On the Effect of Instruction Tuning Loss on Generalization

Anwoy Chatterjee[*†]
Department of Electrical Engineering
IIT, Delhi, India
anwoychatterjee@gmail.com

H. S. V. N. S. Kowndinya Renduchintala[†]
Media and Data Science Research
Adobe Inc., India
rharisrikowndinya333@gmail.com

Sumit Bhatia
Media and Data Science Research
Adobe Inc., India
sumit.bhatia@adobe.com

Tanmoy Chakraborty
Department of Electrical Engineering
IIT, Delhi, India
tanchak@iitd.ac.in

Abstract

Instruction tuning has emerged as a pivotal post-training paradigm that enables pre-trained language models to better *follow* user instructions. Despite its significance, little attention has been given to optimizing the loss function used. A fundamental, yet often overlooked, question is whether the conventional auto-regressive objective—where loss is computed only on response tokens, excluding prompt tokens—is truly optimal for instruction tuning. In this work, we systematically investigate the impact of differentially weighting prompt and response tokens in instruction tuning loss, and propose **W**eighted **I**nstruction **T**uning (WIT) as a better alternative to conventional instruction tuning. Through extensive experiments on five language models of different families and scale, three finetuning datasets of different sizes, and five diverse evaluation benchmarks, we show that the standard instruction tuning loss often yields suboptimal performance and limited robustness to input prompt variations. We find that a low-to-moderate weight for prompt tokens coupled with a moderate-to-high weight for response tokens yields the best-performing models across settings and also serves as a better starting point for the subsequent preference alignment training. These findings highlight the need to reconsider instruction-tuning loss and offer actionable insights for developing more robust and generalizable models. Our code is open-sourced here.

[†]These two authors contributed **equally** to this work.
[*]Work done during an internship at Media and Data Science Research (MDSR) Lab, Adobe Inc.

1 Introduction

Transformer-based language models (LMs) pre-trained using just an auto-regressive objective over massive text corpora (Brown et al., 2020; Touvron et al., 2023) demonstrate remarkable performance across a range of NLP tasks (Zhao et al., 2021; Wang et al., 2022a; Wan et al., 2023; Sun et al., 2023). However, they often struggle to reliably follow user instructions as they are essentially *text-completion* models, whose pre-training objective, i.e., next-token prediction, has a fundamental mismatch with the goal of instruction following.

Instruction tuning aims to bridge this gap by finetuning an LM on a diverse collection of task instances phrased as instructions (Wei et al., 2022; Sanh et al., 2022; Ouyang et al., 2022), where each task instance consists of a task description (i.e., the instruction), an optional input, a corresponding output, and in some cases, a few demonstrations. Instruction tuning has been shown to significantly improve instruction following capability and generalization of LMs to unseen tasks (Wang et al., 2022b; Wei et al., 2022; Sanh et al., 2022; Chung et al., 2024), and hence has emerged as a widely adopted method in adapting pre-trained LMs to better follow user instructions.

While many studies have shown that the effectiveness of instruction tuning is heavily contingent on various factors such as task composition (Wang et al., 2023; Dong et al., 2024; Renduchintala et al., 2024), data quality (Zhou et al., 2023a; Ding et al., 2023), data quantity (Ji et al., 2023; Yuan et al., 2023), and training dynamics (Mukherjee et al., 2023; Pareja et al., 2025),

Transactions of the Association for Computational Linguistics, vol. 13, pp. 1360–1380, 2025. https://doi.org/10.1162/TACL.a.42
Action Editor: Sebastian Ruder. Submission batch: 3/2025; Revision batch: 6/2025; Published 10/2025.
© 2025 Association for Computational Linguistics. Distributed under a CC-BY 4.0 license.

a very fundamental yet under-explored factor is the loss function itself. The most commonly utilized loss function for instruction tuning is an auto-regressive objective where loss on prompt tokens is zeroed out (Aribandi et al., 2022; Li et al., 2024; Touvron et al., 2023; Chiang et al., 2023; Mitra et al., 2023), thereby backpropagating only on response tokens. Although the conventional loss function has been shown to be effective in practice, it is not clear *why* this should be the optimal choice, and to the best of our knowledge, there has not been a comprehensive study on the choice of the loss function to be used for instruction tuning.

Although a couple of recent studies (Huerta-Enochian and Ko, 2024; Shi et al., 2025) explored alternative instruction tuning loss formulations, they still leave out a lot of open questions. For instance, Shi et al. (2025) proposed Instruction Modeling, which does not zero out the loss on prompt tokens and instead employs the same auto-regressive objective used in the pre-training step—effectively treating instruction tuning as continual pre-training. However, this is only found to be beneficial when lengthy prompts are coupled with brief responses or when only a small number of training examples are involved. Similarly, Huerta-Enochian and Ko (2024) proposed using a small non-zero weight on prompt tokens, called prompt loss weight (PLW). The authors found that a non-zero PLW is beneficial when working with instruction-tuning data containing short completions and that it can safely be ignored when working with instruction-tuning data containing longer completions. However, its applicability across diverse training and evaluation datasets remains unexplored. Moreover, the extent to which prompt token weights should depend solely on the relative length of completions to prompts remains unclear.

While both these approaches offer some promising directions, they also reveal a deeper issue: The conventional loss function treats prompt and response tokens in a binary fashion, ignoring the former entirely during loss computation and giving full weight to response tokens. Prompts carry critical task-specific cues and implicit instructions that shape the model's response. Ignoring their learning signal may deprive the model of valuable contextual guidance, while fully emphasizing response tokens can lead to overfitting on response patterns. Recent concerns about models memorizing response patterns (Jain et al., 2024; Shi et al., 2025; Chu et al., 2025) further highlight the need for a more flexible loss formulation for instruction tuning. We hypothesize that by differentially weighting prompts and responses, we can better balance the contributions of contextual understanding and response generation, thereby fostering improved generalization.

To this end, we propose **Weighted Instruction Tuning** (`WIT`) as an alternative to the conventional instruction tuning loss that assigns different weights to prompt and response tokens, enabling more fine-grained control of what the model learns. Figure 1 illustrates this notion of differential weighting and shows how it differs from standard approaches of instruction tuning and continual pre-training. We perform extensive fine-tuning experiments using this new loss function, by training 525 models with different weights on prompt and response tokens across different model families, model sizes, and instruction tuning datasets. Furthermore, in order to investigate the transferability of gains from `WIT` to preference alignment phase, we carry out an additional 525 training runs on top of these models using the Direct Preference Optimization (DPO) algorithm (Rafailov et al., 2023). We evaluate the models on popular benchmarks like MMLU (Hendrycks et al., 2021) and BBH (Suzgun et al., 2023) to measure knowledge and reasoning capabilities, IFEval (Zhou et al., 2023b) to objectively evaluate instruction-following ability, and AlpacaEval (Li et al., 2023) and MT-Bench (Zheng et al., 2023) for judging conversational proficiency. The key insights from our study are as follows:

- The conventional instruction tuning loss *rarely* yields the best-performing model across different configurations.
- Assigning a low-to-moderate weight (0–0.5) to prompt tokens and a moderate-to-high weight (0.5–1) to response tokens consistently results in the best-performing models across various settings—with optimal configuration of prompt and response token weights achieving an average relative gain of $\sim 6.55\%$ over the conventional loss.
- The gains from using `WIT`-loss also transfer to the subsequent preference alignment training using the DPO algorithm, i.e., `WIT`-finetuned models are *better starting*

