MULTI-LEVEL PREFERENCE LEARNING: ENHANCING RLHF FOR COMPLEX MULTI-INSTRUCTION TASKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement Learning from Human Feedback (RLHF) has become a predominant approach for aligning AI systems with human preferences, proving effective in instruction following but struggling with complex multi-instruction tasks. Conventional approaches rely on human annotation or stronger LLMs, incurring high costs or bias, while synthetic augmentation methods often compromise semantic quality. We identify a critical oversight in existing methods: they focus on comparing responses while overlooking latent signals in prompts, and only capture intra-sample preference disparities while neglecting inter-sample discrepancies. To address this, we propose a novel Multi-level Preference Learning (MPL) framework to enhance multi-instruction capabilities. Specifically, for a given response in original preference pairs, we construct varied prompts with preference relations under different conditions to learn intra-sample disparities. For a given original preference pair, we synthesize multi-instruction pairs to capture inter-sample discrepancies. We design two corresponding training objectives and integrate our framework seamlessly into both Reward Modeling and Direct Preference Optimization. Extensive experiments across model families from Qwen2 to the latest Qwen3 series validate the efficacy of our framework on both programmatically verifiable benchmarks and natural complex-instruction benchmarks.

1 INTRODUCTION

Large Language Models (LLMs) (Yang et al., 2024; Grattafiori et al., 2024; Achiam et al., 2023; Team et al., 2024; Liu et al., 2024; Yang et al., 2025) aligned with Reinforcement Learning from Human Feedback (RLHF) (Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022) have achieved remarkable efficacy across diverse tasks. However, although recent LLMs handle simple instructions well, their response quality on complex instructions with multiple constraints often falls short—constraints are omitted or wrongly followed—hindering deployment in realistic scenarios (He et al., 2024b; Jiang et al., 2024; Wen et al., 2024).

Existing efforts fall into three categories: (1) human annotation of multi-instruction data (Conover et al., 2023; Köpf et al., 2023), which is accurate but resource-intensive; (2) LLM-based augmentation that synthesizes multi-instruction queries or responses with stronger models (Xu et al., 2023; Sun et al., 2024; Huang et al., 2025a; He et al., 2024a; Zhang et al., 2024; Qi et al., 2025), which inherits the cost, biases, and generation errors of the teacher model; and (3) programmatic augmentation of existing preference data (Yuan et al., 2024), which is cheap and verifiable but may compromise semantic coherence. Across all three, we identify two overlooked signals: existing methods emphasize *response* comparison while ignoring the latent preference signals embedded in *prompts*, and they model only *intra-sample* preference distinctions while neglecting the *inter-sample* distinctions that exist among preference pairs.

To exploit these overlooked indicators, we propose a novel Multi-level Preference Learning (MPL) framework with two components: 1) **Intra-sample level preference learning.** For an original preference pair (x, y_w, y_l), we construct intra-sample preference data (x_w, x_l, y) where $y \in \{y_w, y_l\}$ and x_w, x_l are derived by augmenting x with instruction sets from a predefined pool, such that $x_w \succ x_l$ given the response y, together with a training objective that teaches the model these prompt-side preference disparities. 2) **Inter-sample level preference learning.** We define the preference gap as the degree to which y_w is preferred over y_l under a given prompt, and augment

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

prompts to systematically expand or reduce this gap, constructing inter-sample preference relations in conjunction with the original pairs, again with a dedicated training objective.

We conduct comprehensive experiments on LLaMA3 (Grattafiori et al., 2024), Qwen2 (Yang et al., 2024), and the latest Qwen3 series (Yang et al., 2025), covering both programmatically verifiable instruction following (IFEval) and natural complex instructions (ComplexBench). Models trained with our framework consistently outperform all compared methods in multi-instruction following, without semantic degradation.

The main contributions are: (1) we propose MPL, a multi-level preference learning framework that exploits prompt-side and inter-sample preference signals overlooked by prior work; (2) we show that MPL integrates seamlessly into both reward modeling and direct preference optimization with minimal modification; (3) extensive experiments, including on recent Qwen3-4B/8B/14B backbones and natural complex-instruction benchmarks, demonstrate consistent superiority over existing approaches.

2 Related Work

Preference Alignment and Optimization Objectives RLHF (Ouyang et al., 2022) aligns LLMs with human preferences through reward model (RM) training followed by policy optimization such as PPO (Schulman et al., 2017). A line of work simplifies or improves this pipeline: DPO (Rafailov et al., 2023) bypasses explicit reward modeling, IPO (Azar et al., 2024) regularizes against overfitting, KTO (Ethayarajh et al., 2024) needs only binary annotations, ORPO (Hong et al., 2024) folds preference optimization into SFT, SimPO (Meng et al., 2024) removes the reference model with an average-log-probability reward, and GRPO (Shao et al., 2024) uses group-relative rewards. Our framework is orthogonal to the choice of optimizer: it enriches preference *data* with multi-level signals, and can be plugged into RM training or any of the objectives above.

Multi-Instruction Following A series of benchmarks measure constraint following: IFEval (Zhou et al., 2023) with programmatically verifiable atomic instructions, and natural complex-instruction benchmarks such as FollowBench (Jiang et al., 2024), InFoBench (Qin et al., 2024), ComplexBench (Wen et al., 2024), CFBench (Zhang et al., 2025), and CELLO (He et al., 2024b). Proposed solutions fall into three categories: (1) human annotation (Conover et al., 2023; Köpf et al., 2023), which is resource-intensive; (2) LLM-based augmentation (Xu et al., 2023; Sun et al., 2024; Huang et al., 2025a; He et al., 2024a; Zhang et al., 2024; Huang et al., 2025b; Qi et al., 2025), e.g., Conifer (Sun et al., 2024) distills GPT-4-written multi-level instructions and MuSC (Huang et al., 2025a) decomposes and recombines constraints with the target LLM itself, but these inherit generator errors; (3) programmatic augmentation such as LIFT (Yuan et al., 2024), which can invert semantic preferences. We instead use programmatically verifiable instructions for both construction and validation, guaranteeing data correctness, and add question-oriented preference relations that mitigate semantic loss.

Preference Modeling with Ranking Objectives Modeling pairwise preference probabilities as a logistic function of score differences dates back to Bradley–Terry (Bradley & Terry, 1952) and RankNet (Burges et al., 2005), and underlies modern reward modeling and DPO. Beyond pairs, PRO (Song et al., 2024) extends the contrast to full rankings and REBEL (Gao et al., 2024) regresses relative reward differences between policies. Our inter-sample objective shares the logistic ranking form but differs in *what is ranked*: rather than responses or policies, we rank *preference gaps of the same response pair under different constraint-augmented prompts*, an orthogonal prompt-side signal that can be combined with any of these methods.

Concurrent Constraint-Backtranslation Construction CRAB (Qi et al., 2025) back-translates constraints from existing responses to form SFT pairs. We differ in three aspects: (i) we construct *preference pairs* (x_w, x_l, y), enabling direct use in RM training and DPO; (ii) we additionally model the *inter-sample* preference gap between augmented prompts; and (iii) our parameterization yields both satisfied and violated variants per response, providing explicit positive and negative prompt-side signals rather than only satisfied pairs.

