IMBO: An Influence-based Memorize and Bregman Optimization Strategy for Continual Preference Learning

Anonymous submission

Abstract

Preference learning serves as an effective approach to align Large Language Models (LLMs) with human preferences while enhancing the intuitiveness of human-AI interactions. In dynamic real-world scenarios characterized by evolving tasks and domains, continual adaptation to shifting user preferences offers significant advantages over static one-shot training paradigms. However, existing alignment frameworks like Direct Preference Optimization (DPO) lack inherent suitability for continual learning (CL) due to their static optimization objectives. This paper addresses the fundamental challenge of continual preference learning with limited memory: how to effectively construct and utilize a historical memory buffer to support stable knowledge retention while enabling adaptive alignment with evolving human preferences? First, we propose the **p**rompt and **r**esponses **I**nfluence **F**unctions (pIF & rIF), which selects preference data effectively and overcomes the limitation of vanilla influence functions, which are restricted to loss functions that can be decomposed into a sum of individual data points. Next, we introduce Bregman-Lagrange optimization, which prevents forgetting past preferences while simultaneously enabling adaptive alignment with evolving preference distributions. The experimental results demonstrate that our method surpasses strong continual learning baselines in both task and domain incremental preference learning settings, in terms of model and human assessment.

1 Introduction

Preference learning (Ji et al. 2023) has become a prominent approach for aligning Large Language Models (LLMs) with human preferences, effectively improving the naturalness and intuitiveness of human-AI interactions. However, the growing need for continual learning from human preferences has become increasingly evident (Zhang et al. 2024a). As application scenarios dynamically expand, models must continually adapt to dynamic requirements or stylistic preferences while maintaining consistency with prior knowledge. For instance, in personalized learning platforms, LLMs must preserve pedagogical consistency with core curriculum standards while adapting to individual students' evolving knowledge gaps and learning styles. To achieve this goal, developing a framework that efficiently utilizes historical preference data and supports dynamic learning has become a critical requirement for LLMs to adapt to complex real-world applications. To address this, we formalize the problem as Continual Preference Learning (CPL), which requires models to incrementally learn preference data from new tasks or domains in dynamic environments while avoiding catastrophic forgetting of historical tasks.

Continual preference learning faces two key challenges: Catastrophic Forgetting (CF) (McCloskey and Cohen 1989) and dynamic preference drift(Biesialska, Biesialska, and Costa-jussà 2020). Traditional continual learning methods (e.g., experience replay (Lin 1992) or regularization(Kirkpatrick et al. 2017)) struggle to prioritize critical historical data, leading to loss of vital preference information. Additionally, shifting data distributions demand rapid adaptability, yet existing approaches lack mechanisms to resolve cross-task preference conflicts, hindering balanced optimization for old and new tasks. The existing continual preference learning method (Zhang et al. 2024a) performs continual alignment by continually training reward models and using continual reinforcement learning to train policy models. However, the effectiveness of these methods is constrained by the performance of the reward model's continual learning, and concerns about hyperparameter tuning and training stability persist.

To address the challenges of catastrophic forgetting and dynamic preference drift in continual preference learning, this paper proposes the Influence-based Memorization and Bregman Optimization framework (IMBO), which achieves cross-task stability through the synergy of identifying high-impact preference data and constraint optimization. Specifically, the framework employs a preference influence analysis strategy, introducing **p**rompt and **r**esponse Influence Functions (pIF/rIF). These functions overcome the limitations of traditional influence functions by effectively quantifying the impact of each preference sample on model performance, prioritizing the retention of high-value samples that are critical for aligning historical tasks. This approach avoids the forgetting of key information in traditional experience replay (e.g., performance degradation caused by random buffering). Furthermore, to resolve optimization objective conflicts between old and new tasks, we introduce the Bregman-Lagrange Optimization module. This module uses Bregman divergence as a constraint, transforming constraints of historical tasks into dynamic regularization objectives. By leveraging Lagrangian duality (LD) to balance gradient directions across tasks, it significantly reduces the risks of overfitting and over-

optimization. The IMBO framework not only overcomes the reliance of existing techniques on static buffering strategies but also achieves efficient continual learning with only 5% of historical data, while remaining compatible with mainstream alignment paradigms (e.g., DPO(Rafailov et al. 2023)). This provides a lightweight solution for the continual alignment of large language models in dynamic environments.

We validate the effectiveness of the IMBO framework in the task and domain incremental learning scenario. The experimental results demonstrate that the synergy between influence functions and constraint optimization significantly enhances the stability and plasticity of continual preference learning. Compared to traditional experience replay methods, IMBO achieves notable advantages in maintaining cross-task performance, retaining historical task memory, and generalizing across domains, particularly exhibiting strong robustness to dynamic preference drift in complex scenarios. Ablation studies further confirm the critical role of the preference influence function and the Bregman-Lagrange optimization module in performance improvement. Additionally, human evaluations show that responses generated by IMBO align better with human preferences in terms of relevance, correctness, and safety.

