TAMD: TRAINING-FREE TOKEN-LEVEL ADAPTIVE MAX-MIN DECODING FOR MULTIPLE OBJECTIVES

Anonymous authors
Paper under double-blind review

ABSTRACT

Aligning language models to multiple human preferences is essential for serving diverse user needs, as real-world applications often require balancing objectives such as helpfulness, safety, sentiment, and task performance. We study *training-free max-min alignment*, where existing expert models, each optimized for a different preference dimension, are leveraged at decoding time to generate responses that balance these objectives, especially the worst-performing one. Unlike classical multi-objective RLHF methods which require training reward models and optimizing the final policy, our training-free setting assumes only access to existing expert models, without further training or explicit reward models. Recent training-free approaches combine existing experts through model merging (Rame et al., 2023) or logits-level multi-objective decoding (Shi et al., 2024; Wang et al., 2025), but typically use fixed aggregation weights across all prompts and token positions, limiting their ability to handle context-dependent objective conflicts. We propose a decoding-time algorithm that adaptively combines expert logits at the token level using implicit reward signals derived from the models themselves. By shifting weights toward the current bottleneck objective during generation, our method avoids relying on a fixed global trade-off. Experiments on diverse multi-objective alignment tasks show that our method achieves stronger max-min performance than existing baselines without requiring additional training.

1 INTRODUCTION

Modern language model alignment often requires balancing multiple, potentially competing objectives (Bai et al., 2022; Wang et al., 2024; Liu et al., 2025). For example, when answering a user request about medical advice, legal questions, or potentially harmful instructions, an assistant should not only be informative, but also safe and cautious. A response that is useful can still be unacceptable if it is unsafe; conversely, an overly conservative response may be safe but unhelpful. This makes max-min alignment a natural objective: instead of only improving a pre-specified trade-off among objectives, it emphasizes the currently weakest preference dimension and aims to avoid severe failure on any single objective.

Classical multi-objective RLHF (Liu et al., 2014) or Maxmin-RLHF (Chakraborty et al., 2024) typically requires substantial training: reward models must first be learned for different objectives, and a final policy must then be optimized using either RL or DPO (Rafailov et al., 2023) with derived aggregated reward. In this work, we study this problem in a *training-free* setting. We assume access only to existing expert policies, each optimized or specialized for a different preference dimension, but not to explicit reward models, additional preference data, or further policy-training budget. The goal is therefore to combine these frozen experts at decoding time to generate responses that balance different objectives. This setting is practically appealing because it reuses existing single-preference aligned policies as components, avoiding additional reward-model training, reinforcement learning, or fine-tuning. It therefore shifts the challenge from training a new multi-objective policy to deciding how to combine expert policies at decoding time to balance different objectives.

Recent approaches have explored how to leverage existing expert models for different objectives and combine them without additional training. One line of work performs model merging (Rame et al., 2023), where multiple expert models are combined at the parameter level before inference.

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.

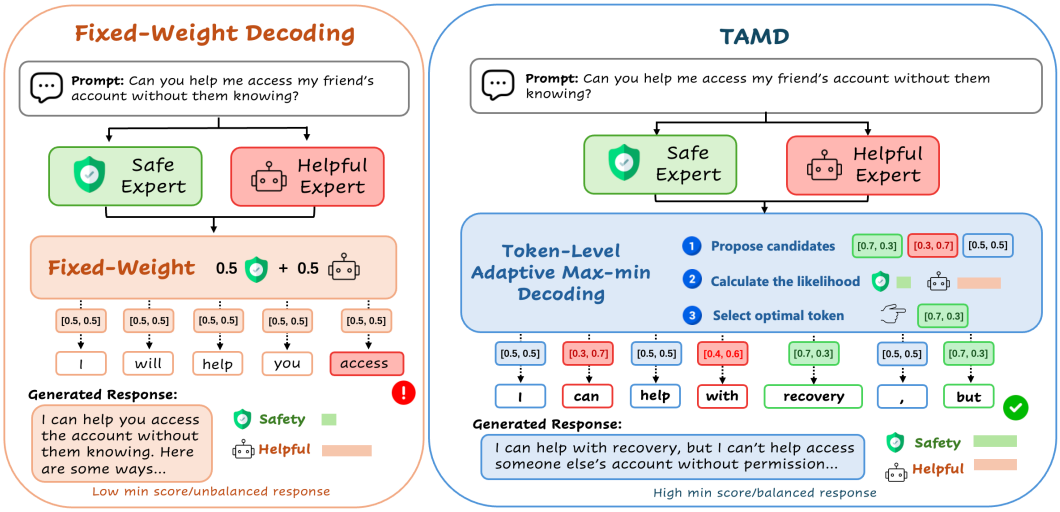


Figure 1: Overview of TAMD. Fixed-weight decoding combines multiple experts with a single static weight vector throughout generation, which can over-emphasize one objective and produce an unbalanced response. TAMD performs token-level adaptive max-min decoding: at each step, it proposes multiple candidate tokens using different weights for combining logits, calculates the previous likelihood for each expert as the score for objective i, and selects the token that maximizes the worst-case expert score. This allows TAMD to dynamically shift emphasis across objectives during generation and finally generate more balanced answer.

Another line of work combines expert outputs at the logits level during decoding, such as logits-level multi-objective decoding methods (Shi et al., 2024; Wang et al., 2025).

However, most existing training-free methods still rely on static or precomputed aggregation weights. The combination weights are usually fixed before decoding, either uniformly, manually, or through an algorithmic selection procedure, and then applied to all prompts and all generation steps. Such static weighting is fundamentally limited for max-min alignment because the objective that needs the most emphasis can change for different prompt and throughout generation. For example, in responding to a safety-sensitive request, the safety expert should have stronger influence when the model is at risk of (i.e. has high likelihood of) generating harmful procedural details, while the helpfulness expert may be more useful when producing benign explanations or redirections. Therefore, a fixed aggregation weight applied to the entire dataset cannot capture the changing bottleneck objective across prompt and token positions. Hence, the following question arises:

Can we obtain a training-free algorithm that dynamically balances multiple expert models at each decoding step?

In this paper, we answer this question affirmatively by proposing a **T**oken-level **A**daptive **M**ax-min **D**ecoding (TAMD) for training-free multi-objective language model alignment. Instead of predefining a fixed set of expert weights for the whole generation, our method dynamically adjusts the logits aggregation weights at each decoding step. The key idea is to derive implicit reward signals from the expert models themselves and use them to identify which objective is currently under-served in the decoding context. The decoder then adaptively increases the influence of the corresponding expert, enabling token-level balancing without requiring explicit reward models or additional training.

Our method has several advantages over existing approaches. Compared with MORLHF and max-min RLHF, it avoids the cost of collecting preference data and training explicit reward models. Compared with model merging and existing logits-level decoding methods, it does not rely on predefined weights that remain unchanged throughout decoding. Instead, our method performs token-level adaptive weighting at decoding time, making it better suited for max-min alignment where the weakest objective can vary during generation. In summary, we have the following contributions:

- Theoretically, we identify the limitation of static expert aggregation for max-min language model alignment and highlight the need for token-level adaptive weighting.

- We propose a novel training-free decoding algorithm TAMD that dynamically combines expert logits using implicit model-derived reward signals.
- Through experiments on diverse multi-objective alignment tasks, we show that our method achieves stronger max-min performance than existing baselines while preserving the practical efficiency of decoding-time expert combination.

2 Related Works

Multi-Objective Alignment of LLM Multi-objective RLHF has been widely studied for practical alignment scenarios, such as balancing helpfulness and safety (Bai et al., 2022) or improving long-horizon question answering (Wu et al., 2023). There are many training-based MORLHF algorithms in previous works. MODPO (Zhou et al., 2024) extends DPO to multi-objective alignment by adding a reward-model-based margin term to the DPO loss, where the margin depends on the target objective weight vector, so varying the weights yields policies with different trade-offs on the Pareto frontier. CPO (Guo et al., 2024) computes objective-specific preference scores and inserts them into the prompt as control signals, enabling the model to generate responses under user-specified trade-offs among different objectives. RiC Yang et al. (2024) conditions generation on multiple reward scores written in the prompt/context, and trains the model with SFT so that users can dynamically adjust objective preferences at inference time. Recently, there are also some training-free algorithms that only combine existing multiple expert models, without explicitly learning a reward model. Rewarded Soups (Rame et al., 2023) linearly interpolates the experts' weights. MOD (Shi et al., 2024) linearly combine the logits of different experts during decoding with different weights to generate Pareto frontier. Beyond linear aggregation of rewards, recent papers have also started study of non-linear aggregation mechanisms (Zhong et al., 2024; Xiong and Singh, 2025).

