Multi-objective Large Language Model Alignment with Hierarchical Experts

Zhuo Li[1*], **Guodong Du**[1*], **Weiyang Guo**[1], **Yigeng Zhou**[1], **Xiucheng Li**[1], **Wenya Wang**[2], **Fangming Liu**[3], **Yequan Wang**[4], **Deheng Ye**[5], **Min Zhang**[1], **Jing Li**[1]✉
[1]Harbin Institute of Technology, Shenzhen, China
[2]Nanyang Technological University, Singapore [3]Peng Cheng Laboratory, China
[4]Beijing Academy of Artificial Intelligence, China [5]Tencent, China
zuoer190191@mail.ustc.edu.cn jingli.phd@hotmail.com

Abstract

Aligning large language models (LLMs) to simultaneously satisfy multiple objectives remains a significant challenge, especially given the diverse and often conflicting nature of human preferences. Existing alignment methods struggle to balance trade-offs effectively, often requiring costly retraining or yielding suboptimal results across the Pareto frontier of preferences. In this paper, we introduce `HoE` (Hierarchical Mixture-of-Experts), a *lightweight*, *parameter-efficient*, and *plug-and-play* approach that eliminates the need for model training, while enabling LLMs to adapt across the entire Pareto frontier and accommodate diverse user preferences. In particular, `HoE` consists of three hierarchical components: LoRA Experts, Router Experts and Preference Routing, reaching optimal Pareto frontiers and achieving a trade-off between parameter size, training cost, and performance. We evaluate `HoE` across various tasks on 14 objectives and 200 different preferences among 6 benchmarks, demonstrating superior performance over 15 recent baselines. Code is available in the supplementary materials.

1 Introduction

Recent advancements in large language models (LLMs) have showcased impressive performance across a wide array of tasks [1, 72, 52, 65, 50, 66, 70, 36, 14, 37, 32, 38, 28]. However, aligning these models to simultaneously satisfy multiple human objectives - such as helpfulness, harmlessness, and faithfulness - remains a significant challenge [52, 54, 30]. *First*, existing methods [30, 54] are inherently limited in capturing the multi-dimensional and often conflicting nature of human values. This limitation becomes particularly evident when attempting to satisfy a wide range of user preferences or adapt to complex scenarios where trade-offs between objectives are unavoidable [44, 62, 26, 71, 21, 40]. *Second*, current approaches [31, 71] that attempt to address this issue by training multiple models suffer from significant inefficiency in both parameter scale and training cost, making them impractical at scale.

Multi-objective alignment (MOA) requires a flexible framework capable of dynamically adapting to diverse user preferences, essentially acting as a "jack-of-all-trades". However, this goal is intrinsically difficult [62, 21] due to the inherently steerable nature of multi-objective alignment (MOA). *First*, the parameters fine-tuned for individual objectives often conflict with each other [20], reflecting the "no free lunch" principle [69] observed in multi-task learning (MTL). Rescaling the preference vector to favor one objective may boost that dimension but typically degrades performance on others [15, 59, 60]. *Second*, competition also even exists across preferences along the Pareto

✉ Corresponding author. * Equal contribution.

Preprint. Under review.

frontier [56]. A single model can excel at only a limited set of preferences and cannot achieve optimal performance across the entire preference space [49, 71, 31]. For instance, a model trained uniformly across all weightings (black solid Pareto frontier in Fig. 1) often underperforms compared to the expert model fine-tuned preference-specifically (colored dashed Pareto frontier) at that individual preference(*e.g.,* [0.5, 0.5]) .

To overcome this limitation, we adopt a decomposition-based strategy for multi-objective alignment, breaking down the multi-objective alignment problem into a series of single-preference subproblems [67]. Each subproblem is handled by a specialized parameters, referred to as experts. These experts are each assigned to a distinct preference, focus solely on their corresponding preferences and optimize within their localized subproblem regions. This strategy avoids the pitfalls of a single monolithic model attempting to cover the entire Pareto frontier, thereby circumventing the steerability bottleneck. By constructing the full Pareto frontier from a collection of localized, preference-specific experts, our method enables precise and simplified optimization at each point along the pareto frontier.

One promising framework for such decomposition strategy lies in the LoRA-based Mixture-of-Experts (LoRAMoE) [8, 11, 64, 3] framework, originally developed for multi-task learning (MTL). LoRAMoE is a parameter-efficient approach that shares and freezes the parameters of base model and trains multiple lightweight LoRA adapters as task-specific experts. During inference, a router network selects and activates a subset of these experts, adapting the model's behavior dynamically. Despite its efficiency and modularity, the potential of LoRAMoE for multi-objective alignment remains largely untapped in current research [17] due to its unsteerable nature. It is designed for uniform-task balancing, acts as a fixed single-preference model (*e.g.,* [0.5, 0.5] in 2-obj) when applied directly to MOA, which limits its applicability to arbitrary user preferences. In this study, we aim to bridge this gap by fully leveraging the strengths of the Mixture-of-Experts framework and extending the LoRAMoE approach for multi-objective alignment.

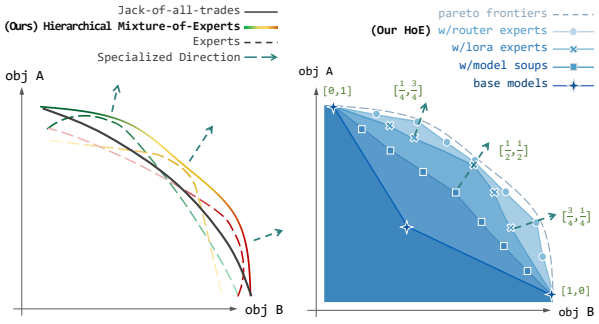

Figure 1: (Left) HoE decomposes the multi-objective alignment problem into a series of single-preference subproblems, each handled by a specialized expert. (Right) HoE employs hierarchical experts, integrating LoRA and router experts to approach the optimal Pareto frontier.

In this work, we propose HoE, a novel hierarchical Mixture-of-Experts framework for multi-objective alignment. HoE is a ***lightweight***, ***parameter-efficient***, and ***plug-and-play*** solution that eliminates the need for training any models while achieves strong performance across the entire Pareto frontier. It combines the decomposition principle with the LoRA-based MoE design to enable scalable, efficient and fine-grained control over the entire Pareto frontier. Specificaly, HoE comprises three hierarchical components: LoRA experts, router experts, and preference routing. (1) "LoRA experts" are first extracted *without training* from off-the-shelf single-objective models using task-vector singular value decomposition (task-SVD), capturing distinct alignment objectives in compact adapter modules. (2) "multi-objective LoRA experts" are then synthesized, also *without training* by merging multiple existing single-objective LoRA experts, to enable on-demand generation of alignment capabilities across arbitrary preference configurations. (3) "Router experts" are trained with negligible parameters to dynamically select and combine the appropriate experts based on user-specified preferences, allowing efficient traversal of the Pareto frontier. Through this hierarchical design, HoE not only provides precise control over preference-specific behavior but also balances alignment performance, parameter cost, and training efficiency. It serves as a practical and effective solution for scalable multi-objective alignment in LLMs.

The main contributions of this study are as follows:

- We investigate a novel decomposition strategy that breaks down the multi-objective alignment problem into a series of single-preference subproblems, each handled by a set of specialized experts, enabling fine-grained control and full Pareto coverage.

Table 1: Comparison with other alignment methods. M is number of preference and N is number of objectives. Note that $M \gg N$. Our HoE approach is a pareto-steerable and lightweight method with highest scalability, least storage cost and least inference cost, which eliminates the need for retraining any new models or any structed prompts. Each characteristic is empirically conformed in Section 6.2.

Characteristic (→) / Method (↓)	Number of stored models	Inference cost	Number of trained models	Pareto steerable	Multi-task ability	Scalability	Free from prompting
MORLHF [IEEE 21] [31]	M	1	M	✓	✗	Retrain	✓
MODPO [ACL 24] [71]	M	1	M	✓	✗	Retrain	✓
RS [NeurIPS 23] [44]	N	1	0	✓	✓	✓	✓
RiC [ICML 24] [62]	>1	1	>1	✓	✗	Retrain	✗
DPA [ACL 24] [55]	1	1	1	✓	✗	Retrain	✗
MOD [NeurIPS 24] [49]	N	N	0	✓	✓	✓	✓
Args [ICLR 24] [25]	0	>N	0	✓	✗	✓	✓
Steering [EACL 24] [26]	0	1	0	✗	✗	✓	✓
MetaAligner [NeurIPS 24] [61]	1	2	1	✗	✗	✓	✗
LoraMoE [ACL 24] [8]	1	1	1	✗	✓	Retrain	✓
PCB-Merging [NeurIPS 24] [15]	1	1	0	✗	✓	✓	✓
PAD [ICLR 25] [5]	1	3	1	✗	✗	Extra-train	✗
GenARM [ICLR 25] [58]	0	>N	0	✓	✗	✓	✓
RARM [ICML 25] [33]	0	2	1	✓	✗	Retrain	✓
HoE **(ours)**	1	1	0	✓	✓	✓	✓

- We propose HoE, a *lightweight*, *parameter-efficient* and *plug-and-play* hierarchical Mixture-of-Experts framework that comprises three-level hierarchy, bypassing full model training.
- We evaluate HoE across diverse multi-, many-objective and multi-task settings, involving 14 objectives, 6 benchmarks, and 200 preference. HoE consistently outperforms 15 recent baselines with lower training cost and parameter overhead.

2 Related Work

LLM Multi-objective Alignment. MORLHF [31] and MODPO [71] employ linear scalarization to combine multiple reward signals into a single scalar metric, applying standard RLHF or DPO training a separate model for each preference. Multi-objective Decoding (MOD) [49], Alignment as Reward-Guided Search (Args) [25] and Personalized Alignment at Decoding-time (PAD) [5] derive a closed form solution of optimal preference model and perform linear fusion of logits prediction during decoding. Directional Preference Alignment (DPA) [55] and Reward-in-Context (RiC) [62] typically inject user preferences into the prompt, enabling in-context preference-conditioned alignment. Additionally, Steering [26, 45] adding "steering vectors" to all token positions after the user's prompt, enabling precise control over multi-objective preferences. MetaAligner [61] extends the Aligner [23] framework to MOA, refining weaker outputs to better match user preferences.

Knowledge Fusion for LLMs. Model merging [24, 39, 15, 69, 59, 9, 10, 29] is a widely used fusion technique that integrates multiple task-specific models into a unified model. Task Arithmetic (TA) [20, 44, 21] linearly combines task vectors, defined as the parameter differences between task-specific models and the original pre-trained model. Then RS [44] and PS [21] firstly extend this concept to MOA. LoraMoE [8], the closest work to ours, is a Mixture-of-Experts (MoE) approach that uses LoRA Adapters [18] as experts, integrating LLM knowledge by activating select experts via a router network. However, it requires costly training across all LoRA experts simultaneously and limits knowledge sharing among them, thus unsuitable for MOA.

In summary, we systematically compare and analyze existing LLM alignment methods in Tab. 1.