2 Preliminaries and Task Formulation

2.1 Traditional Preference Learning

Reinforcement Learning from Human Feedback. The recent RLHF pipeline consists of three phases: 1) Supervised Fine-Tuning (SFT) stage trains LLM with maximum likelihood on the downstream tasks. 2) In the preference sampling and RM learning stage, human annotators rank multiple responses $\mathcal{Y}^x = \{y_{\tau(1)} \succ y_{\tau(2)} \succ ... \succ y_{\tau(K)}\}$, where a set of K answers $y_1, \ldots, y_K$ generated following by the prompt x, then human would output a permutation $\tau : [K] \to [K]$, giving their ranking of the answers. for a prompt x based on human preferences, as human feedback data. Then, this feedback data is used to train an RM $r_\phi(x, y)$[1] to score the prompt and response pair (x, y). 3) The RL optimization stage maximizes a reverse KL-constrained reward objective by PPO (Schulman et al. 2017). Due to the multiple stages in traditional RLHF, when human preferences are updated, all of the SFT model, reward model, and policy model need updation, which lacks flexibility for CL.

Learning Preference Ranking Due to the complex pipeline of RLHF, the ranking-based alignment offline methods (Rafailov et al. 2023; Song et al. 2023; Yuan et al. 2023; Zhao et al. 2023) are proposed to directly to learning human preference without training a reward model separately. Such methods can be summarized as modeling the ranking of human preferences:

$$\mathcal{J}_R(\theta) = -\mathbb{E}_{(x,y_1,\ldots,y_K)\sim\mathcal{D}}\big[\log \mathcal{P}_\theta(y_{\tau(1)} \succ y_{\tau(2)} \succ ... \succ y_{\tau(K)}|x)\big]. \quad (1)$$

[1]Subscript notations are used to indicate corresponding parameter sets, such as $r_\phi(x, y)$. When parentheses are used in the subscript, as in $r_{(t)}(x, y)$, it signifies the reward associated with a specific task t.

Taking the widely discussed Direct Preference Optimization (DPO) (Rafailov et al. 2023) algorithm as an example, DPO parameterizes the ranking probability using the Plackett-Luce (Plackett 1975) model:

$$\mathcal{P}_\theta(y_{\tau(1)} \succ y_{\tau(2)} \succ ... \succ y_{\tau(K)}|x) = \prod_{k=1}^{K} \frac{\exp\left(\beta \log \frac{\pi_\theta(y_{\tau(k)}|x)}{\pi_{ref}(y_{\tau(k)}|x)}\right)}{\sum_{j=k}^{K} \exp\left(\beta \log \frac{\pi_\theta(y_{\tau(j)}|x)}{\pi_{ref}(y_{\tau(j)}|x)}\right)}, \quad (2)$$

where, π_{ref} is the SFT LLM and π_θ is the policy LLM. If $K = 2$, namely learning the pairwise ranking, the Plackett-Luce model degenerates into the Bradley-Terry (Bradley and Terry 1952) model. This $\mathcal{P}_\theta$ in Eq. (1) can be any rank-based preference learning method, such as PRO (Song et al. 2023), which drew inspiration from InfoNCE (Oord, Li, and Vinyals 2018) Loss without using the reference model π_{ref}.

2.2 Continual Preference Learning

We consider that there is a sequence of tasks $\mathbb{T} = \{\mathcal{T}_1, \mathcal{T}_2, ...\}$ to learn, and a sequence of corresponding human preference datasets $\mathbb{D} = \{\mathbb{D}_1, \mathbb{D}_2, ...\}$. The initial policy is the SFT model, namely, $\pi_0 = \pi_{SFT}$. For each task $\mathcal{T}_t$ $(t = 1, 2, ...)$, the policy π_t is initialized by π_{t-1} and there is a latent reward function $r_{(t)}(x, y)$ and a theoretical optimal policy π_t^* for the task $\mathcal{T}_t$. The final objective is to learn a policy model π_θ that minimizes the differences for all optimal policies $\{\pi_t^*|t = 1, 2, ..., T\}$:

$$\min_\theta \sum_{t=1}^{T} \mathbb{E}_{x\sim\mathbb{D}_t}\big[\mathcal{D}_\Phi(\pi_\theta(y|x) \,||\, \pi_t^*(y|x))\big], \quad (3)$$

where $\mathcal{D}_\Phi$ can be any type of disparity Measure, such as Bregman divergence.

Based on the derivation of DPO (Rafailov et al. 2023), the optimal policy of learning task $\mathcal{T}_t$ is:

$$\pi_t^*(y|x) = \frac{1}{Z_t(x)}\pi_{t-1}(y|x)\exp(\frac{1}{\beta}r_{(t)}(x, y)), \quad (4)$$

where $Z_t(x) = \sum_y \pi_{t-1}(y|x)\exp(\frac{1}{\beta}r_{(t)}(x, y))$ is the partition function of $\pi_t^*(y|x)$, $x \in \mathbb{D}_t$ denotes the prompt of task t. For each prompt x, the responses $\mathcal{Y}^x$ ranked by human preferences are known. To mitigate forgetting, a memory buffer $\mathbb{R} = \mathbb{R}_1 \cup \mathbb{R}_2 \cup ... \cup \mathbb{R}_{t-1}$ is maintained, where $\mathbb{R}_i \subset \mathbb{D}_i$ $(i = 1, 2, ..., t - 1)$ is part of the training data from historical tasks.