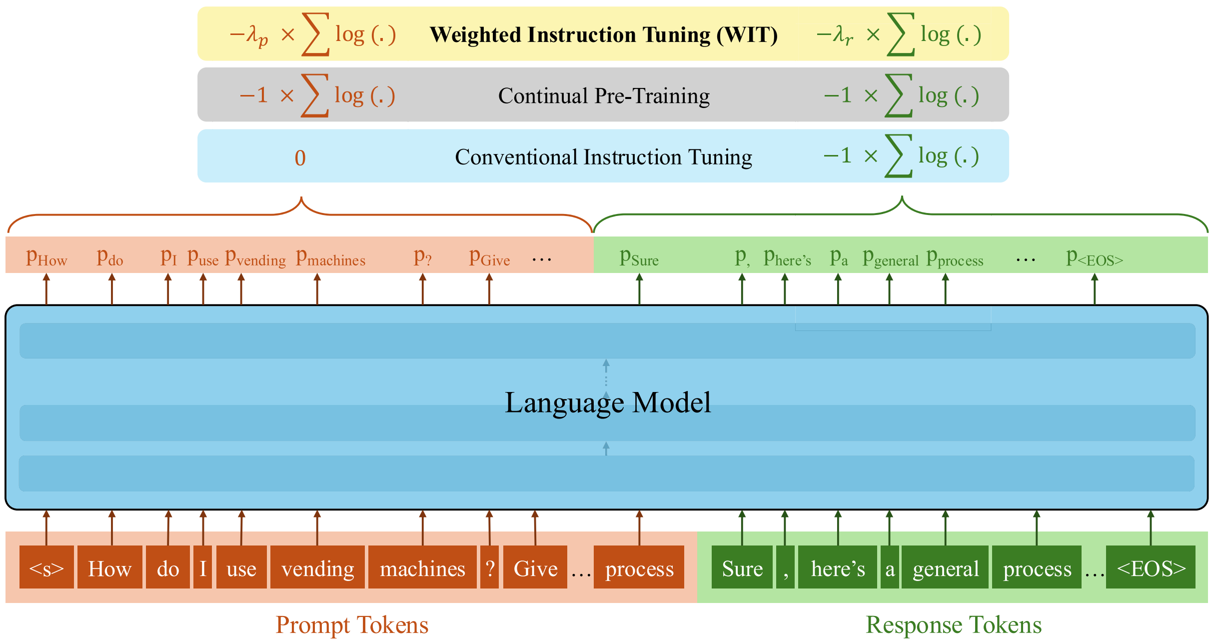

Figure 1: Conventional instruction tuning zeroes out the loss on prompt tokens, while continual pre-training treats prompt and response tokens equally. We find that both approaches are suboptimal and introduce **Weighted Instruction Tuning** (`WIT`), which assigns different weights λ_p and λ_r ($0 \leq \lambda_p, \lambda_r \leq 1$) to prompt and response token losses respectively, as a better alternative.

points compared to conventional instruction-tuned models, for DPO.

- A relatively moderate response-token weight not only enhances performance on standard benchmarks, but also improves model robustness to minor prompt variations.
- In many cases (although not always), fine-tuning solely on prompts also enhances instruction following compared to the base model, suggesting the possibility of instruction tuning the model even in the absence of response annotations.

We also present a post hoc analysis of how prompt characteristics (like length and diversity) correlate with optimal prompt-token weights, offering insights into factors influencing the choice of token weights. We also examine how `WIT` reshapes prompt and response probability distributions, highlighting its impact on model behavior. Our findings aim to aid the research in development of more robust and generalizable models.

2 Proposed Formulation

Let $\mathcal{D} = \{(\mathbf{P}_i, \mathbf{R}_i)\}_{i=1}^{N_\mathcal{T}}$ be an instruction tuning dataset consisting of $N_\mathcal{T}$ $(prompt, response)$ pairs, where each prompt $\mathbf{P}_i$ consists of an instruction (implicit or explicit) and an optional input, while $\mathbf{R}_i$ represents the expected ground-truth response. If $|\mathbf{S}|$ denotes the number of tokens in sequence $\mathbf{S}$, then $\mathbf{P}_i$ and $\mathbf{R}_i$ can be expanded as:

$$\mathbf{P}_i = \{p_i^{(1)}, p_i^{(2)}, \ldots, p_i^{(|\mathbf{P}_i|)}\},$$
$$\mathbf{R}_i = \{r_i^{(1)}, r_i^{(2)}, \ldots, r_i^{(|\mathbf{R}_i|)}\}$$

The conventional instruction tuning, which is an auto-regressive objective that zeroes out the loss on prompt tokens, is given by:

$$\mathcal{L}_{IT} = \frac{-\sum_{i=1}^{N_\mathcal{T}} \sum_{j=1}^{|\mathbf{R}_i|} \log \mathbb{P}_\mathcal{M}\left(r_i^{(j)} \mid \mathbf{P}_i, r_i^{(1)}, \ldots, r_i^{(j-1)}\right)}{\sum_{i=1}^{N_\mathcal{T}} |\mathbf{R}_i|} \tag{1}$$

Here, $\mathbb{P}_\mathcal{M}(.)$ denotes the probability assigned by the language model $\mathcal{M}$.

As discussed in Section 1, ignoring learning signals corresponding to the prompts may lead the model to struggle with comprehending novel prompts, while assigning full weight on response tokens can hamper generalization ability by potentially overfitting on common response patterns in the instruction tuning data. Hence, we propose Weighted Instruction Tuning (`WIT`), which assigns differential weights to the prompt and response tokens, as an alternative to the conventional instruction tuning loss. It is given by:

$$\mathcal{L}_{\texttt{WIT}} = \frac{-1}{\sum_{i=1}^{N_{\mathcal{T}}} \left(\mathbb{I}(\lambda_p \neq 0) \cdot |\mathbf{P}_i| + \mathbb{I}(\lambda_r \neq 0) \cdot |\mathbf{R}_i| \right)} \times \sum_{i=1}^{N_{\mathcal{T}}} \left[\lambda_p \sum_{j=1}^{|\mathbf{P}_i|} \log\ \mathbb{P}_{\mathcal{M}} \left(p_i^{(j)} \mid p_i^{(1)}, \ldots, p_i^{(j-1)} \right) + \lambda_r \sum_{j=1}^{|\mathbf{R}_i|} \log\ \mathbb{P}_{\mathcal{M}} \left(r_i^{(j)} \mid \mathbf{P}_i, r_i^{(1)}, \ldots, r_i^{(j-1)} \right) \right] \quad (2)$$

where the weighting factors λ_p and λ_r, denote the *prompt* and *response token weights*, respectively, while $\mathbb{I}(.)$ is the indicator function. $\mathcal{L}_{\texttt{WIT}}$ computes the weighted sum of log-probabilities—scaling the log-probabilities of prompt tokens by λ_p and those of response tokens by λ_r—and then normalizes by the count of tokens with non-zero weight. The indicator function ($\mathbb{I}$) ensures that the weighted sum is divided exactly by those tokens whose weight is non-zero. Note that the conventional instruction tuning loss $\mathcal{L}_{IT}$ is a special case of $\mathcal{L}_{\texttt{WIT}}$ for $(\lambda_p, \lambda_r) = (0, 1)$.

3 Experimental Setup

3.1 Finetuning Data

Instruction Tuning

We considered the following three commonly used diverse instruction tuning datasets to study the role of prompt and response token weights:

(i) LIMA (Zhou et al., 2023b) is a carefully curated set of $1K$ high-quality ($prompt$, $response$) pairs from sources such as Stack Exchange, wikiHow, and Reddit, along with some manually authored examples.

(ii) Alpaca-Cleaned is a filtered version of the original Alpaca dataset (Taori et al., 2023) after removing problematic instances, with $52K$ ($prompt, response$) pairs generated by `text-davinci-003`.

(iii) Tülu-v2 (Ivison et al., 2023) is a data mixture with instances from diverse sources such as FLAN-v2 (Longpre et al., 2023), Open Assistant (Köpf et al., 2023), GPT4-Alpaca (Peng et al., 2023), and Open-Orca (Lian et al., 2023), containing $326K$ ($prompt, response$) pairs in total, from which we randomly select $150K$ samples to reduce overall experiment cost and runtime.

The above choice of three datasets together covers a small dataset (LIMA), a moderately sized dataset (Alpaca-Cleaned), and a large dataset (Tülu-v2). Furthermore, they also differ in other characteristics such as response length, prompt length and diversity, etc. (Section 5.1).

Preference Alignment Training

For preference alignment training, we use a binarized version of the **UltraFeedback** dataset (Cui et al., 2024), consisting of around $60K$ ($prompt$, $chosen\_response$, $rejected\_response$) tuples.

3.2 Finetuning Procedure

For our experiments, we consider five models spanning different model families and sizes: Llama-3.2-1B, Gemma-2-2B, Llama-3.2-3B, Mistral-7B, and Llama-3-8B. We finetune each model for 1 epoch on Tülu-v2, for 2 epochs on Alpaca-Cleaned, and for 5 epochs on LIMA. Following Touvron et al. (2023), Pang et al. (2024), and other contemporary works, we use a learning rate of 5×10^{-6} for Mistral-7B and a learning rate of 2×10^{-5} for all other models, with batch size 64, weight decay 0.1, and cosine learning rate decay with linear warmup over the first 1% of steps. For preference alignment phase, we apply DPO (Rafailov et al., 2023), similar to Ivison et al. (2023), with a learning rate of 5×10^{-7}, batch size 32, weight decay 0.0, and 0.1 warmup ratio, finetuning each model for 2 epochs. We ran all the experiments on 8 NVIDIA A100-SXM4-80GB GPUs, utilizing Flash Attention 2.0 (Dao, 2024) and for larger models like Mistral-7B and Llama-3-8B, we use the full-sharded data parallel functionality in PyTorch.[1] The code to reproduce all our results is open-sourced here.

3.3 Evaluation Protocol

We assess the performance of our instruction-tuned models across various dimensions by employing the following evaluation suites:

(i) MMLU (Massive Multitask Language Understanding) (Hendrycks et al., 2021) is

[1] `https://pytorch.org/docs/stable/notes/fsdp.html`.

a benchmark spanning 57 tasks across humanities, STEM, and social sciences, with approximately $14K$ multiple-choice ($prompt, response$) pairs. We evaluate models in a *zero-shot setting* using flexible exact match, following LM Evaluation Harness (Gao et al., 2024).