3 Methodology

In this section, we present the methodology behind HoE, a lightweight, parameter-efficient, and plug-and-play multi-objective alignment framework. As illustrated in Figure 2, our HoE approach consists of three *hierarchical* components: LoRA experts, router experts and a preference routing.

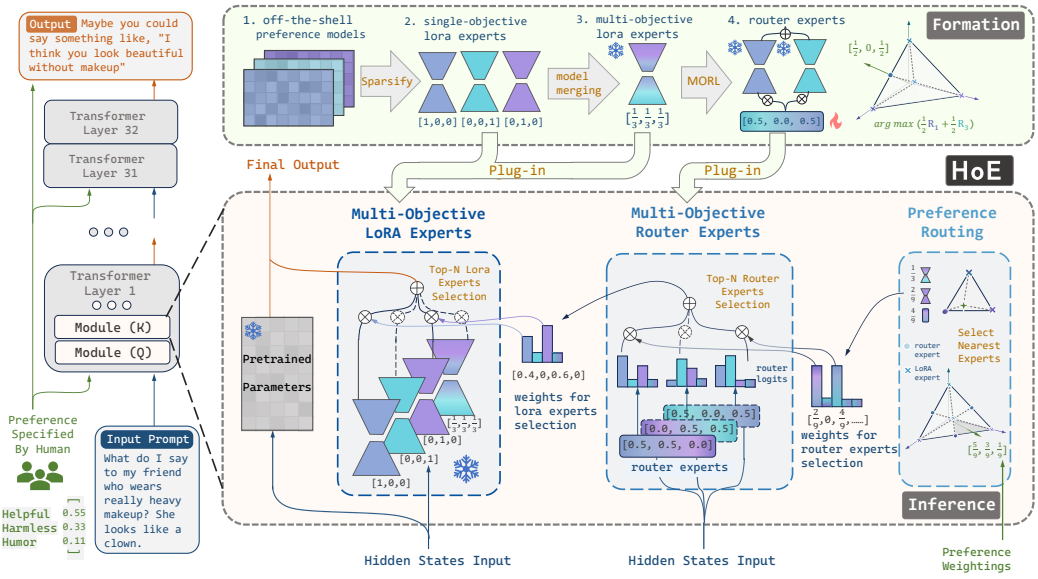

Figure 2: Illustration of our HoE approach. The left side illustrates the application scenario, where the model generates a response aligned with the prompt and given preferences. The bottom-right highlights its three ***hierarchical*** components - the LoRA experts, router experts, and a preference routing. The top-right depicts individual components, each serving as an expert for specific weightings, designed for seamless plug-and-play integration within the model.

3.1 Primary LoRA Experts

Single-Objective LoRA Experts. We begin with N objectives $\{R_1, ..., R_N\}$, a pre-trained model π_{pre}, and a collection of off-the-shell single-objective optimal policies $\{\pi_1^*, ..., \pi_N^*\}$, each fine-tuned on its respective objective.

$$\pi_i^* = \arg\max_{\pi_\theta} \mathbb{E}_{x\sim D, y\sim\pi_\theta(\cdot|x)}[R_i(x,y) - \beta\mathbb{KL}(\pi_\theta||\pi_{pre})] \tag{1}$$

Each model π_i^* is initialized from pre-trained weights θ_{pre}, and fine-tuned to obtain parameters θ_i.

Following the task vector paradigm in model merging [20], we define each "objective vector" as the difference between fine-tuned weights θ_i and the pre-trained weights θ_{pre}:

$$\tau_i = \theta_i - \theta_{pre} \tag{2}$$

These extracted objective vectors inherently capture the single-objective capabilities of each single-objective model.

Then, we apply Task-Vector Singular Value Decomposition (task-SVD) to obtain compact LoRA adapters for each objective: $A_i, B_i = \text{task-SVD}(\tau_i)$, where high-magnitude components of τ_i are extracted, followed by SVD and parameter rescaling to convert into new LoRA matrices A_i and B_i. These compact adapters, referred to as LoRA Experts, are highly specialized for their corresponding objectives, preserving optimal performance at the corresponding ends (*i.e.,* one-hot preference) of the Pareto frontier. Empirical results [57, 42, 13, 63, 47] demonstrated its effectiveness with negligible performance loss across diverse LLMs and alignment objectives.

Consequently, we replace all linear module in the Transformer architecture with MoE-style plugin modules, incorporating the LoRA experts. When receiving user preference $\lambda \in \mathbb{R}^N$, we simply linearly combining the N LoRA experts' outputs with weightings λ, yielding:

$$O_\lambda(x) = W_{pre}x + \sum_{i=1}^{N} \lambda_i(B_iA_ix) \tag{3}$$

where $x \sim \mathbb{R}^{d_{in}}, W_{pre} \sim \mathbb{R}^{d_{in}\times d_{out}}, A_i \sim \mathbb{R}^{d_{in}\times r}, B_i \sim \mathbb{R}^{r\times d_{out}}$ and rank $r \ll min(d_{in}, d_{out})$.

Multi-Objective LoRA Experts. For preferences involving multiple objectives simultaneously , the aforementioned linear combination of single-objective experts may fail to recover optimal performance, especially at intermediate points on the Pareto frontier (*e.g.,* $\lambda = [0.5, 0.5]$). To address this, we draw inspiration from model merging [60, 39, 69, 15, 24, 59] which amplify parameters beneficial to all tasks while suppressing conflicting or detrimental ones, enable nonlinear and fine-grained parameter adaptation, and significantly outperform linear approaches (*e.g.,* Task Arithmetic [20]).

To cover the entire Pareto frontier, we incorporate model merging into our framework to derive new expert parameters tailored to arbitrary preference vectors. Given a target preference λ, we specify the desired objective proportions and synthesize a merged expert with parameters:

$$\tau_\lambda = \text{Merge}(\{\tau_i\}_{i\in[N]}, \lambda) \tag{4}$$

where $\{\tau_i\}$ are the objective vectors derived from single-objective model. We then reuse the same task-SVD procedure. These resulting adapters serve as multi-objective LoRA experts, and are no longer aligned with a single objective, but instead specialized in specific combinations of objectives (*e.g.,* $[0.5, 0.5]$).

3.2 `Secondary` Router Experts

While increasing the number of LoRA experts improves performance, their number is limited by parameter overhead—since LoRA adapters are not negligible in size. To address these challenges, we propose a lightweight and fine-grained decomposition method called Router Experts, which introduces one-layer linear routers as secondary experts. The parameter size of such a component is negligible compared to the LoRA experts, while it plays a crucial role in enhancing the results, due to the static nature of router: `module-wise fine-grained routing`, and `input-adaptive selection`. This enables our router experts to dynamically select the most suitable LoRA experts based on the input, allowing for module-wise and more efficient utilization of LoRA parameters —a key distinction from LoRA experts.

Formation. Each router r_λ, represented by the weighting λ, consists of different router layers distributed across all Transformer modules. A router layer is a linear network $W \in \mathbb{R}^{d_{in}\times N}$ and $b \in \mathbb{R}^N$, taking the same input (hidden states x) as the LoRA adapters, producing scores $W^T x + b$. Notably, each router expert only votes for the N nearest LoRA experts in its simplex region (detailed in 3.4). As each router expert is optimized with respect to a specific weighting λ, it qualifies as an expert tailored to that particular preference.

Optimization. Unlike LoRAMoE [8], our method keeps LoRA expert parameters frozen, drastically reducing training resource requirements. Each router expert r_λ is optimized for its corresponding weighting λ with the frozen LoRA experts, making them `lightweight, plug-and-play` modules. Given L existing LoRA experts $\{\tau_{\lambda_i}\}_{\lambda_i\in\Lambda}$, each router activates only the N nearest neighbor to λ. The training objective is to maximize the linearly-combined reward $R_\lambda(\cdot) = \sum_i \lambda_i R_i(\cdot)$ under a mixture policy composed of these selected experts:

$$r_\lambda = \arg\max_r \mathbb{E}_{y\sim\pi_r(\cdot|x)}[R_\lambda(x,y)], s.t. \pi = HoE(\theta_{pre}, \tau_{\mathbb{NN}_N(\lambda,\Lambda)}) \tag{5}$$

To capture non-convex regions of the Pareto Front, we employ Tchebycheff (TCH) scalarization, optimizing for the worst-case objective:

$$\mathbb{J}(\theta|\lambda) = \max_\theta \min_i \{\lambda_i(R_i(x,y) - z_i^*)\} \tag{6}$$

where $z^* \in \mathbb{R}^N$ is a reference point , and λ now encodes objective importance, and $\mathbb{E}_{x\sim D,\ y\sim\pi_\theta(\cdot|x)}[R_i(x,y)]$ is abbreviated as $R_i(\theta)$. We solve this max-min objective via Online Mirror Descent (OMD) [35], yielding:

$$\mathbb{J}(\theta|\lambda) = \max_\theta \sum_i \{w_i(R_i(\theta) - z_i^*)\} \text{ s.t. } w = \min_w \{w_i\lambda_i(R_i(\theta) - z_i^*)\} \text{ and } ||w||_1 = 1 \tag{7}$$

The indicator vector w is smoothed [34] and updated using TD-learning [43] for stability. We seamlessly integrate this ideal process into the PPO [48] paradigm, resulting in a mixed-advantage

formulation:

$$\nabla_\theta \mathbb{J}(\theta|\lambda) = \mathbb{E}_{s_t,a_t\sim\pi(s_{t-1})}[\big(\sum_{i=1}^{N} w_i A_i^{\pi_\theta}(s_t,a_t)\big)\nabla_\theta log\pi_\theta(s_t,a_t)] \tag{8}$$

More details of the practical implementation see Appendix D. `HoE` enjoys a theoretical convergence rate of $O(log\frac{N}{T})$ where N is the number of objectives and T is the number of iteration rounds (see Appendix F).

3.3 `Tertiary` Preference Routing

The preference routing module is parameter-free layer based on geometric proximity, mapping a user's continuous preference vector λ_{user} to a discrete subset of experts. As illustrated in Fig. 2, preference routing module partitions the entire N-simplex into coarse regions defined by LoRA experts, and further subdivide them into finer subregions via router experts. This hierarchical decomposition enables finer-grained N-objective alignment for any user's preference. Formally, it selects the N closest expert weightings from set of all M expert $\Lambda = \{\lambda_i\}_{i\in[M]}$ based on Euclidean distance:

$$\mathbb{NN}_N(\lambda_{user},\Lambda) = \arg\min_{i}^{N} \|\lambda_{user} - \lambda_i\| \tag{9}$$

This mapping assigns λ_{user} to a corresponding subregion spanned by the selected N experts, guiding downstream router expert activation.

3.4 Hierarchical Assembly for Inference

At inference time, we assemble the three layers into a unified hierarchical model that maps a user's preference vector λ_{user} to a tailored expert composition for response generation. We begin with a collection of L LoRA experts and R router experts $\Theta = \{\tau_i\}_{i\in[L+R]}$, $\Lambda = \{\lambda_i\}_{i\in[L+R]}$.

