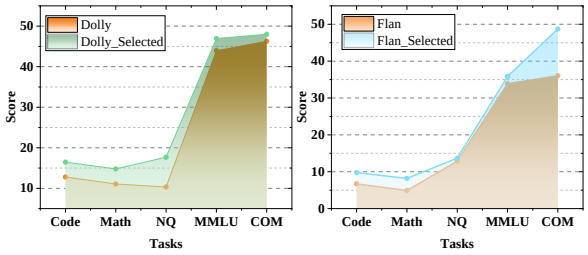

Figure 3: We randomly select a subset from the FLAN dataset that is comparable in size to the Dolly dataset for experiments. In multiple-choice questions, the option corresponding to the sequence with the lowest perplexity is chosen as the model's final inference result for that question.

dataset FLAN (Longpre et al., 2023). The experimental results are shown in Figure 3. In the Dolly dataset experiment, the model trained with our method significantly outperforms the model trained on the full data set across all metrics. In the Flan dataset experiment, the random subset may impact the data quality, leading to lower training results. After using our method, the model performance is further improved, especially in common sense question-answering. The experimental results demonstrate the effectiveness of our method in adapting to different instruction styles and significantly enhance the performance of instruction fine-tuning.

While our method improves performance across stylistically diverse datasets, it remains unclear whether these gains extend to instruction following—a key aspect of LLM utility. To test this, we evaluate on the IFEVAL (Zhou et al., 2023b),

Methods	Pro-strict	Ins-strict	Pro-loose	Ins-loose	Avg
Dolly	16.45	25.90	18.11	27.94	22.10
Selected	18.48	28.90	19.59	30.58	**24.39**
Flan	23.48	35.13	23.84	35.37	29.46
Selected	25.32	36.69	25.69	37.05	**31.19**

Table 2: Performance of our method on datasets with diverse stylistic characteristics under the IFEVAL evaluation framework.

which measures instruction adherence across diverse task formats, allowing us to assess whether our data selection strategy yields better alignment with human instructions. As reported in Table 2, models trained on datasets curated by our method achieve superior performance compared to those trained on the original unfiltered data, under both strict and loose evaluation criteria. These consistent improvements highlight that our approach not only enhances task performance across heterogeneous benchmarks but also strengthens the ability of models to faithfully follow instructions, underscoring the practical significance of our method for real-world instruction-driven applications.

4.3 Ablation Experiments

RQ3: Can the data selected through our method enhance ability of the model to express internal knowledge? We train models on a low-consistency dataset to evaluate its impact on performance, using consistency as the sole selection criterion. For performance evaluation, we integrate the Vicuna testset (Chiang et al., 2023) from open-

Methods	Ability						Open-Domain
	MMLU	Math	Code	COM	NQ	Avg.	Vicuna
Random	45.97	10.99	11.59	52.66	29.14	30.07	48.96
Low-Consistency	48.62	6.22	11.59	60.61	11.65	27.74	8.79
High-Consistency	46.34	13.87	15.24	53.32	29.11	**30.45**	**60.81**

Table 3: The comparison of consistency selection experiments. "Ability" refers to the collective mean of diverse testing capabilities. For the Vicuna test, we utilize "weighted_alpaca_eval_gpt4_turbo" from AlpacaEval2.0 as the annotator.

domain problems into our analysis, assessing with AlpacaEval2.0 (Li et al., 2023b). The comprehensive experimental results are detailed in Table 3. It is apparent that the data selected through low-consistency has not conducive to the unleashing of the model's intrinsic knowledge. While such data may yield higher scores on selected-choice tasks, it underperforms significantly in domains that require knowledge output, such as mathematical problem-solving or open-domain question answering. The scores on these tasks are even far lower than the results of random selection. Conversely, the selection of high-consistency data has demonstrated superior performance across all dimensions. To delve into this phenomenon, we examine the Vicuna-test results and randomly select one question to assess the impact of various selection methods, as detailed in Table 5. Mode trained on High-Consistency data produce answers that are not only richer in content but also more fluent in language, while maintaining better contextual coherence. This result indicates that high-consistency data can more effectively facilitate the expression of knowledge acquired by the model during pre-training.