(ii) BBH (Big-Bench Hard) (Suzgun et al., 2023) is a challenging subset of the BIG-Bench benchmark, comprising 23 tasks with $6.5K$ examples requiring logical deduction and multi-step reasoning. We evaluate BBH in a *zero-shot setting without chain-of-thought (CoT) prompting*, using flexible exact match as the evaluation metric, similar to MMLU.

(iii) AlpacaEval (Li et al., 2023) is an LLM-based evaluation framework with 805 prompts designed to assess conversational ability. Using AlpacaEval-1.0, we report model win rates against `text-davinci-003`, judged by GPT-4o-mini. Unlike MMLU and BBH, which emphasize correctness, AlpacaEval provides a holistic measure by evaluating both response quality and relevance in instruction-tuned models.

(iv) IFEval (Zhou et al., 2023b) is an instruction-following evaluation benchmark that focuses on a set of ''verifiable instructions'' offering an automated yet objective evaluation of instruction-following capability, unlike LLM-as-a-judge.

(v) MT-Bench (Zheng et al., 2023) evaluates multi-turn conversational and instruction-following abilities using 80 high-quality multi-turn questions. We adopt the single-answer grading scheme, with 160 responses rated from 1 to 10 by an LLM judge.[2] For our experiments, we use Llama-3.3-70B as the judge.

4 Results

To study the role of prompt and response tokens in instruction tuning, we finetune five language models of different scales (Section 3.2) on the three instruction tuning datasets (Section 3.1) by varying the prompt and response weight configurations (λ_p, λ_r) in $\{0, 0.2, 0.4, 0.6, 0.8, 1.0\}$. We then evaluate the generalization capability of these instruction-tuned models across five diverse benchmarks: MMLU, BBH, IFEval, AlpacaEval and MT-Bench. Figure 2 depicts the average performance of models across all benchmarks; Figures 6, 7, and 8 in the Appendix contain the individual benchmark performances for Tülu-v2, Alpaca-Cleaned, and LIMA as training data, respectively.

As illustrated in Figure 2, conventional instruction tuning, i.e., $\lambda_p = 0$ and $\lambda_r = 1$, is never the optimal choice. This underscores the critical role of loss function design in instruction tuning. Similarly, $\lambda_p = 1$ and $\lambda_r = 1$, which corresponds to the same auto-regressive objective used in pre-training step, i.e., instruction tuning treated as continual pre-training as suggested by Shi et al. (2025), also performs suboptimally. In fact, it yields optimal average performance in exactly 1 out of 15 (*model*, *training_dataset*) combinations, reinforcing the need to reconsider loss weighting strategies to enhance performance and generalization.

Building on these observations, we further quantify the *relative* performance gains of `WIT` compared to the conventional instruction tuning. As summarized in Table 1, `WIT` yields consistent improvements in average benchmark performance, achieving an average relative gain of around 6.55%. These findings underscore the value of assigning different weights to prompt and response tokens. In some cases, the benefits are especially pronounced—for example, fine-tuning Mistral-7B on the AlpacaCleaned dataset with $\lambda_p = 0.6$ and $\lambda_r = 0.4$ achieves a relative performance gain of approximately 20.25%.

We now present our key *empirical* findings based on the trends observed across different configurations.

4.1 Key Observations

Low-to-Moderate Prompt-Token Weight Yields Best Performing Models. While the optimal prompt-token weight varies based on the specific setting, i.e., the particular combination of model, training dataset, and evaluation benchmark (as also demonstrated in Figures 6, 7, and 8 in the Appendix), we find that in approximately 81% of the cases, i.e., 61 out of the 75 ($model$, $training\_dataset$,

[2]We scale the rating by 10 while averaging with other benchmarks.

Figure 2: Heatmaps depicting average performance across five benchmarks (MMLU, BBH, AlpacaEval, IFEval, and MT-Bench) for different configurations of (λ_p, λ_r) and for different models finetuned on Tülu-v2, Alpaca-Cleaned, and LIMA. Best performing configuration is highlighted with a red circle. The color map is based on relative gain with respect to conventional instruction tuning. Rows correspond to prompt token weights (λ_p) and columns correspond to response token weights (λ_r). Conventional instruction tuning is marked with `IT` and base model performance is marked with `Base`.

Model	Training Data	Conventional Loss	WIT Loss (Optimal λ_p, λ_r)	Relative Gain
Llama-3.2-1B	Tülu-v2	28.60	31.61	+10.49%
	AlpacaCleaned	28.29	32.48	+14.81%
	LIMA	24.57	24.68	+0.45%
Gemma-2-2B	Tülu-v2	47.19	48.42	+2.61%
	AlpacaCleaned	48.85	50.87	+1.04%
	LIMA	38.15	38.97	+2.15%
Llama-3.2-3B	Tülu-v2	44.68	44.94	+0.58%
	AlpacaCleaned	42.08	42.34	+0.62%
	LIMA	40.44	41.07	+1.56%
Mistral-7B	Tülu-v2	63.69	64.69	+1.57%
	AlpacaCleaned	53.24	64.02	+20.25%
	LIMA	52.31	52.99	+1.3%
Llama-3-8B	Tülu-v2	54.11	62.70	+15.88%
	AlpacaCleaned	52.54	59.81	+13.84%
	LIMA	45.48	50.54	+11.13%
			Average Relative Gain =	+6.55%

Table 1: Relative percentage gain of `WIT` (for optimal prompt and response token weights) over conventional instruction tuning on downstream tasks.

evaluation_benchmark) combinations that we considered, the best performance is achieved with a low-to-moderate prompt-token weight in the range of 0 to 0.6. Furthermore, in 56% of the cases, i.e., 43 out of the 75 settings, the optimal prompt-token weight is non-zero, strongly suggesting that ignoring prompt tokens for instruction tuning is suboptimal.

Moderate-to-High Response-Token Weight Yields Optimal Models. Existing instruction-tuning approaches (Shi et al., 2025; Huerta-Enochian and Ko, 2024) assign maximal weight to response tokens (i.e., $\lambda_r = 1$). However, our experiments reveal that $\lambda_r = 1$ is the optimal configuration in only 24% of the cases, i.e., 18 out of the 75 (*model*, *training_dataset*, *evaluation_benchmark*) combinations. And in the remaining 76% of the cases, $\lambda_r < 1$ yields the best performance. Furthermore, in 73.33% of the cases, i.e., 55 out of 75 settings, a moderate-to-high response-token weight, in the range of 0.4 to 1, yielded the best performance. These findings further reinforce that conventional instruction tuning, i.e., $(\lambda_p, \lambda_r) = (0, 1)$, leads to suboptimal performance. We hypothesize that an

Evaluation Benchmark	Average Optimal λ_p	Average Optimal λ_r
MMLU	0.28	0.56
BBH	0.17	0.61
AlpacaEval	0.36	0.64
IFEval	0.48	0.43
MT-Bench	0.23	0.55

Table 2: Optimal prompt-token weight (λ_p) and response-token weight (λ_r) for various evaluation benchmarks, averaged across different ($model$, $training\_dataset$) combinations. The optimal response-token weight varies from moderate to high, with values ranging from 0.43 for IFEval to 0.64 for AlpacaEval, while the optimal prompt-token weight varies from low to moderate, from 0.17 for BBH to 0.48 for IFEval.

extreme response-token weight might encourage memorization of response patterns and hurt generalization, as also noted by Jain et al. (2024) and Shi et al. (2025).

Varying Effects of Response-Token Weight on Instruction Adherence and Conversational Fluency. The results on IFEval, AlpacaEval, and MT-Bench across different models and training datasets, as observed in Figures 6, 7, and 8, reveal a *trade-off* between instruction adherence and conversational fluency. For IFEval, which measures instruction-following ability, lower response weights are favoured—in 60% of cases, i.e., 9 out of 15 ($model$, $training\_dataset$) combinations, $\lambda_r \leq 0.4$ is optimal. In contrast, conversational fluency benchmarks (AlpacaEval and MT-Bench)–prefer relatively higher response weights—in 60% of settings, i.e., 18 out of 30 combinations, $\lambda_r \geq 0.6$ is optimal, and in 80% cases, $\lambda_r \geq 0.4$ yields best performance. Table 2 also reflects this trend: The average optimal λ_r is relatively lower for IFEval (0.43) compared to AlpacaEval (0.64) and MT-Bench (0.55). These findings underscore the importance of tailoring prompt and response weighting in `WIT` to align with the intended downstream behaviour of instruction-tuned models.

Prompt-Only Tuning Also Enhances Base Model Capabilities. As depicted in Figures 6, 7, and 8, training with $\lambda_r = 0$, i.e., computing loss only on prompt tokens, still leads to notable improvements over the base LM across all benchmarks, except AlpacaEval, when using the large and diverse Tülu-v2 dataset for finetuning. In contrast, for smaller datasets like Alpaca-Cleaned and LIMA, improvements appear primarily on IFEval. Thus, even without direct optimization on response tokens, prompt-only finetuning enhances instruction adherence, suggesting that training on unannotated prompts can also impart instruction-following. The observations also indicate that prompt-only tuning may require sufficiently large and diverse data to generalize effectively. Overall, the findings highlight the potential of leveraging large-scale unannotated datasets to boost instruction-following abilities without extensive labeled prompt-response pairs.

