MAIN: Mutual Alignment Is Necessary for instruction tuning

Anonymous ACL submission

Abstract

Instruction tuning has empowered large language models (LLMs) to achieve remarkable performance, yet its success heavily depends on the availability of large-scale, high-quality instruction-response pairs. To meet this demand, various methods have been developed to synthesize data at scale. However, current methods for scaling up data generation often overlook a crucial aspect: the alignment between instructions and responses. We hypothesize that the quality of instruction-response pairs is determined not by the individual quality of each component, but by the degree of mutual alignment. To address this, we propose a Mutual Alignment Framework (MAIN) which enforces coherence between instructions and responses through mutual constraints. We demonstrate that MAIN generalizes well across model architectures and sizes, achieving state-of-the-art performance on LLaMA, Mistral, and Qwen models across diverse benchmarks. This work underscores the critical role of instruction-response alignment in enabling generalizable and high-quality instruction tuning for LLMs. All code is available from our anonymous repository.

1 Introduction

Large Language Models have demonstrated unprecedented capabilities in comprehending human intent and performing cross-task generalization through contextual learning(Brown et al., 2020). A key breakthrough in aligning model behaviors with human expectations is primarily attributed to instruction tuning, a supervised learning paradigm that bridges the gap between pre-trained models' latent knowledge and explicit task requirements (Ouyang et al., 2022). Through multi-task training on (instruction, response) pairs, this approach enables systematic knowledge elicitation while maintaining task-agnostic generalization (Chung et al., 2024). The effectiveness of this process is significantly influenced by the availability of high-quality instruction-response pairs at scale. In essence, the quality of data used in instruction tuning is critical to determining the performance and overall effectiveness of the model.

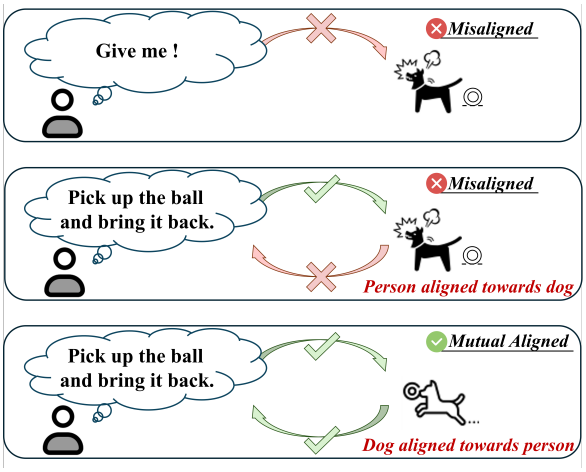

Figure 1: This figure illustrates a common interaction where a person and a dog adjust their behaviors to align instruction with response, evolving through repeated interactions to achieve mutual understanding.

Instruction-tuning methods currently follow two primary approaches. The first involves engaging domain experts (Köpf et al., 2024; Conover et al., 2023; Bach et al., 2022) to manually create instructions for specific tasks, ensuring high precision but facing challenges related to scalability and cost. The second approach (Wang et al., 2022a; Peng et al., 2023) leverages LLMs to generate responses based on given prompts. Although this approach is more scalable, it risks introducing inaccuracies or hallucinations (Zhang et al., 2023).

Recent research has explored an alternative: leveraging human-written documents as typical responses and using LLMs to infer user instructions (Köksal et al., 2023; Li et al., 2023a; Chen et al., 2024; Nguyen et al., 2024), a process known

as instruction back-translation. These approaches primarily focused on making the generated data resemble human data, without considering the inherent relationship between the instruction and the response. We contend that the alignment between the instruction and the response is also essential.

As shown in Figure 1, the interaction between a person and a dog illustrates the bidirectional nature of training. Both the person and the dog adjust their behaviors to achieve mutual alignment. Similar to how a good command to a dog is one that elicits a proper response, generating an instruction-response pair must be aligned for optimal effectiveness. The quality of the instruction is validated by the response it triggers, and the same logic applies in reverse. Generating a high-quality pair requires careful alignment through mutual interaction. The instruction must clearly guide the response, while the response should accurately reflect the instruction, ensuring that both are mutually reinforcing.

The interdependence between instructions and responses introduces a dual-variable optimization problem, where enhancing one component necessitates adjusting the other simultaneously, as neither can be optimized in isolation. Drawing inspiration from the alternating update strategy used in Expectation-Maximization (EM) algorithms (Moon, 1996), we propose **MAIN**, a framework for synthesizing high-quality data. This framework iteratively optimizes both instructions and responses, progressively reinforcing their mutual alignment. Through this co-adaptive process, the alignment between instruction-response pairs improves substantially, which we believe will significantly boost the model's performance. Furthermore, we propose a straightforward but effective filtering strategy, **mutual filter**, which selects pairs with superior alignment, ultimately boosting the quality of the fine-tuning dataset.

To validate the effectiveness of our proposed MAIN framework, we conducted extensive evaluations by fine-tuning models with instruction-response pairs generated by MAIN across multiple benchmarks. Experimental results demonstrate substantial improvements in output preference, instruction-following capability, and reasoning ability. Specifically, for the LLaMA-2-7B model, our framework achieves a 5.85% increase in output preference compared to Dog Instruct (Chen et al., 2024), and a 3.60% improvement in instruction-following ability over Better Alignment (Nguyen et al., 2024). Furthermore, additional analyses, including experiments on filtering strategies and GPT-4-based pairwise evaluations of instruction alignment, confirm that MAIN's mutual alignment enhances the coherence and quality of instruction-response pairs. Our primary contributions are as follows:

- We emphasize the critical importance of mutual alignment between instructions and responses in synthesizing high-quality instruction-tuning data.
- We propose MAIN, a mutual alignment framework that reinforces the inner connection between instructions and responses, and develop a straightforward but efficient data filtering method.
- We conduct extensive evaluations across diverse model families and parameter scales, showing that MAIN outperforms existing methods in enhancing instruction tuning effectiveness.