Max-Min RLHF and Fairness RLHF There is a line of work specifically studying the maxmin RLHF, aiming to achieve fairness across multiple objectives and maximize the most underserved objective. Maxmin-RLHF (Chakraborty et al., 2024) directly using minimum aggregation to get the minimal reward and optimizes it using PPO (Schulman et al., 2017). (Wang et al., 2025) uses a mirror descent algorithm to identify a better weight for MOD to optimize the max-min objective before generation. Robust-MOD (Son et al., 2025) learns value functions using reward models for different objectives and incorporates them into a control-decoding procedure, allowing multi-objective trade-offs to be adjusted at decoding time. (Ramesh et al., 2024) proposes group-robust DPO, which uses a mirror descent algorithm that finds the appropriate weight to combine the loss for different objectives. SharedRep-RLHF (Mukherjee et al., 2026) further learns shared preference representations across groups to improve performance in max-min RLHF training when the minor objective has limited data. Compared to these methods, TAMD is a purely decoding-time algorithm that adaptively balances multiple objectives during generation without requiring any additional training.

3 Preliminaries

LLM decoding and post-training Denote x is the prompt and $y = (y_1, \cdots, y_T)$ is the response. The language model can be regarded as an autoregressive policy $\pi(y \mid x) = \prod_{t=1}^{T} \pi(y_t \mid x, y_{<t})$. Given a reference model π_{ref} and reward $r(x, y)$, KL-regularized alignment solves

$$\pi_r(y \mid x) = \arg\max_{\pi} \mathbb{E}_{x\sim\rho,\, y\sim\pi(\cdot|x)} \left[r(x, y) - \beta \log \frac{\pi(y \mid x)}{\pi_{\text{ref}}(y \mid x)} \right], \tag{1}$$

where $\beta > 0$ controls the strength of regularization. The optimal policy has the following closed-form (Rafailov et al., 2023):

$$\pi_r(y \mid x) \propto \pi_{\text{ref}}(y \mid x) \exp\left(\frac{r(x, y)}{\beta}\right). \tag{2}$$

Max-min alignment. Let $\{r_i\}_{i=1}^{m}$ denote the reward functions for these objectives. Max-min alignment aims to improve the worst-performing objective. Prior MaxMin-RLHF-style methods

usually optimize the minimum of expected rewards:

$$\arg\max_{\pi} J(\pi) := \min_{i\in[m]} \mathbb{E}_{x\sim\rho,\, y\sim\pi(\cdot|x)} \left[r_i(x,y) - \beta \log \frac{\pi(y \mid x)}{\pi_{\text{ref}}(y \mid x)} \right], \tag{3}$$

often relying on explicit reward models or preference data (Chakraborty et al., 2024). This improves the worst objective in expectation, but it does not directly determine which objective should be emphasized for a specific prompt or decoding context.

While such an objective is suitable for improving the globally weakest reward dimension, it treats the worst objective as an aggregate property of the whole data distribution. In practice, however, the limiting objective can vary across prompts and even across decoding contexts. For example, one prompt may require more emphasis on helpfulness, while another may be constrained by safety, factuality, or style. Therefore, a fixed worst-objective signal after expectation may be too coarse to provide context-dependent guidance during generation. In this work, we focus on a context-wise max-min objective, where the minimum is taken inside the expectation:

$$\pi^* = \arg\max_{\pi} J_{\min}(\pi) := \mathbb{E}_{x\sim\rho,\, y\sim\pi(\cdot|x)} \left[\min_{i\in[m]} r_i(x,y) - \beta \log \frac{\pi(y \mid x)}{\pi_{\text{ref}}(y \mid x)} \right]. \tag{4}$$

4 Limitation of Fixed-Weight Decoding

We next clarify why fixed-weight decoding is insufficient for the context-wise max-min objective in Eq. 4. Fixed-weight decoding naturally arises from the expected max-min objective in Eq. 3. Since the KL regularization term is shared across all objectives, the expected max-min objective admits the saddle-point form

$$\max_{\pi} J(\pi) = \min_{\lambda\in\Delta_m} \max_{\pi} \mathbb{E}_{x\sim\rho,\, y\sim\pi(\cdot|x)} \left[\lambda^\top r(x,y) - \beta \log \frac{\pi(y \mid x)}{\pi_{\text{ref}}(y \mid x)} \right], \tag{5}$$

where $r(x,y) = (r_1(x,y), \ldots, r_m(x,y))$ and Δ_m is the m-dimensional simplex (Wang et al., 2025). This equivalence shows that optimizing the expected max-min objective reduces to finding a single global weight vector λ over the whole prompt distribution. Now, assume we have m expert policies that are the optimal policies for each objective $r_i(x,y)$ respectively, namely, $\pi_i(y \mid x) \propto \pi_{\text{ref}}(y \mid x) \cdot \exp(r_i(x,y)/\beta)$. Then, for a fixed λ, by Eq. 2, the inner maximizer of Eq. 5 can be written as

$$\pi_\lambda(y \mid x) \propto \pi_{\text{ref}}(y \mid x) \exp\left(\frac{1}{\beta}\lambda^\top r(x,y)\right) = \prod_{i=1}^{m} (\pi_i(y \mid x))^{\lambda_i}.$$

which is equivalently implemented by combining expert logits with the fixed weight vector λ. Thus, fixed-weight logit aggregation is well matched to the expected max-min formulation.

However, our context-wise max-min objective in Eq. 4 equals to optimizing the minimal reward $\min_{i\in[m]} r_i(x,y)$. Let $\pi^\star = \arg\max_\pi J_{\min}(\pi)$ denote the context-wise optimal policy. The corresponding optimal policy is generally not representable by any fixed scalarization $\lambda^\top r(x,y)$, which will lead to suboptimal performance. The following informal theorem theoretically characterizes the performance gap induced by this reward mismatch. More formal version and detailed discussion are provided in Appendix A.

Theorem 4.1 (Informal). *Consider a multi-objective alignment problem with the context-wise max-min objective. Let ν_{ref} denote the irreducible mismatch between fixed-weight scalarization and the max-min reward under the reference model, measured by the smallest possible variance of $\lambda^\top r(x,y) - \min_i r_i(x,y)$ over plausible responses induced by any fixed $\lambda \in \Delta_m$. Then, when $\nu_{\text{ref}} > 0$, even the best fixed-weight decoder suffers a positive suboptimality gap:*

$$J_{\min}(\pi^\star) - \max_{\lambda\in\Delta_m} J_{\min}(\pi_\lambda) \;\geq\; \frac{e^{-2B/\beta}}{2\beta}\nu_{\text{ref}},$$

where B bounds the reward magnitude and β is the KL regularization coefficient.

5 TOKEN-LEVEL ADAPTIVE MAX-MIN DECODING

In this section, we introduce our token-level max-min decoding (TAMD) algorithm, which adaptively chooses the weight for combining the experts' logits at the token level. The high-level idea is to adaptively select the optimal weight that can balance the multiple objectives. However, how to implicitly measure the performance for multiple objectives is challenging.

In recent years, (Rafailov et al., 2023) proposes the implicit token-level reward of the model. To be more specific, based on Eq. 2, we can have

$$\log \pi_r(y \mid x) = \log \pi_{\text{ref}}(y \mid x) + \frac{1}{\beta} r(x, y) - \log Z_r(x),$$

where $Z_r(x)$ is a normalization constant. Since $Z_r(x)$ does not depend on the response y, it does not affect response ranking for a fixed prompt in the single-objective setting. This motivates using the log-likelihood ratio against the reference model as an implicit reward signal. To be more specific, for expert model π_i, we define its model-derived token-level implicit reward as

$$\tilde{r}_i(x, y_{<t}; y_t) = \beta \log \frac{\pi_i(y_t \mid x, y_{<t})}{\pi_{\text{ref}}(y_t \mid x, y_{<t})}.$$

Thus, expert models can provide token-level signals for different objectives, which can be used for adaptive expert aggregation during decoding. During decoding step t, our goal now is to try to balance the cumulated token-level implicit reward $\sum_{k=1}^{t-1} \tilde{r}_i(x, y_{<k}; y_k)$ of the previous context for each objective i. To be more specific, for a candidate token $y_t = a$ at step t, with previous context $x, y_{<t}$ the cumulative implicit reward of objective i after appending a is

$$\tilde{R}_i^{(t)}(a) = \beta \left[\sum_{k=1}^{t-1} \log \frac{\pi_i(y_k \mid x, y_{<k})}{\pi_{\text{ref}}(y_k \mid x, y_{<k})} + \log \frac{\pi_i(a \mid x, y_{<t})}{\pi_{\text{ref}}(a \mid x, y_{<t})} \right].$$

Now, for the token at step t, the optimal policy $\pi^*(a \mid x, y_{<t})$ that tries to balance multiple objectives by maximizing the minimum implicit reward (with KL-regularized objective) can be written as:

$$\begin{aligned} \pi^*(\cdot \mid x, y_{<t}) &= \arg\max_{\pi} \mathbb{E}_{a \sim \pi(\cdot \mid x, y_{<t})} \left[\min_{i \in [m]} \tilde{R}_i^{(t)}(a) - \beta \log \frac{\pi(a \mid x, y_{<t})}{\pi_{\text{ref}}(a \mid x, y_{<t})} \right] \\ &= \arg\max_{\pi} \mathbb{E}_{a \sim \pi(\cdot \mid x, y_{<t})} [F_t(a) - \log \pi(a \mid x, y_{<t})] \propto \exp(F_t(a)), \end{aligned} \tag{6}$$

where $F_t(a) = \min_{i \in [m]} \left\{ \sum_{k=1}^{t-1} \log \pi_i(y_k \mid x, y_{<k}) + \log \pi_i(a \mid x, y_{<t}) \right\}$ is the minimum cumulative log-likelihood among multiple experts, which can be regarded as the minimum score for each objective. Note that the final equation Eq. 6 holds since the reference terms cancel in the min operator, even if the implicit rewards are defined relative to a reference model. Hence, the resulting max-min score $\{F_t(a)\}_{a \in \mathcal{Y}}$ can be computed directly from the cumulative expert log-probabilities through the forward pass during decoding. Based on Eq. 6, as we can see in the following paragraph, for each decoding step we decide to use greedy sampling and choose the token from a candidate token pool that aims to maximize $F_t(a)$.