4.2 Transferability of Gains from `WIT` to Preference Alignment Phase

To assess whether the gains from `WIT` persist after the preference alignment training phase, we performed DPO on models instruction-tuned with various prompt and response token weights. Figure 3 depicts the average benchmark performance of models that underwent DPO on top of the instruction-tuned models from Figure 2. We find that DPO performed on top of conventional instruction-tuned models still yields suboptimal results when compared to DPO performed on top of weighted instruction tuned models. Table 3 shows the relative performance gain on downstream tasks for DPO on top of `WIT` (with optimal setting of prompt and response token weights) over DPO on conventional instruction tuning. We find that the optimal configuration of prompt and response token weights for DPO yields a relative gain of nearly 8%. Furthermore, while we note that the exact optimal (λ_p, λ_r) configuration for DPO might be different compared to optimal (λ_p, λ_r) of instruction-tuning, we find that DPO performed on top of optimal (λ_p, λ_r) configuration for instruction tuning still yields 2.44% relative gain in performance over DPO on top of conventional instruction tuned models. These findings highlight that models fine-tuned by `WIT` serve as better starting points for the preference alignment training phase, and the performance gains transfer even after DPO training.

4.3 Robustness to Prompt Variations

Beyond achieving high performance on various evaluation benchmarks, a key desirable property of an instruction-tuned LM is its *robustness* to

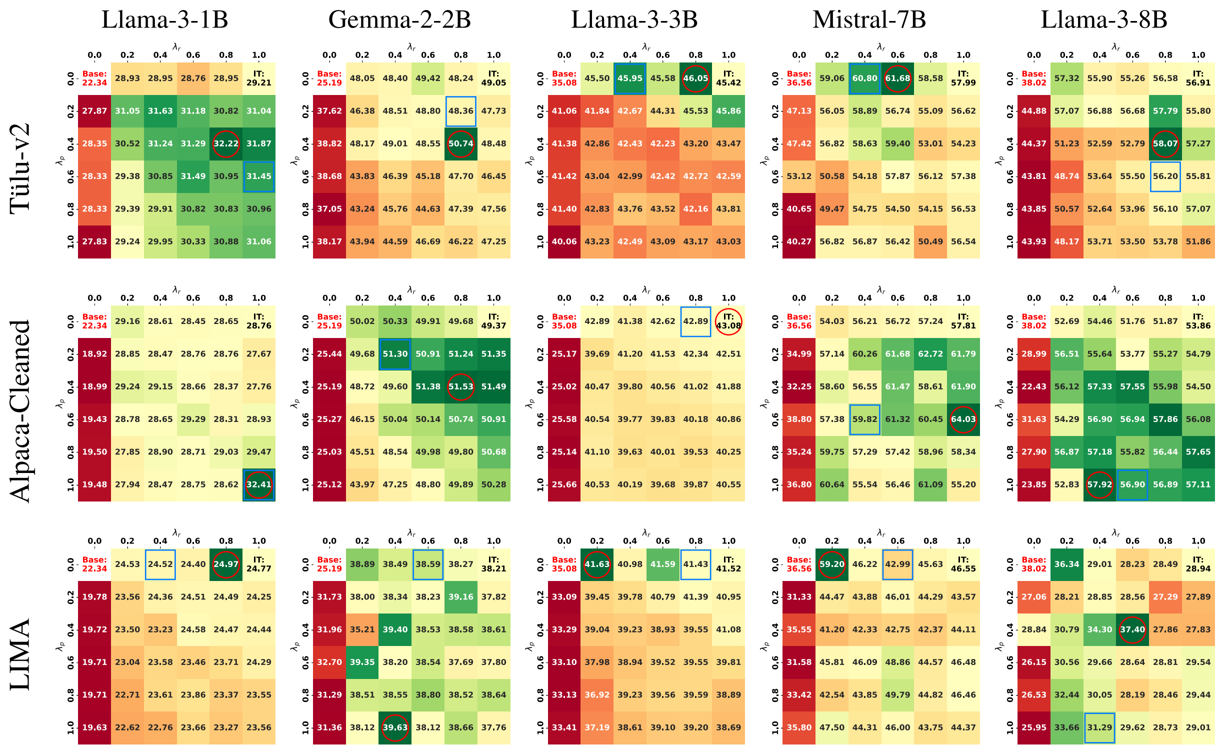

Figure 3: Heatmaps depicting average performance across five benchmarks (MMLU, BBH, AlpacaEval, IFEval, and MT-Bench) for different configurations of (λ_p, λ_r) and for different instruction-tuned models which underwent DPO on UltraFeedback dataset. Best performing configuration after DPO is highlighted with a red circle and best performing configuration from before DPO is highlighted with a blue square. The color map is based on relative gain with respect to conventional instruction tuning. Rows correspond to prompt token weights (λ_p) and columns correspond to response token weights (λ_r). Conventional instruction tuning is marked with `IT` and base model performance is marked with `Base`.

Model	Training Data	DPO on top of conventional instruction tuning	DPO on top of weighted instruction tuning (Optimal λ_p, λ_r)	Relative Gain
Llama-3.2-1B	Tülu-v2	29.21	32.22	+10.31%
	AlpacaCleaned	28.76	32.41	+12.69%
	LIMA	24.77	24.97	+0.81%
Gemma-2-2B	Tülu-v2	49.05	50.74	+3.45%
	AlpacaCleaned	49.37	51.53	+4.38%
	LIMA	38.21	39.63	+3.72%
Llama-3.2-3B	Tülu-v2	45.42	46.05	+1.39%
	AlpacaCleaned	43.08	43.08	0.00%
	LIMA	41.52	41.63	+0.27%
Mistral-7B	Tülu-v2	57.99	61.68	+6.36%
	AlpacaCleaned	57.81	64.03	+10.76%
	LIMA	46.55	59.2	+27.18%
Llama-3-8B	Tülu-v2	56.91	58.01	+2.03%
	AlpacaCleaned	53.86	57.92	+7.54%
	LIMA	28.94	37.4	+29.23%
			Average Relative Gain =	+8.01%

Table 3: Relative percentage gain of DPO on top of `WIT`, for optimal (λ_p, λ_r), over DPO on conventional instruction tuning.

prompt variations. Prior work has shown that these models are often sensitive to minor changes in prompts (Arora et al., 2023; Leidinger et al., 2023; Voronov et al., 2024; Mizrahi et al., 2024; Sclar et al., 2024). To quantify this, Chatterjee et al. (2024) introduced the Prompt Sensitivity Index (POSIX), which measures a model's sensitivity to *intent-preserving* prompt variations, such as spelling errors, re-wordings or prompt format changes. Figure 4 reports POSIX values for our models, using intent-preserving variants of $5K$ randomly sampled AlpacaCleaned prompts as provided by Chatterjee et al. (2024).

In line with our observations in the case of performance on evaluation benchmarks, it can be noted from Figure 4 that the models fine-tuned using the conventional instruction tuning loss almost never are the best in terms of prompt sensitivity (except for 1 out of 15 combinations), and are often more sensitive than even the corresponding base model (e.g., Llama-3-8B across all datasets). Also, lower response-token weights consistently lead to reduced sensitivity to input changes. Taken together with benchmark performance (Figures 2 and 4), these results suggest that a moderate response-token weight offers the best