2 Methodology

In this section, we present our proposed Mutual Alignment Framework, designed to enhance instruction tuning performance by establishing and strengthening the intrinsic alignment between instructions and responses.

2.1 Preliminary

Data The framework utilizes two primary datasets: a limited set of high-quality, human-annotated instruction-response pairs seed data $D_{\text{seed}} = \{(I, R)\}$ and a larger collection of unlabeled responses $D_{\text{unlabeled}} = \{R_u\}$, extracted from web corpus.

Models The forward model $M_f := p(R|I)$ is designed to follow instructions, generating responses given instructions, while the reverse model $M_r := p(I|R)$ learns to generate instructions given responses.

2.2 Data Synthesis Framework: MAIN

We present our data synthesis framework, MAIN, illustrated in Figure 2. Given a base language model, a small set of high-quality seed pairs, and a large collection of unlabeled responses, MAIN constructs a high-quality training dataset through three tightly coupled stages: **Mutual Alignment**, **Data Augmentation**, and **Data Curation**.

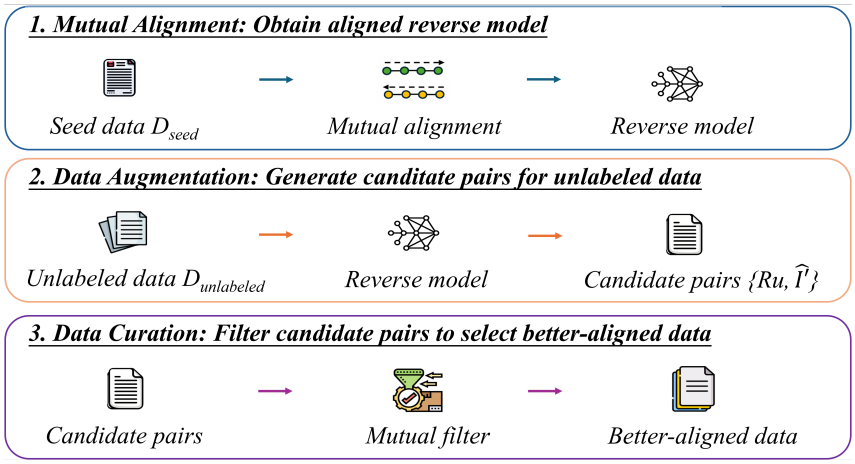

Figure 2: An overview of the data synthesis process, including mutual alignment, data augmentation, and data curation, aimed at creating high-quality, well-aligned instruction-response pairs from both seed and unlabeled data.

- Mutual Alignment: This step is to obtain a reverse model $M_r := p(I|R)$ from the seed data D_{seed} based on the base model M_{base}. This step would align the internal relationship between instruction and response.

- Data Augmentation: With the reverse model M_r trained in the previous step, we apply it to the unlabeled response data $\mathcal{D}_{\text{unlabeled}}$ to generate corresponding pseudo-instructions. This yields a set of candidate pairs $\mathcal{D}_{\text{aug}} = \{(R_u, \hat{I}')\}$, expanding the data space beyond the original seed data.

- Data Curation: Not all augmented pairs are equally reliable. To select high-quality examples, we apply our mutual filter, which uses both the forward and reverse models to assess alignment consistency. Only examples that meet mutual alignment criteria are retained. These filtered pairs, combined with the original seed data, form the final fine-tuning dataset: $\mathcal{D}_{\text{filter}} = \text{filter}(\mathcal{D}_{\text{aug}}) \cup \mathcal{D}_{\text{seed}}$.

2.3 Mutual Alignment

Achieving strong alignment between instructions and responses is critical for effective instruction tuning. However, establishing a robust relationship between these two components presents a challenging dual-variable problem, as neither direction can be optimized in isolation. Inspired by the iterative principles of the Expectation-Maximization algorithm, we propose mutual alignment that treats instruction-to-response and response-to-instruction generation as complementary tasks, modeled as a forward generation process and a reverse generation process, respectively. By alternately optimizing one direction while regulating the other, our method iteratively minimizes discrepancies until convergence is reached, ultimately yielding a model that produces highly aligned instruction–response pairs.

An overview of our approach is provided in Figure 3, and Algorithm 1 details the iterative optimization process.

Algorithm 1 Mutual Alignment

Input: Seed data $\mathcal{D}_{\text{seed}} = \{(I, R)\}$, Unlabeled data $\mathcal{D}_{\text{unlabeled}} = \{R_u\}$, Base model M_{base}, Number of iterations N

Output: Reverse model M_r^N, forward model M_f^N

Initialize $M_f^0 \leftarrow M_{\text{base}}$, $M_r^0 \leftarrow M_{\text{base}}$
for $k = 0$ **to** $N - 1$ **do**
  Generate $\hat{I}$ from R in $\mathcal{D}_{\text{seed}}$ using M_r^k
  Build training set $\mathcal{D}_f = \{(\hat{I}, R)\} \cup \mathcal{D}_{\text{seed}}$
  Update M_f^k on $\mathcal{D}_f$ by minimizing loss $\mathcal{L}_f$ (Equation (2)) to obtain M_f^{k+1}
  Generate pseudo-responses $\hat{R}$ from I in $\mathcal{D}_{\text{seed}}$ using M_f^{k+1}
  Build training set $\mathcal{D}_r = \{(\hat{R}, I)\} \cup \mathcal{D}_{\text{seed}}$
  Update M_r^k on $\mathcal{D}_r$ by minimizing loss $\mathcal{L}_r$ (Equation (4)) to obtain M_r^{k+1}
end for
Return M_r^N and M_f^N