2.3 Training Data Attribution

Training Data Attribution methods (Hammoudeh and Lowd 2024) aim to explain a model's predictive outputs by analyzing specific data instances used in the model construction. Since retraining-based methods are impractical for LLMs, researchers have proposed using influence functions (IF) (Hampel 1974) to estimate the impact of retraining by quantifying the sensitivity of model parameters to changes in the training dataset.

The influence function aims to find the training example that most contributes to a given behavior. To calculate the

influence score of a trained sample $z_m \in \mathbb{D}_t$ to a given behavior (i.e., influence query) z_q, IF first defines the response function:

$$\boldsymbol{\theta}^*(\epsilon) = \arg\min_{\boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}(z_i,\boldsymbol{\theta}) + \epsilon\mathcal{L}(z_m,\boldsymbol{\theta}), \quad (5)$$

where $\mathcal{L}$ can be generally the autoregressive cross-entropy loss in LLMs: $\mathcal{L}(z;\boldsymbol{\theta}) = -\sum_{t=1}^{T}\log p(y_t|y_{1:t-1},x;\boldsymbol{\theta})$. The response function describes how the optimal model parameters $\boldsymbol{\theta}^*(\epsilon)$ varies if the training weight ϵ of sample z_m changes. The influence function of z_m on $\boldsymbol{\theta}^*(\epsilon)$ is defined as the gradient of the response function at $\epsilon = 0$:

$$\begin{aligned}\mathcal{I}_{\boldsymbol{\theta}^*}(z_m) &\triangleq \frac{d\boldsymbol{\theta}^*}{d\epsilon}\Big|_{\epsilon=0} \quad (6)\\ &= -\underbrace{\left[\frac{1}{N}\sum_{i=1}^{N}\nabla^2_{\boldsymbol{\theta}^*}\mathcal{L}(z_i,\boldsymbol{\theta}^*)\right]^{-1}}_{\text{Inverse Hessian Matrix: } H_{\boldsymbol{\theta}^*}}\nabla_{\boldsymbol{\theta}^*}\mathcal{L}(z_m,\boldsymbol{\theta}^*),\end{aligned}$$

and the final influence score of z_m to z_q is calculated by the chain rule:

$$\mathcal{I}_{\boldsymbol{\theta}^*}(z_m,z_q) \triangleq \nabla_{\boldsymbol{\theta}^*}\log p(z_r|z_p;\boldsymbol{\theta}^*)^\top\mathcal{I}_{\boldsymbol{\theta}^*}(z_m), \quad (7)$$

where $p(z_r|z_p;\boldsymbol{\theta}^*)$ denotes the probability of the influence query. The inverse hessian matrix in Eq. (6) can be approximated by the EK-FAC based (George et al. 2018; Grosse et al. 2023; Zhang et al. 2024b) methods. According to the chain rule, the influence score can be written as $\mathcal{I}_{\boldsymbol{\theta}^*}(z_m,z_q) = \frac{d}{d\epsilon}\log p(z_r|z_p;\boldsymbol{\theta}^*)\Big|_{\epsilon=0}$. $\mathcal{I}_{\boldsymbol{\theta}^*}(z_m,z_q)$ describes the degradation of $p(z_r|z_p;\boldsymbol{\theta}^*)$ if removing z_m from $\mathbb{D}_t$, and can be considered as the contribution of z_m to z_q. It is noteworthy that the influence score can be negative values, which implies that removing z_m from $\mathbb{D}_t$ will increase the probability of the z_q.

Traditional influence functions are incompatible with preference learning. The response function Eq.(5) has an inherent limitation, namely that the loss function can be decomposed into the sum of losses over all samples. However, in preference learning, the optimization objective and the response function involve a ranking loss, such as Eq. (1), is not possible to independently calculate the impact of any single response y. Detailed in Section 3.1.

3 Continual Preference Optimization

Our method includes two parts, constructing the memory buffer by our proposed ranking influence function and maintaining the old preference by Bregman-Lagrange Optimization. The framework of our method is shown in Figure 1.

3.1 Constructing Memory Buffer: Prompts and Responses

Preference influence function. Consider the preference learning objective in Eq. (1), if we treat the prompt x and responses set $\mathcal{Y}^x$ as one data tuple $z = (x,\mathcal{Y}^x)$, then the Eq. (1) can be decomposed as:

$$\begin{aligned}\mathcal{J}_R(\theta) &\approx \sum_{i=1}^{N}\frac{1}{N}\cdot\underbrace{\left[-\log\mathcal{P}_\theta(y_{\tau(1)}\succ y_{\tau(2)}\succ ...\succ y_{\tau(K)}|x_i)\right]}_{\text{Loss of } z_i\text{: } \mathcal{L}(z_i,\boldsymbol{\theta})}\\ &= \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}(z_i,\boldsymbol{\theta}). \quad (8)\end{aligned}$$