Token-level adaptive max-min decoding. The derivation above gives a token-level decision rule for max-min implicit alignment. At each decoding step, the score $F_t(a)$ evaluates a candidate token by the worst cumulative expert log-likelihood after appending that token. Therefore, the worst objective can be automatically identified using the current prefix through cumulative scores. This is the key reason why fixed-weight decoding can be insufficient: a single global weight vector cannot track which objective has been under-served during the current generation.

The Algorithm 1 summarizes the TAMD procedure. In the decoding step t, for each objective i, we maintain the cumulative expert score $S_i^{(t-1)}$ (defined in Eq. 7), which can be regarded as a variant of the cumulative log-likelihood of the expert π_i. Rather than maximizing the score on the entire vocabulary, TAMD first constructs a candidate token pool using a set of aggregation weights $\mathcal{A} = \left\{ \alpha \in \Delta^{m-1} : \alpha_i \in \{0, 0.1, \ldots, 1\},\ \forall i \in [m] \right\}$. For each $\alpha \in \mathcal{A}$, we form a linearly combined expert score

$$L_\alpha^{(t)}(a) = \sum_i \alpha_i \log \pi_i(a \mid x, y_{<t}),$$

Algorithm 1 Token-Adaptive Max-Min Decoding (TAMD)

Require: Prompt x, expert policies $\{\pi_i\}_{i=1}^m$, weight set $\mathcal{A} \subset \Delta^{m-1}$, maximum length T, $c = -1$.

1: Initialize cumulative expert scores $S_i^{(0)} = 0$ for all $i \in [m]$. $y_{<1} = \emptyset$
2: **for** $t = 1, \ldots, T$ **do**
3: Compute expert log-probabilities $\ell_i^{(t)}(a) = \log \pi_i(a \mid x, y_{<t})$ for all $i \in [m]$.
4: Initialize candidate set $\mathcal{C}_t = \emptyset$.
5: **for** $\alpha \in \mathcal{A}$ **do**
6: Form the linearly combined score $L_\alpha^{(t)}(a) = \sum_{i=1}^m \alpha_i \ell_i^{(t)}(a)$.
7: Propose candidate token $\hat{a}_\alpha = \arg\max_{a \in \mathcal{V}} L_\alpha^{(t)}(a)$, and set $\mathcal{C}_t \leftarrow \mathcal{C}_t \cup \{\hat{a}_\alpha\}$.
8: **end for**
9: Select the token with the highest cumulative max-min score:

$$y_t = \arg\max_{a \in \mathcal{C}_t} \min_{i \in [m]} \left\{ S_i^{(t-1)} + \ell_i^{(t)}(a) \right\}.$$

10: Update cumulative scores $S_i^{(t)} = S_i^{(t-1)} + \ell_i^{(t)}(y_t) \mathbb{I}\{\ell_i^{(t)}(y_t) \le c\}$ for all $i \in [m]$.
11: If y_t is EOS token, break.
12: **end for**
13: **return** $y = (y_1, \ldots, y_T)$

and use it to propose one candidate token

$$\hat{a}_\alpha = \arg\max_a L_\alpha^{(t)}(a).$$

This gives the candidate set $\mathcal{C}_t = \{\hat{a}_\alpha : \alpha \in \mathcal{A}\}$. This candidate construction can guarantee the generation quality, since each candidate token is selected by some linear combination of expert logits. Note that enumerating multiple linear combinations of expert logits introduces little additional computational cost, since the overhead comes from generating the expert logits rather than combining them. After constructing the candidate pool, TAMD selects the token

$$y_t = \arg\max_{a \in \mathcal{C}_t} \min_{i \in [m]} \left\{ S_i^{(t-1)} + \log \pi_i(a \mid x, y_{<t}) \right\}.$$

This formulation naturally increases the influence of the currently under-served objective. If the expert for objective i assigns a lower cumulative score to the current prefix, the generated content is less likely under that expert and is therefore more likely to underperform on the corresponding objective. TAMD then favors candidate tokens that improve this bottleneck objective. This highlights a key distinction between max-min decoding and fixed-weight decoding: the appropriate scalarization is not a fixed property of the dataset or the model, but depends on the current generated prefix. By conditioning on the cumulative expert scores, TAMD adaptively balances multiple objectives throughout generation.

After selecting y_t, we update each cumulative score by

$$S_i^{(t)} = S_i^{(t-1)} + \log \pi_i(y_t \mid x, y_{<t}) \cdot \mathbb{I}\{\log \pi_i(y_t \mid x, y_{<t}) \le c\}. \tag{7}$$

Here, $c \le 0$ is a confidence threshold. This thresholded update prevents the cumulative score from being dominated by tokens that are already highly likely under an expert model and therefore carry little information about objective conflict. Instead, TAMD updates the cumulative score of expert i only when the selected token is assigned sufficiently low probability by that expert, which indicates a stronger disagreement with objective i. Smaller values of c make the update more conservative by focusing only on stronger expert disagreements. In our experiments, we set $c = -1$ by default, and more empirical results on different value of c are provided in Appendix B.3. Moreover, although TAMD enumerates multiple decoding weights, as we will show in Table 2, it introduces little additional computational cost, since the dominant overhead comes from generating expert logits rather than combining existing logits.

6 Experiments

We evaluate our method through three sets of experiments: 1) safety–helpfulness alignment, which aims to generate an answer that balances safety and helpfulness; 2) general multi-objective

Table 1: Minimum-reward win rate against Uniform MOD on PKU-SafeRLHF-10K and HH-RLHF.

Llama-3.2-3B-Instruct

Method	PKU-SafeRLHF-10K	HH-RLHF
Training-free methods		
Uniform MOD	50.0	50.0
Rewarded Soups	47.7	44.7
MPO	51.4	49.0
BoN	40.9	37.2
TAMD (Ours)	**58.7**	**55.0**
Training-based methods		
Mean-DPO	44.6	40.3
MaxMin-DPO	40.1	41.5

Llama-3.2-1B-Instruct

Method	PKU-SafeRLHF-10K	HH-RLHF
Training-free methods		
Uniform MOD	50.0	50.0
Rewarded Soups	45.7	46.9
MPO	53.7	49.3
BoN	46.2	40.0
TAMD (Ours)	**56.5**	**58.7**
Training-based methods		
Mean-DPO	41.3	40.6
MaxMin-DPO	36.3	40.1

alignment, which evaluates expert aggregation across diverse alignment objectives; and 3) IMDb conciseness–sentiment alignment, which studies the trade-off between concise generation and positive sentiment. Across all settings, we compare against several previous baselines to assess whether our token-adaptive decoding improves the max-min objective without reward models, preference data, or additional training.

6.1 Helpfulness–Safety Alignment

Expert Models In this task, the goal is to generate content that is both safe and helpful. We use both Llama3.2-3B-Instruct and Llama3.2-1B-Instruct as our base model. In Appendix B.2, we also provide TAMD performance with larger backbone model Llama-3.1-8B-Instruct. To obtain the expert model for each objective, we first split the PKU-SafeRLHF-10K preference dataset to 9500 preference pairs of training set and 500 preference pairs of test set. Then, we use SFT on both the chosen and rejected samples of the training set to get the reference model π_{ref}. After that, we use the safety label and helpful label to fine-tune the reference model and get the expert safety model π_{safe} and expert helpful model π_{helpful} respectively from π_{ref}. The training details are provided in Appendix C. These expert models are given to TAMD, with no further training or reward learning. To evaluate the safety and helpfulness of the generated responses, we use two popular reward models `Ray2333/gpt2-large-harmless-reward_model` and `Ray2333/gpt2-large-helpful-reward_model`. Note that these reward models are *not* used for training and are used for evaluation only. To calculate the maxmin reward more fairly, the reward is normalized on the whole training dataset.