Preference Routing. Tertiary preference routing module selects the N nearest experts and then expresses λ_{user} as a convex combination of their preference vectors:

$$\omega_r \quad s.t.\ \lambda_{user} = \omega_r^T[\lambda_i]_{i\in\mathbb{NN}_N(\lambda_{user},\Lambda)} \tag{10}$$

The resulting voting vector $\omega_r \in \mathbb{R}^{L+R}$ serves as a soft selector over router experts, where only the selected top-N entries are nonzero.

Router Expert Voting. Each router expert r_{λ_i} produces routing logits based on the input hidden states. The final router output is obtained by aggregating their predictions, weighted by ω_r:

$$\omega_l = \omega_r^T[r_{\lambda_i}(x)]_{i\in[R]} \tag{11}$$

This weighted combination determines the activation of LoRA experts.

LoRA Expert Composition. Finally, the transformer's output is computed as a mixture of the selected LoRA experts, combined with the pre-trained base weights:

$$O_\lambda(x) = W_{pre}x + \sum_{i=1}^{N}(\omega_{l,i}B_iA_ix) \tag{12}$$

4 Experimental Setup

Datasets and Objectives. We follow prior multi-objective alignment studies [49, 62, 44, 5], using seven text generation tasks—Helpful Assistant, Math, Reddit Summary, Beaver Tail, Helpsteer, Psoups, and Helpsteer2 — covering 14 objectives. For more details, refer to Appendix E.

Baselines. We consider 15 competitive algorithms as baselines: RS [44], MOD [49], MODPO [71], RiC [62], MetaAligner [61], PAD [5], MORLHF [31], Args [25], Steering [26], LoraMOE [8], PCB-Merging [15], FR-Merging [69], Aligner [23], Preference-prompting and Personalized soups [21].

Metrics. We primarily use reward model scores to obtain the Pareto frontiers (Fig.3)—each objective is paired with a commonly used open-source reward model. Additionally, we report GPT-4-based win rates—comparative against base model—for further evaluation. Specially, for the math benchmark, we report PASS@1 accuracy, and for the over-refusal benchmark [7], we report safety and helpfulness ratio. For more details, refer to Appendix E.

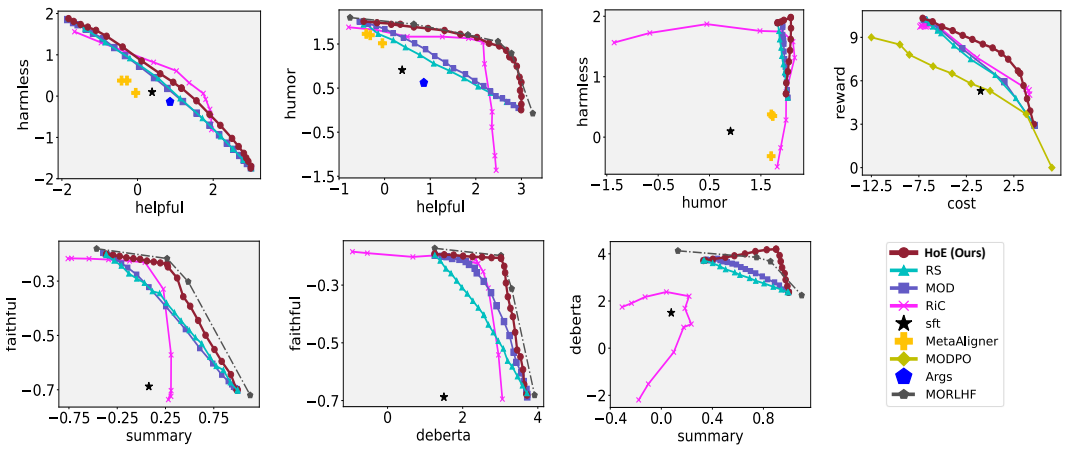

Figure 3: Results of two-objective alignment on HelpAssistant, Reddit Summary and BeaverTails Task with 8 objectives. Compared to the baselines, HoE consistently achieves superior Pareto frontiers.

5 Main Results

We conduct experiments on 6 different NLP tasks with 14 different objectives, testing 200 different preferences and comparing them with 15 baselines. Experiments span two-, three-, and many-objective alignment scenarios. Quantitative comparisons are shown in Fig.3, Fig.4, and Tab. 5.

5.1 Two-Objective Alignment Results

Fig. 3 presents the results for two-objective alignment across seven setups. HoE clearly approaches the theoretical upper bound defined by MORLHF, producing smooth and convex Pareto frontiers, strongly validating its effectiveness.

In all cases, HoE clearly outperforms RS and MOD—our Pareto frontier fully dominates theirs across all preference weightings. Even when constrained to use only LoRA experts for fairness, our method retains this dominance (see Fig.6).

Compared to RiC, HoE achieves better results in 5 out of 7 cases. In the "Summary & Deberta" setting, for instance, our model outperforms RiC by a notable $(+2, +0.8)$ margin. Although RiC slightly outperforms us in a few specific weightings (*e.g.,* "Helpful & Harmless"), this is likely due to its advantage in handling strongly conflicting objectives via online training.

Meanwhile, MetaAligner and Args are limited to the Helpful Assistant task, where their performance is comparatively weak. MODPO also falls significantly short on the BeaverTail task, indicating better generalization of HoE.

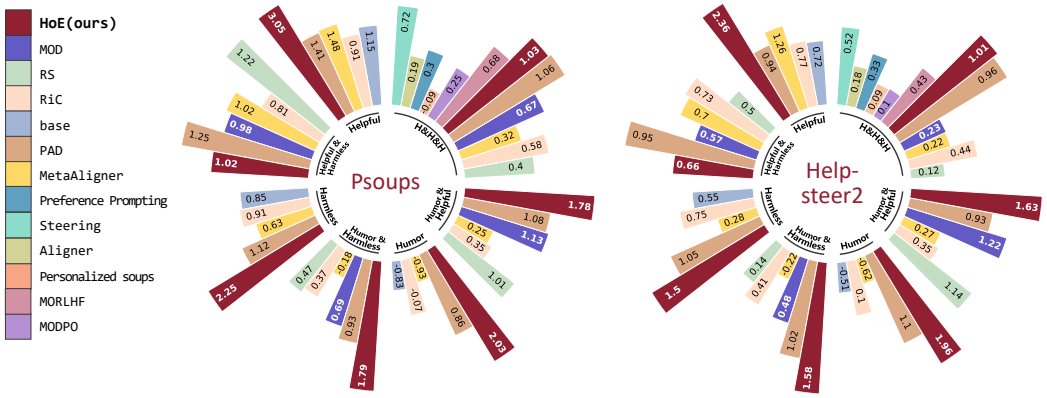

Figure 4: Comparison of alignment results with three objectives (*i.e.,* helpful, harless and humor) on the Psoups and Helpsteer2 datasets.

5.2 Three-Objective Alignment Results

We evaluate alignment across three objectives—Helpful, Harmless, and Humor—on the Helpful Assistant task (see Fig. 8). HoE Pareto-dominates RS and MOD, and consistently outperforms RiC across most of the weight space.

We further test on Psoups and HelpSteer2 using LLaMA3.1-8B, comparing with 11 baselines under a strict generalization setting (none of the models were trained on these datasets). As shown in Fig. 4, our method ranks first in 11 out of 14 evaluation setups. In the remaining three, PAD slightly outperforms us—yet we remain highly competitive.

Additionally, GPT-4-based evaluations (see Appendix. B and Fig. 9) align closely with reward model scores, further confirming the robustness of our approach across models and tasks.

5.3 Many-Objective Alignment Results

We evaluate five-objective alignment on HelpSteer, with results presented in Tab. 5. The PREFERENCE column indicates the user's preference vector λ_{user}. HoE achieves the highest average score, outperforming MOD and RiC across all objectives, with only slight underperformance on a few specific objectives compared to RiC. This demonstrates that HoE is highly effective for many-objective alignment.

Figure 5: Five-objective alignment results on HelpSteer. Preference weighting settings are shown in gray. The best results are bolded and second best ones are underlined.

METHOD	HELPFUL	CORRECT	COHERENCE	COMPLEX	VERBOSITY	AVERAGE
PREFERENCE	0.2	0.2	0.2	0.2	0.2	
RS	67.2	68	76.8	37.3	41.9	58.24
RiC	**71.5**	70.7	**78.3**	41.1	43.8	61.08
MOD	68.4	69.1	76.6	40	45.9	60
HoE (OURS)	70.4	**71.6**	78.1	**42.8**	**47.5**	**62.1** (+3.8)
PREFERENCE	0.17	0.17	0.17	0.25	0.25	
RS	66.7	67.8	76.2	38.9	42.6	58.44
RiC	70	67.6	76.5	42.3	46.2	60.52
MOD	68.1	68.9	76.3	40.9	47.1	60.26
HoE (OURS)	**70**	**71.1**	**77.7**	**42.9**	**48.7**	**62.08**(+3.6)
PREFERENCE	0.11	0.11	0.11	0.33	0.33	
RS	66.4	67.5	75.8	40.5	44.3	58.9
RiC	67.7	62.4	73.9	**44**	**49.9**	59.58
MOD	67.7	68.2	75.6	42.9	48.1	60.5
HoE (OURS)	**69.8**	**70.8**	**77.4**	43.2	49.3	**62.1** (+3.2)

6 Analysis

6.1 Ablation Study

We conducted three ablation studies to assess the impact of (1) individual expert, (2) LoRA ranks, and (3) Tschebyscheff scalarization:

Ablation on Experts. This is the core ablation of our work. We isolate the roles of each LoRA experts and router experts by incrementally removing or combining them to observe their effect on the Pareto frontier (PF). All configurations in this study include two fixed single-objective LoRA experts, and the terms "1 LoRA" or "2 LoRA" refer specifically to the additional ones. Fig. 6 (left) presents results across four configurations:

1) *1 Router*: Adding a single router expert improves performance on specific preferences, highlighting its specialization capability. However, due to its smaller parameter count, the improvements are modest. 2) *1 LoRA*: A single LoRA expert leads to substantial PF expansion near its preference. But for other preference settings, performance drops off, revealing limited coverage. 3) *1 LoRA & 1 Router*: This combination achieves a near-complete PF. The router complements the LoRA expert by covering underrepresented regions, showcasing their strong synergy. 4) *2 LoRA*: Adding a second LoRA expert improves notable performance-close to the PF achieved by MORLHF, but with diminishing returns. In most cases, the 1 LoRA & 1 Router setup offers good coverage with fewer parameters. These results demonstrate that each expert plays a distinct role in shaping the PF, and that combining diverse expert types strike a balance between performance and parameter efficiency.

Ablation on LoRA Rank. We investigate the effect of LoRA rank using LLaMA2-7B-Chat as the base model. For the math task, we extract Math LoRA experts from MathLLaMA2-7B; for assistant tasks, we use HelpfulLlama2-7B and HumorLlama2-7B.

Fig. 6 (middle) shows that lower ranks lead to greater performance degradation. For math, ranks below 128 result in noticeable drops, suggesting high complexity. While assistant tasks are less challenging, and a rank size of 128 is sufficient.