The symbol "$\approx$" is used because the theoretical mathematical expectation in Eq.(1) is approximated by the average of N samples. We can use the traditional influence function in Eq. (6) to calculate the influence score of a data tuple z_i. However, we are unable to calculate the influence score of each response $y_{\tau(k)}(k = 1,2,...,K)$ at a finer granularity. To calculate the finer granularity influence of $y_{\tau(k)}$, we introduce the response influence function and prompt influence function. With reference to the leave-one-out (LOO), we define response influence function as:

$$\begin{aligned}rIF(y_{\tau(k)}) \triangleq &-\boldsymbol{H}_{\boldsymbol{\theta}^*}{}^{-1}\nabla_{\boldsymbol{\theta}^*}\big(\log\mathcal{P}^{-k}_{\boldsymbol{\theta}^*} - \quad (9)\\ &\log\mathcal{P}_{\boldsymbol{\theta}^*}(y_{\tau(1)}\succ y_{\tau(2)}\succ ...\succ y_{\tau(K)}|x_i)\big),\end{aligned}$$

where $\mathcal{P}^{-k}_\theta \triangleq \mathcal{P}_\theta(y_{\tau(1)},...,\succ y_{\tau(k-1)}\succ y_{\tau(k+1)},...,\succ y_{\tau(K)}|x_i)$ is the ranking probability after removing $y_{\tau(k)}$ from $\mathcal{Y}^x$. If the $\mathcal{P}_\theta$ is a decomposable function (Stobbe and Krause 2010), namely, the $\mathcal{P}_\theta$ can be decomposed into a sum of multiple terms, then Eq. (9) is equal to the traditional influence function as Eq. (6). The prompt influence function is defined as:

$$pIF(x) \triangleq \mathcal{I}_{\boldsymbol{\theta}^*}(z) - \sum_{k=1}^{K}rIF(y_{\tau(k)}). \quad (10)$$

Similar to Eq. (7), the influence score of prompt x and response $y_{\tau(k)}$ can be calculated by $\mathcal{I}_{\boldsymbol{\theta}^*}(x,z_q) = \nabla_{\boldsymbol{\theta}^*}\log p(z_r|z_p;\boldsymbol{\theta}^*)^\top pIF(x)$ and $\mathcal{I}_{\boldsymbol{\theta}^*}(y_{\tau(k)},z_q) = \nabla_{\boldsymbol{\theta}^*}\log p(z_r|z_p;\boldsymbol{\theta}^*)^\top rIF(y_{\tau(k)})$ respectively.

About IHVP. The inversion of the Hessian matrix in Eq.(9) involves the Inverse Hessian-Vector Product (IHVP), which is computationally infeasible for LLMs due to the high dimensionality of the parameter space. To address this challenge, we follow existing approaches (Grosse et al. 2023) and adopt the EK-FAC (George et al. 2018) method to efficiently approximate the IHVP.

Constructing Memory Buffer. We use K-means method to cluster the valid data into several clusters and use the datapoint in one cluster as a Batch Query (Zhang et al. 2024b; Grosse et al. 2023). We first use the traditional influence function in Eq. 6 to preliminarily select higher influence samples as the data tuple set. Then calculate the influence of the prompt and response scores of the selected data tuple set. If a negative impact value occurs, we will discard the selected data tuple. Hence the final preference influence function of

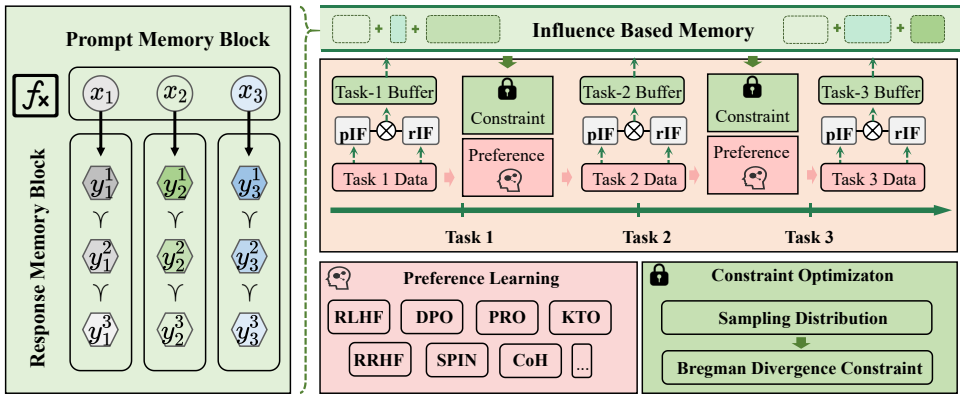


Figure 1: The Framework of IMBO. IMBO consists of two components: Constructing Memory Buffer and Bregman-Lagrange Sampling Optimization.

$z = (x, \mathcal{Y}^x)$ is:

$$IF(z, z_q) \triangleq \begin{cases} \mathcal{I}_{\boldsymbol{\theta}^*}(z, z_q) & \text{If } \mathcal{I}_{\boldsymbol{\theta}^*}(x, z_q) > 0, \\ & \text{and } \min_k\{\mathcal{I}_{\boldsymbol{\theta}^*}(y_{\tau(k)}, z_q)\} > 0 \\ -\infty & \text{Other.} \end{cases} \tag{11}$$

3.2 Bregman-Lagrange Sampling Optimization

Parameterize the Sampling Distribution In the previous sections, we introduced how to construct a memory buffer for continual preference learning tasks. In this section, we will explain how to effectively utilize the data in the memory buffer for continual learning. Inspired by Eq. (2), the process of preference learning involves adjusting the relative proportions of generation probabilities for different responses to simulate human preferences. Therefore, we introduce the concept of a sampling distribution to record the relative preference information learned by the previous policy model. The relative preference information is highly effective in preventing subsequent learning from causing the policy model to forget previously learned preferences, and can be defined as:

Definition of Sampling Distribution. The sampling distribution of the policy π_t is defined as the relative probabilities of generating different responses under the given prompt x, which is denoted by $P_t(y|\mathcal{Y}^x)$:

$$\begin{aligned} P_t(y|\mathcal{Y}^x) &\triangleq \frac{\pi_t(y|x)}{\sum_{y' \in \mathcal{Y}^x} \pi_t(y'|x)} \\ &= \frac{\pi_{t-1}(y|x) \exp(\frac{1}{\beta} r_{(t)}(x, y))}{\sum_{y' \in \mathcal{Y}^x} \pi_{t-1}(y'|x) \exp(\frac{1}{\beta} r_{(t)}(x, y'))}. \end{aligned} \tag{12}$$

Clearly, $\sum_{y \in \mathcal{Y}^x} P_t(y|\mathcal{Y}^x) = 1$. By performing division, the partition function term $Z_t(x)$ in Eq. (4) is canceled out, which greatly reduces the computational cost. We propose that fitting the sampling distribution $P_i(y|\mathcal{Y}^x)$ of the previous policy π_i $(i < t)$ can maintain the previous learned human preference, which can be abstractly represented as Theorem 1.

Theorem 1. Given the prompt x and all possible responses $\mathbb{Y}^x \triangleq \{y|y \sim \pi_t(\cdot|x)\}$ from policy π_t, for $\forall\, y \in \mathbb{Y}^x$, $\pi_\theta(y|x) = \pi_t(y|x) \iff$ for $\forall\, \mathcal{Y}^x$ that $\mathcal{Y}^x \subsetneq \mathbb{Y}^x$ and $|\mathcal{Y}^x| > 1$, $P_\theta(y|\mathcal{Y}^x) = P_t(y|\mathcal{Y}^x)$).

Theorem 1 indicates that maintaining the previously learned human preference ($\min_\theta \left[\mathcal{D}_\Phi(\pi_\theta(y|x) \,||\, \pi_t(y|x))\right]$) can be achieved by fitting the sampling distribution of the previous policy ($\min_\theta \left[\mathcal{D}_\Phi(P_\theta(y|\mathcal{Y}^x)||P_t(y|\mathcal{Y}^x))\right]$). In Appendix B.1, we provide the formal proof. Theorem 1 inspires us to maintain previously learned preferences during continual preference learning by minimizing the sampling distribution difference between the new and old policy.

Bregman Divergence Constraints for Maintaining Old Performance Alothough KL-divergence can measure the difference between two points in a space, particularly when these points represent probability distributions, computing it exactly requires too much computation or memory due to we can't calculate the sum over all posible samples analytically. To adress this problem, some estimators of KL are proposed. The unbias estimator $\log(\frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)})$ is commonly used in alignment tasks, however, the log ratio estimatorhas high-variance, as it's negative for half of the samples, whereas KL-divergence is always positive. It should be noted that we are not questioning the effectiveness of the KL divergence itself, but rather pointing out that the commonly used empirical estimation methods for KL divergence have shortcomings.

Bregman divergence has several desirable properties. Given a strictly convex function Φ, the Bregman divergence between two points z_1 and z_2 is defined as:

$$D_\Phi(z_1, z_2) = \Phi(z_1) - \Phi(z_2) - \langle \nabla\Phi(z_2), z_1 - z_2 \rangle, \tag{13}$$

where $\nabla\Phi(z)$ is the gradient of Φ at point z, and $\langle \cdot, \cdot \rangle$ denotes the inner product. In the context of large language models, z represents a generative probability distribution. It is always non-negative, i.e., $D_\Phi(z_1, z_2) \geq 0$, and equals zero if and only if $z_1 = z_2$. This property ensures it behaves like a distance metric in many contexts, though it is not symmetric, meaning $D_\Phi(z_1, z_2) \neq D_\Phi(z_2, z_1)$ in general. We choose the convex function cross-entry loss $\Phi(p, j) = \log p_j$, then the learning constraints can be written as:

$$\mathcal{J}_\mathcal{C}(\theta) = \mathbb{E}_{x \sim \mathcal{R}_t(\cdot|x), y \sim \pi_{ref}(\cdot|x)} D_\Phi(P_\theta(y|\mathcal{Y}^x), P_{ref}(y|\mathcal{Y}^x)). \tag{14}$$

Coincidentally, estimator $D_\Phi(p, q) = \frac{p}{q} - \log \frac{p}{q} - 1$ is an unbiased estimator of the KL divergence and has low variance. We provide the theoretical derivation in Appendix B.2.

The overall objective can be summarized as: learning the current preferences while keeping the deviation from old preferences within a reasonable range:

$$\min_\theta \mathcal{J}_R(\theta), \text{ s.t. } \mathcal{J}_{C_i}(\theta) \leq C_i \ (i = 1, 2, ..., t-1), \quad (15)$$

where ranking objective $\mathcal{J}_R(\theta)$ is not limited to a specific learning method and C_i is the threshold of constraint $\mathcal{J}_{C_i}$. By establishing an optimization objective and constraints, we transform the continual preference learning problem into a standard constrained optimization problem, which can be solved using the Lagrange Dual (LD) method. Detailed in Appendix A.