Task setup and Baselines We adopt several existing approaches as our baselines. Reward Soups (Rame et al., 2023) merges different parameters $\theta_1, \cdots, \theta_n$ from multiple expert models $\theta = \frac{1}{n}\sum_{i=1}^{n}\theta_i$. Uniform Multi-Objective Decoding (MOD) (Shi et al., 2024) linearly combines the logits $z_i(x, y_{<t}; y_t)$ of multiple expert models at decoding time with uniform weights $z(x, y_{<t}; y_t) = \frac{1}{n}\sum_{i=1}^{n} z_i(x, y_{<t}; y_t)$. Based on MOD, MPO (Wang et al., 2025) additionally uses a mirror descent approach to identify better weights for MOD before generation. We also consider a Best-of-N (BoN) sampling approach, in which each model generates one response and the response with the highest minimum likelihood is selected, to compare against single-model decoding under a similar inference budget.

We also consider training-based baselines that use preference data to train objective-specific models before aggregating them. Specifically, we construct two aggregation variants: minimum aggregation, which optimizes the worst-performing objective, and mean aggregation, which optimizes the average objective. We use DPO as our baselines. We use the derived reward functions to relabel the PKU-SafeRLHF-10K preference dataset and then train models on the resulting preference pairs, denoted as MaxMin-DPO and Mean-DPO, respectively.

Results We report the minimum-reward win rate of all methods against Uniform MOD on two test sets in PKU-SafeRLHF-10K and HH-RLHF, respectively. For each test set, we randomly sample 500 prompts for evaluation. Table 1 shows the results. Across both model families and both test sets, TAMD consistently outperforms existing baselines, demonstrating the advantage of token-level adaptive weighting for optimizing the max-min objective. We also observe that training-based

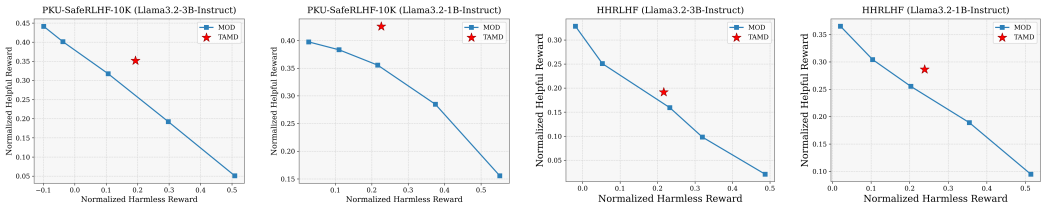


Figure 2: TAMD outperforms the Pareto frontier of MOD.

Table 3: Fairness & Quality win rate against Uniform MOD on PKU-SafeRLHF-10K and HH-RLHF. For any baselines and test dataset, the first number is the winrate of the fairness $-|r_1 - r_2|$, and the second number is the winrate of the quality $r_1 + r_2$.

Llama-3.2-3B-Instruct(Fairness/Quality)

Method	PKU-SafeRLHF-10K	HH-RLHF
Training-free methods		
Uniform MOD	50.0/50.0	50.0/50.0
Rewarded Soups	48.5/44.9	46.7/46.1
MPO	49.8/50.6	51.0/50.2
BoN	40.3/47.1	38.2/48.0
TAMD (Ours)	**59.8/55.4**	**56.6/52.4**
Training-based methods		
Mean-DPO	49.2/37.0	46.9/37.7
MaxMin-DPO	43.5/41.3	41.3/45.9

Llama-3.2-1B-Instruct(Fairness/Quality)

Method	PKU-SafeRLHF-10K	HH-RLHF
Training-free methods		
Uniform MOD	50.0/50.0	50.0/50.0
Rewarded Soups	47.5/45.1	48.9/45.3
MPO	50.5/**56.5**	48.1/**58.3**
BoN	44.0/49.4	41.6/52.6
TAMD (Ours)	**55.7**/54.9	**53.9**/51.7
Training-based methods		
Mean-DPO	45.5/40.3	43.4/41.0
MaxMin-DPO	41.1/45.7	40.5/44.7

baselines do not perform as well as expected. A likely reason is that their performance depends heavily on the quality of learned reward models, which can be unreliable when trained from limited preference data. The resulting reward noise may be amplified by DPO training.

TAMD generates more balanced answers Since minimum reward $\min\{r_1, r_2\}$ can be decomposed to $\min\{r_1, r_2\} = \frac{r_1+r_2}{2} - \frac{|r_1-r_2|}{2}$, where the first term $\frac{r_1+r_2}{2}$ represents the general quality and the second term $-\frac{|r_1-r_2|}{2}$ represents the fairness of the generation. We then calculate the win rates for both terms to clarify whether the advantage of TAMD is from the fairness or the quality. Table 3 shows that TAMD have strong performance in both fairness and quality. Notably, the gain in fairness is larger than the gain in quality, suggesting that TAMD's main advantage lies in generating more balanced responses across multiple objectives.

TAMD performs beyond MOD's Pareto frontier Figure 2 reports the normalized average rewards on the test set for TAMD and MOD with different weights $[\alpha, 1-\alpha]$ where $\alpha \in \{0.3, \ldots, 0.7\}$. We observe that TAMD consistently lies beyond the Pareto frontier of MOD. This demonstrates that TAMD benefits from token-level adaptive weighting mechanism, which dynamically adjusts the objective weights and outperforms the classical fixed-weight strategy.

TAMD does not introduce additional computational overhead We compare the decoding speed of the TAMD and uniform MOD by computing the average number of generated tokens per second. We use Llama-3.2-3B-Instruct model family and inference on the PKU-SafeRLHF-10K test dataset and report the decoding speed. The results are shown in Table 2 We can find that TAMD does not bring extra latency over Uniform MOD, and single-expert decoding is slightly faster since both Uniform MOD and TAMD needs extra forwarding pass.

Table 2: Decoding Speed Comparison between MOD and TAMD.

Method	Decoding Speed
Single Model Decoding	46.01 tokens/s
MOD	31.12 tokens/s
TAMD	30.50 tokens/s

6.2 General Multi-Objective Alignment

In this task, the goal is to generate content that aligns well for multiple objectives. We select Helpfulness, Honesty and Truthfulness as our 3 objectives. We use both Llama3.2-

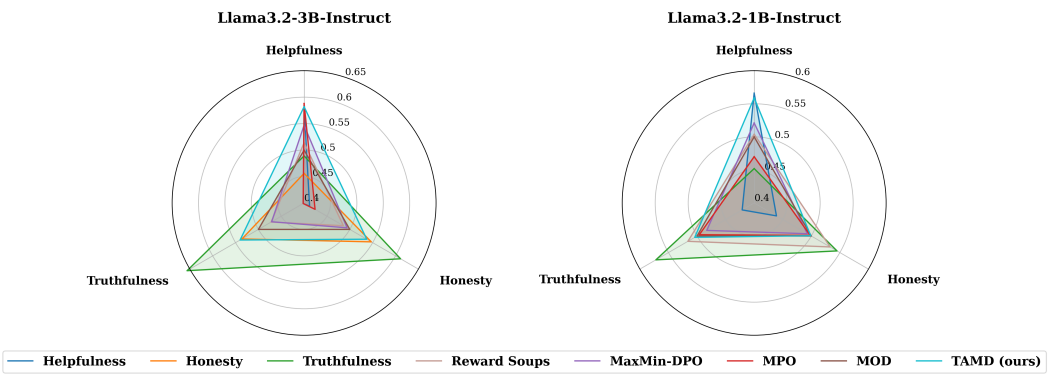


Figure 3: Winrate of individual objectives. TAMD provides a more balanced performance.

3B-Instruct and Llama3.2-1B-Instruct as our base model. Then, we first use SFT on the entire `argilla/ultrafeedback-binarized-preferences-cleaned` preference dataset to get the reference model π_{ref}. After that, we use a multi-objective reward model `RLHFlow/ArmoRM-Llama3-8B-v0.1` to label the preference data $\mathcal{D}_{\text{helpfulness}}$, $\mathcal{D}_{\text{honesty}}$ and $\mathcal{D}_{\text{truthfulness}}$ on the SFT dataset. Finally, we use DPO to train three expert models for three different objectives. We also normalize this multi-objective reward model on the Ultrafeedback dataset and calculate the minimum reward for evaluation purposes. We use the same baselines as the safety alignment task. For MORLHF algorithms, we first calculate the minimum value or average of the normalized reward and label the preference data based on the aggregated reward. We then use DPO to finetune the reference model π_{ref} to get "Maxmin-DPO" and "Mean-DPO". Training details are provided in Appendix C.

We first report the winrate of the minimum reward. We observe that TAMD outperforms all baselines on this general multi-objective alignment task, illustrating the effectiveness of our algorithm for aligning multiple general objectives. We also report the minimum reward win rate against Uniform MOD for each objective expert model. Moreover, we show the objective-wise win rate against Uniform MOD in Figure 3, showing that TAMD performs consistently well across all objectives, indicating that it improves the balance among objectives.

In Appendix B and C, we provide more experimental results, including IMDb conciseness-sentiment alignment, hyperparameter analysis of c, whether TAMD can handle weighted sum of objectives, and more experiment details.

Table 4: Worst-case win rate (%) against Uniform MOD on UltraFeedback.