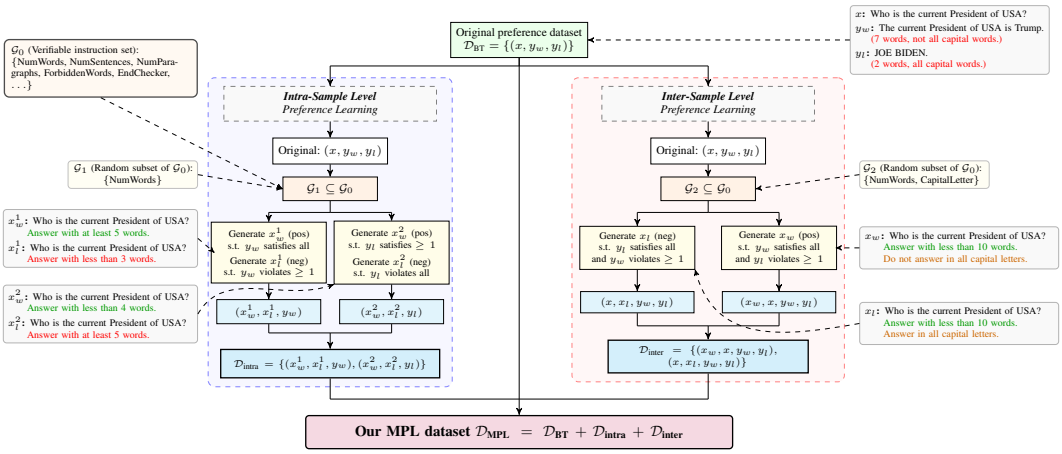

Figure 1: Overview of the data augmentation of MPL framework with examples. In the figure, "satisfies/violates all" within "Generate x s.t. y satisfies/violates all" indicates that response y satisfies/violates all instructions in x; if this were "satisfies/violates ≥ 1", it would indicate that response y satisfies/violates at least one instruction in x. In the examples, green text denotes instructions that all responses satisfy, red denotes instructions that all responses violate, and orange denotes instructions that at least one response fails to satisfy.

3 METHOD

3.1 PRELIMINARY

In RLHF, the Bradley-Terry (BT) model (Bradley & Terry, 1952) models human preferences over a preference tuple (x, y_w, y_l), where $y_w \succ y_l$, by positing a latent reward $r^*(x, y)$:

$$p^*(y_w \succ y_l|x) = \frac{\exp(r^*(x, y_w))}{\exp(r^*(x, y_w)) + \exp(r^*(x, y_l))}. \tag{1}$$

A parameterized reward model r_ϕ is then optimized via maximum likelihood estimation (MLE) to approximate r^*:

$$\mathcal{L}_{BT} = -\mathop{\mathbb{E}}_{(x,y_w,y_l)\sim\mathcal{D}_{BT}}[\log p_\phi(y_w \succ y_l|x)], \tag{2}$$

where p_ϕ replaces r^* in Eqn. 1 with r_ϕ and $\mathcal{D}_{BT} = \{(x, y_w, y_l)\}$ is the preference dataset[1]. The Response-Conditioned BT model (Rc-BT) (Cai et al., 2025) instead compares *prompts* conditioned on a response, capturing latent prompt-side signals:

$$\begin{aligned} p^*(x_w \succ x_l|y) &= \frac{\exp(r^*(x_w, y))}{\exp(r^*(x_w, y)) + \exp(r^*(x_l, y))}, \\ \mathcal{L}_{Rc\text{-}BT} &= -\mathop{\mathbb{E}}_{(x_w,x_l,y)\sim\mathcal{D}_{Rc\text{-}BT}}[\log p_\phi(x_w \succ x_l|y)]. \end{aligned} \tag{3}$$

3.2 INTRA-SAMPLE LEVEL PREFERENCE LEARNING

As discussed in Section 1, existing preference learning methods primarily focus on comparing responses while overlooking the latent preference signals embedded within prompts. To illustrate this insight, consider the following example (also shown in Fig. 1): given prompt x: "*Who is the current President of USA?*", with chosen response y_w: "*The current President of USA is Trump.*" (7 words, not all capital letters) and rejected response y_l: "*JOE BIDEN.*" (2 words, all capital letters). Conventional approaches exclusively exploit the preference relation between y_w and y_l. However, we observe that by augmenting the original prompt x with different instructions, we can construct an alternative form of preference data. For instance, if we append "*Answer with at least 5 words*" to form x_w, and "*Answer with less than 3 words*" to form x_l, then (x_w, x_l, y_w) naturally constitutes a preference triple where $x_w \succ x_l$ given response y_w, since y_w satisfies the constraint in x_w but violates the constraint in x_l. This observation motivates our intra-sample level preference learning framework, formalized in Algorithm 1.

[1]We omit the superscript $^{(i)}$ where no ambiguity exists.

Algorithm 1 Intra-Sample Level Preference Data Generation

Require: Instruction pool $\mathcal{I} = \{I_1, \ldots, I_n\}$, original preference dataset $\mathcal{D}_{BT} = \{(x, y_w, y_l)\}$
Ensure: Intra-sample level preference dataset $\mathcal{D}_{intra}$
$\mathcal{D}_{intra} \leftarrow \emptyset$
for all $(x, y_w, y_l) \in \mathcal{D}_{BT}$ **do**
  Sample instruction subset $\mathcal{I}' \subseteq \mathcal{I}$
  {Generate preference pair for y_w}
  $x_w^1 \leftarrow x$ + instructions from $\mathcal{I}'$ with params s.t. y_w satisfies all
  $x_l^1 \leftarrow x$ + instructions from $\mathcal{I}'$ with params s.t. y_w violates ≥ 1
  {Generate preference pair for y_l}
  $x_w^2 \leftarrow x$ + instructions from $\mathcal{I}'$ with params s.t. y_l satisfies ≥ 1
  $x_l^2 \leftarrow x$ + instructions from $\mathcal{I}'$ with params s.t. y_l violates all
  $\mathcal{D}_{intra} \leftarrow \mathcal{D}_{intra} \cup \{(x_w^1, x_l^1, y_w), (x_w^2, x_l^2, y_l)\}$
end for
return $\mathcal{D}_{intra}$

Preference Relation Formulation. Given an original preference pair (x, y_w, y_l), we construct intra-sample level preference data (x_w, x_l, y) where $y \in \{y_w, y_l\}$, and x_w, x_l are derived by augmenting x with different instruction sets. We define two valid preference relations: 1) response y satisfies **all** instructions in x_w, while y violates **at least one** instruction in x_l; 2) response y violates **all** instructions in x_l, while y satisfies **at least one** instruction in x_w. Under either condition, we establish a clear preference ordering $x_w \succ x_l$ given response y, as x_w represents a prompt whose constraints are better aligned with the response.

Instruction Pool and Data Construction. We construct an instruction pool $\mathcal{I} = \{I_1, I_2, \ldots, I_n\}$ consisting of n atomic instructions. These instructions are mutually independent and non-conflicting, and each is *programmatically verifiable*: we can check with certainty whether a response satisfies it, and we can configure its parameters (e.g., word-count thresholds) so that a given response satisfies or violates it as required (implementation details in Appendix C). For each original preference pair (x, y_w, y_l), we randomly sample a subset of instructions from $\mathcal{I}$ and generate two augmented prompts x_w and x_l by configuring instruction parameters such that one of the two relations above holds. We generate one intra-sample preference pair for each response, yielding two pairs (x_w^1, x_l^1, y_w) and (x_w^2, x_l^2, y_l) per original sample, so the resulting intra-sample dataset is twice the size of the original dataset. The detailed workflow is illustrated in the left part of Fig. 1.

Training Objective. Following the Response-Conditioned Bradley–Terry model (Cai et al., 2025), we model $p^*(x_w \succ x_l|y)$ with r^* as in Eqn. 3 and derive the maximum-likelihood objective over $\mathcal{D}_{intra} = \{(x_w^1, x_l^1, y_w)\} \cup \{(x_w^2, x_l^2, y_l)\}$:

$$\mathcal{L}_{intra} = -\mathop{\mathbb{E}}_{(x_w^1, x_l^1, y_w) \sim \mathcal{D}_{intra}} [\log p_\phi(x_w^1 \succ x_l^1|y_w)] - \mathop{\mathbb{E}}_{(x_w^2, x_l^2, y_l) \sim \mathcal{D}_{intra}} [\log p_\phi(x_w^2 \succ x_l^2|y_l)]. \tag{4}$$

By optimizing this objective, the model learns to distinguish prompts based on instruction–response alignment, effectively capturing preference signals at the intra-sample level and enhancing multi-instruction following capabilities.