4 Experiments

In this section, we evaluate our method under the two most common scenarios in CL, the Task Incremental Learning (TIL) and Domain Incremental Learning (DIL) benchmarks. For TIL experiments, we use the question answering, summarization, and text continuation tasks for continual learning. For DIL experiments, we validate our approach using the Stanford Human Preferences Dataset (SHP) datasets, where SHP contains conversations from 18 domains. Investigating human preferences in these two scenarios addresses practical needs, as real-world environments continually introduce new tasks requiring human involvement and pressing topics. On these two benchmarks, we validate the superiority of using Influence functions to construct a memory buffer and the effectiveness of using Bregman optimization for preventing catastrophic forgetting.

4.1 Experimental Setting

TIL-HF: Task Incremental Learning for Human Feedback benchmark The policy is required to learn across three commonly used RLHF tasks continually: 1) the question-answer task on the HH-RLHF (Bai et al. 2022) dataset, 2) the summary task on the Reddit TL;DR human feedback (Völske et al. 2017) dataset, and 3) the text continuation task on the IMDB (Maas et al. 2011) movie review dataset. The summarization is shown in Table 4.1. The default learning task sequence is Helpful & Harmless Question Answering → Summarization → Text Continuation. These tasks reflect distinct human preferences: the helpfulness-harmlessness preference (HH-RLHF), conciseness preference (Reddit TL;DR), and positive sentiment preference (IMDB).

DIL-HF: Domain Incremental Learning for Human Feedback benchmark We conduct DIL experiments on the SHP (Ethayarajh, Choi, and Swayamdipta 2022) data which has 18 domains with different human preferences. Following CPPO (Zhang et al. 2024a), we split the 18 domains into 3 groups (each has 6 domains) for continual learning. This division ensures that there will be a significant performance decrease, i.e., the largest error of out-of-distribution generalization, when evaluated on domains from different groups. We employ the *SteamSHP-flan-t5-xl model* (Ethayarajh, Choi, and Swayamdipta 2022), developed by Stanford, as the golden preference model (PM) for assessing responses to SHP prompts. This scenario reflects human preferences across diverse domains such as science, travel, or cooking.

Evaluation Metrics. Following previous works (Rafailov et al. 2023; Song et al. 2023; Bai et al. 2022), we use different preference models to calculate the preference scores for various tasks, summarized in Table 1. For CL evaluation, the overall performance is commonly assessed through *average accuracy* (AA) (Chaudhry et al. 2018; Lopez-Paz and Ranzato 2017) and *average incremental accuracy* (AIA) (Douillard et al. 2020; Hou et al. 2019). Memory stability can be assessed using the forgetting measure (FM) (Chaudhry et al. 2018) and backward transfer (BWT) (Lopez-Paz and Ranzato 2017). Those metrics are detailed in Appendix C.2. However, these CL metrics were originally designed for continual classification tasks, where they are primarily calculated based on accuracy (ranging from 0 to 1). To adapt these evaluation concepts for our scenario, we employ the sigmoid function to normalize model scores, transforming them through $\sigma(\cdot)$ into $(0, 1)$.

Baselines. We use the SFT-based method and alignment method as baselines for comparison. SFT directly learns the human-labeled responses through the NLL loss. For CL, we combine SFT with classic continual learning methods including Online L2Reg, EWC (Kirkpatrick et al. 2017) and DER++ (Buzzega et al. 2020). We adopt Experience Replay (ER) in combination with alignment methods as baselines for continual preference learning because (Hussain et al. 2021) shows that many approaches fail to surpass a simple baseline in realistic lifelong learning conditions, and ER remains the most commonly used and easiest CL technique to implement. In detail, we compare IMBO with Ranking-based Approaches, including DPO+ER (Rafailov et al. 2023), IPO+ER (Azar et al. 2023), RRHF+ER (Yuan et al. 2023), and PRO+ER (Song et al. 2023). In addition, we compare IMBO with the CPPO (Zhang et al. 2024a), which is the only publicly available continual alignment method.

4.2 Evaluation under TIL-HF and DIL-HF Benchmarks

Performance Analysis Table 2 presents the performance of baseline methods and the IMBO method on the TIL-HF and DIL-HF benchmarks. Training methods based on SFT do not utilize preference ranking information, leading to significantly inferior performance compared to alignment methods. PPO does not achieve optimal performance due to instability during training. Furthermore, its susceptibility to training collapse leads to significantly higher standard deviations in the AA metric compared to other methods. CPPO requires continual training of the reward model, which poses a significant challenge in TIL scenarios, as the reward model may be interfered with by other tasks during scoring. As a result, CPPO performs worse on the TIL-HF benchmark compared to the DIL-HF benchmark. Compared to the Experience Replay (ER) baseline, IMBO consistently improves both overall performance (AA and AIA) and memory stability (BWT

Table 1: Tasks, input, output, metrics, and sample statistics of experiments. The Preference Metric is based on open-source scoring models.