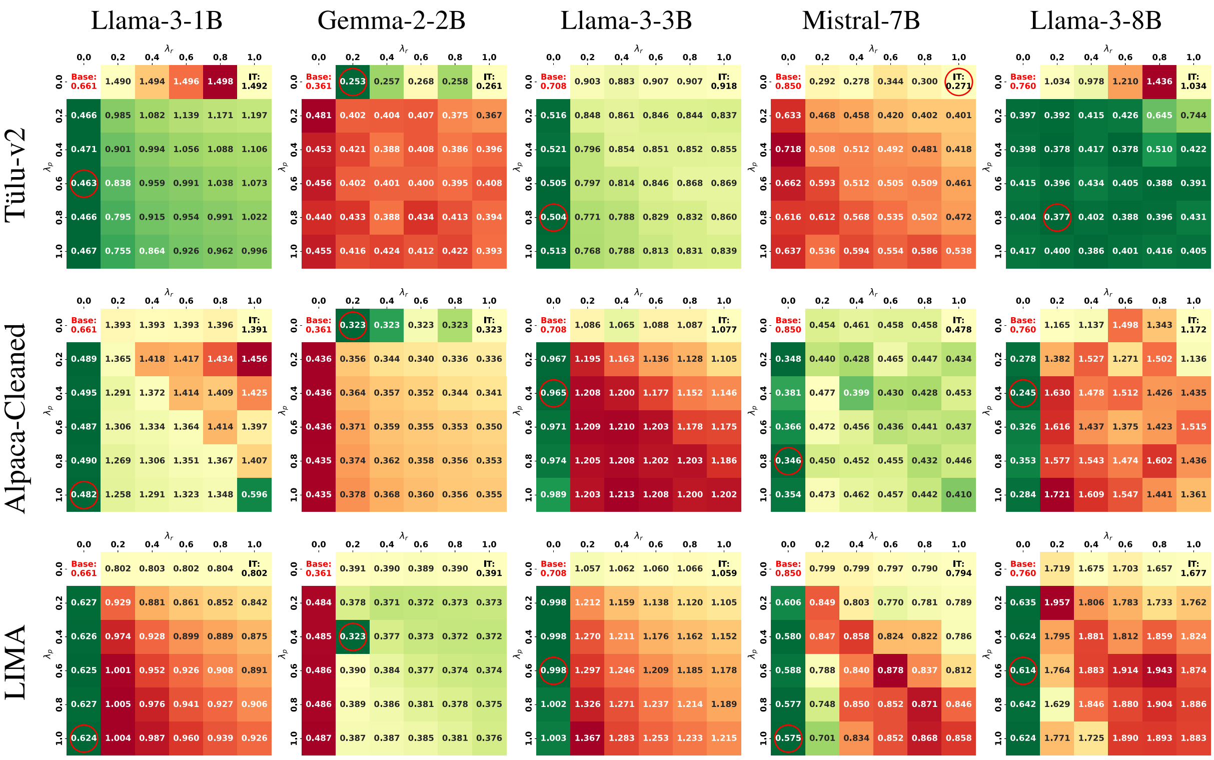

Figure 4: Heatmaps depicting Prompt Sensitivity Index (POSIX) for various (λ_p, λ_r) across models finetuned on Tülu-v2, Alpaca-Cleaned, and LIMA. Least prompt sensitivity configuration is highlighted with a red circle. The color map is based on relative reduction in prompt sensitivity with respect to conventional instruction tuning. Rows correspond to prompt token weights (λ_p) and columns correspond to response token weights (λ_r). Conventional instruction tuning is marked with `IT` and base model performance is marked with `Base`.

trade-off between robustness and performance, further highlighting the limitations of extreme response weighting.

5 Discussions

Building on above empirical results, we discuss broader patterns and preliminary insights that could inspire future studies on the interplay between task characteristics and token weighting.

5.1 Prompt-Token Weight: When and Why?

As shown in Figures 6, 7, and 8 in the Appendix, the optimal prompt-token weight varies with the combination of language model, training dataset, and the evaluation benchmark. To gain insights that may help us understand when and why a non-zero prompt-token weight is beneficial, we conduct a correlation analysis between various prompt characteristics (e.g., prompt length) and the optimal prompt-token weight, by varying one variable at a time.

Role of Finetuning Data in Selection of Prompt-Token Weight. Table 4 reports the optimal prompt-token weight (λ_p) and response-token weight (λ_r) for different finetuning datasets averaged across various (*model*, *evaluation_benchmark*) combinations. This helps us study how the optimal prompt-token weight varies with finetuning data. While the average optimal prompt-token weight for all finetuning datasets is in the low-to-moderate range, it is comparatively lower for Tülu-v2 compared to Alpaca-Cleaned or

Finetuning Data	**Average Optimal λ_p**	**Average Optimal λ_r**
Tülu-v2	0.20	0.58
Alpaca-Cleaned	0.36	0.49
LIMA	0.35	0.6

Table 4: Optimal prompt-token weight (λ_p) and response-token weight (λ_r) for various training datasets averaged across different (*model*, *evaluation_benchmark*) combinations. A relatively low prompt-token weight, along with a relatively moderate response-token weight, yields the best performance for all three training datasets.

Correlation	Train Prompt Characteristics				Eval Prompt Characteristics			Model Characteristics		
	Avg. Gen. Ratio	N-gram Div.	Avg. Parse Tree Depth	Avg. Prompt Len.	N-gram Div.	Avg. Parse Tree Depth	Avg. Prompt Len.	Model Size	Avg. log-prob of train prompt tokens	Avg. log-prob of eval prompt tokens
Spearman	0.50	−0.50	−0.50	−0.50	0.40	−0.50	−0.70	0.20	0.50	0.50
Kendall's τ	0.33	−0.33	−0.33	−0.33	0.40	−0.20	−0.60	0.20	0.20	0.20

Table 5: Correlation coefficients (Spearman and Kendall's τ) between the optimal prompt-token weight (λ_p) and various characteristics of the finetuning datasets, evaluation benchmarks, and language models.

LIMA. To better understand the possible dataset characteristics contributing to these trends, we study the prompt characteristics in the finetuning datasets, such as the average prompt length and the average generation ratio (i.e., the ratio of response length and prompt length) to capture the length characteristics, n-gram diversity (Meister et al., 2023) of prompts to capture lexical diversity, and the average depth of prompts' dependency parse tree to capture syntactic complexity.

Table 5 shows that the average generation ratio is positively correlated with the optimal prompt-token weight, while the average prompt length exhibits a negative correlation. This indicates that higher prompt-token weights tend to be preferred when the finetuning data contains longer completions relative to prompts, but not necessarily when the prompts themselves are longer. Furthermore, both lexical diversity, as measured by n-gram diversity, and syntactic complexity of the prompts are observed to negatively influence the optimal prompt-token weight.

Role of Evaluation Benchmark in Selection of Prompt-Token Weight. The optimal prompt- and response-token weights for different evaluation benchmarks averaged across various ($model$, $training\_dataset$) combinations are presented in Table 2. This helps us study how the optimal prompt-token weight varies with evaluation benchmarks. We observe that the optimal prompt-token weight varies from low to moderate, ranging from 0.17 for BBH to 0.48 for IFEval. To investigate the possible underlying benchmark characteristics contributing towards the observed optimal prompt-token weights, we obtain prompt characteristics of evaluation benchmarks (similar to those extracted in the case of finetuning data) whose correlation with the optimal prompt-token weight is presented in Table 5. As with finetuning data, a lower prompt-token weight yields better performance on benchmarks with longer prompts; syntactic complexity of the prompts also has a negative correlation with optimal prompt-token weight. However, unlike with training data, we observe that the lexical diversity of evaluation benchmarks is positively correlated with the optimal prompt-token weight.

Language Model	Average Optimal λ_p	Average Optimal λ_r
Llama-3-1B	0.33	0.63
Gemma-2-2B	0.42	0.57
Llama-3-3B	0.20	0.57
Mistral-7B	0.32	0.53
Llama-3-8B	0.35	0.50

Table 6: Optimal prompt-token weight (λ_p) and response-token weight (λ_r) for various language models averaged across different ($training\_dataset$, $evaluation\_benchmark$) combinations. A relatively lower prompt-token weight, coupled with a comparatively moderate response-token weight, yields the best performance for all five models.

Role of Language Model in Selection of Prompt-Token Weight. To study how the optimal prompt-token weight varies across language models, Table 6 reports the optimal prompt-token weight (λ_p) and response-token weight (λ_r) for different language models, averaged across various ($training\_dataset$, $evaluation\_benchmark$) combinations. We observe that the optimal prompt-token weight varies from low to moderate, ranging from 0.20 for Llama-3-3B to 0.42 for Gemma-2-2B. To better understand the potential factors contributing to these variations, we obtain model-dependent characteristics of train datasets and evaluation benchmarks, such as the average next-token