Forward Model Alignment. To capture the alignment from responses to instructions, we let

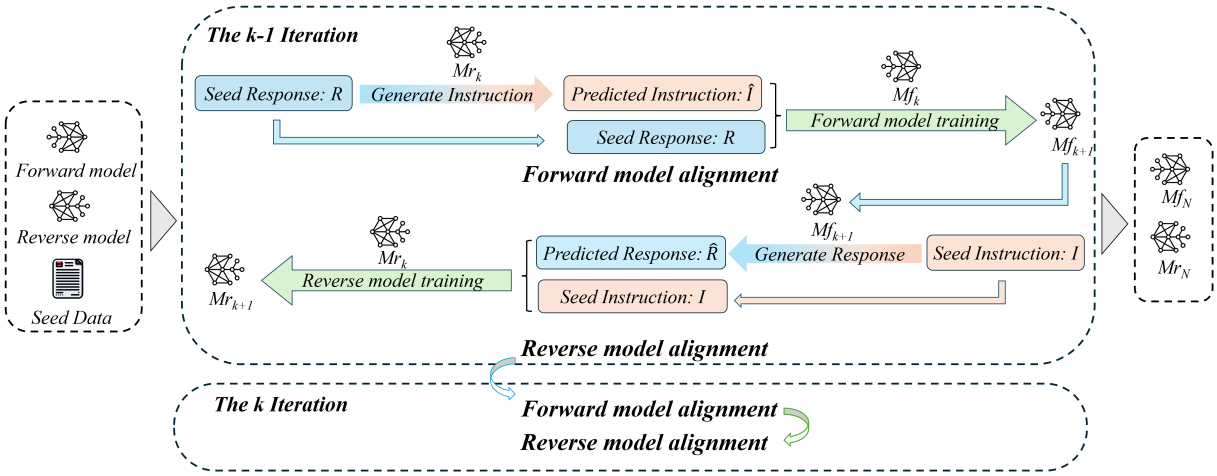

Figure 3: An overview of our method for iteratively aligning instructions and responses through mutual optimization.

the forward model learn the distribution that the reverse model simulates. Specifically, at each iteration k, the reverse model generates synthetic instructions $\hat{I}$ for the responses, forming a target distribution that reflects how instructions should ideally relate to responses. The forward model is then trained to approximate this distribution.

$$\hat{I} = M_r^k(R), \quad \forall R \in \mathcal{D}_{\text{seed}}. \tag{1}$$

These synthetic pairs $(\hat{I}, R)$ are merged with the original seed data (I, R) to form the training set. The forward model is then updated to M_f^{k+1} by optimizing a weighted loss function:

$$\mathcal{L}_f = \alpha \cdot \mathcal{L}(\hat{I}, R) + (1 - \alpha) \cdot \mathcal{L}(I, R). \tag{2}$$

The first loss term $\mathcal{L}(\hat{I}, R)$ aligns the forward model with the synthetic instructions generated by the reverse model, ensuring that the forward model learns how responses correspond to instructions as modeled by the reverse model. The second loss term $\mathcal{L}(I, R)$ maintains consistency with the original human-annotated instructions, thereby preventing the forward model from overfitting to synthetic data. The parameter α controls the balance between synthetic and human-annotated instructions, with its dynamic adjustment described in Dynamic Weighting. This process encourages the forward model to adapt to the instruction distribution induced by the reverse model.

Reverse Model Alignment. Similarly, the reverse model is trained to capture the alignment from instruction to response as guided by the forward model. The reverse model now is updated based on the latest forward model M_f^{k+1} that generates synthetic responses $\hat{R}$ conditioned on the seed instructions:

$$\hat{R} = M_f^{k+1}(I), \quad \forall I \in \mathcal{D}_{\text{seed}}. \tag{3}$$

And it is optimized using similar weighted loss function:

$$\mathcal{L}_r = \alpha \cdot \mathcal{L}(\hat{R}, I) + (1 - \alpha) \cdot \mathcal{L}(R, I). \tag{4}$$

Dynamic Weighting To balance the influence of synthetic and seed data, we adopt a dynamic weighting strategy that adaptively adjusts their contributions during training. The weighting coefficient $\alpha \in [0, 1]$ controls this balance, where static settings may lead to suboptimal outcomes: over-reliance on synthetic data can introduce noise, while overemphasis on seed data may hinder generalization. To address this, we update α at each step based on the relative loss of the two data sources. For forward model training, the update rule is:

$$\alpha = \frac{\mathcal{L}(\hat{I}, R)}{\mathcal{L}(\hat{I}, R) + \mathcal{L}(I, R)}. \tag{5}$$

This formulation ensures that the contribution of synthetic data is modulated according to its training loss, enabling the model to incorporate novel patterns from synthetic pairs while preserving the stability offered by seed data.

2.4 Data Augmentation

Following mutual alignment optimization, we expand the training corpus by generating synthetic instructions for unlabeled responses. Specifically,

for each unlabeled response $R_u \in \mathcal{D}_{\text{unlabeled}}$, the reverse model generates a corresponding synthetic instruction $\hat{I}'$, forming a set of candidate instruction–response pairs $\{(R_u, \hat{I}')\}$. These pairs approximate how users might naturally formulate instructions for the given responses. However, as the quality of generated pairs may vary, a subsequent curation step is required to ensure alignment consistency and data reliability.