To further assess the impact of diversity in instruction styles in data selection, we conduct a set of experimental comparisons. The outcomes are displayed in Table 4. The results show that a quality-centric approach may neglect data diversity, possibly constraining the proficiency of model in specific domains. Although a diversity-centric selection expands the data range, it risks incorporating lower-quality data, which could impair model performance. However, models that balance both quality and diversity in selection show enhanced performance in our tests. Quality guarantees that the model learns the interaction style of instructions, while diversity enables the model to master various styles, thereby improving its generalization and adaptability across different situations.

4.4 Effect of Noise

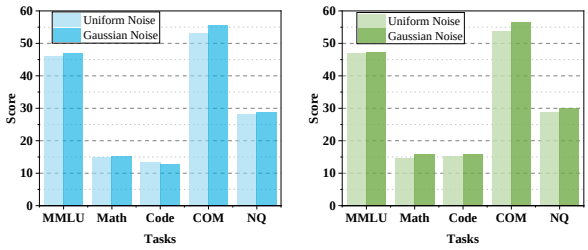

Figure 4: Examining the impacts of varying noise types and varying noise intensities on experimental outcomes. In the figure on the left, $\beta = 1$ signifies the initial intensity of the noise. Conversely, in the figure on the right, $\beta = 10$ suggests a tenfold increase in the noise intensity.

RQ4: How does the noise affect the performance of our method? We conduct an in-depth exploration of the data filtering effect under different noise intensities. Specifically, we select 5%-15% of the original dataset as subsets under noise levels of $\beta = 1$ and $\beta = 10$, and train Llama2 based on these subsets. The results are shown in Figure 5. The model trained with the filtered subset generally outperforms the results of training with the full dataset under the two noise intensities, confirming the effectiveness of our proposed approach. Especially under the condition of $\beta = 10$ and a 12% selection ratio, the model performance reaches the optimal level. Additionally, we observe an overall trend toward better model performance at higher noise levels, which may be due to the fact that low noise intensity is not sufficient to cause effective interference in the data.

We replace the conventional Gaussian noise with uniform noise to investigate the impact on model performance. The results are presented in Figure 4. The figure clearly illustrates that, across various noise levels, Gaussian noise yields significantly superior experimental outcomes compared to uniform

Methods	MMLU	Math	Code	COM	NQ	Average
Alpaca-All	47.93	13.12	13.41	55.04	20.83	30.07
Diversity	46.34	13.87	15.24	53.32	29.11	31.58
Consistency	45.26	15.39	15.85	44.80	30.94	30.45
Diversity+ Consistency	47.12	15.69	15.85	56.51	29.83	**33.00**

Table 4: The importance of combining quality and diversity(Llama2).

Question	What are the main differences between Python and JavaScript programming languages?
High-Consistency-Output	The primary difference between Python and JavaScript is their syntax, data types, and use cases. Python is a high-level language with a simpler syntax than JavaScript which makes it easier to read and write code. It has fewer built-in data types compared to JavaScript, but Python provides more extensive library support for working with complex data structures such as lists and dictionaries. On the other hand, JavaScript is primarily used in client-side scripting and runs on the browser, while Python can be used both on the server side and client side. Additionally, Python supports multiple paradigms like object-oriented programming and functional programming, whereas JavaScript follows a single-paradigm approach of imperative programming.
Low-Consistency-Output	Python is an interpreted language, while JavaScript is a compiled language.

Table 5: A comparative analysis of model generation capabilities under two different selecting approaches.

noise. A meticulous comparison of the images within the figure reveals a notable trend: as noise intensity rises, both methods exhibit considerable performance gains. Our research concludes that a moderate increase in noise intensity aids in refining the identification of data quality. This effect might stem from the fact that moderate noise levels effectively accentuate key data features while diminishing the relevance of less critical details, thus enhancing the efficiency of data quality differentiation.

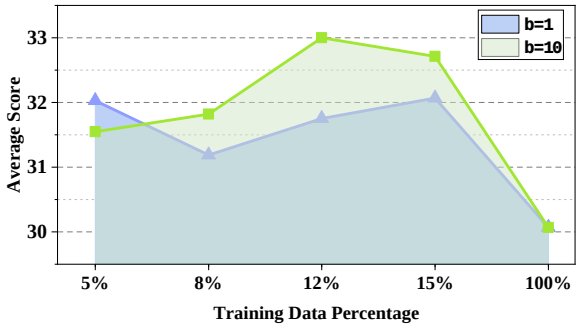

Figure 5: Compare various dataset sizes within the alpaca dataset to assess how our method's performance varies. Additional experiments on noise intensity can be found in Appendix B