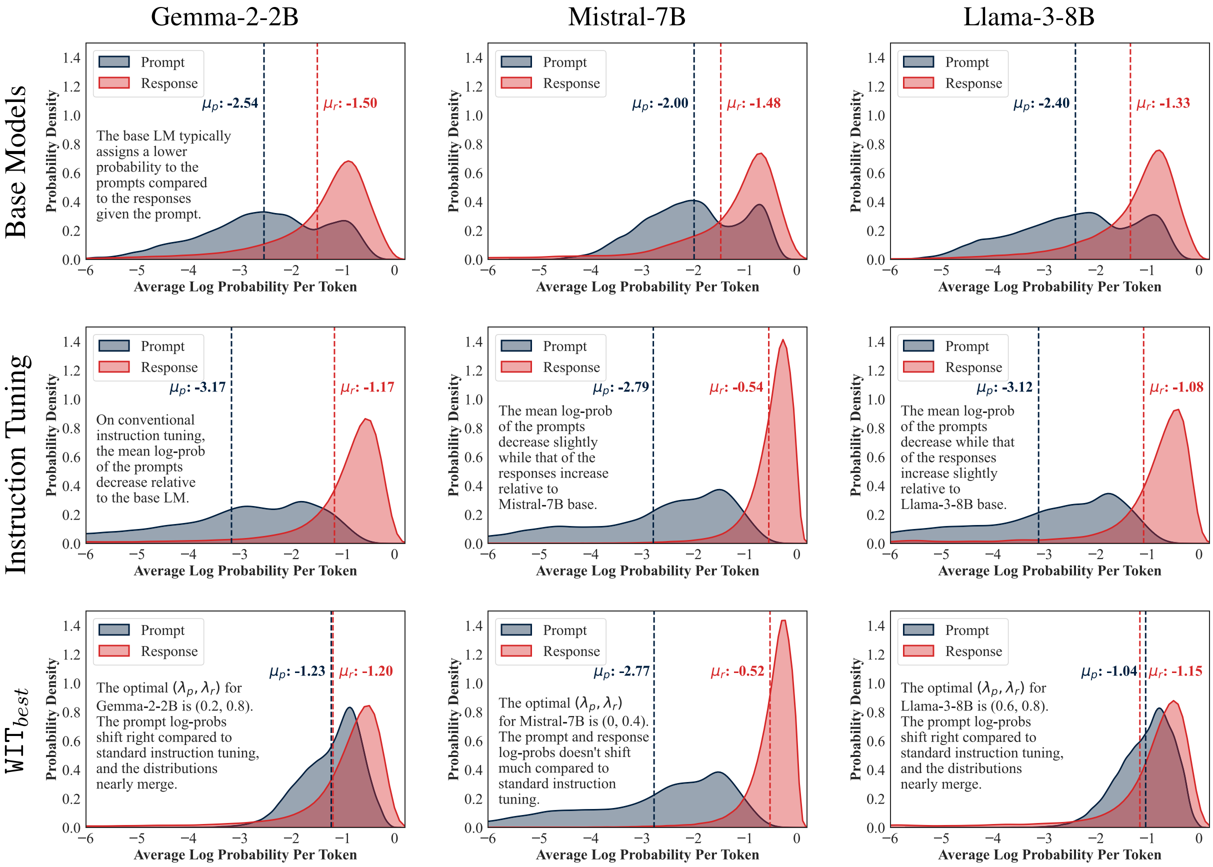

Figure 5: Distribution of average log probabilities for prompts and responses (given the corresponding prompts) from training samples of Tülu-v2, comparing base models with their instruction-tuned counterparts trained using the conventional response-only loss and the `WIT` loss (with optimal token weights).

log probabilities of prompts from finetuning datasets and evaluation benchmarks. The average next-token log probability is observed to be positively correlated with prompt-token weight (c.f. Table 5), suggesting that if a model has higher perplexity on prompts of a certain dataset, then a lower prompt-token weight can be more suitable. Furthermore, model size has a weak positive correlation with optimal λ_p.

In Summary. It is important to note that, as observed in our analysis, multiple factors influence the optimal prompt-token weight, often in different directions. Thus, considering the combined effect of these characteristics should be more effective than focusing on any single property when selecting prompt-token weights for `WIT`.

5.2 Impact of Instruction Tuning on Prompt and Response Probabilities

To assess how instruction tuning alters model behavior, we analyze the shifts in the log-probability distributions for prompt and response tokens. For this, we compute the length-normalized average log-probabilities for the training instances in Tülu-v2 across the base and two instruction-tuned variants of Gemma-2-2B, Mistral-7B, and Llama-3-8B (see Figure 5). The 1B and 3B variants of Llama-3 exhibit similar trends as the 8B model and are omitted for brevity. For each instance, we compute the *average log-probability per token* for (i) the prompt, and (ii) the response given the prompt, enabling fair comparison across different sequence lengths.

Behavior of the Base LMs. Across all model families, we observe that base LMs assign lower probabilities to prompts in isolation compared to responses conditioned on prompts, as evidenced by a leftward shift in prompt probability distributions relative to responses (first row in Figure 5). This aligns with expectations, as models pretrained on naturally occurring text develop a stronger prior over plausible completions than over standalone queries.

Effect of Conventional Instruction Tuning. When models are instruction-tuned using the conventional response-only loss, we observe that while the probability distribution of responses remains largely unchanged compared to the base LM (except in the case of Mistral), the probability assigned to prompt tokens shifts further left, indicating a decrease in their likelihood (middle row in Figure 5). This reveals an interesting insight on the effect of response-only loss: While the probability of the correct response given the prompt remains almost unchanged, the likelihood of the prompt itself decreases. Thus, the conventional instruction tuning loss, though it doesn't explicitly consider the prompt tokens, negatively affects the prediction of the input prompt tokens. We hypothesize that this degradation in prompt modeling might hurt the instruction comprehension ability of the models, potentially leading to a drop in performance on instruction-following benchmarks like IFEval, as observed in Figure 6.

Effect of `WIT`. When trained with the `WIT` loss using optimal prompt-response weights, the prompt probability distribution shifts rightward and aligns closely with that of the responses, especially for Llama and Gemma models (bottom row in Figure 5). For Mistral, however, this shift is negligible as the optimal `WIT` setting involves a null prompt weight. These observations indicate that `WIT` encourages the model to assign relatively higher likelihood to prompts, while the average log-likelihoods of responses remain similar or, in some cases, even decrease relative to conventional instruction tuning, likely improving instruction comprehension and mitigating overfitting on response patterns. This balanced treatment of prompts and responses contributes to better generalization across downstream tasks as well as enhanced robustness, as demonstrated in Figures 2 and 4.

6 Related Work

We review the prior work on instruction tuning across three main dimensions: instruction tuning algorithms, finetuning data, and evaluation.

Instruction Tuning Algorithms. Conventional instruction tuning uses an auto-regressive objective with loss zeroed on prompt tokens—a practice that, as recent work suggests, can encourage overfitting to response patterns (Jain et al., 2024; Shi et al., 2025). To mitigate this, Jain et al. (2024) proposed *NEFTune*, which adds noise to input embeddings to improve response quality, but offers no gains on OpenLLM benchmarks. Another approach, introduced by Shi et al. (2025) as *Instruction Modeling*, is akin to continual pre-training and applies loss to both prompt and response tokens; this benefits low-resource settings but underperforms on OpenLLM benchmarks. Assigning a small weight to prompt-token loss has also shown promise for datasets with short responses (Huerta-Enochian and Ko, 2024), though its effectiveness has primarily been validated on Alpaca variants. Other studies leverage large proprietary models for phased training or fine-tuning on GPT-4-generated completions (Pang et al., 2024; Xie et al., 2024). Recent findings even suggest that instruction-following can emerge from response-only training (Hewitt et al., 2024; An and Kim, 2024), though this requires further validation.

Instruction Tuning Data. The effectiveness of instruction tuning has been found to heavily depend upon task composition (Wang et al., 2023; Dong et al., 2024; Renduchintala et al., 2024), data quality (Zhou et al., 2023a; Ding et al., 2023), and data quantity (Ji et al., 2023; Yuan et al., 2023). Notable instruction tuning datasets include FLAN (Wei et al., 2022), Super-Natural Instructions (Wang et al., 2022b), Alpaca (Taori et al., 2023), Tülu (Ivison et al., 2023), Dolly (Conover et al., 2023), and LIMA (Zhou et al., 2023a) to name a few. For a more comprehensive review of data management for instruction tuning, we refer the reader to the survey by Wang et al. (2024).

Evaluation of Instruction Tuned Models. Evaluation of instruction tuned models can be broadly classified into two categories: close-ended and open-ended evaluations. Close-ended evaluations offer more objective evaluations—these include multiple-choice questions (MCQs) based benchmarks like MMLU (Hendrycks et al., 2021), BBH-Hard (Suzgun et al., 2023), as well as benchmarks like IFEval (Zhou et al., 2023b), which contain verifiable prompts that can be evaluated using a program, for instance. Open-ended evaluations, on the other hand, attempt to assess the quality of the output. The most common method is to use LLM-as-a-judge, where an LLM like GPT-4 is used to perform comparisons of responses to assess their quality. AlpacaEval (Li et al., 2023) is one such approach. For a comprehensive review of evaluation methods, we refer the reader to the survey by Zhang et al. (2023).

7 Conclusions

We proposed `WIT` as an alternative to conventional instruction tuning and analyzed the effects of differentially weighting prompt and response token losses. Our experiments on various models, datasets, and benchmarks show that both conventional instruction tuning and continual pre-training are generally suboptimal. While prior work (Wei et al., 2022; Ivison et al., 2023; Zhou et al., 2023a; Shi et al., 2025; Huerta-Enochian and Ko, 2024) consistently assigns maximal weight to response tokens, our results highlight the advantages of reducing response-token loss and including prompt-token loss. This overlooked balance offers new directions for robust instruction tuning. We also observe that the gains with `WIT` transfer even to the preference alignment phase. Moreover, we find that finetuning solely on prompts, though not always optimal, can still impart instruction following ability, highlighting potential for instruction tuning without response annotations.

Beyond performance, our findings suggest that instruction tuning loss functions influence model robustness and may shape biases. This highlights loss function design as a potential tool for aligning LMs with ethical and safety objectives, mitigating adversarial vulnerabilities, and improving reliability in real-world applications.