3.3 Inter-Sample Level Preference Learning

Beyond modeling preference relations within individual preference data, we observe that valuable preference signals also exist *between* preference data pairs. Given a preference pair (x, y_w, y_l), we define the *preference gap* as the degree to which y_w is preferred over y_l under a given prompt, where a larger gap corresponds to a higher probability $p^*(y_w \succ y_l|x)$. By augmenting prompts with additional instructions, we can systematically manipulate this gap to create inter-sample preference signals, formalized in Algorithm 2.

To illustrate, we revisit the example from Section 3.2 (also shown in Fig. 1). Consider augmenting x to form x_w: "*Who is the current President of USA? Answer with less than 10 words. Do not answer in all capital letters.*" Here y_w satisfies both the original semantic requirement and all added instructions, while y_l fails both the semantic requirement and at least one new instruction. This widens the preference gap: $p^*(y_w \succ y_l|x_w) > p^*(y_w \succ y_l|x)$. We call this a *gap-expanding*

Algorithm 2 Inter-Sample Level Preference Data Generation

Require: Instruction pool $\mathcal{I} = \{I_1, \ldots, I_n\}$, original preference dataset $\mathcal{D}_{BT} = \{(x, y_w, y_l)\}$
Ensure: Inter-sample level preference dataset $\mathcal{D}_{inter}$
$\mathcal{D}_{inter} \leftarrow \emptyset$
for all $(x, y_w, y_l) \in \mathcal{D}_{BT}$ **do**
  Sample instruction subset $\mathcal{I}' \subseteq \mathcal{I}$
  {Gap-expanding: y_w satisfies all, y_l violates ≥ 1}
  $x_w \leftarrow x$ + instructions from $\mathcal{I}'$ with params configured accordingly
  {Gap-reducing: y_l satisfies all, y_w violates ≥ 1}
  $x_l \leftarrow x$ + instructions from $\mathcal{I}'$ with params configured accordingly
  $\mathcal{D}_{inter} \leftarrow \mathcal{D}_{inter} \cup \{(x_w, x, y_w, y_l), (x, x_l, y_w, y_l)\}$
end for
return $\mathcal{D}_{inter}$

relationship. Conversely, consider x_l: "*Who is the current President of USA? Answer with less than 10 words. Answer in all capital letters.*" Now y_l satisfies all added instructions while y_w violates one of them. Although y_l remains semantically inferior, its better alignment with the augmented instructions narrows the preference gap: $p^*(y_w \succ y_l|x) > p^*(y_w \succ y_l|x_l)$. We call this a *gap-reducing* relationship. Our inter-sample level preference learning framework leverages both relationships to capture richer preference signals across preference data pairs.

Preference Relation Formulation. For gap-expanding inter-sample data (x_w, x, y_w, y_l), where x_w is derived by augmenting x with additional instructions, we require: y_w satisfies **all** instructions in x_w, while y_l violates **at least one** instruction in x_w. Under this condition we establish $p^*(y_w \succ y_l|x_w) > p^*(y_w \succ y_l|x)$, indicating that the preference for y_w over y_l is *stronger* under the augmented prompt. Similarly, for gap-reducing inter-sample data (x, x_l, y_w, y_l), we require: y_l satisfies **all** instructions in x_l, while y_w violates **at least one** instruction in x_l, which yields $p^*(y_w \succ y_l|x) > p^*(y_w \succ y_l|x_l)$, indicating a *weaker* preference under the augmented prompt.

Instruction Pool and Data Construction. Following Section 3.2, we use the same instruction pool $\mathcal{I} = \{I_1, I_2, \ldots, I_n\}$. For each original preference pair (x, y_w, y_l), we randomly sample instructions from $\mathcal{I}$ and configure their parameters to generate both x_w (satisfying the gap-expanding condition) and x_l (satisfying the gap-reducing condition). This yields two inter-sample data relations per original sample, resulting in an inter-sample dataset twice the size of the original dataset. The detailed workflow is illustrated in the right part of Fig. 1.

Training Objective. We extend the Bradley–Terry framework to compare preference pairs: applying the logit transform $\gamma(p) = \log \frac{p}{1-p}$ to preference probabilities turns them into comparable scores and, since $\gamma(\sigma(a)) = a$, yields the compact comparisons

$$\begin{aligned} p^*((x_w, y_w, y_l) \succ (x, y_w, y_l)) &= \sigma\big(\Delta(x_w) - \Delta(x)\big), \\ p^*((x, y_w, y_l) \succ (x_l, y_w, y_l)) &= \sigma\big(\Delta(x) - \Delta(x_l)\big), \end{aligned} \tag{5}$$

where $\Delta(x) = r^*(x, y_w) - r^*(x, y_l)$ and σ is the logistic function (the derivation from the explicit BT form is in Appendix B). Applying maximum likelihood estimation over $\mathcal{D}_{\text{inter}} = \{(x_w, x, y_w, y_l)\} \cup \{(x, x_l, y_w, y_l)\}$ gives:

$$\begin{aligned} \mathcal{L}_{inter} = &- \mathop{\mathbb{E}}_{(x_w, x, y_w, y_l) \sim \mathcal{D}_{inter}} [\log p_\phi((x_w, y_w, y_l) \succ (x, y_w, y_l))] \\ &- \mathop{\mathbb{E}}_{(x, x_l, y_w, y_l) \sim \mathcal{D}_{inter}} [\log p_\phi((x, y_w, y_l) \succ (x_l, y_w, y_l))], \end{aligned} \tag{6}$$

where p_ϕ follows Eqn. 5 with r^* replaced by r_ϕ. By optimizing this objective, the model learns to discern preference gaps across different prompts, effectively capturing inter-sample level preference signals and further enhancing multi-instruction following capabilities.

3.4 Multi-Level Reward Modeling and Direct Preference Optimization

Overall MPL Objective. Combining the BT objective (Eqn. 2), the intra-sample objective (Eqn. 4), and the inter-sample objective (Eqn. 6):

$$\mathcal{L}_{MPL} = \mathcal{L}_{BT} + \mathcal{L}_{intra} + \mathcal{L}_{inter}. \tag{7}$$

Loss Combination and Mini-batch Sampling. Each mini-batch is built from *linked* data groups: for every original pair (x, y_w, y_l), the batch simultaneously contains the original pair (for $\mathcal{L}_{BT}$), its two intra-sample pairs (for $\mathcal{L}_{intra}$), and its two inter-sample relations (for $\mathcal{L}_{inter}$). All three objectives are binary logistic (Bradley–Terry) losses on pairwise preferences and hence share the same scale; each is averaged over its own pairs in the batch and the per-term means are summed with unit weights, so no extra normalization is needed.

Remark on Constraint Parameterization. Instruction parameters (e.g., word-count thresholds) are configured *conditioned on the response's own attributes*, by design: it guarantees every constructed pair a programmatically certain satisfy/violate label. Two factors mitigate the concern that the model exploits parameter–response correlations as a shortcut: (i) each instruction type appears in both satisfy and violate variants with thresholds sampled in both directions, balancing the marginal correlation; and (ii) gains on unseen instruction types (Section 4.2) and on natural instructions (Section 4.4) cannot be explained by parameter shortcuts alone.

Applying MPL to RM and DPO. Substituting the BT form $p_\phi(y_w \succ y_l|x) = \sigma(r_\phi(x, y_w) - r_\phi(x, y_l))$ into Eqns. 2, 4 and 6 yields the component RM losses as plain logistic regressions on reward differences (with $\Delta(x) = r_\phi(x, y_w) - r_\phi(x, y_l)$ for the inter-sample term); their sum $\mathcal{L}_{r_\phi}(\mathcal{D}_{MPL})$ is the MPL-RM objective. For DPO, expressing the reward as a function of the optimal policy (Rafailov et al., 2023) and substituting likewise yields the MPL-DPO objective. Full derivations and closed-form losses are given in Appendix B.