4.5 Selected Data Analysis

RQ5: Can our method consider the relationship between the pre-trained model and the instruction dataset, thereby selecting data that align with the model's preferences? Directly analyzing this research question is challenging. Therefore, we utilize GLM-4 (Zeng et al., 2024) to classify the dataset into nine categories, in order to depict the trends in data selection among different pre-trained models, and further analyze the selection preferences of the models. A smaller value indicates a greater tendency to select this type of data. Detailed experimental results are shown in Table 6. It is evident that on the Alpaca dataset, the Llama2 and Qwen2 models exhibit unique preferences in data selection. In particular, Qwen2-0.5B and Qwen2-1.5B, which have the same model architecture, show similar selection preferences for data types. This preference is due to models within the same category using similar corpora during the pre-training phase. This observation confirms that our selection method does take into account the knowledge learned by the model during pre-training, thereby filtering out data that aligns with the model's own preferences.

To conduct an in-depth analysis of the data types our method typically selects and whether the chosen data maintains diversity, we employ the Self-instruct (Wang et al., 2023) to analyse. The findings are illustrated in Figure 6, indicating that the filtered dataset has enhanced task distribution while preserving the diversity present in the original data. More specifically, the filtered datasets exhibited a tendency to include creative and interpretive tasks such as "generate," "write," "create," "explain," and "describe," while tasks involving revisions such as "rewrite" and "edit" showed a relative decrease. In terms of semantic information content, "rewrite" instructions are significantly inferior to "generate" instructions. This result reveals that the data filtering method adopted in this study effectively removes instructions with low semantic information content.

4.6 Scale Generalization

RQ6: Can our method be effectively applied to different model versions and larger-scale models? In the previous experimental analysis, our evaluation was primarily conducted on relatively smaller models and earlier versions of LLMs. However, as the landscape of foundation models is

Category	Alpaca					Dolly		
	All	Selected	Δ	Selected	Δ	All	Selected	Δ
Moldel	—	Llama2-7B	—	Qwen2-0.5B/1.5B	—	—	Llama2-7B	—
Discipline	2193	242	88.96%	277/283	87.37%/87.10%	561	169	69.88%
Language	5855	72	**98.77%**	80/78	**98.63%/98.67%**	113	36	68.14%
Konwledge	15761	2012	87.23%	2567/2537	83.71%/83.90%	10651	3639	65.83%
Comprehension	3860	669	**82.67%**	767/817	80.13%/78.83%	626	252	59.74%
Reasoning	837	94	88.77%	118/89	85.90%/89.37%	208	92	**55.77%**
Creation	12758	2103	83.52%	2565/2780	**79.89%/78.21%**	856	339	60.40%
Code	626	59	90.58%	82/90	86.90%/85.62%	5	1	**80.00%**
Mathematics	3195	99	96.90%	89/84	97.21%/97.37%	162	53	67.28%
Other	5874	697	88.13%	796/810	86.45%/86.21%	1520	572	62.37%

Table 6: Using GLM-4 to classify the data before and after selection. Here, Δ is calculated as $\frac{(\text{ALL}-\text{Selected})}{\text{ALL}}$.

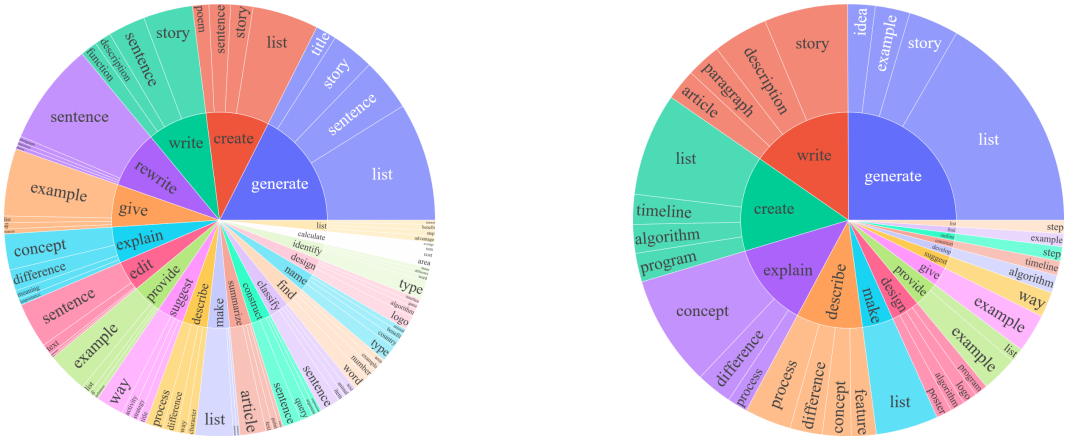

Figure 6: Comparing the diversity of instructions between the original alpaca data (left) and the filtered data (right) involves analyzing the verb-noun structure of the instructions. The inner circle displays the top 20 most common root verbs found in the instructions, while the outer circle lists their corresponding first four direct noun objects.