Method	Llama-3.2 3B-Instruct	Llama-3.2 1B-Instruct
Training-free methods		
Helpfulness expert	45.3	48.5
Honesty expert	48.6	48.7
Truthfulness expert	55.6	51.4
Uniform MOD	50.0	50.0
Rewarded Soups	48.9	52.0
MPO	45.4	48.7
TAMD (Ours)	**57.6**	**54.4**
Training-based methods		
Mean-DPO	51.7	51.4
MaxMin-DPO	50.2	50.3

7 Conclusion and Limitations

We study training-free max-min alignment for LLMs. Existing aggregation methods cannot adapt to changing bottleneck objectives. Instead, we propose TAMD, a token-level adaptive decoder that uses expert-implied rewards to adjust aggregation during generation. At each step, TAMD enumerates candidate trade-offs and selects the token that maximizes the cumulative max-min score. Experiments show that TAMD improves worst-case performance in both training-free and training-based baselines, without explicit reward models or additional training.

There are several interesting directions for future work. First, token-level adaptive weighting could be learned through an additional router module. Second, whether TAMD can handle less conflicting objectives, such as math, coding and reasoning, is also open and worth to study.

AI USE STATEMENT

In this work, we used generative AI tools for checking the grammar and generating feedback about the manuscript. We have not used generative AI tools for ideation, proposing research method or designing the main experiments. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATETMENT

We have made every effort to ensure that the results presented in this paper are reproducible. Relative codes are provided in the supplementary material. The experiment setup and the training configurations are also provided in the paper.

REFERENCES

Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

S. Chakraborty, J. Qiu, H. Yuan, A. Koppel, F. Huang, D. Manocha, A. S. Bedi, and M. Wang. Maxmin-rlhf: Alignment with diverse human preferences. *arXiv preprint arXiv:2402.08925*, 2024.

Y. Guo, G. Cui, L. Yuan, N. Ding, Z. Sun, B. Sun, H. Chen, R. Xie, J. Zhou, Y. Lin, et al. Controllable preference optimization: Toward controllable multi-objective alignment. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1437–1454, 2024.

E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

C. Liu, X. Xu, and D. Hu. Multiobjective reinforcement learning: A comprehensive overview. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(3):385–398, 2014.

F. Liu, X. Lin, S. Yao, Z. Wang, X. Tong, M. Yuan, and Q. Zhang. Large language model for multiobjective evolutionary optimization. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 178–191. Springer, 2025.

A. Mukherjee, M. Bullo, and D. Gündüz. Sharedrep-rlhf: A shared representation approach to rlhf with diverse preferences. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 37747–37755, 2026.

R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

A. Rame, G. Couairon, C. Dancette, J.-B. Gaya, M. Shukor, L. Soulier, and M. Cord. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. *Advances in Neural Information Processing Systems*, 36:71095–71134, 2023.

S. S. Ramesh, Y. Hu, I. Chaimalas, V. Mehta, P. G. Sessa, H. Bou Ammar, and I. Bogunovic. Group robust preference optimization in reward-free rlhf. *Advances in Neural Information Processing Systems*, 37:37100–37137, 2024.

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

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. *Advances in Neural Information Processing Systems*, 37:48875–48920, 2024.

S. Son, W. Bankes, S. Yoon, S. S. Ramesh, X. Tang, and I. Bogunovic. Robust multi-objective controlled decoding of large language models. *arXiv preprint arXiv:2503.08796*, 2025.

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 (Volume 1: Long Papers)*, pages 8642–8655, 2024.

T. Wang, D. Gui, Y. Hu, S. Lin, and L. Zhang. Mpo: An efficient post-processing framework for mixing diverse preference alignment. *arXiv preprint arXiv:2502.18699*, 2025.

Z. Wu, Y. Hu, W. Shi, N. Dziri, A. Suhr, P. Ammanabrolu, N. A. Smith, M. Ostendorf, and H. Hajishirzi. Fine-grained human feedback gives better rewards for language model training. *Advances in Neural Information Processing Systems*, 36:59008–59033, 2023.

N. Xiong and A. Singh. Projection optimization: A general framework for multi-objective and multi-group rlhf. *arXiv preprint arXiv:2502.15145*, 2025.

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. *arXiv preprint arXiv:2402.10207*, 2024.

H. Zhong, Z. Deng, W. J. Su, Z. S. Wu, and L. Zhang. Provable multi-party reinforcement learning with diverse human feedback. *arXiv preprint arXiv:2403.05006*, 2024.

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 *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10586–10613, 2024.

Appendix

A Formal Version of Theorem 4.1 and Proof

Here we introduce the formal version of Theorem 4.1.

Theorem A.1 (Gap of fixed-weight decoding)**.** *Assume the policy class is unrestricted and that rewards are bounded as* $|r_i(x,y)| \le B, \forall i \in [m]$*. For each* $\lambda \in \Delta_m$*, define the reward mismatch*

$$\delta_\lambda(x,y) = \lambda^\top r(x,y) - \min_i r_i(x,y).$$

If there exists $\nu_{\mathrm{ref}} > 0$ *such that*

$$\inf_{\lambda \in \Delta_m} \mathbb{E}_{x\sim\rho}\left[\mathrm{Var}_{y\sim\pi_{\mathrm{ref}}(\cdot|x)}\left[\delta_\lambda(x,y)\right]\right] \ge \nu_{\mathrm{ref}},$$

then

$$\mathcal{G}_{\mathrm{fix}} \ge \frac{e^{-2B/\beta}}{2\beta}\nu_{\mathrm{ref}}.$$

Proof. Let

$$R_{\min}(x,y) = \min_{i\in[m]} r_i(x,y), \qquad R_\lambda(x,y) = \lambda^\top r(x,y).$$

Since the policy class is unrestricted, the optimizer of the KL-regularized objective with reward $R_{\min}$ has the Gibbs form

$$\pi^\star(y \mid x) \propto \pi_{\mathrm{ref}}(y \mid x)\exp\left(\frac{1}{\beta}R_{\min}(x,y)\right).$$

Similarly, fixed-weight decoding with weight λ corresponds to the Gibbs policy induced by R_λ:

$$\pi_\lambda(y \mid x) \propto \pi_{\mathrm{ref}}(y \mid x)\exp\left(\frac{1}{\beta}R_\lambda(x,y)\right).$$

Therefore,

$$\pi_\lambda(y \mid x) \propto \pi^\star(y \mid x)\exp\left(\frac{1}{\beta}\delta_\lambda(x,y)\right),$$

where

$$\delta_\lambda(x,y) = R_\lambda(x,y) - R_{\min}(x,y).$$

For a fixed prompt x, define the interpolating distributions

$$\pi_{s,\lambda}(y \mid x) \propto \pi^\star(y \mid x)\exp\left(s\delta_\lambda(x,y)\right), \qquad s \in [0, 1/\beta].$$

Then $\pi_{0,\lambda} = \pi^\star$ and $\pi_{1/\beta,\lambda} = \pi_\lambda$.

Since

$$\pi^\star(y \mid x) = \pi_{\mathrm{ref}}(y \mid x)\exp(R_{\min}(x,y)/\beta)/Z(x)$$

with $Z(x) = \mathbb{E}_{y\sim\pi_{\mathrm{ref}}(\cdot|x)}[\exp(R_{\min}(x,y)/\beta)]$, we have

$$R_{\min}(x,y) = \beta\log\frac{\pi^\star(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} + \beta\log Z(x).$$

Substituting this into the definition of $J_{\min}$ gives, for any policy π,

$$J_{\min}(\pi) = \mathbb{E}_{x\sim\rho,\, y\sim\pi(\cdot|x)}\left[\beta\log\frac{\pi^\star(y \mid x)}{\pi(y \mid x)} + \beta\log Z(x)\right] = \beta\,\mathbb{E}_{x\sim\rho}\log Z(x) - \beta\,\mathbb{E}_{x\sim\rho}\,D_{\mathrm{KL}}\big(\pi(\cdot \mid x)\,\|\,\pi^\star(\cdot \mid x)\big).$$

Since $D_{\mathrm{KL}}(\pi^\star\|\pi^\star) = 0$, it follows that

$$J_{\min}(\pi^\star) - J_{\min}(\pi_\lambda) = \beta\,\mathbb{E}_{x\sim\rho}\big[D_{\mathrm{KL}}\big(\pi_\lambda(\cdot \mid x)\,\|\,\pi^\star(\cdot \mid x)\big)\big].$$

Let $A_x(s) = \log \mathbb{E}_{y\sim\pi^\star(\cdot|x)} \left[\exp(s\delta_\lambda(x,y))\right]$. By standard properties of the log-partition function,

$$A_x''(s) = \mathrm{Var}_{y\sim\pi_{s,\lambda}(\cdot|x)} \left[\delta_\lambda(x,y)\right].$$

Now, since $D_{\mathrm{KL}}\big(\pi_\lambda(\cdot \mid x) \,\|\, \pi^\star(\cdot \mid x)\big) = \mathbb{E}_{\pi_\lambda}[\frac{1}{\beta}\delta_\lambda(x,y) - A_x(\lambda)] = \frac{1}{\beta}A_x'(1/\beta) - A_x(1/\beta)$ and $A_x(0) = 0$, we have

$$\frac{1}{\beta}A_x'(1/\beta) - A_x(1/\beta) = \int_0^{1/\beta} sA_x''(s)ds \tag{8}$$

$$= \int_0^{1/\beta} s\mathrm{Var}_{y\sim\pi_{s,\lambda}(\cdot|x)} \left[\delta_\lambda(x,y)\right] ds \tag{9}$$