4 Experiments

4.1 Experimental Settings

Dataset. We extract the first turn of English conversations from OpenAssistant (Köpf et al., 2023), designating rank 0 as "chosen" (y_w) and rank 1 as "rejected" (y_l), and partition the pairs into a training set $\mathcal{D}_{BT}$ (90%) and an evaluation set $\mathcal{D}_{eval}$ (10%). We further build a multi-instruction evaluation set $\mathcal{D}^{if}_{eval}$ from $\mathcal{D}_{eval}$ following Section 3.2. The instruction pool $\mathcal{I}$ contains 10 of the 25 atomic instructions from IFEval (Zhou et al., 2023) (full list in Appendix A), with the remaining 15 used as OOD instructions; the MPL workflow then generates $\mathcal{D}_{MPL}$ from $\mathcal{D}_{BT}$ and $\mathcal{I}$.

Models and Training. We employ six models: Qwen2-1.5B-Instruct, Qwen2-7B-Instruct (Yang et al., 2024), Llama-3.1-8B-Instruct (Grattafiori et al., 2024), and three recent Qwen3-family backbones (Yang et al., 2025): Qwen3-4B-Instruct-2507, Qwen3-8B, and Qwen3-14B. RM/DPO models trained on $\mathcal{D}_{MPL}$ are denoted MPL-RM and MPL-DPO. For MPL-RM we use learning rate 9.65×10^{-6}, batch size 256; for MPL-DPO, 9.65×10^{-7} and 32; both use a cosine schedule with 10-step warm-up and train for 5 epochs. All experiments use DeepSpeed (Aminabadi et al., 2022), DeepSpeed-Chat (Yao et al., 2023) and Huggingface Transformers (Wolf et al., 2020) on A100 (80GB) and H20 (141GB) GPUs with an identical global batch size across runs.

Compared Methods. Baselines are Vanilla RM/DPO trained on $\mathcal{D}_{BT}$, M-LIFT (a multi-instruction extension of LIFT (Yuan et al., 2024)), MuSC (Huang et al., 2025a), and, on the Qwen3 backbones, SimPO (Meng et al., 2024) trained on the same $\mathcal{D}_{BT}$ (details of all compared methods in Appendix D). For a base dataset of size n, Vanilla uses n samples, M-LIFT $3n$, and ours $5n$ due to the additional preference relations; Section 4.3 verifies that our gains persist under equal data volume.

Evaluation. Two complementary tracks: RM discrimination and DPO generation. • *Reward Model: SQ* (accuracy on $\mathcal{D}_{eval}$) and *IF* (accuracy on $\mathcal{D}^{if}_{eval}$). • *DPO Instruction-Following:* IFEval (Zhou et al., 2023) ($\sim$500 prompts, 25 verifiable instruction types): *Prompt-level Strict Acc* (Prompt), *Instruction-level Strict Acc* (Inst), and *Inst (OOD)* over the 15 held-out instruction types. • *Natural Complex Instructions:* ComplexBench (Wen et al., 2024) overall score (DRFR, combining rule- and LLM-based evaluation). • *DPO Semantic Quality:* AlpacaEval (Dubois et al., 2024) *SQ Win Rate* (GPT-4o comparison against the corresponding Instruct model) and MT-Bench (Zheng et al., 2023) *SQ Score* (1–10, GPT-4o judged).

Table 1: Reward-model results (SQ and IF accuracy) across four backbones.

	Qwen2-1.5B		**Qwen2-7B**		**Llama-3.1-8B**		**Qwen3-8B**	
Methods	IF	SQ	IF	SQ	IF	SQ	IF	SQ
Vanilla RM	64.17	60.56	68.75	61.94	63.75	62.22	70.23	99.35
M-LIFT	77.22	60.00	81.67	60.56	82.92	61.67	88.39	99.52
MuSC	69.03	55.28	72.50	61.39	75.28	60.28	75.38	97.92
Ours	**91.81**	**61.94**	**94.31**	**63.89**	**94.31**	**63.33**	**98.19**	**99.56**

Table 2: DPO results on IFEval across three backbones (all Instruct variants). *Prompt*: prompt-level strict accuracy; *Inst*: instruction-level strict accuracy; *Inst (OOD)*: instruction-level strict accuracy on out-of-distribution instructions.

	Qwen2-1.5B			**Qwen2-7B**			**Llama-3.1-8B**		
Methods	Prompt	Inst	Inst (OOD)	Prompt	Inst	Inst (OOD)	Prompt	Inst	Inst (OOD)
Vanilla DPO	26.30	36.78	38.50	49.26	59.13	58.90	67.04	75.00	74.85
M-LIFT	35.00	46.15	46.48	56.30	64.54	59.05	70.40	78.52	75.46
MuSC	27.39	38.83	40.06	48.53	58.90	57.32	64.88	74.46	73.78
Ours	**42.65**	**54.77**	**47.86**	**57.54**	**66.11**	**64.18**	**71.51**	**79.24**	**76.07**

Table 3: DPO results on IFEval for the recent Qwen3 model family. Qwen3-8B Vanilla DPO and MPL rows are averaged over three data-order seeds; other rows are single runs. SimPO is trained on the same base data $\mathcal{D}_{BT}$.

	Qwen3-4B-2507			**Qwen3-8B**			**Qwen3-14B**		
Methods	Prompt	Inst	Inst (OOD)	Prompt	Inst	Inst (OOD)	Prompt	Inst	Inst (OOD)
Vanilla DPO	74.63	81.73	81.75	77.22	83.53	82.98	79.44	85.34	85.12
M-LIFT	77.04	83.41	82.82	77.78	83.89	83.13	79.81	85.70	85.12
SimPO	76.85	83.05	82.67	77.96	84.01	83.44	78.70	85.10	85.12
MuSC	76.48	83.17	82.82	78.33	84.50	84.05	80.19	85.58	85.28
Ours	**77.41**	**83.53**	**83.13**	**80.56**	**86.06**	**85.74**	**81.30**	**86.54**	**86.04**

4.2 Main Results

Table 1 reports RM results. M-LIFT, MuSC, and MPL all improve IF over Vanilla RM, with MPL achieving the largest gains: on Qwen2-7B-Instruct we surpass M-LIFT by 12.64% and MuSC by 21.81%, while M-LIFT and MuSC degrade SQ (-1.38% and -0.55%) and ours preserves it. The same trend holds on the recent Qwen3-8B backbone, where MPL-RM again achieves the best IF while preserving SQ.

Table 2 reports DPO results on IFEval. MPL achieves the largest improvements on all three models: on Qwen2-7B-Instruct it outperforms M-LIFT by 1.24% and MuSC by 9.01% on Prompt, by 1.57% and 7.21% on Inst, and by 5.13% and 6.86% on Inst (OOD). A case study comparing MPL-DPO with M-LIFT on a three-constraint prompt is provided in Appendix E.1 (Fig. 2).

Results on recent Qwen3 backbones. To examine whether MPL remains effective on current state-of-the-art open-weight families, we repeat the DPO experiments on Qwen3-4B-Instruct-2507, Qwen3-8B and Qwen3-14B, additionally including SimPO as a stronger preference-optimization baseline. Table 3 reports the results (averaged over three data-order seeds for Qwen3-8B). MPL attains the best prompt-level accuracy on all three backbones (tied with M-LIFT on Qwen3-4B) and the best instruction-level accuracy on Qwen3-4B/8B/14B, with gains over Vanilla DPO consistent across seeds, confirming that the benefits of multi-level preference learning transfer to recent, stronger base models.

Semantic quality is preserved. Existing augmentation methods like M-LIFT can degrade semantic quality; our intra- and inter-sample constructions are both designed to preserve the original semantic

Table 4: Semantic quality of DPO models: SQ Win Rate (AlpacaEval, % no-worse vs. the corresponding base model) and SQ Score (MT-Bench).

	Llama-3.1-8B		**Qwen2-7B**		**Qwen3-8B**	
Methods	SQ Win Rate	SQ Score	SQ Win Rate	SQ Score	SQ Win Rate	SQ Score
Vanilla DPO	49.25	7.73	54.86	**7.26**	71.7	8.26
M-LIFT	55.11	7.45	44.51	6.99	75.6	8.43
SimPO	53.9	7.61	51.8	7.13	76.4	8.34
MuSC	52.7	7.56	43.9	6.92	75.8	8.34
Ours	**66.33**	**7.75**	**58.73**	7.22	**78.1**	**8.46**

Table 5: Ablation of the intra- and inter-sample levels (Qwen2-7B-Instruct, IFEval).