rapidly evolving, with new versions and larger-scale models being continuously released, it is essential to examine whether our method can generalize under these more challenging settings. To this end, we extend our experiments to three representative updated models of different scales: Llama3-8B (Dubey et al., 2024), Qwen2-7B (Yang et al., 2024a), and Qwen2.5-14B (Yang et al., 2024b). Furthermore, to provide a more comprehensive assessment of model performance, we additionally introduce BBH (Suzgun et al., 2023) benchmark into the evaluation suite, thereby enabling us to evaluate not only the effectiveness of our approach on updated models but also its robustness when faced with more demanding inference tasks. As shown in Table 7, our method consistently outperforms the baselines across different models and parameter sizes, yielding substantial improvements in average performance with gains of +7.25, +6.36, and +3.70 points on Llama3-8B, Qwen2-7B, and Qwen2.5-14B, respectively. These results demonstrate that our approach generalizes well across newer versions of LLMs while scaling effectively to larger models, underscoring its robustness and its potential applicability in real-world scenarios where high-capacity models are increasingly adopted.

5 Related Work

Instruction Dataset Previous research has focused on improving the model's ability to follow instructions using an extensive instruction dataset (Ouyang et al., 2022; Chung et al., 2022). FLAN (Ouyang et al., 2022) effectively boosted model performance by transforming traditional NLP tasks into instruction datasets using instruction templates. Alpaca employs the self-instruct technique, utilizing advanced LLMs to generate a varied collection of 52k instructions (Taori et al., 2023; Wang et al., 2023). Humpack (Li et al., 2023a) employs instruction reverse translation and self-filtering for fine-tuning. WizardLm (Xu et al., 2023a) create more complex instructions, thereby enhancing the performance of large

Models	Methods	External Model	MMLU	Math	Code	COM	NQ	BBH	Average	Δ
Llama3-8B	Alpaca-All	—	59.77	26.08	40.85	70.84	30.06	36.36	43.99	—
	AlpacGasus	✓	60.89	30.63	34.15	73.46	33.91	57.51	48.42	+4.43
	Ours	✗	60.92	42.23	40.24	73.55	33.63	56.87	**51.24**	**+7.25**
Qwen2-7B	Alpaca-All	—	66.59	60.12	62.80	81.24	28.45	44.59	57.30	—
	AlpacGasus	✓	69.69	75.51	64.02	82.23	30.14	57.61	63.20	+5.90
	Ours	✗	70.00	76.04	67.68	80.67	29.45	58.10	**63.66**	**+6.36**
Qwen2.5-14B	Alpaca-All	—	75.17	73.77	62.80	84.28	33.46	67.27	66.13	—
	AlpacGasus	✓	77.56	82.87	69.51	84.36	34.43	56.96	67.62	+1.49
	Ours	✗	77.52	83.17	71.34	84.19	33.60	69.15	**69.83**	**+3.70**

Table 7: Experimental results on the updated and larger LLMs, showing that our method consistently generalizes across newer model versions and larger parameter scales.

language models. LIMA (Zhou et al., 2023a) demonstrates that with just 1,000 meticulously curated high-quality data points, LLMs can exhibit significant improvements in command-following capabilities.

Noise Utilization NAT (Namysl et al., 2020) boosts the robustness of sequence labeling models via noise-aware training without sacrificing input accuracy. LNSR (Hua et al., 2022) strengthens pre-trained models for follow-up tasks by injecting noise during training. NEFTune (Jain et al., 2023) enhances model performance by adding noise to embeddings during fine-tuning. Our method introduces a novel approach by applying noise to instruction data selection, a previously unexplored area.