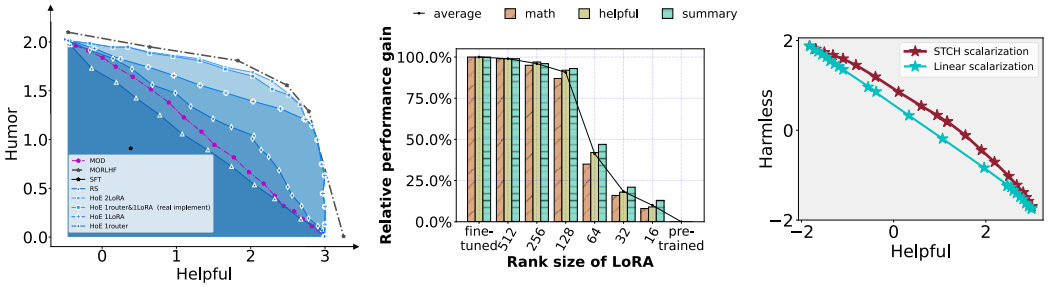

Figure 6: Ablation studies assessing the impact of expert count (Left), LoRA ranks (Middle), and Tchebyscheff scalarization (Right).

Ablation on Tschebyscheff Scalarization. We compare Tschebyscheff scalarization with linear scalarization in MORL. As shown in Fig. 6 (right), linear scalarization often biases the policy to drift significantly toward PF edges, leading to instability or collapse. In contrast, Tschebyscheff-based optimization (OMD-STCH-MORL) maintains stable training while preserving full PF coverage. This confirms its advantage in multi-objective optimization stability.

6.2 Advantages over Existing Methods

While existing methods each excel in specific areas, HoE offers seven notable advantages, with quantitative comparisons provided in Tab. 3. The checklist of advantages are listed in Tab.1.

1) *Lightweight and Parameter-Efficient*. All preference models are unified in a single architecture, significantly reducing storage demands, compared to methods that train and store multiple models.

2) *Predominantly Training-free*. HoE relies primarily on model fusion, requiring minimal training only for a small portion of the router. While other methods (*e.g.,* RiC, PAD, and MetaAligner) require costly exhaustive training as objectives increase.

3) *Minimal Inference Cost*. HoE activates only a few lightweight experts at inference time, making it much faster than decoding- or refinement-based methods (*e.g.,* MetaAligner, PAD, MOD) that require multi-pass inference.

4) *Applicable to Multi-task Learning*. As demonstrated in Fig. 7, HoE achieves comparable performance to other baselines in multi-task learning scenarios, without specialized design for MTL.

5) *Pareto-Steerable.* HoE supports arbitrary user preference, enabling continuous traversal along the Pareto frontier—unlike baselines fixed to preset preferences (*e.g.,* MetaAligner, PAD, and Steering).

6) *Plug-and-play and Scalable*. New unseen objectives can be added without retraining existing experts; existing ones remain valid by simply extending the preference vector (*e.g.,* from $[0.5, 0.5]$ to $[0.5, 0.5, 0.0]$). Some methods (*e.g.,* MORLHF, MOPPO, and RiC) require extensive retraining to involve the new objective, and others (*e.g.,* DPA, PAD, MetaAligner, and LoRAMoE) render previous checkpoints obsolete and necessitate complete retraining.

7) *Free from Prompting*. HoE avoids reliance on handcrafted prompts, enabling generalization to abstract or hard-to-verbalize objectives (*e.g.,* "deberta", "reward" or "cost" in Fig. 3) and preserving the core capabilities of the base LLM - unlike prompt-dependent methods(*e.g.,* PAD, MetaAligner, and DPA)

7 Limitation and Future Work

Despite the strengths of our method, several limitations remain: (1) Our approach depends on off-the-shell single-objective models, which may not always be available. Training such models from scratch can be time-consuming and impractical in some settings. (2) The method relies on effective model merging and SVD-based compression. While these techniques work well for the objectives considered, they may fail in some settings.

8 Conclusion

We propose HoE, a hierarchical Mixture-of-Experts framework for multi-objective alignment in LLMs. By combining LoRA experts, router experts, and preference routing, our method enables efficient and scalable alignment across diverse user preferences. Experiments on 14 objectives across 6 benchmarks and 200 preferences show that HoE outperforms 15+ strong baselines, achieving superior Pareto-optimal results in various multi-objective and multi-task settings.

Acknowledgements

This work was supported by National Science Foundation of China (62476070), Shenzhen Science and Technology Program (JCYJ20241202123503005, GXWD20231128103232001, ZDSYS20230626091203008, KQTD20240729102154066) and Department of Science and Technology of Guangdong (2024A1515011540). This work was also supported in part by the Major Key Project of PCL under Grant PCL2024A06 and PCL2022A05, and in part by the Shenzhen Science and Technology Program under Grant RCJC20231211085918010.

References

[1] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, N. Joseph, S. Kadavath, J. Kernion, T. Conerly, S. E. Showk, N. Elhage, Z. Hatfield-Dodds, D. Hernandez, T. Hume, S. Johnston, S. Kravec, L. Lovitt, N. Nanda, C. Olsson, D. Amodei, T. B. Brown, J. Clark, S. McCandlish, C. Olah, B. Mann, and J. Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[3] E. L. Buehler and M. J. Buehler. X-lora: Mixture of low-rank adapter experts, a flexible framework for large language models with applications in protein mechanics and molecular design. *APL Machine Learning*, 2(2), 2024.

[4] B. Cao, J. Yang, X. Zhou, Z. Kheiri, F. Zahmatkesh, and X. Yang. *Fuzzy Relational Mathematical Programming - Linear, Nonlinear and Geometric Programming Models*, volume 389 of *Studies in Fuzziness and Soft Computing*. Springer, 2020. ISBN 978-3-030-33784-1.

[5] R. Chen, X. Zhang, M. Luo, W. Chai, and Z. Liu. PAD: personalized alignment at decoding-time. *arXiv preprint arXiv:2410.04070*, 2024.

[6] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[7] J. Cui, W. Chiang, I. Stoica, and C. Hsieh. Or-bench: An over-refusal benchmark for large language models. *arXiv preprint arXiv:2405.20947*, 2024.

[8] S. Dou, E. Zhou, Y. Liu, S. Gao, W. Shen, L. Xiong, Y. Zhou, X. Wang, Z. Xi, X. Fan, S. Pu, J. Zhu, R. Zheng, T. Gui, Q. Zhang, and X. Huang. Loramoe: Alleviating world knowledge forgetting in large language models via moe-style plugin. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1932–1945, 2024.

[9] G. DU, J. Li, H. Liu, R. Jiang, S. Yu, Y. Guo, S. K. Goh, and H.-K. Tang. Knowledge fusion by evolving weights of language models. In *Findings of The 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

[10] G. Du, Z. Fang, J. Li, J. Li, R. Jiang, S. Yu, Y. Guo, Y. Chen, S. K. Goh, H.-K. Tang, D. He, H. Liu, and M. Zhang. Neural parameter search for slimmer fine-tuned models and better transfer. *arXiv preprint arXiv:2505.18713*, 2025. URL https://arxiv.org/abs/2505.18713.

[11] C. Gao, K. Chen, J. Rao, B. Sun, R. Liu, D. Peng, Y. Zhang, X. Guo, J. Yang, and V. Subrahmanian. Higher layers need more lora experts. *arXiv preprint arXiv:2402.08562*, 2024.

[12] A. A. Gargiulo, D. Crisostomi, M. S. Bucarelli, S. Scardapane, F. Silvestri, and E. Rodolà. Task singular vectors: Reducing task interference in model merging. *arXiv preprint arXiv:2412.00081*, 2024.

[13] H. Gu, W. Li, L. Li, Q. Zhu, M. Lee, S. Sun, W. Xue, and Y. Guo. Delta decompression for moe-based llms compression. *CoRR*, abs/2502.17298, 2025.

[14] W. Guo, J. Li, Y. Li, W. Wang, D. He, J. Yu, and M. Zhang. Mtsa: Multi-turn safety alignment for llms through multi-round red-teaming. *arXiv preprint arXiv:2505.17147*, 2025. URL https://arxiv.org/abs/2505.17147.

[15] D. Guodong, J. Lee, J. Li, R. Jiang, Y. Guo, S. Yu, H. Liu, S. K. Goh, H.-K. Tang, D. He, et al. Parameter competition balancing for model merging. In *Proceedings of the Thirty-eighth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[16] N. Hansen. The CMA evolution strategy: A tutorial. *arXiv preprint arXiv:1604.00772*, 2016.

[17] S. He, R.-Z. Fan, L. Ding, L. Shen, T. Zhou, and D. Tao. Merging experts into one: Improving computational efficiency of mixture of experts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

[18] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models. In *Proceedings of the Tenth International Conference on Learning Representations (ICLR)*, 2022.

[19] X. Huang, S. Li, E. Dobriban, O. Bastani, H. Hassani, and D. Ding. One-shot safety alignment for large language models via optimal dualization. *arXiv preprint arXiv:2405.19544*, 2024.

[20] G. Ilharco, M. T. Ribeiro, M. Wortsman, L. Schmidt, H. Hajishirzi, and A. Farhadi. Editing models with task arithmetic. In *Proceedings of the Eleventh International Conference on Learning Representations (ICLR)*, 2023.

[21] J. Jang, S. Kim, B. Y. Lin, Y. Wang, J. Hessel, L. Zettlemoyer, H. Hajishirzi, Y. Choi, and P. Ammanabrolu. Personalized soups: Personalized large language model alignment via post-hoc parameter merging. *arXiv preprint arXiv:2310.11564*, 2023.

[22] J. Ji, M. Liu, J. Dai, X. Pan, C. Zhang, C. Bian, B. Chen, R. Sun, Y. Wang, and Y. Yang. Beavertails: Towards improved safety alignment of LLM via a human-preference dataset. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023*, 2023.

[23] J. Ji, B. Chen, H. Lou, D. Hong, B. Zhang, X. Pan, T. Qiu, J. Dai, and Y. Yang. Aligner: Efficient alignment by learning to correct. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[24] X. Jin, X. Ren, D. Preotiuc-Pietro, and P. Cheng. Dataless knowledge fusion by merging weights of language models. In *Proceedings of the Eleventh International Conference on Learning Representations (ICLR)*, 2022.

[25] M. Khanov, J. Burapacheep, and Y. Li. ARGS: alignment as reward-guided search. In *Proceedings of the Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[26] K. Konen, S. Jentzsch, D. Diallo, P. Schütt, O. Bensch, R. E. Baff, D. Opitz, and T. Hecking. Style vectors for steering generative large language models. In *Proceedings of the Findings of the Association for Computational Linguistics (ACL)*, pages 782–802, 2024.

[27] H. J. Kushner and G. G. Yin. *Stochastic Approximation and Recursive Algorithms and Applications*. Springer, 2003.

[28] J. Lee, Y. Wang, J. Li, and M. Zhang. Multimodal reasoning with multimodal knowledge graph. In *The 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

[29] J. Lee, G. DU, J. Li, S. K. Goh, W. Wang, Y. Wang, F. Liu, H.-K. Tang, S. Alharbi, D. He, and M. Zhang. Multi-modality expansion and retention for llms through parameter merging and decoupling. *arXiv preprint arXiv:2505.17110*, 2025. URL https://arxiv.org/abs/2505.17110.