Limitations

One limitation of our approach is the use of fixed weights, i.e., one for all prompt tokens and another for all response tokens, throughout training. However, our preliminary analysis shows that optimal weights likely depend on factors like prompt and response likelihood from the lens of the model, which evolve during training. Moreover, no universal values of optimal prompt and response token weights exist across models or datasets. Future work exploring adaptive loss weighting strategies that dynamically adjust based on model predictions or training dynamics may be key to developing more robust and generalizable models.

Acknowledgments

We thank the reviewers and the action editor for their time and effort in reviewing our manuscript and for their thoughtful feedback, which considerably improved the clarity and overall presentation of our work. Anwoy Chatterjee gratefully acknowledges the support of the Google PhD Fellowship. Tanmoy Chakraborty acknowledges the Rajiv Khemani Young Faculty Chair Professorship in AI.

References

Seokhyun An and Hyounghun Kim. 2024. Response tuning: Aligning large language models without instruction. *arXiv preprint arXiv:2410.02465v1*.

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q. Tran, Dara Bahri, Jianmo Ni, Jai Gupta, Kai Hui, Sebastian Ruder, and Donald Metzler. 2022. ExT5: Towards extreme multi-task scaling for transfer learning. In *International Conference on Learning Representations*.

Simran Arora, Avanika Narayan, Mayee F. Chen, Laurel Orr, Neel Guha, Kush Bhatia, Ines Chami, and Christopher Re. 2023. Ask me anything: A simple strategy for prompting language models. In *The Eleventh International Conference on Learning Representations*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Anwoy Chatterjee, H. S. V. N. S. Kowndinya Renduchintala, Sumit Bhatia, and Tanmoy Chakraborty. 2024. POSIX: A prompt sensitivity index for large language models. In

Findings of the Association for Computational Linguistics: EMNLP 2024, pages 14550–14565, Miami, Florida, USA. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.findings-emnlp.852

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. *See* https://vicuna.lmsys.org *(accessed 14 April 2023)*.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. 2025. SFT memorizes, RL generalizes: A comparative study of foundation model post-training. In *Forty-second International Conference on Machine Learning*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

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

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2024. ULTRAFEEDBACK: Boosting language models with scaled AI feedback. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 9722–9744. PMLR.

Tri Dao. 2024. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations*.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.emnlp-main.183

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2024. How abilities in large language models are affected by supervised fine-tuning data composition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 177–198, Bangkok, Thailand. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.acl-long.12

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. 2024. A framework for few-shot language model evaluation.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

John Hewitt, Nelson F. Liu, Percy Liang, and Christopher D. Manning. 2024. Instruction following without instruction tuning. *arXiv preprint arXiv:2409.14254v1*.

Mathew Huerta-Enochian and Seung Yong Ko. 2024. Instruction fine-tuning: Does prompt loss matter? In *Proceedings of the 2024*

Conference on Empirical Methods in Natural Language Processing, pages 22771–22795, Miami, Florida, USA. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.emnlp-main.1267

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023. Camels in a changing climate: Enhancing LM adaptation with tulu 2. *arXiv preprint arXiv:2311.10702v2*.

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. 2024. NEFTune: Noisy embeddings improve instruction finetuning. In *The Twelfth International Conference on Learning Representations*.

Yunjie Ji, Yong Deng, Yan Gong, Yiping Peng, Qiang Niu, Lei Zhang, Baochang Ma, and Xiangang Li. 2023. Exploring the impact of instruction data scaling on large language models: An empirical study on real-world use cases. *arXiv preprint arXiv:2303.14742v1*.

Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi-Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, 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. *Advances in Neural Information Processing Systems*, 36:47669–47681.

Alina Leidinger, Robert van Rooij, and Ekaterina Shutova. 2023. The language of prompting: What linguistic properties make a prompt successful? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9210–9232, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.findings-emnlp.618

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Omer Levy, Luke Zettlemoyer, Jason E. Weston, and Mike Lewis. 2024. Self-alignment with instruction backtranslation. In *The Twelfth International Conference on Learning Representations*.

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

Wing Lian, Bleys Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, and ''Teknium''. 2023. OpenOrca: An open dataset of GPT augmented flan reasoning traces. https://huggingface.co/datasets/Open-Orca/OpenOrca

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 *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 22631–22648. PMLR.

Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. 2023. Locally typical sampling. *Transactions of the Association for Computational Linguistics*, 11:102–121. https://doi.org/10.1162/tacl_a_00536

Arindam Mitra, Luciano Del Corro, Shweti Mahajan, Andres Codas, Clarisse Simoes, Sahaj Agarwal, Xuxi Chen, Anastasia Razdaibiedina, Erik Jones, Kriti Aggarwal, Hamid Palangi, Guoqing Zheng, Corby Rosset, Hamed Khanpour, and Ahmed Awadallah. 2023. Orca 2: Teaching small language models how to reason. *arXiv preprint arXiv:2311.11045v2*.

Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. 2024. State of what art? A call for multi-prompt LLM evaluation. *Transactions of the Association for Computational Linguistics*, 12:933–949. https://doi.org/10.1162/tacl_a_00681

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of GPT-4. *arXiv preprint arXiv:2306.02707v1*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll 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 Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Wei Pang, Chuan Zhou, Xiao-Hua Zhou, and Xiaojie Wang. 2024. Phased instruction fine-tuning for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5735–5748, Bangkok, Thailand. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.findings-acl.341

Aldo Pareja, Nikhil Shivakumar Nayak, Hao Wang, Krishnateja Killamsetty, Shivchander Sudalairaj, Wenlong Zhao, Seungwook Han, Abhishek Bhandwaldar, Guangxuan Xu, Kai Xu, Ligong Han, Luke Inglis, and Akash Srivastava. 2025. Unveiling the secret recipe: A guide for supervised fine-tuning small LLMs. In *The Thirteenth International Conference on Learning Representations*.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with GPT-4. *arXiv preprint arXiv:2304.03277v1*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741.

H. S. V. N. S. Kowndinya Renduchintala, Sumit Bhatia, and Ganesh Ramakrishnan. 2024. SMART: Submodular data mixture strategy for instruction tuning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 12916–12934, Bangkok, Thailand. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.findings-acl.766

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

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *The Twelfth International Conference on Learning Representations*.

Zhengxiang Shi, Adam Yang, Bin Wu, Laurence Aitchison, Emine Yilmaz, and Aldo Lipani. 2025. Instruction tuning with loss over instructions. *Advances in Neural Information Processing Systems*, 37:69176–69205.

Xiaofei Sun, Xiaoya Li, Jiwei Li, Fei Wu, Shangwei Guo, Tianwei Zhang, and Guoyin Wang. 2023. Text classification via large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8990–9005, Singapore. Association for Computational Linguistics.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed 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*, pages 13003–13051, Toronto, Canada. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.findings-acl.824

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.

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, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288v2*.

Anton Voronov, Lena Wolf, and Max Ryabinin. 2024. Mind your format: Towards consistent evaluation of in-context learning improvements. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6287–6310, Bangkok, Thailand. Association for Computational Linguistics. `https://doi.org/10.18653/v1/2024.findings-acl.375`

Zhen Wan, Fei Cheng, Zhuoyuan Mao, Qianying Liu, Haiyue Song, Jiwei Li, and Sadao Kurohashi. 2023. GPT-RE: In-context learning for relation extraction using large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3534–3547, Singapore. Association for Computational Linguistics. `https://doi.org/10.18653/v1/2023.emnlp-main.214`

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022a. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171v4*.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023. How far can camels go? Exploring the state of instruction tuning on open resources. *Advances in Neural Information Processing Systems*, 36:74764–74786.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A., Sumanta Patro, Tanay Dixit, and Xudong Shen. 2022b. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. `https://doi.org/10.18653/v1/2022.emnlp-main.340`

Zige Wang, Wanjun Zhong, Yufei Wang, Qi Zhu, Fei Mi, Baojun Wang, Lifeng Shang, Xin Jiang, and Qun Liu. 2024. Data management for training large language models: A survey. *arXiv preprint arXiv:2312.01700v3*.

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

Juncheng Xie, Shensian Syu, and Hung-yi Lee. 2024. Non-instructional fine-tuning: Enabling instruction-following capabilities in pre-trained language models without instruction-following data. *arXiv preprint arXiv:2409.00096v1*.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2023. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825v2*.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2023. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792v8*.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini 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. *Advances in Neural Information Processing Systems*, 36:55006–55021.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023b. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911v1*.

Figure 6: Heatmaps depicting performance on MMLU (first row), BBH (second row), AlpacaEval (third row), IFEval (fourth row), and MT-Bench (fifth row) for different configurations of (λ_p, λ_r) and for different models finetuned on **Tülu-v2**. In each heatmap, the best performance is highlighted with a red circle. The color map is based on relative gain with respect to conventional instruction tuning. Each row of a heatmap corresponds to a prompt-token weight, and each column corresponds to a response-token weight. Conventional instruction tuning is marked with `IT`, and base model performance is marked with `Base`.