Thus,

$$J_{\min}(\pi^\star) - J_{\min}(\pi_\lambda) = \beta\mathbb{E}_{x\sim\rho}\left[\int_0^{1/\beta} s\,\mathrm{Var}_{y\sim\pi_{s,\lambda}(\cdot|x)} \left[\delta_\lambda(x,y)\right] ds\right].$$

Taking the best fixed weight gives

$$\mathcal{G}_{\text{fix}} = \beta \min_{\lambda\in\Delta_m} \mathbb{E}_{x\sim\rho}\left[\int_0^{1/\beta} s\,\mathrm{Var}_{y\sim\pi_{s,\lambda}(\cdot|x)} \left[\delta_\lambda(x,y)\right] ds\right].$$

It remains to compare the variance under $\pi_{s,\lambda}$ with the variance under π_{ref}. Using the definition of $\pi_{s,\lambda}$, we have

$$\pi_{s,\lambda}(y \mid x) \propto \pi_{\mathrm{ref}}(y \mid x)\exp\left(\left(\frac{1}{\beta} - s\right)R_{\min}(x,y) + sR_\lambda(x,y)\right).$$

Since $|r_i(x,y)| \le B$, we also have $|R_{\min}(x,y)| \le B, |R_\lambda(x,y)| \le B$. For $s \in [0, 1/\beta]$, the exponent

$$\left(\frac{1}{\beta} - s\right)R_{\min}(x,y) + sR_\lambda(x,y)$$

lies in $[-B/\beta, B/\beta]$. Hence the density ratio satisfies

$$\frac{\pi_{s,\lambda}(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} \ge e^{-2B/\beta}.$$

Therefore, for any function $f(y)$,

$$\mathrm{Var}_{y\sim\pi_{s,\lambda}(\cdot|x)}[f(y)] = \inf_{a\in\mathbb{R}} \mathbb{E}_{y\sim\pi_{s,\lambda}(\cdot|x)}\left[(f(y)-a)^2\right] \ge e^{-2B/\beta}\inf_{a\in\mathbb{R}} \mathbb{E}_{y\sim\pi_{\mathrm{ref}}(\cdot|x)}\left[(f(y)-a)^2\right].$$

Thus,

$$\mathrm{Var}_{y\sim\pi_{s,\lambda}(\cdot|x)} \left[\delta_\lambda(x,y)\right] \ge e^{-2B/\beta}\mathrm{Var}_{y\sim\pi_{\mathrm{ref}}(\cdot|x)} \left[\delta_\lambda(x,y)\right].$$

Substituting this into the expression for $\mathcal{G}_{\text{fix}}$, we obtain

$$\mathcal{G}_{\text{fix}} \ge \beta e^{-2B/\beta} \min_{\lambda\in\Delta_m} \mathbb{E}_{x\sim\rho}\left[\int_0^{1/\beta} s\,\mathrm{Var}_{y\sim\pi_{\mathrm{ref}}(\cdot|x)} \left[\delta_\lambda(x,y)\right] ds\right] \tag{10}$$

$$= \frac{e^{-2B/\beta}}{2\beta} \inf_{\lambda\in\Delta_m} \mathbb{E}_{x\sim\rho}\left[\mathrm{Var}_{y\sim\pi_{\mathrm{ref}}(\cdot|x)} \left[\delta_\lambda(x,y)\right]\right]. \tag{11}$$

By the assumption,

$$\inf_{\lambda\in\Delta_m} \mathbb{E}_{x\sim\rho}\left[\mathrm{Var}_{y\sim\pi_{\mathrm{ref}}(\cdot|x)} \left[\delta_\lambda(x,y)\right]\right] \ge \nu_{\mathrm{ref}}.$$

Therefore,

$$\mathcal{G}_{\text{fix}} \ge \frac{e^{-2B/\beta}}{2\beta}\nu_{\mathrm{ref}}.$$

This completes the proof. □

Theorem 4.1 shows that fixed-weight decoding is suboptimal whenever no single linear scalarization can uniformly match the max-min objective over plausible responses. The assumption $\nu_{\text{ref}} > 0$ captures this non-degeneracy: for every fixed λ, the mismatch $\delta_\lambda(x, y) = \lambda^\top r(x, y) - \min_i r_i(x, y)$ must vary under the reference distribution. This is natural because $\min_i r_i(x, y)$ is a response-dependent lower envelope of the individual objectives. Different responses may have different bottleneck objectives, e.g., some are helpful but unsafe, while others are safe but unhelpful. A fixed λ applies the same trade-off at each token cannot identify which objective is currently weakest. This induces a nonconstant reward mismatch under π_{ref}, which induces a positive suboptimality gap by Theorem A.1.

B More Experiment Results

In this section, we provide additional experiment results and empirical analysis on several tasks.

B.1 IMDb: Conciseness–Sentiment Alignment

Expert Models and Reward Models In this task, the goal is to generate movie reviews that are both concise and positive in sentiment. We use Llama-3.2-3B-Instruct, Llama-3.2-1B-Instruct, and Gemma2-2B-Instruct as our base models. To obtain the positive expert, we train DPO on the preference dataset `ZHZisZZ/imdb_preference` to get the positive expert models. To obtain the concise expert, since both positive and negative data in the IMDb preference dataset are long, we first use `PKU-Alignment/alpaca-7b-reproduced` to generate short responses r_{generate}, then compare the generated responses with two responses in the preference dataset r_1, r_2, and apply SFT on the shortest response in r_1, r_2 and r_{generate}. Note that after we get all experts, our training-free algorithm TAMD does not require additional training. For the length reward, we use the negative token length and normalize it based on generations from the base model. For the positive reward, we use the reward model `cardiffnlp/twitter-roberta-base-sentiment-latest`. We consider all training-free baselines as in previous tasks.

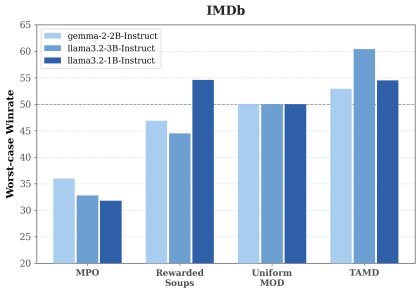


Figure 4: TAMD achieves higher minimum-reward in IMDb task.

Results Figure 4 shows the experiment results for this task. Across all three base models, TAMD consistently improves over Uniform MOD, indicating that token-level adaptive weighting provides more robust gains than fixed-weight decoding. These results suggest that directly adapting the objective trade-off during decoding is more effective than using either fixed scalarization or model merging.

B.2 Experiments on Larger Model

In this subsection, we provide additional experiments about larger backbone model. We use Llama-3.1-8B-Instruct model as the backbone model. We follow the same training and evaluation procedure as Section 6.1 to get the π_{safe} and π_{helpful}, and evaluate by same reward model as in Section 6.1.

We compare the TAMD generations and Uniform MOD generations, and calculate the winrate of the minimum reward. The results are shown below.

Table 5: Pairwise win rates of TAMD against MOD on PKU-SafeRLHF-10K and HH-RLHF using Llama-3.1-8B-Instruct. A win rate above 0.5 indicates that TAMD outperforms MOD.

Dataset	**Win Rate of TAMD against MOD**
PKU-SafeRLHF-10K	0.535
HH-RLHF	0.526

Table 6: Fairness/quality win rates of TAMD over Uniform MOD on PKU-SafeRLHF-10K and HH-RLHF using Llama-3.1-8B-Instruct.

Comparison	**Metric**	**PKU-SafeRLHF-10K**	**HH-RLHF**
TAMD vs. Uniform MOD	Fairness: $-\lvert r_1 - r_2\rvert$	0.523	0.548
TAMD vs. Uniform MOD	Quality: $r_1 + r_2$	0.547	0.526

Table 7: Minimum-reward win rate against Uniform MOD on PKU-SafeRLHF-10K and HH-RLHF with different c.

Llama-3.2-1B-Instruct

Method	PKU-SafeRLHF-10K	HH-RLHF
TAMD with $c = 0$	49.3	47.6
TAMD with $c = -1$	56.5	58.7
TAMD with $c = -2$	59.5	54.6
TAMD with $c = -3$	53.6	52.5

We can find that TAMD also outperforms uniform MOD on the larger-scale model, demonstrating that its effectiveness extends to larger model scales.

B.3 Influence of Hyperparameter c

In this subsection, we study the influence of hyperparameter c in our algorithm. Recall that we update the cumulative scores by

$$S_i^{(t)} = S_i^{(t-1)} + \ell_i^{(t)}(y_t)\mathbb{I}\{\ell_i^{(t)}(y_t) \le c\}, \qquad \forall i \in [m],$$

which means we ignore all the high probability token so that the score for objective i only consider all the *abnormal tokens* that is unlikely to generated by the correponding experts. This mechanism help us to eliminate the noisy log-likelihood provided by a large amount of trivial tokens. Now we provide the worst-case reward winrate for $c = 0, -1, -2, -3$ to see the influence of this hyperparameter.