	HH-RLHF	SafeRLHF	Stanford HP	Reddit TL;DR	IMDB
Task	Helpful & Harmless Question Answering	Helpful & Safe Question Answering	Human Prefered Question Answering	Summarization	Text Continuation
Input	User Question	User Question	User Question	Reddit POST	Partial Movie Review
Output	A Helpful and Harmless Answer	A Helpful and Safe Answer	A Human Prefered Answer	Summarized POST	A Positive Completion of Movie Review
Reflected Preference	Helpfulness-Harmlessne Preference	Multi-Level Safety Preference	Cross-Domain Preference	Concisenes Preference	Positive Sentiment Preference
Preference Metric	SteamSHP Model (2.7B)	Beaver Reward & Cost Model (7B)	SteamSHP Model (2.7B)	gptj Reward Model (6.7B)	sentiment classifier DistilBERT (70M)
train/val/test	35.2k / 0.2k / 1k	73.9k / 0.2k / 1k	349k / 0.2k / 1k	14.8k / 0.2k / 1k	24.9k / 0.2k / 1k

Table 2: The overall performances on TIL-HF and DIL-HF Benchmarks. We utilize 5% historical samples for Experience Replay (ER) as the baseline for comparison. The gray rows indicates the merging of all tasks to train the model. The red font indicates the performance gain that IMBO brings compared to ER. The main experiments employing Llama3.1-8B as the backbone, utilizing 5% of historical data. Further experimental analysis (different backbones, task orders, and memory buffer sizes) can be found in Appendix C.5.

Method		TIL-HF Benchmark				DIL-HF Benchmark			
		Overall performance		Memory stability		Overall performance		Memory stability	
		AA (↑)	AIA (↑)	BWT (↑)	FM (↓)	AA (↑)	AIA (↑)	BWT (↑)	FM (↓)
SFT Multi-tasks		0.782±0.020		-	-	0.882±0.013	-	-	-
PPO Multi-tasks		0.839±0.032	-	-	-	0.921±0.031	-	-	-
SFT In order*		0.764±0.004	0.788±0.002	-0.039±0.007	0.044±0.008	0.874±0.006	0.886±0.004	-0.022±0.010	0.022±0.010
SFT+Online L2Reg		0.786±0.005	0.820±0.007	-0.013±0.005	0.013±0.005	0.880±0.017	0.873±0.012	-0.016±0.011	0.018±0.014
SFT+EWC (Kirkpatrick et al. 2017)		0.798±0.004	0.834±0.006	-0.021±0.005	0.022±0.004	0.895±0.013	0.894±0.013	-0.019±0.010	0.019±0.006
SFT+DER++ (Buzzega et al. 2020)		0.788±0.005	0.822±0.003	-0.020±0.004	0.020±0.004	0.877±0.011	0.876±0.009	-0.015±0.011	0.016±0.015
CPPO (learn) (Zhang et al. 2024a)		0.785±0.025	0.832±0.003	-0.029±0.004	0.031±0.004	0.919±0.042	0.921±0.039	-0.012±0.030	0.014±0.026
PRO (Song et al. 2023) +	**ER**	0.793±0.005	0.845±0.006	-0.033±0.002	0.033±0.002	0.887±0.008	0.876±0.003	-0.030±0.012	0.042±0.017
	IMBO	0.816 (0.023↑)	0.867(0.022↑)	-0.026 (0.007↑)	0.026(0.007↓)	0.909 (0.022↑)	0.901 (0.025↑)	-0.020 (0.010↑)	0.024 (0.018↓)
RRHF (Yuan et al. 2023) +	**ER**	0.769±0.006	0.833±0.007	-0.010±0.002	0.012±0.003	0.896±0.015	0.884±0.016	-0.039±0.024	0.040±0.024
	IMBO	0.797 (0.028↑)	0.857 (0.024↑)	-0.005 (0.005↑)	0.007 (0.005↓)	0.917 (0.021↑)	0.913 (0.029↑)	-0.026 (0.013↑)	0.027(0.013↓)
IPO (Azar et al. 2023) +	**ER**	0.778±0.008	0.842±0.007	-0.029±0.003	0.029±0.003	0.899±0.010	0.894±0.012	-0.028±0.018	0.029±0.019
	IMBO	0.909 (0.031↑)	0.876 (0.034↑)	-0.011 (0.018↑)	0.012 (0.017↓)	0.929 (0.030↑)	0.927 (0.033↑)	-0.012 (0.016↑)	0.012 (0.017↓)
DPO (Rafailov et al. 2023) +	**ER**	0.804±0.008	0.860±0.004	-0.025±0.004	0.028±0.003	0.906±0.022	0.893±0.039	-0.023±0.015	0.027±0.016
	IMBO	0.842 (0.038↑)	0.901 (0.041↑)	-0.006 (0.019↑)	0.011 (0.019↓)	0.949 (0.043↑)	0.940 (0.047↑)	-0.009 (0.014↑)	0.010 (0.017↓)

and FM). Take DPO + ER/IMBO as an example, under the TIL-HF benchmark, IMBO achieves a significant AA gain of 0.038 (0.804 → 0.842) and AIA improvement of 0.041 (0.860 → 0.901), while reducing 67.9% of forgetting measure (FM: 0.028 → 0.011) . Similarly, under the DIL-HF benchmark, IMBO boosts AA by 0.043 (0.906 → 0.949) and AIA by 0.047 (0.893 → 0.940) and mitigates 63.0% of forgetting measure (FM: 0.027 → 0.010). In addition, IBMO shows improvements over several other baseline methods, such as PRO/RRHF/IPO+ER. These improvements highlight IMBO's ability to balance task adaptation and knowledge retention through dynamic sample selection and parameter-efficient updates.