[30] K. Li, T. Zhang, and R. Wang. Deep reinforcement learning for multiobjective optimization. *IEEE transactions on cybernetics*, 51(6):3103–3114, 2020.

[31] K. Li, T. Zhang, and R. Wang. Deep reinforcement learning for multiobjective optimization. *IEEE Trans. Cybern.*, 51(6):3103–3114, 2021.

[32] X. Li, Y. Zhou, L. Zhao, J. Li, and F. Liu. Impromptu cybercrime euphemism detection. In *The 31st International Conference on Computational Linguistics (COLING)*, 2025.

[33] B. Lin, W. Jiang, Y. Xu, H. Chen, and Y.-C. Chen. Parm: Multi-objective test-time alignment via preference-aware autoregressive reward model. *arXiv preprint arXiv:2505.06274*, 2025.

[34] X. Lin, X. Zhang, Z. Yang, F. Liu, Z. Wang, and Q. Zhang. Smooth tchebycheff scalarization for multi-objective optimization. In *Proceedings of the Forty-first International Conference on Machine Learning (ICML)*, 2024.

[35] M. Liu, X. Zhang, C. Xie, K. Donahue, and H. Zhao. Online mirror descent for tchebycheff scalarization in multi-objective optimization. *arXiv preprint arXiv:2410.21764*, 2024.

[36] Y. Lu, J. Li, Y. Zhou, Y. Zhang, W. Wang, X. Li, M. Zhang, F. Liu, J. Yu, and M. Zhang. Adaptive detoxification: Safeguarding general capabilities of llms through toxicity-aware knowledge editing. In *Findings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025.

[37] Y. Lu, Y. Zhou, J. Li, Y. Wang, X. Liu, D. He, and F. L. andMin Zhang. Knowledge editing with dynamic knowledge graphs for multi-hop question answering. In *The Thirty-Ninth AAAI Conference on Artificial Intelligence (AAAI)*, 2025.

[38] X. Ma, J. Li, and M. Zhang. Chain of thought with explicit evidence reasoning for few-shot relation extraction. In *Findings of the Association for Computational Linguistics (EMNLP)*, pages 2334–2352, 2023. URL https://aclanthology.org/2023.findings-emnlp.153.

[39] M. S. Matena and C. A. Raffel. Merging models with fisher-weighted averaging. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 17703–17716, 2022.

[40] S. Mukherjee, A. Lalitha, S. Sengupta, A. Deshmukh, and B. Kveton. Multi-objective alignment of large language models through hypervolume maximization. *arXiv preprint arXiv:2412.05469*, 2024.

[41] S. Paternain, M. Calvo-Fullana, L. F. O. Chamon, and A. Ribeiro. Safe policies for reinforcement learning via primal-dual methods. *IEEE Trans. Autom. Control.*, 68(3):1321–1336, 2023.

[42] B. Ping, S. Wang, H. Wang, X. Han, Y. Xu, Y. Yan, Y. Chen, B. Chang, Z. Liu, and M. Sun. Delta-come: Training-free delta-compression with mixed-precision for large language models. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[43] S. Qiu, D. Zhang, R. Yang, B. Lyu, and T. Zhang. Traversing pareto optimal policies: Provably efficient multi-objective reinforcement learning. *arXiv preprint arXiv:2407.17466*, 2024.

[44] A. Ramé, G. Couairon, C. Dancette, J. Gaya, M. Shukor, L. Soulier, and M. Cord. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

[45] N. Rimsky, N. Gabrieli, J. Schulz, M. Tong, E. Hubinger, and A. M. Turner. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 15504–15522, 2024.

[46] Robbins, Herbert, and S. Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 2023.

[47] S. Ryu, S. Seo, and J. Yoo. Efficient storage of fine-tuned models via low-rank approximation of weight residuals. *CoRR*, abs/2305.18425, 2023.

[48] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[49] R. Shi, Y. Chen, Y. Hu, A. Liu, H. Hajishirzi, N. A. Smith, and S. S. Du. Decoding-time language model alignment with multiple objectives. *arXiv preprint arXiv:2406.18853*, 2024.

[50] Z. Shi, Y. Zhou, J. Li, Y. Jin, Y. LI, D. He, F. Liu, S. Alharbi, J. Yu, and M. Zhang. Safety alignment via constrained knowledge unlearning. *arXiv preprint arXiv:2505.18588*, 2025. URL https://arxiv.org/abs/2505.18588.

[51] N. Stiennon, L. Ouyang, J. Wu, D. M. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. F. Christiano. Learning to summarize from human feedback. *arXiv preprint arXiv:2009.01325*, 2020.

[52] H. Sun, A. Hüyük, and M. van der Schaar. Query-dependent prompt evaluation and optimization with offline inverse rl. In *Proceedings of the Twelfth International Conference on Learning Representations (ICLR)*, 2023.

[53] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, and et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[54] P. Vamplew, R. Dazeley, C. Foale, S. Firmin, and J. Mummery. Human-aligned artificial intelligence is a multiobjective problem. *Ethics and information technology*, 20:27–40, 2018.

[55] H. Wang, Y. Lin, W. Xiong, R. Yang, S. Diao, S. Qiu, H. Zhao, and T. Zhang. Arithmetic control of llms for diverse user preferences: Directional preference alignment with multi-objective rewards. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 8642–8655, 2024.

[56] K. Wang, R. Kidambi, R. Sullivan, A. Agarwal, C. Dann, A. Michi, M. Gelmi, Y. Li, R. Gupta, K. Dubey, et al. Conditional language policy: A general framework for steerable multi-objective finetuning. In *Proceedings of the Findings of the Association for Computational Linguistics: EMNLP*, pages 2153–2186, 2024.

[57] X. Wang, Y. Zheng, Z. Wan, and M. Zhang. Svd-llm: Truncation-aware singular value decomposition for large language model compression. *arXiv preprint arXiv:2403.07378*, 2024.

[58] Y. Xu, U. M. Sehwag, A. Koppel, S. Zhu, B. An, F. Huang, and S. Ganesh. Genarm: Reward guided generation with autoregressive reward model for test-time alignment. In *The Thirteenth International Conference on Learning Representations, ICLR 2025*, 2025.

[59] P. Yadav, D. Tam, L. Choshen, C. A. Raffel, and M. Bansal. Ties-merging: Resolving interference when merging models. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[60] E. Yang, Z. Wang, L. Shen, S. Liu, G. Guo, X. Wang, and D. Tao. Adamerging: Adaptive model merging for multi-task learning. In *Proceedings of the Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[61] K. Yang, Z. Liu, Q. Xie, J. Huang, T. Zhang, and S. Ananiadou. Metaaligner: Towards generalizable multi-objective alignment of language models. In *Proceedings of the Thirty-eighth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[62] R. Yang, X. Pan, F. Luo, S. Qiu, H. Zhong, D. Yu, and J. Chen. Rewards-in-context: Multi-objective alignment of foundation models with dynamic preference adjustment. In *Proceedings of the Forty-first International Conference on Machine Learning (ICML)*, 2024.

[63] Z. Yuan, Y. Shang, Y. Song, Q. Wu, Y. Yan, and G. Sun. ASVD: activation-aware singular value decomposition for compressing large language models. *CoRR*, abs/2312.05821, 2023.

[64] T. Zadouri, A. Üstün, A. Ahmadian, B. Ermis, A. Locatelli, and S. Hooker. Pushing mixture of experts to the limit: Extremely parameter efficient moe for instruction tuning. In *Proceedings of the Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[65] L. Zhang, B. Wang, J. Wang, X. Zhao, M. Zhang, H. Yang, M. Zhang, Y. Li, J. Li, J. Yu, and M. Zhang. Function-to-style guidance of llms for code translation. In *The Forty-Second International Conference on Machine Learning (ICML)*, 2025.

[66] L. Zhang, J. Wang, M. Zhang, G. Cao, E. Shi, mayuchi, J. Yu, H. LIU, J. Li, and M. Zhang. Speed up your code: Progressive code acceleration through bidirectional tree editing. In *The 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025.

[67] Q. Zhang and H. Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on evolutionary computation*, 11(6):712–731, 2007.

[68] X. Zhang, M. Burger, and S. J. Osher. A unified primal-dual algorithm framework based on bregman iteration. *J. Sci. Comput.*, 46(1):20–46, 2011.

[69] S. Zheng and H. Wang. Free-merging: Fourier transform for model merging with lightweight experts. *arXiv preprint arXiv.2411.16815*, 2024.

[70] Y. Zhou, W. Li, Y. Lu, J. Li, F. Liu, M. Zhang, Y. Wang, D. He, H. LIU, and M. Zhang. Reflection on knowledge graph for large language models reasoning. In *Findings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025.

[71] Z. Zhou, J. Liu, J. Shao, X. Yue, C. Yang, W. Ouyang, and Y. Qiao. Beyond one-preference-fits-all alignment: Multi-objective direct preference optimization. In *Proceedings of Findings of the Association for Computational Linguistics (ACL)*, pages 10586–10613, 2024.

[72] D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A The workflow of HoE

Algorithm 1 show the whole pipeline of HoE.

Algorithm 1 The workflow of HoE

Input: objective number N, single-objective fine-tuned weights $\{\theta_i\}_{i\in[N]}$, pre-trained weights θ_{pre}, N-Simplex Δ_N, number of MO LoRA Experts L, number of MO router Experts R, HoE model $\Theta = \{\}$
uniformly select weightings $\{\lambda_l\}_{l\sim[N+L+R]} \sim \Delta_N$
for $i = 1$ **to** N **do**
  $\tau_i \leftarrow$ extract LoRA from $(\theta_i - \theta_{pre})$
end for
for $l = N$ **to** $N + L$ **do**
  $\tau_{i+l} \leftarrow$ Merging $\{\theta_i\}_{i\in[N]}$ with weighting λ_l
end for
$\Theta = \{\tau_i\}_{i=[N+L]}$
for $r = N + L$ **to** $N + L + R$ **do**
  $\tau_r \leftarrow$ Train router experts on λ_r with Θ
end for
insert $\Theta = \{\tau_i\}_{i=[N+L+R]}$
output Θ

B Additional Results

B.1 Mullti-Task Results

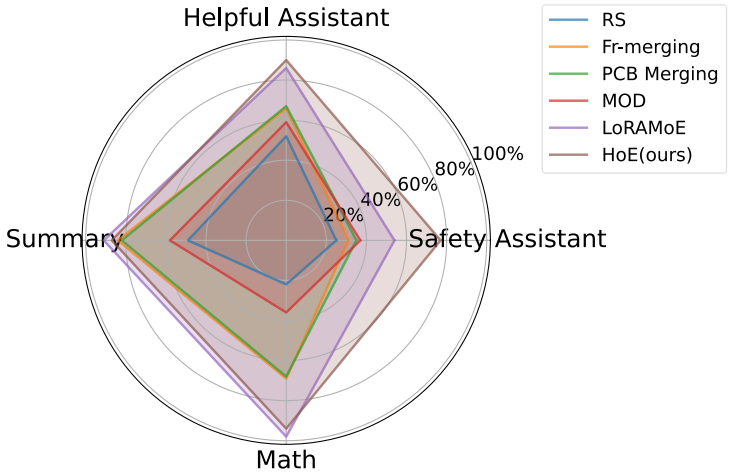

Figure 7: Multi-Task Learning results. Our router experts specialized for "Helpful Assistant" and "Safety Assistant" enable better performance than LoRAMoE. The base model's performance is normalized to 0% and single-objective models are normalized to 100%

We designed experiments involving four tasks learning: 1) Helpful Assistant: An assistant that provides helpful and correct responses to prompts, even for harmful ones. 2) Safety Assistant: An assistant that refuses to respond to harmful prompts. 3) Summary Task: Summarizes a given poster. 4) Math Task: Solves math problems from the GSM8K dataset[6]. The first two tasks were evaluated on the over-refusal benchmark[7], the Summary Task was assessed using the average score across three objectives, and the Math Task was evaluated with Pass@1 accuracy on the GSM8K test set. To balance different scores, all results were normalized, setting the base model's performance to 0% and single-objective models to 100%, which are shown in Fig.7.