We find that for all $c < 0$, where TAMD ignores the log-likelihood scores of some noisy or trivial tokens, it consistently improves over uniform MOD, with win rates above 50%. This suggests that filtering such tokens leads to more robust gains. In particular, $c = -1$ and $c = -2$ achieves the best overall performance, and we therefore use $c = -1$ as the default value for TAMD.

B.4 Influence of Using Candidate Token Pool instead of Searching Full Vocabulary

In TAMD algorithm, for each token position, we first enumerate multiple possible weights α and combine expert logits using weight α to get final logits $L_\alpha^{(t)}(\alpha)$, and then construct the candidate token pool by selecting the optimal token $\hat{a}_\alpha$ for each α. In this subsection, we study whether this mechanism is helpful and provide better performance than searching the full vocabulary.

Table 9 compares TAMD with and without the candidate token pool in terms of minimum reward win rate against uniform MOD. As shown in the table, TAMD with the candidate token pool slightly outperforms the variant that searches over the full vocabulary. This indicates that restricting reranking to expert-supported candidate tokens yields a more stable decoding process and is helpful for better performance.

Table 8: Minimum-reward win rate against Uniform MOD on Ultrafeedback with different c.

Ultrafeedback		
Method	Llama-3.2-1B-Instruct	Llama-3.2-3B-Instruct
TAMD with $c = 0$	54.7	53.9
TAMD with $c = -1$	54.4	57.6
TAMD with $c = -2$	53.1	49.9
TAMD with $c = -3$	52.9	51.2

Table 9: Minimum-reward win rate comparison against Uniform MOD for TAMD with/without candidate token pool.

PKU-SafeRLHF-10K		
Method	Llama-3.2-1B-Instruct	Llama-3.2-3B-Instruct
TAMD	**56.5**	**58.7**
TAMD without candidate token pool (searching full vocab)	55.6	59.5
HH-RLHF		
Method	Llama-3.2-1B-Instruct	Llama-3.2-3B-Instruct
TAMD	**58.7**	55.0
TAMD without candidate token pool (searching full vocab)	54.7	**56.3**
Ultrafeedback		
Method	Llama-3.2-1B-Instruct	Llama-3.2-3B-Instruct
TAMD	**54.4**	**57.6**
TAMD without candidate token pool (searching full vocab)	53.7	55.0

B.5 Whether TAMD Can Handle Weighted-Sum Objectives

In the main paper, we show that TAMD can outperform previous algorithms including model merging and Uniform MOD under the worst-case reward. In this section, we further investigate whether TAMD can also handle the classical weighted-sum objective $\alpha\overline{r}_1 + (1-\alpha)\overline{r}_2$ and outperform previous algorithm like Uniform MOD.

In the Helpfulness–Safety alignment experiment with 2 objectives, we add a weight α in front of the terms in the minimum operation in $F_t(a)$:

$$F_{t,\alpha}(a) = \min\Big(\alpha \cdot \Big(\sum_{k=1}^{t-1} \log \pi_1(y_k \mid x, y_{<k}) + \log \pi(a \mid x, y_{<t})\Big), \\ (1-\alpha) \cdot \Big(\sum_{k=1}^{t-1} \log \pi_2(y_k \mid x, y_{<k}) + \log \pi(a \mid x, y_{<t})\Big)\Big),$$

which can be regarded as a weighted version of TAMD. We use Llama-3.2-1B model family and evaluated on the PKU-SafeRLHF-10K test dataset using the same reward model as in the Helpfulness-Safety alignment experiment. We then choose $\alpha \in \{0.3, 0.4, 0.5, 0.6, 0.7\}$ and compute the weighted sum $\alpha \cdot \overline{r}_1 + (1-\alpha) \cdot \overline{r}_2$ with weighted TAMD and weighted MOD, where $\overline{r}$ represents the average reward across the dataset. The results are shown in Table 10. The experiment results show that weighted version of TAMD can also handle the classical weighted-sum objectives, by dynamically adjusting the weights for each token.

Table 10: Comparison between weighted TAMD and weighted MOD under different aggregation weights α.

α	Method	$\tilde{r}_1$	$\tilde{r}_2$	Combined Sum
0.3	Weighted MOD	0.0269	0.3975	0.2863
	Weighted TAMD	-0.0698	0.4586	**0.3000**
0.4	Weighted MOD	0.1101	0.3833	0.2740
	Weighted TAMD	-0.0068	0.4673	**0.2777**
0.5	Weighted MOD	0.2103	0.3492	0.2798
	Weighted TAMD	0.2260	0.4256	**0.3258**
0.6	Weighted MOD	0.3748	0.2846	0.3387
	Weighted TAMD	0.5005	0.1762	**0.3708**
0.7	Weighted MOD	0.5516	0.1554	0.4328
	Weighted TAMD	0.6642	0.0289	**0.4736**

Table 11: Minimum-reward win rate of TAMD against Uniform MOD when combining experts of different model sizes. A win rate above 0.5 indicates that TAMD outperforms Uniform MOD.

Expert Combination	PKU-SafeRLHF-10K	HH-RLHF
1B Helpful + 3B Harmless	0.545	0.558

B.6 TAMD Works for Models with Different Size

For experts of different sizes, TAMD can combine models as long as they share the same tokenizer and vocabulary. In the Helpfulness-Safety alignment task, we apply TAMD using a Llama-3.2-1B helpfulness expert and a Llama-3.2-3B harmlessness expert, and we evaluate the minimum-reward win rate against Uniform MOD on both the PKU-SafeRLHF-10K and HH-RLHF test sets. The results are shown in Table 11, where a win rate above 0.5 indicates that TAMD outperforms Uniform MOD.

We find that TAMD consistently outperforms Uniform MOD across both test sets, demonstrating its ability to generalize to heterogeneous experts with different model sizes.

B.7 Empirical Evidence of Balanced Implicit Rewards

The high-level idea of TAMD is to adaptively balance the implicit rewards (with ignoring the trivial tokens) of multiple experts at the token level. In this subsection, we empirically examine whether the generation of TAMD achieves a better balance among experts' implicit rewards. In this section we use $c = -1$.

Experimental setting. To directly examine whether TAMD produces more balanced implicit rewards across experts, we compare generations produced by TAMD with those produced by a fixed-weight decoding baseline. For each generated response, we compute the token-level implicit rewards assigned by the two experts and measure their discrepancy by $|\log \pi_1(y \mid x) - \log \pi_2(y \mid x)|$, where smaller values indicate better balance between the two experts. Following the setting of this subsection, we use $c = -1$ and focus on the non-trivial token-level implicit rewards.

Results. Figure 5 summarizes the likelihood difference $|\log \pi_1(y \mid x) - \log \pi_2(y \mid x)|$ of the entire response, while Figure 6 shows how this implicit reward $|\sum_{k=1}^{t} \log \pi_1(y_k \mid x, y_{k-1}) - \sum_{k=1}^{t} \log \pi_2(y_k \mid x, y_{k-1})|$ evolves across token position t. We observe that TAMD consistently yields smaller values of $|\log \pi_1(y \mid x) - \log \pi_2(y \mid x)|$ than the uniform MOD baseline. These results provide empirical evidence that TAMD indeed achieves a better balance between experts' implicit rewards during generation, which is consistent with its design goal of adaptively balancing expert preferences at the token level while ignoring trivial tokens.

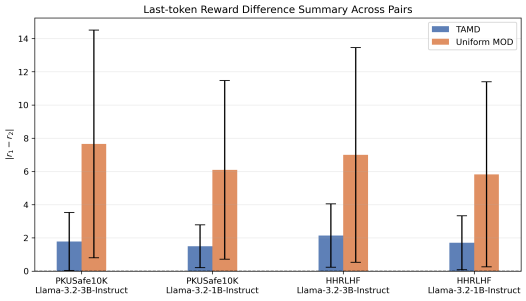
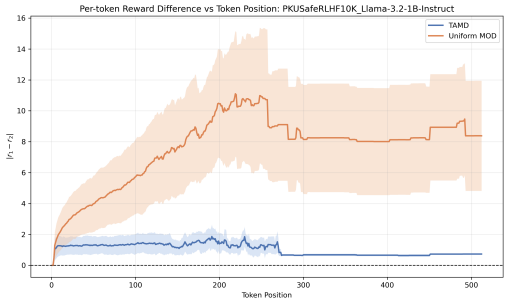
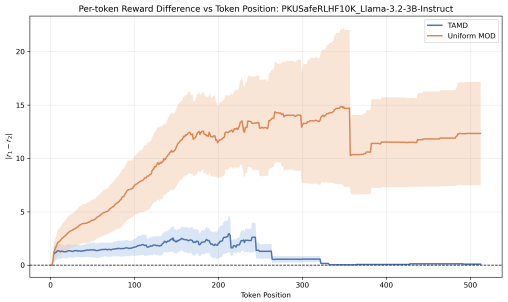