2.5 Data Curation

To further improve data alignment, we introduce an effective filtering mechanism. We assume that high-quality instruction-response pairs should be well-aligned, where the predicted instruction generated by the reverse model can be decoded by the forward model to recover the response, which should closely resemble the original. This process is akin to the interaction between an encoder and a decoder (Cho et al., 2014). Thus, we select the most well-aligned pairs. Using the candidate pairs $\{R_u, \hat{I}'\}$ from the Augmentation stage, we then employ the forward model to generate synthetic responses $\hat{R}'$ based on $\hat{I}'$:

$$\hat{R}' = M_f^N(\hat{I}'). \tag{6}$$

We compute the Cross-Entropy between the synthetic responses $\hat{R}'$ and the original unlabeled responses R_u:

$$\mathcal{L}_{\text{CE}}(\hat{R}', R_u) = -\sum_i \log p(\hat{R}' \mid \hat{I}', R_u). \tag{7}$$

Candidate pairs are sorted in ascending order by their values, and only those with the smallest values—indicating the highest degree of mutual alignment are retained.:

$$\mathcal{D}_{\text{filter}} = \left[\text{filter}(\mathcal{D}_{\text{aug}}), \mathcal{D}_{\text{seed}}\right] \tag{8}$$

This straightforward mechanism, relying solely on our mutual alignment model, effectively curates a high-quality subset of data for fine-tuning.

3 Experiment

3.1 Experimental Setup

Data. The seed data consists of 3,200 human-annotated (instruction, response) examples from the Open Assistant dataset (Köpf et al., 2024), serving as a reliable baseline for fine-tuning. The unlabeled data is Falcon RefinedWeb (Penedo et al., 2023) that is a massive English web dataset containing raw responses without paired instructions. We sampled 502k segments.

Mutual Alignment Framework. For mutual alignment experiments, we adopt LLaMA-2-7B as the base model, and additionally evaluate the generalization of our approach on Mistral and Qwen models. In each iteration, both the forward and reverse models are trained for one epoch. We use a learning rate of 1×10^{-5} with a linear decay schedule. The adaptation weight α is dynamically updated according to Equation 5. The batch size is set to 32. For data curation, we select the top 16,800 instruction-response pairs from the unlabeled set based on cross-entropy scores and combine them with the seed data to form the final fine-tuning dataset.

Base model & fine-tuning. We use the pre-trained LLaMA, Mistral and Qwen as the base models respectively. Detailed hyperparameter configurations are provided in Appendix A.

3.2 Benchmarks

To evaluate our framework, we conduct experiments across three benchmarks that assess different aspects of model performance.

AlpacaEval. We assess output preference using 805 instructions from the AlpacaEval dataset (Li et al., 2023b). Model outputs are compared against text-davinci-003 in a pairwise setting, with GPT-4-based judgments determining win rates.

IFEval. Instruction-following ability is evaluated with IFEval (Zhou et al., 2023), which reports accuracy across four metrics: Prompt-level Strict (P-S), Instruction-level Strict (I-S), Prompt-level Loose (P-L), Instruction-level Loose (I-L) ensuring a comprehensive assessment of instruction adherence.

OpenLLM. Reasoning ability is measured via the Open LLM Leaderboard (Beeching et al., 2023) using the Language Model Evaluation Harness (Gao et al., 2023). We evaluate on ARC (Clark et al., 2018), HellaSwag (Zellers et al., 2019), Winogrande (Sakaguchi et al., 2021), MMLU (Hendrycks et al., 2020), and TruthfulQA (Lin et al., 2021).

3.3 Baselines

We compare our framework to several baseline approaches, fine-tuned on 3.2k seed data and 16.8k generated data from Falcon-RefinedWeb dataset.

Longform. This method (Köksal et al., 2023) prompts a large language model to generate instructions for human-written texts.

Humpback. This method (Li et al., 2023a) is a two-stage curation process that filters and selects high-quality pairs before fine-tuning.

Dog Instruct. This method (Chen et al., 2024) involves a post-processing step that refines the responses to align with standard AI-generated output.

Better Alignment. This method (Nguyen et al., 2024) generates instructions via back-translation, then filters pairs to obtain high-quality response.

4 Experimental Results

This section presents the experimental results, including quantitative results, an ablation study, data alignment analysis and a case study, to assess the effectiveness of our approach.

4.1 Quantitative Results

We conduct experiments across three benchmarks, each assessing a different aspect of the MAIN: AlpacaEval evaluates output preference, IFEval measures instruction-following ability, and OpenLLM tests reasoning capability, with additional experiments detailed in Appendix B

Output Preference. As shown in Table 1, our method achieves the highest win rate in AlpacaEval dataset, surpassing the leading baseline. Specifically, on Llama-2-7B, our method achieves a win rate of 58.20%, representing a 5.85% improvement over the best baseline Dog Instruct (Chen et al., 2024). On Mistral-7B, our method outperforms Better alignment (Nguyen et al., 2024) by 3.15%, reaching a win rate of 48.94% compared to 45.79%. On Qwen2.5-14B, MAIN achieves the highest win rate of 71.30%. These results confirm that our MAIN method enhances instruction-response alignment more effectively than previous approaches, leading to outputs that better align with human expectations.