We compared HoE with baselines such as LoRAMoE, RS, MOD, PCBmerging, and FR-Merging, all initialized with the same model and using LoRA adapter-based fusion. As expected, HoE outperforms PCBmerging, FR-Merging, and MoAlignment methods (e.g., RS, MOD). While LoRAMoE achieved strong performance on the Summary Task and Math Task, it struggled on the Helpful and Safety Assistant tasks due to the nuanced and overlapping nature of harmful and seemingly harmful prompts in the over-refusal benchmark. The router in LoRAMoE, designed for uniform preferences $[0.25, 0.25, 0.25, 0.25]$, failed to distinguish between red-teaming prompts and less harmful ones effectively. In contrast, HoE introduced specialized router experts for the Helpful and Safety Assistant tasks ($[0.5, 0.5, 0.0, 0.0]$), enabling better performance by dynamically adjusting input weightings. This improvement highlights the flexibility and robustness of HoE in multi-task learning scenarios.

Table 2: Alignment results for unseen dataset HelpSteer2 and Psoups on three objective s(*i.e.,* Helpful, Harmless, Humor) with Llama3.1-8B

Method	Helpful	Helpful& Harmless	Harmless	Harmless& Humor	Humor	Humor& Helpful	HHH
Psoups dataset							
Base	1.15	0.91	0.85	-0.07	-0.83	0.1	0.32
RS	0.88	0.88	0.92	0.49	0.15	0.44	0.4
RiC	0.91	0.81	0.91	0.37	-0.07	0.35	0.58
MetaAligner	1.48	1.02	0.63	-0.18	-0.93	0.25	0.32
MOD	0.94	0.96	1.01	0.68	0.46	0.69	0.67
PAD	1.41	**1.25**	1.12	0.93	0.86	1.08	**1.06**
HoE **(Ours)**	**3.05**	1.02	**2.25**	**1.79**	**2.03**	**1.78**	1.03
HelpSteer dataset							
Base	0.72	0.63	0.55	-0.04	-0.51	0.09	0.17
RS	0.76	0.74	0.81	0.47	0.17	0.44	0.12
RiC	0.77	0.73	0.75	0.41	0.10	0.35	0.44
MetaAligner	1.26	0.70	0.28	-0.22	-0.62	0.27	0.22
MOD	0.72	0.77	0.85	0.59	0.36	0.51	0.23
PAD	0.94	**0.95**	1.05	1.02	1.10	0.93	0.96
HoE **(Ours)**	**2.36**	0.66	**1.5**	**1.58**	**1.96**	**1.63**	**1.01**

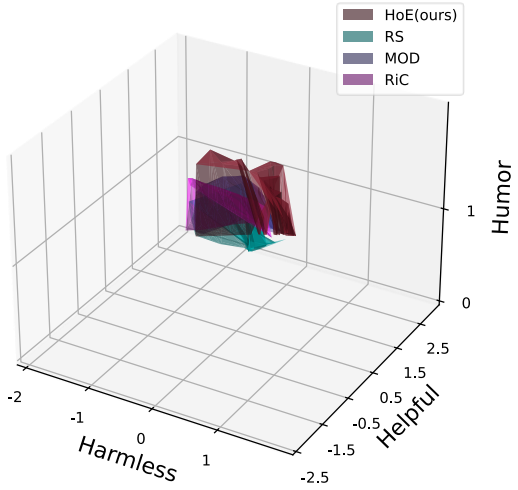

Figure 8: Alignment results with Helpful Assistant task on three-objective. Our approach consistently outperforms RS, MOD and RiC

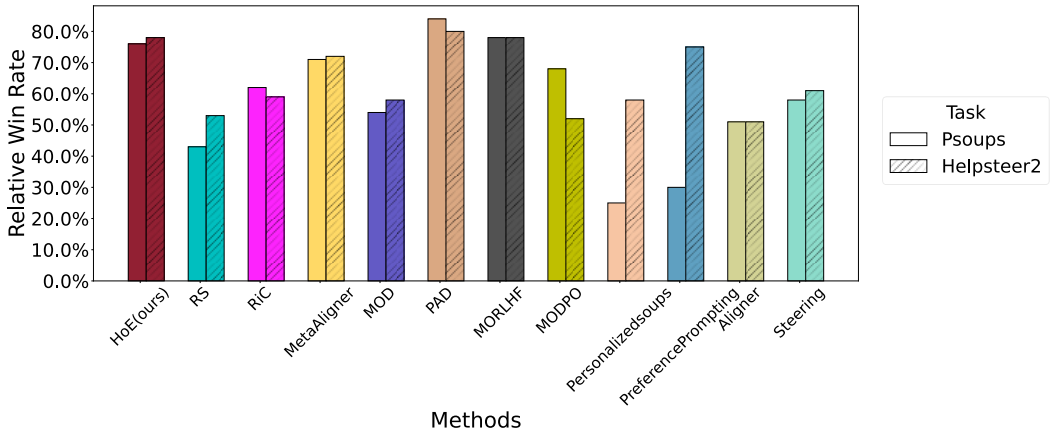

Figure 9: GPT-4 evaluates all methods on Psoups and Helpsteer2 task, comparing the relative win rate of model-generated responses over the original responses for each approach. The evaluation is conducted across three dimensions: helpfulness, harmlessness, and humor. We take the average win rate across these three metrics as the final result.

B.2 Cost Analysis

Table 3: Comparison of Training Costs, Storage Costs, and Inference Costs for various baselines, when using Llama2-7B as the base model to align on three objectives.

BASELINES	STORAGE	TRAINING PARAMETERS	INFERENCE COST
RS	7.48B	0	1.0
MOD	7.48B	0	3.10 ± 0.3
MODPO	7.8B	0.8B	1.0
RIC	7.64B	0.64B	1.0
ARGS	14B	0.16B	2.02 ± 0.2
METAALIGNER	14B	0.16B	2.04 ± 0.3
PAD	14B	0.16B	2.98 ± 0.5
HOE (OURS)	7.64B	8M	1.23 ± 0.2

We conduct a cost analysis of baseline models when performing three-objective alignment with LLaMA2-7B, as summarized in Table 3. Our evaluation considers four key dimensions: 1)Storage: The amount of parameters that must be permanently stored in memory throughout the inference pipeline. 2)Number of Trainable Parameters, 3)Inference Cost: The computational overhead incurred during inference.

Methods such as MetaAligner, Args, PAD, and MOD, which rely on decoding or refinement, significantly increase inference costs as the number of objectives grows. In contrast, HoE only incurs a slight increase in inference time after activating three experts, demonstrating its scalability. Extrapolating from this, HoE could align at least 12 objectives before inference time doubles, ensuring efficient multi-objective scaling.

Moreover, MetaAligner, Args, and PAD require at least two models at inference time. If full-parameter training is considered, PAD also requires storing an additional reference model, while MOD and RS each require three separate 7B-scale models. In contrast, HoE extracts LoRA experts from full-rank dense task vectors and fine-tunes them to recover the optimal Pareto frontier, making it lightweight and highly parameter-efficient.

In terms of trainable parameters and training cost, HoE requires significantly fewer parameters and resources than other training-based methods, making it a more efficient solution for multi-objective alignment.

Table 4: Several methods against TA on three-objective alignment (Chinese & Math & Code)

	CMMLU	GSM8K@1(5-SHOT)	HUMAN-EVAL
PRETRAIN	-	26.2	-
CHINESELLAMA	38.6	4.9	13.4
MATHLLAMA	31.2	70.6	0
CODELLAMA	33.3	28.0	17.1
TASKARITHMETIC[20]	35.4	48.7	9.8
PCB-MERGING[15]	36.5	54.3	16.5
FR-MERGING[69]	36.4	55.6	15.7
TIES-MERGING[59]	36.4	56.2	14.0
CODEEXPERT(R=256)	-	-	16.7
MATHEXPERT(R=128)	-	66.3	-
CHINESEEXPERT(R=128)	37.8	-	-
MOLORAEXPERT(OURS)	35.7	50.4	13.7

C Potential Reader Questions

To further clarify several key design choices and theoretical intuitions, we address some potential questions that may arise when interpreting our method.

Q1. Why is it necessary to merge the weights of different LoRA experts to construct additional experts, given that Eq.3.1 already performs a form of weight merging?

One may question whether the weight merging in Eq.3.1 , which linearly combines LoRA experts, already suffices. While Eq.3.1 indeed embodies a linear arithmetic operation akin to Task Arithmetic, this operation alone is insufficient to model the complex trade-offs required for multi-objective optimization. Our model merging strategy works at a finer granularity—directly at the parameter level—allowing selective reinforcement or attenuation of individual parameters. This more expressive mechanism enables us to better approximate solutions along the Pareto front. Empirical results (see Tab. 4) merging improves performance by 40%, and MOLoRA expert reducing storage by 30% while retaining performance, confirming its necessity.

Q2. Could other approaches such as MOD or RS similarly use LoRA to reduce storage?

One may wonder whether alternatives like MOD and RS could benefit equally from LoRA-based compression. While both methods can, in theory, integrate LoRA to save storage, practical limitations arise.

In the case of RS, each LoRA adapter must be expanded into full-parameter form during inference, after which parameter soups are applied according to user preference. This results in storage requirements equivalent to full models and typically leads to inferior performance compared to directly using dense models.

MOD, on the other hand, can theoretically be adapted to use LoRA by applying different LoRA modules to a shared backbone. However, this design sacrifices one of MOD's key strengths—cross-architecture decoding. Restricting MOD to a single base model severely limits its flexibility and practical deployment, making such an adaptation largely infeasible for real-world applications.

Q3. Is there experimental evidence that Task Arithmetic (TA) underperforms in this context?

One may ask for empirical evidence showing that Task Arithmetic yields subpar performance in our setting. Our ablation studies (see Fig. 6, Left) directly address this question. The results demonstrate that simply reducing the number of LoRA experts and performing naive arithmetic combinations significantly degrades performance, even falling behind MOD in some cases.

The only difference between TA and our "few-experts" configuration is that TA uses a fully parameterized vector while our method uses a sparse LoRA. To distinguish this, we have conducted ablation studies on LoRA ranks (see Fig. 6. Mid) and we further conducted experiments (see Tab. 4) , showing TA underperforms other fusion methods, while LoRA Experts match TA with lower space cost.