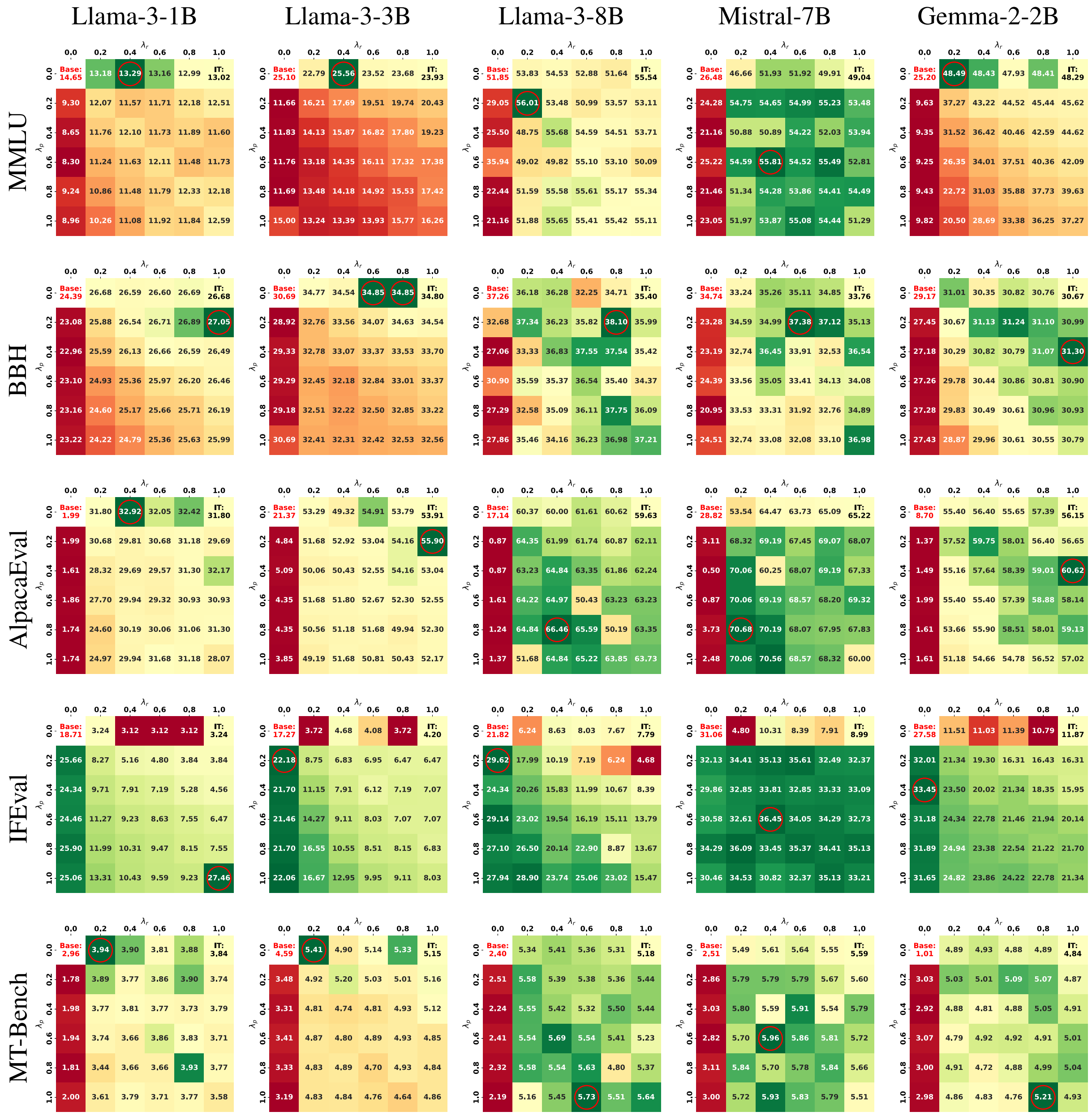

Figure 7: Heatmaps depicting performance on MMLU (first row), BBH (second row), AlpacaEval (third row), IFEval (fourth row), and MT-Bench (fifth row) for different configurations of (λ_p, λ_r) and for different models finetuned on **Alpaca-Cleaned**. In each heatmap, the best performance is highlighted with a red circle. The color map is based on relative gain with respect to conventional instruction tuning. Each row of a heatmap corresponds to a prompt-token weight, and each column corresponds to a response-token weight. Conventional instruction tuning is marked with `IT`, and base model performance is marked with `Base`.

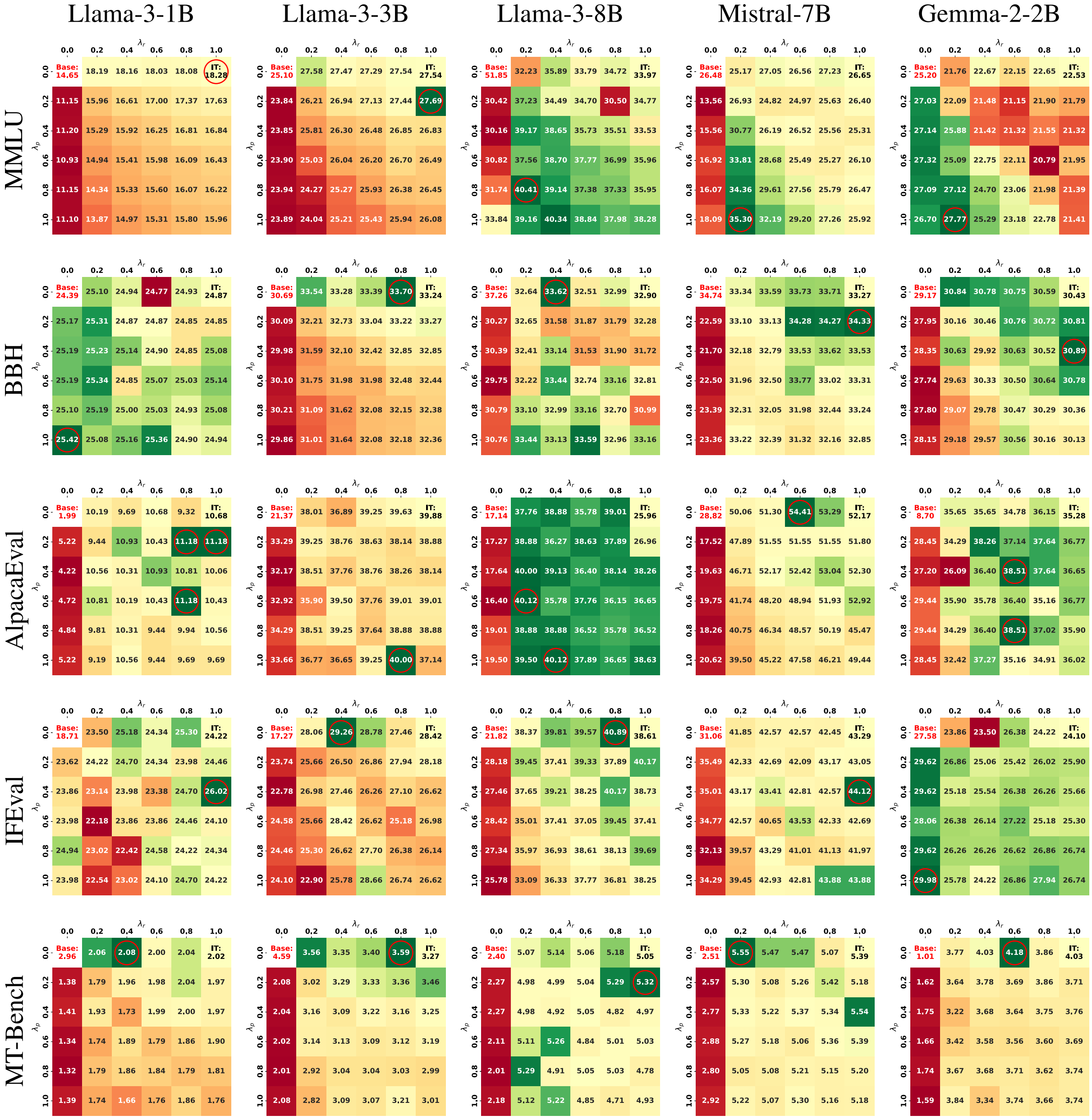

Figure 8: Heatmaps depicting performance on MMLU (first row), BBH (second row), AlpacaEval (third row), IFEval (fourth row), and MT-Bench (fifth row) for different configurations of (λ_p, λ_r) and for different models finetuned on **LIMA**. In each heatmap, the best performance is highlighted with a red circle. The color map is based on relative gain with respect to conventional instruction tuning. Each row of a heatmap corresponds to a prompt-token weight, and each column corresponds to a response-token weight. Conventional instruction tuning is marked with `IT`, and base model performance is marked with `Base`.