Instruction Following. Table 1 illustrates the results of our method in IFEval dataset where MAIN achieves state-of-the-art performance across all three model backbones. Compared to the best-performing baseline, our approach achieves consistent improvements across all evaluation metrics. Specifically, on Llama-2-7B, we see an increase of 2.59% in P-S and 3.42% in I-S. For Mistral-7B, we observe a 5.12% improvement in P-S and a 4.84% improvement in I-S over Better alignment (Nguyen et al., 2024). For Qwen2.5-14B, MAIN also leads with a +2.07% and +2.99% gain in P-S and I-S respectively. These results highlight the crucial role of enhanced data alignment in fine-tuning, which allows our model to better interpret and respond to user instructions, thereby driving its superior performance in instruction-following tasks.

Reasoning Ability. As shown in Table 1, our method demonstrates strong improvements in reasoning and factual accuracy across multiple downstream tasks. On Llama-2-7B, our approach shows a 2.02% improvement over the best baseline, Better Alignment, on ARC-Challenge and a 1.63% improvement on TruthfulQA. In particular, ARC-Challenge benefits from our method's ability to better capture common-sense reasoning patterns, which likely leads to more accurate responses. On Mistral-7B, the most significant improvements are observed in TruthfulQA, where our method outperforms Better Alignment by 5.03%, and in MMLU, with a 2.65% increase. We further include evaluation on Qwen2.5-14B, which follows a similar trend. These benchmarks that require accurate factual recall and complex reasoning, show how our method strengthens the model's ability to provide correct and contextually appropriate answers.

4.2 Ablation Study

We conduct additional ablation study to analyze the impact of our filtering strategy. Further ablation experiments are provided in Appendix C.

Filtering Stragety. Effective filtering is critical for improving alignment quality by removing noisy or misaligned instruction–response pairs. Table 2 compares models trained with no filtering, score-based filtering, and our proposed mutual-filter method. Among these, our mutual-filter achieves the highest win rate and delivers the most consistent gains in instruction-following accuracy.

In contrast, score-based filtering provides little benefit and even underperforms compared to unfiltered data. This is due to the score-based approach, used in Humpback (Li et al., 2023a) and Better Alignment (Nguyen et al., 2024), relying on a ranking model fine-tuned on seed data rather than a dedicated scoring model. Without a clear optimization objective for instruction-response alignment, it struggles to identify high-quality pairs, leading to suboptimal fine-tuning.

Base Model	Method	Output preference	Instruction following				Reasoning ability				
		AlpacaEval	IFEval				ARC_C	MMLU	HellaSwag	Winogrande	TruthfulQA
			P-S	I-S	P-L	I-L					
Llama-2-7B	Humpback (Li et al., 2023a)	41.02	15.46	26.42	18.39	29.91	55.90	44.91	79.42	73.32	44.48
	Longform (Köksal et al., 2023)	35.64	15.23	26.10	17.56	29.29	55.72	45.02	78.98	73.21	45.07
	Dog Instruct(Chen et al., 2024)	52.35	15.52	28.17	19.40	32.01	56.06	45.62	79.89	74.13	45.77
	Better Alignment (Nguyen et al., 2024)	50.37	16.82	27.70	19.69	31.52	55.92	**45.84**	80.33	74.12	45.30
	MAIN	**58.20**	**20.22**	**31.17**	**23.36**	**35.37**	**57.08**	45.47	**81.22**	**74.51**	**47.40**
	Δ over Best Result	+5.85	+3.40	+3.00	+3.67	+3.36	+1.02	-0.37	+0.89	+0.38	+1.63
Mistral-7B	Humpback (Li et al., 2023a)	40.48	17.19	28.05	20.88	32.37	54.01	49.26	79.12	**73.24**	45.48
	Longform (Köksal et al., 2023)	37.62	16.98	27.89	20.75	32.10	53.98	48.12	78.25	71.60	44.88
	Dog Instruct(Chen et al., 2024)	45.34	18.23	28.48	21.32	33.47	53.15	49.10	79.07	73.21	45.98
	Better Alignment(Nguyen et al., 2024)	45.79	18.35	29.45	21.49	34.10	54.20	50.28	78.37	71.48	44.73
	MAIN	**48.94**	**23.47**	**34.29**	**26.60**	**38.84**	**55.12**	**52.93**	**79.38**	72.38	**49.76**
	Δ over Best Result	+3.15	+5.12	+4.84	+5.11	+4.74	+0.92	+2.65	+0.31	-0.86	+3.78
Qwen2.5-14B	Humpback (Li et al., 2023a)	49.83	69.45	79.92	74.28	82.25	65.25	77.04	83.81	78.91	57.63
	Longform (Köksal et al., 2023)	45.42	69.12	79.37	74.05	82.46	65.12	76.92	83.04	76.55	55.72
	Dog Instruct (Chen et al., 2024)	50.27	72.83	81.89	76.64	84.72	64.71	78.33	84.19	79.12	56.34
	Better Alignment (Nguyen et al., 2024)	53.93	71.37	81.55	76.32	84.14	65.36	78.01	**84.25**	79.63	55.91
	MAIN	**61.30**	**74.90**	**84.03**	**79.93**	**87.61**	**66.93**	**80.26**	84.07	**80.72**	**59.49**
	Δ over Best Result	+7.37	+2.07	+2.14	+2.29	+2.89	+1.57	+1.93	-0.18	+1.09	+3.15

Table 1: Benchmarking results of different methods on Llama-2-7B, Mistral-7B and Qwen2.5-14B using Falcon RefinedWeb dataset given same data quantity (20k samples). Δ over Best Result quantify improvements relative to the strongest baseline method across evaluation categories.