Methods	Prompt	Inst	Inst(OOD)
Vanilla DPO	49.26	59.13	58.90
w/o Intra-sample Level	54.04	62.53	60.67
w/o Inter-sample Level	56.25	65.99	62.80
MPL (Ours)	**57.54**	**66.11**	**64.18**

preference ordering. Table 4 validates this: while M-LIFT drops 0.27 SQ points on Llama-3.1-8B-Instruct, MPL matches or exceeds Vanilla DPO on both SQ metrics. On the recent Qwen3-8B backbone the same ordering holds: MPL attains the highest SQ Win Rate (78.1%) and SQ Score (8.46) among all trained variants. MuSC's training data is generated by the target model under its own distribution, which is not comparable to the OpenAssistant-based models; its IF results appear in Tables 1 and 2.

4.3 Ablation Studies

$\mathcal{D}_{intra}$ and $\mathcal{D}_{inter}$ are complementary. Table 5 ablates the two levels on Qwen2-7B-Instruct. Each component alone already outperforms Vanilla DPO; removing the inter-sample level causes modest drops, while removing the intra-sample level degrades more, indicating that the two capture complementary within-pair and between-pair preference relations.

Data efficiency. For a base preference dataset of size n, Vanilla DPO, M-LIFT, and MPL use n, $3n$, and $5n$ samples, respectively. These volumes reflect method-specific augmentation strategies and objectives; baselines cannot directly use our full augmented dataset without adopting our objectives. To isolate the effect of data volume, we downsample MPL to $1/5$ and M-LIFT to $1/3$ of their original sizes, matching Vanilla DPO on Qwen2-7B-Instruct. Under equal data volume, MPL outperforms both baselines on all three IFEval metrics (Table 6), supporting the benefit of multi-level preference learning beyond data scaling alone.

Table 6: Ablation study with equal data volume (Qwen2-7B-Instruct, IFEval).

Methods	Prompt	Inst	Inst (OOD)
Vanilla DPO	49.26	59.13	58.90
M-LIFT	50.37	59.55	56.55
Ours	**53.49**	**63.96**	**62.20**

4.4 Additional Results

Results on natural complex instructions. To examine whether the gains transfer beyond programmatically verifiable atomic instructions, we evaluate on ComplexBench (Wen et al., 2024), whose naturally written instructions cover lexical, format, semantic, and utility constraints with complex compositions. As shown in Table 7, MPL-DPO again achieves the best overall score.

Table 7: Overall scores (DRFR, %) on ComplexBench with natural complex instructions.

Method	Qwen3-8B
Vanilla DPO	64.11
M-LIFT	63.58
Ours	**64.47**

Table 8: Experiment results comparing PPO models based on different reward models in IFEval.

Reward Models	Prompt	Inst	Inst(OOD)
Vanilla RM	47.24	57.40	56.25
MuSC	47.61	58.71	57.16
M-LIFT	47.43	58.71	56.71
Ours	**49.82**	**59.31**	**58.08**

Table 9: Experiment results on the MathIF benchmark using Qwen2-7B-Instruct, averaged across single, dual, and triple constraint settings.

	MATH500		**GSM8K**	
Method	HAcc	SAcc	HAcc	SAcc
Vanilla DPO	27.78	52.40	32.22	54.26
MuSC	28.89	51.85	33.33	55.37
M-LIFT	38.89	61.30	37.78	60.92
Ours	**48.89**	**69.63**	**46.67**	**67.78**

4.5 PPO Results

We use each trained RM as the reward signal for PPO optimization of a Qwen2-7B-Instruct policy. Table 8 shows that our RM yields the best instruction-following performance across all three IFEval metrics, validating its utility for downstream policy optimization in the full RLHF pipeline.

4.6 MathIF Results

We evaluate instruction-constrained mathematical reasoning on MathIF (Fu et al., 2025) using Qwen2-7B-Instruct, averaging HAcc and SAcc across single, dual, and triple constraint settings. Our method outperforms all baselines on both MATH500 and GSM8K (Table 9), extending the gains beyond general instruction following.

5 Conclusion and Future Work

We presented Multi-level Preference Learning (MPL), a preference learning framework that integrates intra-sample and inter-sample preference signals to improve complex multi-instruction following. MPL constructs these signals from existing preference data without relying on advanced LLMs or reversing the original semantic preference ordering. Experiments on reward modeling and direct preference optimization across multiple benchmarks demonstrate the effectiveness of our framework. Evaluations on recent Qwen3 backbones and the natural complex-instruction benchmark ComplexBench further support its applicability across different model architectures and instruction settings.

Future work will focus on reducing the training cost of MPL and extending its instruction function set to support instructions that cannot be verified programmatically. We also plan to explore the application of multi-level preference learning to other alignment tasks.

AI USE STATEMENT

Large language models only assisted with language revision, manuscript organization and visualization. Large language models were not involved in designing or conducting the experiments.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, Section 3 describes the proposed Multi-level Preference Learning (MPL) framework, including the data construction algorithms and training objectives. Appendix B provides the mathematical formulations and derivations for applying MPL to reward modeling and direct preference optimization. Section 4.1 describes the dataset preparation, model backbones, training settings, and evaluation protocols. Appendix A lists the instruction types used for data augmentation and out-of-distribution evaluation, while Appendix C details the instruction pool implementation with example code. Appendix D describes the compared methods, and Appendix E presents additional experimental results.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Reza Yazdani Aminabadi, Samyam Rajbhandari, Ammar Ahmad Awan, Cheng Li, Du Li, Elton Zheng, Olatunji Ruwase, Shaden Smith, Minjia Zhang, Jeff Rasley, et al. Deepspeed-inference: enabling efficient inference of transformer models at unprecedented scale. In *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–15. IEEE, 2022.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pp. 4447–4455. PMLR, 2024.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Chris JC Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, pp. 89–96, 2005.

Jianfeng Cai, Jinhua Zhu, Ruopei Sun, Yue Wang, Li Li, Wengang Zhou, and Houqiang Li. Disentangling length bias in preference learning via response-conditioned modeling, 2025. URL https://arxiv.org/abs/2502.00814.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. Free dolly: Introducing the world's first truly open instructiontuned llm. 2023. URL https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Kto: Model alignment as prospect theoretic optimization, 2024. URL https://arxiv.org/abs/2402.01306.

Tingchen Fu, Jiawei Gu, Yafu Li, Xiaoye Qu, and Yu Cheng. Scaling reasoning, losing control: Evaluating instruction following in large reasoning models. *arXiv preprint arXiv:2505.14810*, 2025.

Zhaolin Gao, Jonathan D Chang, Wenhao Zhan, Owen Oertell, Gokul Swamy, Kianté Brantley, Thorsten Joachims, J Andrew Bagnell, Jason D Lee, and Wen Sun. Rebel: Reinforcement learning via regressing relative rewards. In *Advances in Neural Information Processing Systems*, volume 37, pp. 52354–52400, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Qianyu He, Jie Zeng, Qianxi He, Jiaqing Liang, and Yanghua Xiao. From complex to simple: Enhancing multi-constraint complex instruction following ability of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 10864–10882, 2024a.

Qianyu He, Jie Zeng, Wenhao Huang, Lina Chen, Jin Xiao, Qianxi He, Xunzhe Zhou, Jiaqing Liang, and Yanghua Xiao. Can large language models understand real-world complex instructions? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 18188–18196, 2024b.

Jiwoo Hong, Noah Lee, and James Thorne. Orpo: Monolithic preference optimization without reference model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 11170–11189, 2024.

Hui Huang, Jiaheng Liu, Yancheng He, Shilong Li, Bing Xu, Conghui Zhu, Muyun Yang, and Tiejun Zhao. Musc: Improving complex instruction following with multi-granularity self-contrastive training, 2025a. URL https://arxiv.org/abs/2502.11541.