Instruction Data Selection Recent research aims to minimize the required data for instruction tuning. Intuitively, instruction mining (Cao et al., 2023) has established linear rules using specific natural language metrics for assessing the quality of instruction datasets. AlpaGasus (Chen et al., 2024) rely on other exceptional LLMs for assessing and selecting high-quality instruction data . The AIT (Kung et al., 2023) proposes prompt uncertainty for filtering novel/informative instructions. Q2Q (Li et al., 2024b) uses a fine-tuned model to calculate the Instruction-Following Difficulty (IFD) index for each point. Superfiltering (Li et al., 2024a) employs smaller exceptional models to calculate IFD. Additionally, research has indicated that using the length of instruction outputs as a criterion for data filtering can also achieve significant optimization results (Zhao et al., 2024) .

6 Conclusion

Our method selects data suitable for the pre-trained model itself through noise injection, without relying on additional models and rules. Initially, we assess the value of different data points by introducing noise, which helps us precisely identify the most beneficial data for model training. Subsequently, we reduce data bias by increasing diversity within and between classes. Empirical evaluations across multiple datasets and models show that our innovative technique not only exceeds the performance of full datasets, but also significantly exceeds current state-of-the-art baselines. Our strategy not only reduces the resources required for training but also significantly improves model performance.

Limitations

Our data selecting method involves comparing the output differences of the model before and after adding noise to the data, which inevitably leads to additional inference costs. However, we found that models of the same type tend to have similar types of filtered data, which may be their pre-training data is roughly the same. Therefore, future research can perform data filtering on smaller models and then apply the filtering results to larger models of the same type.

Acknowledgement

This work was supported in part by the National Key R&D Program of China under Grant 2023YFF0905503, National Natural Science Foundation of China under Grants No.62472188.

References

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

Loredana Caruccio, Stefano Cirillo, Giuseppe Polese, Giandomenico Solimando, Shanmugam Sundaramurthy, and Genoveffa Tortora. 2024. Claude 2.0 large language model: Tackling a real-world classification problem with a new iterative prompt engineering approach. *Intell. Syst. Appl.*, 21:200336.

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

Qianglong Chen, Guohai Xu, Ming Yan, Ji Zhang, Fei Huang, Luo Si, and Yin Zhang. 2023. Distinguish before answer: Generating contrastive explanation as knowledge for commonsense question answering. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13207–13224. Association for Computational Linguistics.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Y. Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *CoRR*, abs/2210.11416.

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.

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

Free Dolly. 2023. Introducing the world's first truly open instruction-tuned llm. databricks. com.

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.

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.

Hang Hua, Xingjian Li, Dejing Dou, Cheng-Zhong Xu, and Jiebo Luo. 2022. Fine-tuning pre-trained

language models with noise stability regularization. *CoRR*, abs/2206.05658.

Srinivasan Iyer, Xi Victoria Lin, Ramakanth Pasunuru, Todor Mihaylov, Daniel Simig, Ping Yu, Kurt Shuster, Tianlu Wang, Qing Liu, Punit Singh Koura, Xian Li, Brian O'Horo, Gabriel Pereyra, Jeff Wang, Christopher Dewan, Asli Celikyilmaz, Luke Zettlemoyer, and Ves Stoyanov. 2022. OPT-IML: scaling language model instruction meta learning through the lens of generalization. *CoRR*, abs/2212.12017.

Neel Jain, Ping-yeh Chiang, Yuxin Wen, John Kirchenbauer, Hong-Min Chu, Gowthami Somepalli, Brian R. Bartoldson, Bhavya Kailkhura, Avi Schwarzschild, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2023. Neftune: Noisy embeddings improve instruction finetuning. *CoRR*, abs/2310.05914.

Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, Shahul ES, Sameer Suri, David Glushkov, Arnav Dantuluri, Andrew Maguire, Christoph Schuhmann, Huu Nguyen, and Alexander Mattick. 2023. Openassistant conversations - democratizing large language model 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*.

Po-Nien Kung, Fan Yin, Di Wu, Kai-Wei Chang, and Nanyun Peng. 2023. Active instruction tuning: Improving cross-task generalization by training on prompt sensitive tasks. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 1813–1829. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*, 7:452–466.

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. 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. 2024b. 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.

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Luke Zettlemoyer, Omer Levy, Jason Weston, and Mike Lewis. 2023a. Self-alignment with instruction backtranslation. *CoRR*, abs/2308.06259.

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

Stuart P. Lloyd. 1982. Least squares quantization in PCM. *IEEE Trans. Inf. Theory*, 28(2):129–136.

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.

Marcin Namysl, Sven Behnke, and Joachim Köhler. 2020. NAT: noise-aware training for robust neural sequence labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 1501–1517. Association for Computational Linguistics.