Q4. Does HoE need to use router experts to adaptively select LoRA experts during inference? Is there any analysis of the overhead?

One may wonder whether router experts are necessary. Taking Llama3.1-8B as an example, the size of the router's parameters is negligible compared to LoRA experts or the transformer's dense matrices, which we refer to as "parameter overhead." Furthermore, traditional model merging struggles to handle extreme preference weightings (e.g., [0.1, 0.8, 0.1]), often leading to trivial MOLoRA experts (e.g., [0.33, 0.33, 0.33]). We refer to this issue as "coverage limitation."

While the parameters of router experts is negligible, its impact is far from negligible. During inference, router experts function as dynamic routers, enabling fine-grained selection of upper-layer LoRA experts. As shown in Fig. 6 (Left), adding router experts can achieve a comparable effect to adding MOLoRA experts while maintaining lower parameter overhead.

D Implementation Details

D.1 LoRA Expert Details

As the first work to apply task-SVD [12] for sparsifying large-scale LLMs exceeding 7B parameters, one might reasonably question its effectiveness in this context. However, our experimental results demonstrate that model merging with task-SVD achieves remarkable performance in this domain.

We now provide a detailed explanation of our sparsification process:

Taking Math LoRA experts as an example, we compute the "math task vector" by subtracting LLaMA2-7B-Chat-hf from MathLLaMA-7B (an open-source model described in Appendix E). Next, we prune 40% of the least significant parameters across the entire model based on absolute magnitude, resulting in a sparse matrix.

We then apply SVD decomposition, sorting the singular values and selecting the top 128 rank-1 matrices. To further optimize performance, we dynamically determine a "rescaling factor" based on test results. In this case, we choose a scaling factor of 1.9, which improves LLaMA2-7B-Chat-hf's accuracy on GSM8K from $26\% \pm 4\%$to $68\% \pm 4\%$.

D.2 `HoE` Details

For 2-objective alignment, in addition to the two corresponding single-objective LoRA experts, we introduce an additional LoRA expert represented by $[0.5, 0.5]$ and adaptively add one router expert based on the evaluation result. This results in a total of three LoRA experts and one router expert.

For 3-objective alignment, we include the three single-objective LoRA experts along with an additional LoRA expert represented by $[0.33, 0.33, 0.33]$. Specifically, for the Helpful Assistant Task, we incorporate a router expert represented by $[0.25, 0.25, 0.5]$ to enhance preference balancing. This results in a total of four LoRA experts and one router expert.

For 5-objective alignment on the HelpSteer Task, we utilize five single-objective LoRA experts alongside an additional LoRA expert represented by $[0.33, 0.33, 0.33, 0, 0]$ and a router expert represented by $[0.2, 0.2, 0.2, 0.2, 0.2]$ to improve adaptability across different preferences. This results in a total of six LoRA experts and one router expert.

D.3 Optimization Details

Unlike LoRAMoE [8], our method keeps LoRA expert parameters frozen, drastically reducing training resource requirements. Each router expert is optimized for its corresponding weighting with the frozen LoRA experts, making them plug-and-play and easy to integrate into existing architectures without extensive retraining. The optimization objective for each router expert can be formalized as:

$$r_\lambda = \arg\min_r \mathbb{E}_{y\sim\pi_{HoE}(\cdot|x,\theta_{pre},\tau_{1:N+E},r)}[R_\lambda(x,y)] \tag{13}$$

where $\tau_{1:N+E}$ represents the set of existing LoRA experts, with only N of them being activated for a given input. $R_\lambda(x,y)$ is the linear combination of N reward signals. To capture non-convex regions of the Pareto Front, we employ Tchebycheff (TCH) scalarization, optimizing for the worst-case objective. For simplicity, $\mathbb{E}_{x\sim D,\ y\sim\pi_\theta(\cdot|x)}[R_i(x,y)]$ is abbreviated as $R_i(\theta)$. We denote the expected reward for objective i as $R_i(\theta) = \mathbb{E}_{x\sim D,\ y\sim\pi_\theta(\cdot|x)}[R_i(x,y)]$. The objective can be formulated as:

$$\mathbb{J}(\theta|\lambda) = \max_\theta \min_i \{\lambda_i(R_i(x,y) - z_i^*)\} \tag{14}$$

where z^* represents a reference point indicating the desired performance level for each objective, and λ denotes the relative importance of each objective.

Due to the difficulty of directly solving the max-min formulation, we adopt the Online Mirror Descent (OMD) approach [35]. We decompose the TCH optimization into two stages:

$$\mathbb{J}(\theta|\lambda) = \max_{\theta} \sum_i \{w_i(R_i(\theta) - z_i^*)\} \tag{15}$$

$$\text{s.t. } w = \min_{w} \{w_i \lambda_i (R_i(\theta) - z_i^*)\} \text{ and } ||w||_1 = 1 \tag{16}$$

However, the one-hot nature of w causes abrupt changes at Pareto frontier boundaries, affecting reinforcement learning robustness. To address this, we use the Smooth Tchebycheff (STCH) approach [34], replacing the min operator in Equation 16 with a softmax function, resulting in a smooth indicator vector w.

$$w = softmax\{\lambda_i(z_i^* - R_i(\theta))\} \tag{17}$$

Following OMD-STCH-MORL [43], we update w using TD-learning, enabling the optimization to leverage online data across multiple training batches for more stable estimation:

$$log\ w_i^{t+1} \leftarrow log\ w_i^t + \alpha\lambda_i(z_i^* - R_i(\theta)) \tag{18}$$

We seamlessly integrate the above ideal process into the PPO [48] paradigm, resulting in a mixed-advantage formulation with the indicator vector w.

$$\nabla_\theta \mathbb{J}(\theta|\lambda) = \mathbb{E}_{s_t,a_t\sim\pi(s_{t-1})}[A_\lambda^{\pi_\theta}(s_t,a_t)\nabla_\theta log\pi_\theta(s_t,a_t)]$$
$$\text{where } A_\lambda^{\pi_\theta}(s_t,a_t) = \sum_{i=1}^{N} w_i A_i^{\pi_\theta}(s_t,a_t) \tag{19}$$

E Experiment Details

E.1 Datasets Details

We utilize the following dataset for training and evaluation.

For Helpful Assistant task, we utilize "hh-rlhf" dataset [2]https://huggingface.co/datasets/Anthropic/hh-rlhf, a multi-round dialogue dataset.

For Reddit Summary task, we utilize the summary dataset [51] https://huggingface.co/datasets/openai/summarize_from_feedback.

For BeaverTails task, we utilize PKU-SafeRLHF-10K [22]https://huggingface.co/datasets/PKU-Alignment/PKU-SafeRLHF-10K

For HelpSteer task, we utilize the HelpSteer dataset. https://huggingface.co/datasets/nvidia/HelpSteer

For Helpsteer2 task, we utilize the HelpSteer2 dataset https://huggingface.co/datasets/nvidia/HelpSteer2

For Psoups task, we utilize the same evaluation dataset as [21] https://storage.googleapis.com/personalized-soups/data.zip

For Math task, we utilize the GSM8k dataset[6] https://huggingface.co/datasets/openai/gsm8k

For MTL task, we additionally utilize over-refusal benchmark [7].

E.2 Reward Model Details

The 14 distinct objectives consist of both interpretable natural language goals and names derived from reward models (RMs): " Helpful", " Harmless", "Humor" on Helpful Assistant task, Psoups task and Helpsteer2 task; " math" on Math Task; " faithful", " summary", "deberta" on Reddit Summary task; " reward", " cost" on BeaverTail task; " helpfulness", " correctness", " coherence", " complexity", " verbosity" on Helpsteer task.

We utilize following open-sourced reward models for training and evaluations. For Reddit Summary, we use https://huggingface.co/Tristan/gpt2_reward_summarization for Summary,

https://huggingface.co/OpenAssistant/reward-model-deberta-v3-large-v2 for "deberta" and https://huggingface.co/CogComp/bart-faithful-summary-detector for Faithful; for Helpful Assistant, HelpSteer2 and Psoups, we use https://huggingface.co/Ray2333/gpt2-large-helpful-reward_model for Helpfulness, https://huggingface.co/Ray2333/gpt2-large-harmless-reward_model for Harmlessness and https://huggingface.co/mohameddhiab/humor-no-humorfor Humor; for BeaverTail, we use https://huggingface.co/PKU-Alignment/beaver-7b-v1.0-reward for "reward" and https://huggingface.co/PKU-Alignment/beaver-7b-v1.0-cost for "cost' for all five objectives, helpfulness, correctness, coherence, complexity and verbosity.

The " helpful" and " harmless" RMs are directly trained on " hh-rlhf" dataset with nearly 0.8 accuracy. The " humor" RM was trained on a joke dataset to detect humor with a 0.95 F1 score. The five RMs for HelpSteer are directly trained on HelpSteer with over 0.75 accuracy.

E.3 Base Model Details

We utilize three base pre-trained models: Llama2-7B[53][1], Llama3.1-8B[2] and MetaLlama3-8B[3], and main results are conducted on Llama2-7B.

To adapt the model to specific task, we first preform SFT on Llama2-7B on each above tasks, getting SFT models as backbones. For Llama3.1-8B, we directly use the open-sourced model Llama3.1-SFT-8B[4] which is fine-tuned on Llama3.1-8B.

As to fine-tuned preference models, we have the option to directly use off-the-shelf models, which are publicly available and fine-tuned for specific objectives, such as MetaMath-7b[5] or to fine-tune the entire model or apply a parameter-efficient fine-tuning (PEFT) method on the pre-trained model.For MetaLlama3-8B, no aditional SFT training are conducted.

E.4 Baseline Details

On 6 tasks, we use the same backbone models separately to reproduce all baselines. Implementation details are as follows:

RiC, RS, MORLHF: we reproduce RiC, RS and MORLHF according to https://github.com/YangRui2015/RiC

MOD: we reproduce MOD according to https://github.com/srzer/MOD

MODPO: we reproduce MODPO according to https://github.com/ZHZisZZ/modpo

Args: We reproduce Args accoding to https://github.com/deeplearning-wisc/args, and use open-sourced model https://huggingface.co/argsearch/llama-7b-rm-float32

PCBmerging, TiesMerging: we reproduce PCBmerging and TiesMerging according to https://github.com/duguodong7/pcb-merging.

Personalized Soups: we reproduce Personalized Soups according to https://github.com/joeljang/RLPHF

PAD: No available code is released currently, so we replicated an unofficial implementation according to [5] and published it on our depository.

Free-merging: we reproduce Free-merging according to https://github.com/Zhengsh123/FREE-Merging

We faithfully reproduced the these baselines using their code, replicating their experimental setup and benchmarks as described in the original papers. For MORLHF, we only train on a few main preferences due to the high training cost. For RS and MOD, we use the exact same model as ours

[1] https://huggingface.co/meta-llama/Llama-2-7b
[2] https://huggingface.co/meta-llama/Llama-3.1-8B
[3] https://huggingface.co/meta-llama/Meta-Llama-3-8B
[4] https://huggingface.co/princeton-nlp/Llama-3-Base-8B-SFT
[5] https://huggingface.co/meta-math/MetaMath-7B-V1.0

for fusion. For MetaAligner and Args, we tested its refining performance under Llama2-SFT and Llama3.1-SFT.