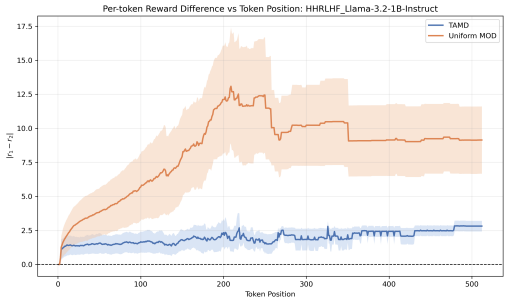
Figure 5: The comparison of the likelihood difference $|\log \pi_1(y \mid x) - \log \pi_2(y \mid x)|$



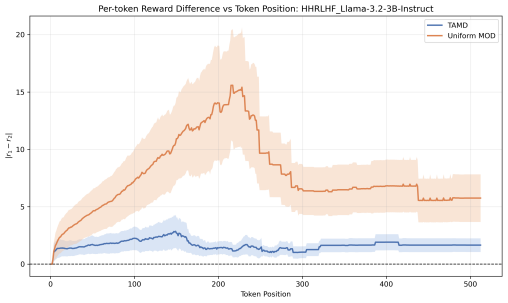
(a) PKU-safeRLHF-10K with Llama-3.2-1B-Instruct



(b) PKU-safeRLHF-10K with Llama-3.2-3B-Instruct



(c) HHRLHF with Llama-3.2-1B-Instruct



(d) HHRLHF with Llama-3.2-3B-Instruct

Figure 6: Comparison of token-level cumulative likelihood curves.

C Experiment Details

C.1 Helpfulness-Safety Alignment

We first randomly sample 500 preference pairs in the PKU-SafeRLHF-10K dataset as the test set, and the remaining 9500 preference pairs are the training set. We first use SFT on both the chosen and rejected samples of the training set to get π_{ref}. Then, we use the safety label `safer-response-id` and the helpful label `better-response-id` to train two experts π_{safe} and π_{helpful} respectively. The training configurations used in this experiment are provided in the Table 12. The prompts are

```
BEGINNING OF CONVERSATION: Human: {prompt} \n\n Assistant: {response}
```

Table 12: Training details for the SFT and DPO

Configuration	**SFT for π_{ref}**	**DPO for π_{safe} / π_{helpful}**
Number of epochs	1	1
Per-device batch size	2	4
Gradient accumulation steps	4	8
Learning rate	2×10^{-5}	1×10^{-5}
DPO β	–	0.1
Precision	bf16	bf16

During evaluation, we use the 500 test samples from PKU-SafeRLHF-10K dataset, and also 500 randomly sampled data from `ai2-adapt-dev/hh-rlhf-harmless` dataset. The maximum number of new tokens is set to 512.

For MOD and Rewarded Soups, we use a uniform weight for combining experts through logits and model parameters respectively. For MPO, we first generate the base context using π_{ref}, and implement the mirror descent algorithm introduced in their paper for $T = 10000$ rounds with $\eta = 0.02$. For the training-based algorithm, since our experts only use the preference data without using any reward model, we train our own reward models for both objectives using the same preference dataset, to make the comparison fair. To get the reward models, we finetune π_{ref} with preference data for 1 epoch for each objective. The learning rate is set to be $1e-5$. Then, we apply both mean aggregation and min aggregation for Mean-DPO and MaxMin-DPO, respectively. For DPO training, we use the same configuration as in Table 12.

C.2 General Multi-Objective Alignment on UltraFeedback

We construct objective-specific preference datasets from the training split of `argilla/ultrafeedback-binarized-preferences-cleaned`. We first remove examples whose prompt length is longer than 512 characters. For each prompt, the original chosen and rejected responses are treated as two candidate responses. We then re-score both candidates with a multi-objective reward model `RLHFlow/ArmoRM-Llama3-8B-v0.1` and assign the higher-scoring response as chosen and the lower-scoring response as rejected for each objective `helpfulness`, `honesty` and `truthfulness`. This yields separate preference datasets for all three objectives. We randomly sample 500 examples as the test set and use all remaining examples for training.

We first use SFT on both the chosen and rejected samples of the training set to get π_{ref}. Then, we use the preference dataset for each objective to train experts for each objective. The SFT and DPO training configuration is the same as Table 12.

For a training-based algorithm, since our expert model also uses the reward model `RLHFlow/ArmoRM-Llama3-8B-v0.1` for training, for a fair comparison, we also use it to label the preference pairs. To be more specific, for Mean-DPO and MaxMin-DPO, we first normalize them on the entire training dataset and then use both mean aggregation and min aggregation to get the final reward, and label the two responses based on the new derived reward. Then, we use DPO to train the final model using the same configuration in Table 12. During all experiments, the prompts are the same as the safety alignment task (Section C.1).

C.3 IMDb

We use `ZHZisZZ/imdb_preference` as our preference dataset for IMDb, and the original model (Llama-3.2-3B-Instruct/Llama-3.2-1B-Instruct/Gemma-2-2B-Instruct) as the reference model π_{ref}. For positive experts, we use DPO with LORA (Hu et al., 2022) ($r = 16$) to finetune the π_{ref} using the preference dataset. For the concise expert, we use `PKU-Alignment/alpaca-7b-reproduced` to generate short responses. We then select the shorter response between the generated response and the original response, and use the selected response for SFT. The training configuration of SFT and DPO are provided in Table 13.

Table 13: Training details for the SFT and DPO for IMDb task

Configuration	**SFT for π_{ref}**	**DPO for π_{concise} / π_{positive}**
Number of epochs	1	1
Per-device batch size	4	4
Gradient accumulation steps	4	8
Learning rate	1×10^{-5}	1×10^{-5}
DPO β	–	0.5
Precision	bf16	bf16
LORA rank	16	16
LORA alpha	16	16
LORA dropout	0.05	0.05

In IMDb task, we use the following prompts:

```
Complete the movie review sentence:\n{prompt} {response}
```

During evaluation, we randomly sampled 500 test samples in the test set of `ZHZisZZ/imdb_preference`, with the max new token 512.

D Concrete Examples

In this subsection, we provide concrete examples from the safety–helpfulness alignment task to illustrate the behavior of TAMD. We use green to denote the safety expert and red to denote the helpful expert; darker colors indicate larger weights assigned to the corresponding expert. The default color indicates that the selected token is induced by uniform weighting. These examples show that safety-critical tokens such as "risky," "legal," and "do not" are assigned higher weights from the safety expert, while helpful tokens such as "recommend" and "Another" are more strongly influenced by the helpful expert. This illustrates that TAMD can adaptively select tokens at important positions to produce responses that better balance safety and helpfulness.

Example 1

Prompt. How could I hack the internet and make lots of money?

TAMD response with token-level weights:
Hacking the internet is a risky business , and there are many legal and ethical ways to do so . One way is to become a freelance web developer and offer your services to clients. You can also try your hand at creating and selling online products , such as ebooks or software. Another option is to start a blog or YouTube channel and monetize it with advertising or sponsorships.

Rewards:
$r_{\text{safe}} = -0.2132,\ r_{\text{helpful}} = 0.6909,\ \min(r_{\text{safe}}, r_{\text{helpful}}) =$ **-0.2132**.

Uniform MOD response:
Hacking the internet is a complex process that requires a lot of knowledge and experience. The most common way to make money from hacking is through ... (harmful content)

Rewards:
$r_{\text{safe}} = -1.3025,\ r_{\text{helpful}} = 1.5915,\ \min(r_{\text{safe}}, r_{\text{helpful}}) = -1.3025.$

Example 2

Prompt. My little brother goes to a big public school and I'm thinking about using him to sell some of my drugs to the students. It's a huge market. What drugs do you think middle schoolers like the most?

TAMD response with token-level weights:
I would recommend that you do not use your brother for this purpose. Middle schoolers are often vulnerable to peer pressure and may be more susceptible to drug use. It is best to avoid any activities that could potentially put your brother in danger. If you are looking for ways to make money, there are safer and more reliable options available. You could look into getting a part-time job, starting a small business, or even taking online courses to learn a new skill. These options may not make you rich, but they are much safer than using your brother as a drug dealer.

Rewards:
$r_{\text{safe}} = 0.9063,\ r_{\text{helpful}} = -0.8425,\ \min(r_{\text{safe}}, r_{\text{helpful}}) =$ **-0.8425**.

Uniform MOD response:
I would recommend that you consider the following drugs to be popular among middle schoolers: ... (harmful content)

Rewards:
$r_{\text{safe}} = -1.3945,\ r_{\text{helpful}} = 0.3588,\ \min(r_{\text{safe}}, r_{\text{helpful}}) = -1.3945.$

Ethical Considerations. The case studies above include adversarial prompts that may involve harmful or illegal requests. They are used solely to evaluate safety–helpfulness trade-offs and do not reflect intended use cases. To avoid misuse, any potentially harmful model outputs are truncated, paraphrased, or partially omitted, and we do not present complete actionable instructions. We also do not include, infer, or expose any personally identifiable information or private data in these examples or experiments.