Xiang Huang, Ting-En Lin, Feiteng Fang, Yuchuan Wu, Hangyu Li, Yuzhong Qu, Fei Huang, and Yongbin Li. Reverse preference optimization for complex instruction following. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 20700–20723, 2025b.

Yuxin Jiang, Yufei Wang, Xingshan Zeng, Wanjun Zhong, Liangyou Li, Fei Mi, Lifeng Shang, Xin Jiang, Qun Liu, and Wei Wang. Followbench: A multi-level fine-grained constraints following benchmark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4667–4688, 2024.

Andreas Köpf, Yannic Kilcher, Dimitri Von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, et al. Openassistant conversations-democratizing large language model alignment. *Advances in Neural Information Processing Systems*, 36:47669–47681, 2023.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. In *Advances in Neural Information Processing Systems*, volume 37, pp. 124198–124235, 2024.

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

Yunjia Qi, Hao Peng, Xiaozhi Wang, Bin Xu, Lei Hou, and Juanzi Li. Constraint back-translation improves complex instruction following of large language models. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM)*, 2025.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. Infobench: Evaluating instruction following ability in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13025–13048, 2024.

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

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. Preference ranking optimization for human alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 18990–18998, 2024.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021, 2020.

Haoran Sun, Lixin Liu, Junjie Li, Fengyu Wang, Baohua Dong, Ran Lin, and Ruohui Huang. Conifer: Improving complex constrained instruction-following ability of large language models, 2024. URL https://arxiv.org/abs/2404.02823.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

Bosi Wen, Pei Ke, Xiaotao Gu, Lindong Wu, Hao Huang, Jinfeng Zhou, Wenchuang Li, Binxin Hu, Wendy Gao, Jiaxin Xu, et al. Benchmarking complex instruction-following with multiple constraints composition. *Advances in Neural Information Processing Systems*, 37:137610–137645, 2024.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

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

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, et al. Qwen2 technical report, 2024. URL https://arxiv.org/abs/2407.10671.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhewei Yao, Reza Yazdani Aminabadi, Olatunji Ruwase, Samyam Rajbhandari, Xiaoxia Wu, Ammar Ahmad Awan, Jeff Rasley, Minjia Zhang, Conglong Li, Connor Holmes, et al. Deepspeed-chat: Easy, fast and affordable rlhf training of chatgpt-like models at all scales. *arXiv preprint arXiv:2308.01320*, 2023.

Weizhe Yuan, Ilia Kulikov, Ping Yu, Kyunghyun Cho, Sainbayar Sukhbaatar, Jason Weston, and Jing Xu. Following length constraints in instructions, 2024. URL https://arxiv.org/abs/2406.17744.

Tao Zhang, Chenglin Zhu, Yanjun Shen, Wenjing Luo, Yan Zhang, Hao Liang, Fan Yang, Mingan Lin, Yujing Qiao, Weipeng Chen, et al. Cfbench: A comprehensive constraints-following benchmark for llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 32926–32944, 2025.

Xinghua Zhang, Haiyang Yu, Cheng Fu, Fei Huang, and Yongbin Li. Iopo: Empowering llms with complex instruction following via input-output preference optimization, 2024. URL https://arxiv.org/abs/2411.06208.

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

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023. URL https://arxiv.org/abs/2311.07911.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A SEEN AND OUT-OF-DISTRIBUTION INSTRUCTION LISTS

Table 10 lists the 10 atomic instruction types used to construct the instruction pool $\mathcal{I}$ for data augmentation (seen instructions), and the remaining 15 IFEval instruction types used as out-of-distribution (OOD) instructions for generalization evaluation.

Table 10: The 10 seen atomic instruction types used for data augmentation and the 15 OOD instruction types used for evaluation.

Seen (10, pool $\mathcal{I}$)	OOD (15, evaluation only)
`length_constraints:number_sentences`	`keywords:existence`
`length_constraints:number_paragraphs`	`keywords:frequency`
`length_constraints:number_words`	`keywords:letter_frequency`
`multi-turn:constrained_start`	`language:response_language`
`keywords:forbidden_words`	`length_constraints:nth_paragraph_first_word`
`combination:repeat_prompt`	`detectable_content:number_placeholders`
`startend:end_checker`	`detectable_content:postscript`
`startend:quotation`	`detectable_format:number_bullet_lists`
`change_case:english_lowercase`	`detectable_format:constrained_response`
`change_case:capital_word_frequency`	`detectable_format:number_highlighted_sections`
	`detectable_format:multiple_sections`
	`detectable_format:json_format`
	`detectable_format:title`
	`change_case:english_capital`
	`punctuation:no_comma`

B DERIVATION OF APPLYING MPL FRAMEWORK TO RM AND DPO

B.1 DERIVATION OF APPLYING MPL FRAMEWORK TO RM

Derivation of inter-sample level RM objective This subsection derives the inter-sample level RM objective. For notational convenience, we unify the two formulations from Eqn. 6 in Section 3.3 into the following representation:

$$p^*((x_1, y_w, y_l) \succ (x_2, y_w, y_l)) = \frac{\exp(\gamma(p^*(y_w \succ y_l|x_1)))}{\exp(\gamma(p^*(y_w \succ y_l|x_1))) + \exp(\gamma(p^*(y_w \succ y_l|x_2)))}. \tag{8}$$

Simultaneously, based on the BT model (Eqn. 1), we can derive:

$$\begin{aligned} p^*(y_w \succ y_l|x) &= \frac{\exp\left(r^*(x, y_w)\right)}{\exp(r^*(x, y_w)) + \exp(r^*(x, y_l))} \\ &= \sigma(r^*(x, y_w) - r^*(x, y_l)). \end{aligned} \tag{9}$$

Substituting Eqn. 9 into Eqn. 8, we can derive:

$$\begin{aligned} &p^*((x_1, y_w, y_l) \succ (x_2, y_w, y_l)) \\ &= \frac{\exp(\gamma(p^*(y_w \succ y_l|x_1)))}{\exp(\gamma(p^*(y_w \succ y_l|x_1))) + \exp(\gamma(p^*(y_w \succ y_l|x_2)))}. \\ &= \frac{\exp(\gamma(\sigma(r^*(x_1, y_w) - r^*(x_1, y_l))))}{\exp(\gamma(\sigma(r^*(x_1, y_w) - r^*(x_1, y_l)))) + \exp(\gamma(\sigma(r^*(x_2, y_w) - r^*(x_2, y_l))))} \\ &= \frac{\exp(r^*(x_1, y_w) - r^*(x_1, y_l))}{\exp(r^*(x_1, y_w) - r^*(x_1, y_l)) + \exp(r^*(x_2, y_w) - r^*(x_2, y_l))} \\ &= \frac{1}{1 + \exp(-[(r^*(x_1, y_w) - r^*(x_1, y_l)) - (r^*(x_2, y_w) - r^*(x_2, y_l))])} \\ &= \sigma((r^*(x_1, y_w) - r^*(x_1, y_l)) - (r^*(x_2, y_w) - r^*(x_2, y_l))). \end{aligned} \tag{10}$$

Finally, we derive the loss function for inter-sample level RM for a parameterized policy r_ϕ using the inter-sample level preference dataset $\mathcal{D}_{inter} = \{(x_w, x, y_w, y_l)\} \cup \{(x, x_l, y_w, y_l)\}$ by applying maximum likelihood estimation:

$$\begin{aligned}\mathcal{L}_{r_\phi}(\mathcal{D}_{inter}) = &- \mathop{\mathbb{E}}_{(x_w,x,y_w,y_l)\sim\mathcal{D}_{inter}} [\log\sigma((r_\phi(x_w,y_w) - r_\phi(x_w,y_l)) - (r_\phi(x,y_w) - r_\phi(x,y_l)))] \\ &- \mathop{\mathbb{E}}_{(x,x_l,y_w,y_l)\sim\mathcal{D}_{inter}} [\log\sigma((r_\phi(x,y_w) - r_\phi(x,y_l)) - (r_\phi(x_l,y_w) - r_\phi(x_l,y_l)))].\end{aligned} \tag{11}$$