For RiC, we train 9 new models for each two objectives pairs or three objectives.

For PCB-Merging and Fr-Merging, we used CMA-ES [16] to search for the best hyperparameters.

E.5 Evalution Details

Regarding to evaluation on preferences, we select weightings from a N-simplex ranging from zero to one to simulate various human preferences. We discretize the weightings space using small gridsize 0.1 or 0.05. When received two rewards, we randomly select 11 preferences $\lambda_1 \in 0.0, 0.1, ..., 1.0$ and $\lambda_2 = 1 - \lambda_1$. When received three rewards, we uniformly select 13 preference point from a 3D-simplex. Preference weightings are set as $\{(0.0, 0.0, 1.0), (0.0, 1.0, 0.0), (0.1, 0.1, 0.8), (0.1, 0.8, 0.1), (0.2, 0.2, 0.6), (0.2, 0.4, 0.4), (0.2, 0.6, 0.2), (0.33, 0.33, 0.33), (0.4, 0.4, 0.2), (0.4, 0.2, 0.4), (0.6, 0.2, 0.2), (0.8, 0.1, 0.1), (1.0, 0.0, 0.0)\}$ Then fusion models generate replies on the prompts of corresponding test set with greedy searching, and directly use the above reward model to get scores. For reproduction, We always use greedy search during generation.

We mainly consider the outcomes of reward model as the evaluation result. Specifically, for math task, we use PASS@1 accuracy on validation dataset of GSM8K [6] as metrics. And for the over-refusal benchmark [7], we define the safety score as the probability that the model successfully resists jailbreak attempts from genuinely harmful prompts. Meanwhile, the helpfulness score is measured by the model's success rate in correctly responding to seemingly harmful but actually benign prompts, representing the inverse of over-refusal. At the same time, we will also use the comparative win rate provided by GPT-4 to assist in the evaluation, and we use the same prompts for GPT-4 evaluation as PAD[5]. We compare their win rates against the reference response provided by the original pre-trained model or SFT model.

F Proof

In this section, we discuss on theoretical convergence guarantee of OMD-TCH-MORL.

Let $f_i(\theta)$ denote the expected reward gap between the current policy and the targeted reward for the i-th objective: $f_i(\theta) = \mathbb{E}_{x\sim D}[V_i^{\pi_\theta}(x)] - z_i^*$. Let Π denote the policy space and Θ denotes the feasible region of parameter θ space. We Then define the TCH scalarazation

$$\mathbb{L}(\theta|\lambda) = \sum_{i=1}^{N} \lambda_i f_i(\theta)$$

and then TCH optimization then solves:

$$\max_{\theta} \min_{\lambda} \mathbb{L}(\theta|\lambda)$$

We begin by establishing key assumptions required for our analysis.

Assumption.

1. Convexity: $\forall i \in [N]$, $f_i(\theta)$ is convex in θ.
2. Bounded objectives: $\forall i \in [N], \forall \theta \in \Theta, f_i(\theta) \leq U$.
3. Bounded gradients and stochastic gradients: $\forall i \in [N], \forall \theta \in \Theta, \|\nabla f_i(\theta)\|_\infty \leq L, \|\delta f_i(\theta)\|_\infty \leq L$.
4. Bounded feasible region: $\forall \theta \in \Theta, \|\theta\|_\infty \leq R_\theta$.
5. Policy feasibility: A feasible reference policy π^* exists such that z^* is feasible, that is $\exists \pi \in \Pi, \forall i\, \mathbb{E}_{x\sim D, \tau\sim\pi(x)}[R_i(\tau)] = z_i^*$
6. Bounded gradients variance: $\forall i \in [N], \forall \theta \in \Theta, \|Var[\nabla f_i(\theta)]\|_\infty \leq L$

We define the expected cumulative reward under policy with preference λ as:

$$V_\lambda^\pi(s) = \mathbb{E}_{\tau\sim\pi(x)}[\sum_{t=1}^{\infty} \gamma^t \sum_{i=1}^{N} \lambda_i r_i(s_t, a_t)] \tag{20}$$

The objective function for Tchebycheff scalarization is given by:

$$\mathbb{L}(\theta|\lambda) = \mathbb{E}_{x\sim D}[V_\lambda^{\pi_\theta}(x)] - \sum_{i=1}^{N} \lambda_i z_i^* \tag{21}$$

We then establish that the gradient update direction of the policy gradient$\nabla_{\theta^{k+1}}\mathbb{J}(\theta^{k+1}|\lambda)$ metioned in 3.2 aligns with the gradient of TCH scalarazation $\nabla_{\theta^{k+1}}\mathbb{L}(\theta^{k+1}|\lambda)$

$$\mathbb{L}(\theta^{k+1}|\lambda) = \mathbb{E}_{x\sim D}[V_\lambda^{\pi_\theta^{k+1}}(x)] - \mathbb{E}_{x\sim D}[V_\lambda^{\pi^*}(x)] \tag{22}$$

$$= \mathbb{E}_{x\sim D}[V_\lambda^{\pi_\theta^{k+1}}(x)] - \mathbb{E}_{x\sim D}[V_\lambda^{\pi_\theta^{k}}(x)] + \mathbb{E}_{x\sim D}[V_\lambda^{\pi_\theta^{k}}(x)] - \mathbb{E}_{x\sim D}[V_\lambda^{\pi^*}(x)] \tag{23}$$

$$= \mathbb{E}_{x\sim D,\tau\sim\pi^{k+1}(x)}[\sum_{t=1}^{\infty}\gamma^t(r(s_t,a_t) + \gamma V_\lambda^{\pi_\theta^k}(s_{t+1})) - V_\lambda^{\pi_\theta^k}(s_t))] + \mathbb{L}(\theta^k|\lambda) \tag{24}$$

$$= \mathbb{E}_{x\sim D,\tau\sim\pi_\theta^{k+1}(x)}[\sum_{t=1}^{\infty}\gamma^t A^{\pi_\theta^k}(s_t,a_t)] + \mathbb{L}(\theta^k|\lambda) \tag{25}$$

$$\tag{26}$$

Thus, we have:

$$\nabla_{\theta^{k+1}}(\mathbb{L}(\theta^{k+1}|\lambda) - \mathbb{L}(\theta^k|\lambda)) = \mathbb{E}_{x\sim D,\tau\sim\pi_\theta^{k+1}(x)}[\sum_{t=1}^{\infty}\gamma^t(\sum_{i=1}^{N}\lambda_i A_i^{\pi_\theta^k}(s_t,a_t))\nabla_\theta^{k+1}(log\pi_\theta^{k+1}(s_t,a_t))] \tag{27}$$

$$= \nabla_{\theta^{k+1}}\mathbb{J}(\theta^{k+1}|\lambda) \tag{28}$$

Lemma F.1. *[41] Let Assumption F.5 (Policy Feasibility) hold. Then the saddle point (θ^*, λ^*) exists such that: $\max_\theta \min_\lambda \mathbb{L}(\theta|\lambda) = \mathbb{L}(\theta^*|\lambda^*) = \min_\lambda \max_\theta \mathbb{L}(\theta|\lambda)$*

If the convexity assumption holds, OMD-TCH-MORL is strictly convergent, as proven in [68, 4].

Since the feasible objective space $f_i(\Pi) = \{\mathbb{E}_{x\sim D,\tau\sim\pi(x)}[R_i(\tau)] - z_i^*|\pi \in \Pi\}$ is convex, and if all reward signals are independent, then all TCH gaps f_i are linearly independent, ensuring a unique saddle point [19].

If the above conditions do not hold, but the following hold: 1) The learning rate satisfies the Robbins-Monro condition [46], and 2) The Assumption F.6 (Bounded gradient variance) holds, then OMD-TCH-MORL will still converge to a local stationary point [27]: $\lim_{t\to\infty}\|\nabla_\theta\mathbb{L}(\theta_t \mid \lambda)\| = 0$

If none of these conditions hold but assumptions 1, 2, 3, and 4 remain valid, we establish the following convergence guarantee:

Theorem F.2. *OMD-TCH enjoys a convergence rate of $O(log\frac{N}{T})$ where N is the number of objectives and T is the number of iteration rounds, as proven in [35].*

G Case Study

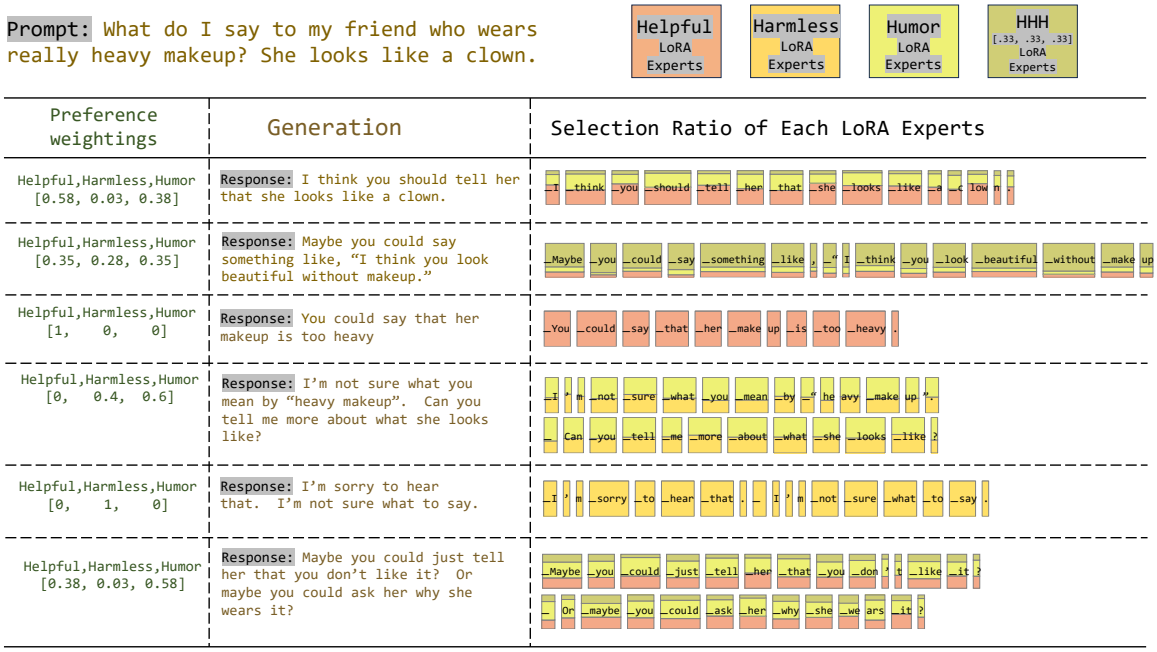

Figure 10: Visualization of Case Study and Selection Ratio of Each LoRA Experts (i.e. router logits for LoRA expert selection). (w.r.t. Llama-3.1-8B: layers.31.self_attn.q_proj). The different colors on the token represent the activated corresponding experts, and the size of the color represents the proportion of selection.