Our mutual filter, by leveraging mutual-alignment models, directly favors instruction-response pairs with strong coherence. By eliminating misaligned samples without requiring additional supervision, it ensures a more effective training dataset, resulting in improved instruction-following and generalization capabilities.

Filtering Method	Win Rate	P-S	I-S	P-L	I-L
Ours w/o filtering	56.40	19.41	23.11	29.74	34.17
Ours w/ score-based filtering	55.26	17.63	20.21	29.11	33.72
Ours w/ mutual filter	58.20	20.22	23.36	31.17	35.37

Table 2: Performance evaluation of LLaMA-2-7B fine-tuned on the Falcon-RefinedWeb dataset (20K samples) under three filtering conditions. All conditions operate on instruction-response pairs generated by the same reverse model.

4.3 Data Alignment Analysis

We assessed the alignment quality of instructions generated by our MAIN method compared to baselines using blind pairwise evaluations conducted by GPT-4. Specifically, we randomly selected 1000 responses from the Falcon RefinedWeb dataset. For each response, GPT-4 evaluated two candidate instructions—one from MAIN and one from a baseline—in random order to avoid positional bias. We then calculated Win, Tie, and Loss rates based on GPT-4's judgments: Win indicates GPT-4 preferred MAIN, Tie indicates no clear preference, and Loss indicates GPT-4 preferred the baseline. Using the Qwen2.5-14B as the base model, MAIN consistently outperformed all baselines, with win rates ranging from 61.7% to 81.6%, demonstrating superior instruction alignment capability (see Table 3). The evaluation prompt used by GPT-4 is detailed in Appendix F.

Baseline	Win Rate	Tie Rate	Loss Rate	Δ
Humpback	69.3%	18.6%	12.1%	+56.2%
Longform	81.6%	8.6%	8.8%	+72.8%
Dog Instruct	61.7%	15.1%	23.2%	+38.5%
Better Alignment	64.7%	12.4%	22.9%	+41.8%

Table 3: GPT-4 pairwise evaluation comparing **MAIN** with four baseline methods on instruction alignment quality. Each comparison is based on 1000 examples sampled from the Falcon RefinedWeb dataset. Δ denotes the win rate margin of MAIN over each baseline.

4.4 Case Study

As shown in Figure 4, the two examples, both extracted from unlabeled data, illustrate the effectiveness of our approach. In the first case, the MAIN instruction explicitly requests specific details about the victim, the shooter, and the events surrounding the shooting, providing clear guidance for the response. In contrast, the baseline instruction is more general, asking for a brief article about the shooting without specifying key details. In the second case, the MAIN instruction emphasizes a critical event: a wave of car burglaries in the suburbs, while the baseline instruction remains vague, simply requesting a summary of events in suburban areas.

In both cases, MAIN Instructions are more fo-

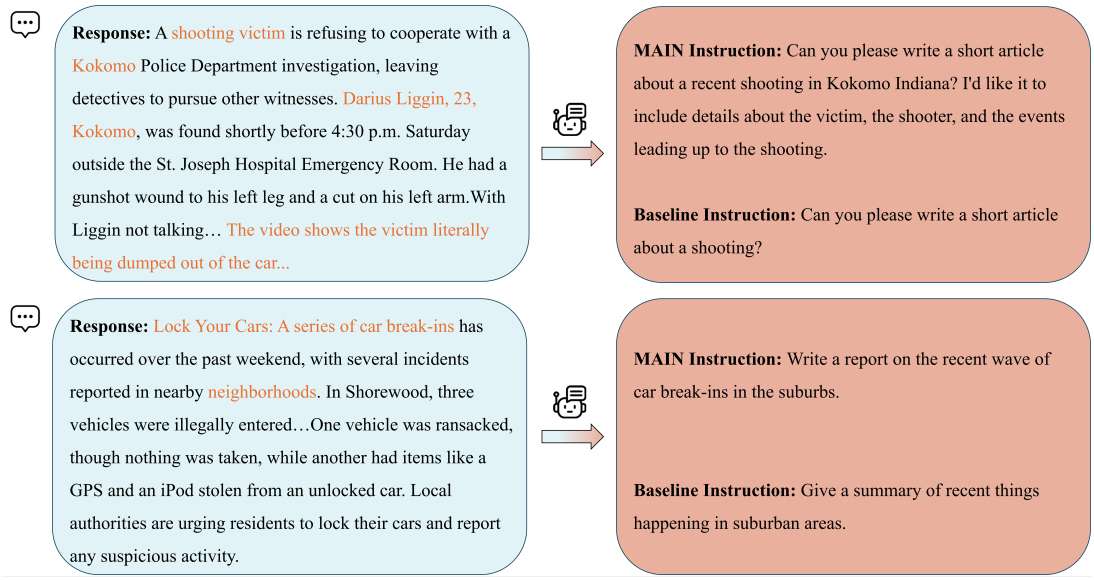

Figure 4: Method Comparison for Instruction Generation: A Case Study on the Effectiveness of Reverse Model Approaches in Aligning Instructions with Responses

cused and specific, resulting in responses that are better aligned with the intended context. In contrast, the baseline instructions are more general. These examples demonstrate that our method generates instructions that are more closely aligned with the responses.

5 Related Work

5.1 Instruction Tuning

Instruction tuning fine-tunes pre-trained LLMs on instruction-response pairs, enabling models to generalize across tasks without task-specific fine-tuning (Wei et al., 2021; Mishra et al., 2021; Wang et al., 2022b). Subsequent work (Mishra et al., 2021; Sanh et al., 2021) focused on cross-task generalization through diverse inputs.