B.2 Derivation of Applying MPL Framework to DPO

Overall Objective The MPL objective of direct preference modeling is achieved through derivations of RM and RL objective. According to the derivation of Appendix A in DPO (Rafailov et al., 2023) and Appendix B in Rc-BT (Cai et al., 2025), by expressing the reward function as a function of the optimal policy and substituting it into Eqn. 2 and 4 respectively, we derive the expressions for original DPO and Intra-sample Level DPO as follows:

$$\begin{aligned}&\mathcal{L}^{BT}_{DPO}(\pi_\theta;\pi_{\text{ref}}) = - \mathop{\mathbb{E}}_{(x,y_w,y_l)\sim\mathcal{D}_{MPL}} \Big[\log\sigma\big(\beta\Delta(y_w,y_l|x)\big)\Big] \\ &\mathcal{L}^{intra}_{DPO}(\pi_\theta;\pi_{\text{ref}}) = \\ &- \mathop{\mathbb{E}}_{(x^1_w,x^1_l,y_w)\sim\mathcal{D}_{MPL}} \Big[\log\sigma\big(\beta\Delta(x^1_w,x^1_l|y_w)\big)\Big] \\ &- \mathop{\mathbb{E}}_{(x^2_w,x^2_l,y_l)\sim\mathcal{D}_{MPL}} \Big[\log\sigma\big(\beta\Delta(x^2_w,x^2_l|y_l)\big)\Big],\end{aligned} \tag{12}$$

where

$$\begin{aligned}\Delta(y_w,y_l|x) &= \log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \log\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)} \\ \Delta(x_w,x_l|y) &= \log\frac{\pi_\theta(y|x_w)}{\pi_{\text{ref}}(y|x_w)} - \log\frac{\pi_\theta(y|x_l)}{\pi_{\text{ref}}(y|x_l)}.\end{aligned} \tag{13}$$

Similarly, by incorporating the preference probabilities $p^*(y_w \succ y_l|x)$ and $p^*(y_w \succ y_l|x_w)$ (or $p^*(y_w \succ y_l|x_l)$) from Eqn. 6, and substituting the reward function expressed as a function of the optimal policy, we obtain the expression for Inter-sample Level DPO as follows (detailed derivation provided in the following paragraph):

$$\begin{aligned}\mathcal{L}^{inter}_{DPO}(\pi_\theta;\pi_{\text{ref}}) = &- \mathop{\mathbb{E}}_{(x_w,x,y_w,y_l)\sim\mathcal{D}_{MPL}} \left[\log\sigma\left(\beta\Big[\Delta(x_w) - \Delta(x)\Big]\right)\right] \\ &- \mathop{\mathbb{E}}_{(x,x_l,y_w,y_l)\sim\mathcal{D}_{MPL}} \left[\log\sigma\left(\beta\Big[\Delta(x) - \Delta(x_l)\Big]\right)\right],\end{aligned} \tag{14}$$

where

$$\Delta(x) = \log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \log\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}. \tag{15}$$

Ultimately, the loss function for our MPL framework applied to DPO is formulated as:

$$\mathcal{L}^{MPL}_{DPO}(\pi_\theta;\pi_{\text{ref}}) = \mathcal{L}^{BT}_{DPO}(\pi_\theta;\pi_{\text{ref}}) + \mathcal{L}^{intra}_{DPO}(\pi_\theta;\pi_{\text{ref}}) + \mathcal{L}^{inter}_{DPO}(\pi_\theta;\pi_{\text{ref}}). \tag{16}$$

Derivation of inter-sample level DPO objective According to Eqn. 5 in the DPO paper, we derive the expression of the reward function in terms of policy as follows:

$$r^*(x,y) = \beta\log\frac{\pi^*(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta\log Z(x). \tag{17}$$

For $(r^*(x_1, y_w) - r^*(x_1, y_l))$ and $(r^*(x_2, y_w) - r^*(x_2, y_l))$ in Eqn. 10, we can substitute Eqn. 17 and simplify:

$$\begin{aligned}
&r^*(x_1, y_w) - r^*(x_1, y_l) \\
&\quad = \beta \log \frac{\pi^*(y_w \mid x_1)}{\pi_{\text{ref}}(y_w \mid x_1)} + \beta \log Z(x_1) - \beta \log \frac{\pi^*(y_l \mid x_1)}{\pi_{\text{ref}}(y_l \mid x_1)} - \beta \log Z(x_1) \\
&\quad = \beta \log \frac{\pi^*(y_w \mid x_1)}{\pi_{\text{ref}}(y_w \mid x_1)} - \beta \log \frac{\pi^*(y_l \mid x_1)}{\pi_{\text{ref}}(y_l \mid x_1)} \\
&r^*(x_2, y_w) - r^*(x_2, y_l) \\
&\quad = \beta \log \frac{\pi^*(y_w \mid x_2)}{\pi_{\text{ref}}(y_w \mid x_2)} + \beta \log Z(x_2) - \beta \log \frac{\pi^*(y_l \mid x_2)}{\pi_{\text{ref}}(y_l \mid x_2)} - \beta \log Z(x_2) \\
&\quad = \beta \log \frac{\pi^*(y_w \mid x_2)}{\pi_{\text{ref}}(y_w \mid x_2)} - \beta \log \frac{\pi^*(y_l \mid x_2)}{\pi_{\text{ref}}(y_l \mid x_2)}
\end{aligned} \tag{18}$$

Substituting the simplified result from Eqn. 18 into Eqn. 10, we obtain:

$$\begin{aligned}
&p^*((x_1, y_w, y_l) \succ (x_2, y_w, y_l)) = \\
&\sigma\left(\left(\beta \log \frac{\pi^*(y_w \mid x_1)}{\pi_{\text{ref}}(y_w \mid x_1)} - \beta \log \frac{\pi^*(y_l \mid x_1)}{\pi_{\text{ref}}(y_l \mid x_1)}\right) - \left(\beta \log \frac{\pi^*(y_w \mid x_2)}{\pi_{\text{ref}}(y_w \mid x_2)} - \beta \log \frac{\pi^*(y_l \mid x_2)}{\pi_{\text{ref}}(y_l \mid x_2)}\right)\right)
\end{aligned} \tag{19}$$

Finally, we derive the loss function for inter-sample level DPO for a parameterized policy π_θ using the inter-sample level preference dataset $\mathcal{D}_{\textit{inter}} = \{(x_w, x, y_w, y_l)\} \cup \{(x, x_l, y_w, y_l)\}$ by applying maximum likelihood estimation, which yields $\mathcal{L}_{\textit{DPO}}^{\textit{inter}}$ as presented in Eqn. 14:

$$\begin{aligned}
&\mathcal{L}_{\textit{DPO}}^{\textit{inter}}(\pi_\theta; \pi_{\text{ref}}) = \\
&- \mathop{\mathbb{E}}_{(x_w, x, y_w, y_l) \sim \mathcal{D}_{\textit{inter}}} [\log \sigma((\beta \log \frac{\pi_\theta(y_w \mid x_w)}{\pi_{\text{ref}}(y_w \mid x_w)} - \beta \log \frac{\pi_\theta(y_l \mid x_w)}{\pi_{\text{ref}}(y_l \mid x_w)}) \\
&\qquad\qquad - (\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}))] \\
&- \mathop{\mathbb{E}}_{(x, x_l, y_w, y_l) \sim \mathcal{D}_{\textit{inter}}} [\log \sigma((\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}) \\
&\qquad\qquad - (\beta \log \frac{\pi_\theta(y_w \mid x_l)}{\pi_{\text{ref}}(y_w \mid x_l)} - \beta \log \frac{\pi_\theta(y_l \mid x_l)}{\pi_{\text{ref}}(y_l \mid x_l)}))].
\end{aligned} \tag{20}$$

C Instruction Pool Implementation

This section describes the implementation details of the instruction pool used in both intra-sample and inter-sample level data construction.