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

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. 2022. 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*.

Xiaofei Sun, Linfeng Dong, Xiaoya Li, Zhen Wan, Shuhe Wang, Tianwei Zhang, Jiwei Li, Fei Cheng, Lingjuan Lyu, Fei Wu, and Guoyin Wang. 2023. Pushing the limits of chatgpt on NLP tasks. *CoRR*, abs/2306.09719.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi,

Denny Zhou, and Jason Wei. 2023. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13003–13051. Association for Computational Linguistics.

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

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

Hugo Touvron, 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.

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

Jason Wei, 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.

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

Canwen Xu, Daya Guo, Nan Duan, and Julian J. McAuley. 2023b. Baize: An open-source chat model with parameter-efficient tuning on self-chat data. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 6268–6278. Association for Computational Linguistics.

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, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024b. Qwen2.5 technical report. *CoRR*, abs/2412.15115.

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.

Hao Zhao, Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. 2024. Long is more for

alignment: A simple but tough-to-beat baseline for instruction fine-tuning. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

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. 2023a. 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*.

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

A Experiment Details

Train Details In light of the fact that the majority of existing baseline experiments are conducted on Llama2-7b (Touvron et al., 2023), we adopt Llama2-7b as a unified evaluation platform to ensure fairness and comparability. Based on this setup, we conduct a systematic comparative analysis of all baseline methods. Additionally, to delve into the performance discrepancies among various architectural designs and across model sizes, we expand our experimental study to include analyses of the Qwen2-0.5B and Qwen2-1.5B (Yang et al., 2024a). Unless otherwise specified, during training, we fine-tune the model for 3 epochs, with the batch size of 256. We utilize the AdamW optimization algorithm with a learning rate set to 2×10^{-5}. To enhance the model's performance, we extend the maximum length of input sentences to 4096 tokens. For testing the various capabilities of the model, we use the Opencompass (Contributors, 2023) framework. For MMLU, we utilize 5-shots, and for CommonsenseQA, we use 8-shots. We rent 4 $\times$ NVIDIA A6000 for model training. During the training process, we adapt a full parameter fine-tuning strategy and utilized gradient accumulation techniques. Despite the fact that most of the instruction data is short, we still set the maximum data length to 4096 tokens.

Main Experiments Details Main experiments are conducted with Gaussian noise at $\beta = 10$. Due to differences in parameter scale and embedding distributions, the proportion of data used varies slightly across models—about 12% for Llama2, 14% for Qwen2-0.5B, and 15% for Qwen2-1.5B—all substantially below the full dataset. For AlpaGasus and LIMA, we follow the officially reported optimal ratios, while other baselines are trained on datasets of the same scale as ours to ensure fair comparison. To further assess generality, we include both manually written (Dolly) and template-converted (Flan) instruction datasets, subsampling 15K examples from Flan to match Dolly. For Flan's multiple-choice tasks, perplexity (PPL) is used as the evaluation metric.

Noise Injection We inject noise parameters only in the region from instruction to input, while the other parts of the template remain undisturbed. In our main experiment, the injected Gaussian noise involves the configuration of two key parameters: mean and variance. Given that the information content of different instructions varies, it is clearly unreasonable to use fixed parameter values. Therefore, we have adopted an adaptive parameter setting method. For each instruction, after embedding, we calculate the specific variance and mean of the region where noise is to be injected, and use these calculated values for initialization to achieve an appropriate semantic shift.

B More Experiments about Noise Intensity

In this paper, we primarily report the data filtering results with β set to 1 and 10. To provide a more comprehensive study, we further include experiments with a wider range of β values, as shown in Table 8. The comparative analysis reveals that when the noise intensity is either very low or very high, the model's performance is close to the baseline. This can be attributed to the fact that overly small noise fails to introduce meaningful perturbations, whereas excessively large noise overwhelms the signal, making it difficult for the model to effectively identify the relevant problems.

	MMLU	Math	Code	COM	NQ	Average
Alpaca_all	47.93	13.12	13.41	55.04	20.83	30.07
Ours($\beta = 3$)	46.13	11.90	15.24	53.48	26.59	30.67
Ours($\beta = 5$)	45.28	14.10	17.07	51.76	28.86	31.41
Ours($\beta = 10$)	47.12	15.69	15.85	56.51	29.83	**33.00**
Ours($\beta = 15$)	43.13	13.57	15.24	53.89	28.37	30.90

Table 8: More about the impact of different noise intensities on performance