5.2 Data Generation

Effective instruction tuning relies on large-scale, high-quality datasets, typically generated in two ways: human-crafted or model-generated.

Human-Crafted Data Datasets curated by domain experts, like OpenAssistant Conversations (Köpf et al., 2024) and Databricks Dolly-15k (Conover et al., 2023), are high quality but costly. Crowd-sourced platforms like ShareGPT (Chiang et al., 2023) also contribute valuable data, especially user-uploaded conversations.

Model-Generated Data To reduce manual annotation costs, methods like Self-Instruct (Wang et al., 2022a) and Alpaca-GPT4 (Peng et al., 2023) generate instruction-response pairs automatically. However, issues like hallucinations (Zhang et al., 2023) persist. New approaches, such as Better Alignment (Nguyen et al., 2024) and Dog-Instruct(Chen et al., 2024), pair human responses with inferred instructions to reduce hallucinations and improve scalability. Our proposed MAIN builds on this by iteratively optimizing instruction-response alignment to ensure high-quality data.

6 Conclusion

In this paper, we highlight the critical role of instruction-response alignment in instruction tuning for LLMs. We introduce the Mutual Alignment Framework, which iteratively optimizes both instructions and responses to improve their coherence, along with a mutual filtering strategy to select high-quality pairs. Experiments across multiple benchmarks show that our framework enables state-of-the-art performance. These findings highlight the importance of mutual alignment in instruction tuning and offer a new perspective for refining instruction-response pairs, paving the way for more effective and principled instruction tuning in future LLM development.

Limitations

Our experiments cover a broad range of model families and parameter scales, demonstrating the robustness of the proposed framework across ar-

chitectures and sizes. However, we have not yet evaluated MAIN on very large models due to resource limit, which may exhibit qualitatively different behaviors. This limits our ability to fully assess the potential of mutual alignment at extreme scales. Investigating its effectiveness in such settings remains an important direction for future work.

References

Stephen H Bach, Victor Sanh, Zheng-Xin Yong, Albert Webson, Colin Raffel, Nihal V Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, et al. 2022. Promptsource: An integrated development environment and repository for natural language prompts. *arXiv preprint arXiv:2202.01279*.

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. 2023. Open llm leaderboard (2023-2024). https://huggingface.co/spaces/open-llm-leaderboard-old/open_llm_leaderboard.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Yongrui Chen, Haiyun Jiang, Xinting Huang, Shuming Shi, and Guilin Qi. 2024. Dog-instruct: Towards premium instruction-tuning data via text-grounded instruction wrapping. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4125–4135.

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.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

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

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

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

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

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Abdullatif Köksal, Timo Schick, Anna Korhonen, and Hinrich Schütze. 2023. Longform: Effective instruction tuning with reverse instructions. *arXiv preprint arXiv:2304.08460*.

Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, et al. 2024. Openassistant conversations-democratizing large language model alignment. *Advances in Neural Information Processing Systems*, 36.

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

Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2021. Cross-task generalization via natural language crowdsourcing instructions. *arXiv preprint arXiv:2104.08773*.

Todd K Moon. 1996. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60.

Thao Nguyen, Jeffrey Li, Sewoong Oh, Ludwig Schmidt, Jason Weston, Luke Zettlemoyer, and Xian Li. 2024. Better alignment with instruction back-and-forth translation. *arXiv preprint arXiv:2408.04614*.

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

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022a. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. 2022b. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *arXiv preprint arXiv:2204.07705*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A Smith. 2023. How language model hallucinations can snowball. *arXiv preprint arXiv:2305.13534*.

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

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

A Training Details

Hyperparameter	Assignment
Computing Infrastructure	8 A100-80GB GPUs
Number of epochs	2
Batch size per GPU	64
Maximum sequence length	1024
Maximum learning rate	2e-5
Optimizer	Adam
Adam epsilon	1e-8
Adam beta weights	0.9, 0.999
Learning rate scheduler	warmup linear
Weight decay	0.1
Warmup steps	100
Learning rate decay	linear

Table 4: Hyperparameters used in the experiments.

Training is conducted with hyperparameters aligned to established supervised fine-tuning (SFT) practices (Zhou et al., 2024; Touvron et al., 2023). The learning rate is set to 2×10^{-5}, with a weight decay of 0.1, a batch size of 64, and a dropout rate of 0.1. Additionally, each iterative phase of training is limited to one epoch. For text generation, we apply nucleus sampling (Holtzman et al., 2019) with a temperature (T) of 0.7 and a top-p value of 0.9. These settings balance diversity and relevance in the generated outputs. More hyperparameters listed in Table 4

B Generalization Results

Table 5 presents extended results that demonstrate both the scalability across varying model sizes across diverse architectures of our MAIN method. These capabilities are evidenced by MAIN's enhanced performance compared to the baseline (Nguyen et al., 2024) on the IFEval benchmark when evaluated on models such as Qwen2.5-3B and LLaMA-2-13B.

Model	Method	P-S (%)	I-S (%)	P-L (%)	I-L (%)
Qwen2.5-3B	Baseline	32.13	35.98	44.02	47.10
	MAIN	35.78	39.12	47.42	51.21
LLaMA-2-13B	Baseline	16.12	18.72	28.54	30.20
	MAIN	18.98	22.63	30.87	34.85

Table 5: Comparative instruction-following performance of MAIN and Baseline on the IFEval dataset across diverse models.

Further evaluations in Table 6 demonstrate that our MAIN framework consistently outperforms Baseline (Nguyen et al., 2024) across diverse NLP benchmarks (BLEU, ROUGE-L, SQuADv2).