Atomic Instructions with Parameters. We select 10 atomic instructions from IFEval to construct our instruction pool $\mathcal{I}$, covering four categories: length constraints (number of words/sentences/paragraphs), keyword constraints (forbidden words, prompt repetition), start/end constraints (start phrase, end phrase, quotation wrapping), and case constraints (all-lowercase, capital word frequency). The complete instruction list, together with the out-of-distribution instructions used for evaluation, is provided in Appendix A. Each atomic instruction is defined as a template with configurable parameters. For example, the word count instruction is defined as:

```
"Answer with {relation} {num_words} words."
```

where `relation` $\in$ {`"less than"`, `"at least"`} specifies the comparison operator, and `num_words` $\in \mathbb{Z}^+$ specifies the word count threshold. An atomic instruction becomes a complete, verifiable instruction only when all its parameters are specified.

Verification and Generation Functions. For each atomic instruction, we implement two core functions: (1) a *verification function* (`check_following`) that takes a response as input and returns whether the response satisfies the instruction with its current parameters; (2) a *generation function* (`get_following_prompt`) that takes a response and a target satisfaction status as input, and returns a new prompt with instruction parameters configured such that the response satisfies or violates the instruction accordingly. Algorithm 3 provides an example implementation for the word count instruction. All verification and generation functions are manually implemented to ensure correctness, avoiding potential errors from LLM-based generation methods.

Algorithm 3 Implementation of the Word Count Instruction.

```
import nltk
_COMPARISON_RELATION = ("less␣than", "at␣least")
_RANGE_MIN, _RANGE_MAX = 1, 100
def count_words(text):
    """Counts␣the␣number␣of␣words."""
    tokenizer = nltk.tokenize.RegexpTokenizer(r"\w+")
    tokens = tokenizer.tokenize(text)
    return len(tokens)
class NumberOfWords(Instruction):
    """Checks␣the␣number␣of␣words."""
    def build_description(self, *, num_words=None,
                          relation=None):
        self._description_pattern = (
            "Answer␣with␣{relation}␣{num_words}␣words.")
        self._comparison_relation = relation
        self._num_words = num_words
        return self._description_pattern.format(
            relation=self._comparison_relation,
            num_words=self._num_words)
    def check_following(self, value):
        num_words = count_words(value)
        if self._comparison_relation == \
                _COMPARISON_RELATION[0]:
            return num_words < self._num_words
        elif self._comparison_relation == \
                _COMPARISON_RELATION[1]:
            return num_words >= self._num_words
    def get_following_prompt(self, response,
                             condition, prompt):
        num_words = count_words(response)
        relation = random.choice(
            _COMPARISON_RELATION)
        if relation == _COMPARISON_RELATION[0]:
            if condition == True:
                ret = num_words + random.randint(
                    _RANGE_MIN, _RANGE_MAX)
            else:
                ret = max(num_words - random.randint(
                    _RANGE_MIN, _RANGE_MAX), 1)
        elif relation == _COMPARISON_RELATION[1]:
            if condition == True:
                ret = max(num_words - random.randint(
                    _RANGE_MIN, _RANGE_MAX), 1)
            else:
                ret = num_words + random.randint(
                    _RANGE_MIN, _RANGE_MAX)
        return prompt + "␣" + \
            self.build_description(
                num_words=ret, relation=relation)
```

Scalability and Overhead. Our framework remains effective when the number of atomic instructions per augmented prompt is small: even a single appended instruction already yields valid multi-level preference relations, and increasing the pool size mainly improves the diversity of generated multi-instruction prompts. In our experiments, we use 10 atomic instructions, providing a good balance between diversity and implementation effort. The verification and generation functions involve only basic string operations (e.g., word counting), so the computational overhead is negligible compared to model training.

D Details of Compared Methods

M-LIFT. M-LIFT extends LIFT to multi-instruction settings by incorporating multiple atomic instructions, where LIFT augments preference datasets by prepending length constraints to prompts and treats the original positive response as negative when it exceeds the length limits in the new training sample. Specifically, given a preference dataset $\mathcal{D} = \{(x, y_w, y_l)\}$, for each preference data (x, y_w, y_l), we randomly select functions $G_1 = \{g_{i_1}, \ldots, g_{i_k}\}$ from function set $G_0 = \{g_1, g_2, ..., g_n\}$, provide conditions $\{(y_w, True/False), (y_l, True/False)\}$ for each g_{i_k}, and combine all returned instructions with x to generate a multi-instruction prompt x', forming (x', y_w, y_l). We generate two categories of data: 1) y_w satisfies all instructions and y_l fails to satisfy at least one, with the generated prompt denoted as x^1, yielding (x^1, y_w, y_l). 2) y_l satisfies all instructions and y_w fails to satisfy at least one, with the generated prompt denoted as x^2. Since y_l performs better in instruction following, we reverse the preference, yielding (x^2, y_l, y_w). The resulting dataset is $\mathcal{D}_{\textit{M-LIFT}} = \{(x, y_w, y_l)\} \cup \{(x^1, y_w, y_l)\} \cup \{(x^2, y_l, y_w)\}$.

MuSC. MuSC decomposes complex instructions into individual constraint sets and constructs negative variants by dropping a subset. Preference pairs are then formed by contrasting responses generated by an LLM under the full constraint set (positive) against those generated under the reduced constraint set (negative).

SimPO. SimPO (Meng et al., 2024) is a strong reference-free preference-optimization method that replaces the reference-model reward in DPO with the average log-probability of the response under the policy, and introduces a target reward margin between chosen and rejected responses. We train SimPO on the same base preference data $\mathcal{D}_{BT}$ with its recommended hyper-parameters.

E Additional Results

E.1 Case Study

Fig. 2 presents a comparison between MPL-DPO and M-LIFT on Qwen2-7B-Instruct. The prompt imposes three constraints: 1) "highlight 6 sections with **", 2) "mention the name Sarah", and 3) "mention the name Sarah only once". Our response successfully satisfies all constraints, while M-LIFT's response fails on constraints 2 and 3 (error locations marked in red), demonstrating that our framework's multi-level preference learning yields superior multi-instruction following capabilities.

E.2 14B-Scale Results

To investigate the scalability of our approach to larger architectures, we conduct experiments using the Qwen2.5-14B-Instruct and Qwen3-14B models on the OpenAssistant dataset. As shown in Table 11, our method consistently outperforms all baseline approaches at this scale across all three IFEval metrics. The consistent performance gains observed from the 7B to 14B scale suggest that the benefits of MPL are not limited to smaller models but remain effective as model capacity increases.

Prompt: Write a limerick about a woman named Sarah who lives in a town where it's always 90°F. Highlight at least 6 sections in your answer with markdown, example: *highlighted section*. Mention the name Sarah only once.

Ours Response:
In a town where temps soared high,
Where the sun was ever-nigh,
Sarah danced, her dress awhirl,
Underneath the blazing chill.
Her laughter echoed, clear and bright,
As she basked in ninety degrees, all the day long.

Prompt: Write a limerick about a woman named Sarah who lives in a town where it's always 90°F. Highlight at least 6 sections in your answer with markdown, example: *highlighted section*. Mention the name Sarah only once.

M-LIFT Response:
Sarah from the sunny town,
lives in a place with heat so profound.
In her little house, it's always ninety degrees,
Under the sun's fierce rays, she does conceive.
She's known by all as Sarah, in this hot climate she glides.

Figure 2: Case study on models based on Qwen2-7B-Instruct and trained through our framework and M-LIFT.

Table 11: Experiment results on IFEval using 14B-scale models, demonstrating the scalability of our method to larger model architectures.

	Qwen2.5-14B-Instruct			**Qwen3-14B**		
Method	Prompt	Inst	Inst (OOD)	Prompt	Inst	Inst (OOD)
Vanilla DPO	75.00	81.86	80.18	79.44	85.34	85.12
MuSC	74.45	81.15	79.72	80.19	85.58	85.28
M-LIFT	74.45	80.67	78.96	79.81	85.70	85.12
Ours	**76.11**	**82.34**	**80.95**	**81.30**	**86.54**	**86.04**