Base Model	Method	BLEU	ROUGE-L	SQuADv2 (EM / F1)
Llama-2-7B	Baseline	43.94	43.82	10.82 / 19.62
	MAIN	47.37	46.39	**12.99 / 21.30**
Mistral-7B	Baseline	40.75	39.41	14.53 / 21.72
	MAIN	43.15	44.70	**17.11 / 23.49**

Table 6: Comparative performance of MAIN and Baseline on diverse NLP benchmarks.

C Ablations

Training iterations. To analyze the impact of iteration count N, we vary N from 1 to 20 and evaluate its effect on AlpacaEval and IFEval dataset. As shown in Table 7, increasing N initially improves performance, as iterative refinement enables the forward and reverse models to progressively align their outputs, enhancing instruction-response consistency.

However, beyond a certain point, performance begins to decline. Excessive iterations reinforce suboptimal patterns leading to overfitting. This underscores the necessity of selecting an optimal N that balances refinement and generalization. Our results emphasize the importance of properly tuning N to maximize the benefits of mutual alignment.

Iterations N	Win Rate
$N = 1$	50.11
$N = 2$	55.72
$N = 3$	58.20
$N = 4$	55.89
$N = 5$	55.60
$N = 10$	54.41
$N = 15$	54.29
$N = 20$	54.52

Table 7: Ablation study on the effect of iteration count N. We analyze the influence of varying the number of training iterations ($N = 1, 2, 3, 4, 5, 10, 15, 20$) on Llama-2-7B fine-tuned on Falcon-RefinedWeb.

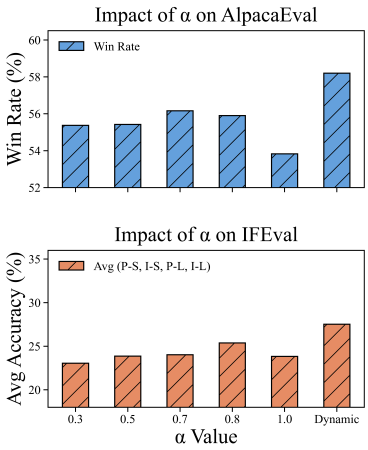

Figure 5: Evaluation of dynamic weighting strategies on LLaMA-2-7B training, comparing fixed and adaptive α values using the Falcon-RefinedWeb dataset, with performance assessed on AlpacaEval and IFEval.

Dynamic Weighting. Balancing the contribution of aligned instruction-response pairs is crucial for achieving both strong alignment and robust generalization. The weighting parameter α controls this balance during training by adjusting the relative influence of synthetic and seed data. To evaluate its effectiveness, we compare fixed values ($\alpha = 0.3, 0.5, 0.7, 0.8, 1.0$) with our adaptive approach ($\alpha =$ Dynamic), which continuously updates α throughout training.

As shown in Figure 5, increasing α generally improves instruction-following ability and output preference by emphasizing well-aligned pairs. However, excessively high α makes the model overly reliant on generated instruction-response pairs, leading to unstable training and degraded performance.

To mitigate this, our dynamic weighting strategy adaptively balances aligned and seed data, preventing instability while maintaining strong alignment. The results show that this approach significantly improves output preference and instruction-following.

D Detailed Benchmark Settings

We have evaluated our method under both few-shot and zero-shot conditions. Specifically, tasks such as ARC_C, HellaSwag, Winogrande, and MMLU were tested with a few-shot setup, whereas AlpacaEval, TruthfulQA, and IFEval benchmarks were evaluated under zero-shot conditions. Detailed settings and results are provided in Table 8

Dataset	Metric	Number of Shots
ARC_C	Acc_norm	25
TruthfulQA	Mc2	Zero-shot
Winogrande	Acc	5
HellaSwag	Acc	10
MMLU	Acc_norm	5
AlpacaEval	Win_rate	Zero-shot
IFEval	P-S, etc.	Zero-shot

Table 8: Evaluation settings and key metrics for benchmark datasets under few-shot and zero-shot conditions.

E Computational Cost Analysis

This section details the computational demands of our proposed method MAIN. While MAIN involves an iterative training process, the overall compute cost is carefully managed to remain modest. In each iteration, only one model is trained for a single epoch on the seed data, while the other performs inference—a lightweight operation. Typically, three iterations suffice for convergence, resulting in six training epochs across the forward and reverse models. In comparison, baseline methods (Li et al., 2023a; Nguyen et al., 2024) also train a reverse model and a ranking model for a similar number of epochs. We summarize estimated GPU hours in Table 9.

Method	Models Trained	Inference Needed	GPU Hours
Baseline	Ranking model + Reverse model	No	5.0
MAIN (Ours)	Forward model + Reverse model	Yes	5.3

Table 9: Estimated computational cost comparison for MAIN against baselines.

F GPT-4 Evaluation Prompt

The following table presents the exact prompt used to instruct GPT-4 during blind pairwise comparisons:

GPT-4 Pairwise Evaluation Prompt
Please act as an expert evaluator of instruction-response alignment. You are given a response and two candidate instructions: Instruction A and Instruction B. Your task is to decide which instruction is better aligned with the response. Evaluate based on the following aspects: - Alignment between instruction and response - logical consistency - natural language fluency Please output only one of the following: A win — if Instruction A is clearly better aligned with the response. B win — if Instruction B is clearly better aligned with the response. Tie — if both instructions are equally good or equally poor.

Table 10: Prompt template used in GPT-4 pairwise evaluation of instruction-response alignment.