4.3 Ablation Study

In this section, we perform an ablation experiment to assess the impact of the following factors on our method: 1) The impact of our proposed prompt and response influence function to construct a memory buffer. 2) The impact of learning constraints by omitting $\mathcal{J}_{\mathcal{C}_i}(\theta)$ in Eq. (15). 3) The consequences of employing the LD method by enforcing $\boldsymbol{\lambda} \equiv 1$. 4) The impact of the Bregman Divergence on the learning constraints.

Results Analysis The ablation experiments in Table 3 demonstrate the effectiveness of each core module in the IMBO framework. The influence function (IF) is crucial for selecting high-impact historical samples; replacing it with random sampling ("w/o IF") leads to performance drops in TIL-HF (AA: 0.842 → 0.806, AIA: 0.901 → 0.859). The constraint term $\mathcal{J}_{\mathcal{C}_i}(\theta)$ is essential for preserving old knowledge—removing it causes severe degradation (TIL-HF AA: 0.822, AIA: 0.855), highlighting its role in preventing catastrophic forgetting. Lagrangian duality (LD) also improves performance; fixing $\boldsymbol{\lambda} \equiv 1$ ("w/o LD") reduces AA to 0.800 and worsens BWT (-0.027). Since the optimization objectives of different tasks are sensitive to the values of $\boldsymbol{\lambda}$, setting $\boldsymbol{\lambda} \equiv 1$ uniformly across all tasks reduces the adaptability of the objective function. Consequently, the performance under this setting is even worse compared to the case w/o $\mathcal{J}_{\mathcal{C}_i}(\theta)$. Replacing Bregman divergence with KL divergence lowers the AA metric from 0.842 to 0.829 in TIL-HF, while the forgetting measure increases by 100% (FM: 0.011 → 0.022), demonstrating the superiority of Bregman divergence in balancing flexibility and stability. In DIL-HF, removing IF or the constraint term still causes significant drops (AA: 0.949 → 898, AIA: 0.940 → 0.920). For the FM metric, different ablation experiments have a significant impact, increasing

Table 3: The ablation experiments for IMBO. w/o $\mathcal{J}_{\mathcal{C}_i}(\theta)$ means learning new tasks without using memory buffer. w/o IF means using a construct memory buffer by random sampling. $D_\Phi \to KL$ denotes using KL Divergence instead of Bregman Divergence.

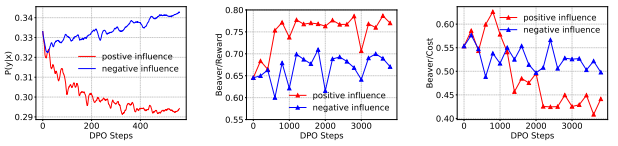
Method	TIL-HF Benchmark				DIL-HF Benchmark			
	Overall performance		Memory stability		Overall performance		Memory stability	
	AA (↑)	AIA (↑)	BWT (↑)	FM (↓)	AA (↑)	AIA (↑)	BWT (↑)	FM (↓)
DPO+IMBO	**0.842**±0.002	**0.901**±0.003	**-0.006**±0.001	**0.011**±0.002	**0.949**±0.004	**0.940**±0.003	**-0.009**±0.002	**0.010**±0.002
w/o IF	0.806±0.006	0.859±0.005	-0.021±0.004	0.021±0.004	0.898±0.004	0.920±0.003	-0.026±0.002	0.029±0.002
w/o $\mathcal{J}_{\mathcal{C}_i}(\theta)$	0.822±0.007	0.855±0.006	-0.068±0.006	0.062±0.004	0.926±0.004	0.928±0.003	-0.051±0.003	0.057±0.003
w/o LD ($\lambda \equiv 1$)	0.800±0.007	0.848±0.005	-0.027±0.004	0.030±0.003	0.910±0.004	0.900±0.003	-0.052±0.002	0.052±0.002
$D_\Phi \to KL$	0.829±0.006	0.870±0.004	-0.020±0.003	0.022±0.003	0.920±0.004	0.914±0.003	-0.041±0.002	0.046±0.002

the FM metric by 190% (w/o IF) to 470% (w/o $\mathcal{J}_{\mathcal{C}_i}(\theta)$).

5 Further Discussion of the Effectiveness of the Influence Function

Table 4: Analysis of the Impact on Recall and Model Retraining Effectiveness. We use 5% training data as memory buffer to retrain the model for comparing different memory constructing methods. (S) and (B) denote single and batch influence queries, respectively.

Memory Buffer	SafeRLHF		Stanford HP
	Reward Score (↑)	Cost Score (↓)	SteamSHP-XL (↑)
Qwen2.5-1B-Base	0.640±0.006	0.555±0.007	0.579±0.002
Full Training Data	0.821±0.003	0.389±0.005	0.725±0.004
Random	0.712±0.002	0.532±0.007	0.612±0.001
K-means	0.719±0.003	0.545±0.008	0.617±0.004
EK-FAC IF (S)	0.732±0.004	0.473±0.006	0.645±0.004
EK-FAC IF (B)	0.744±0.005	0.444±0.002	0.665±0.003
Preference IF (S)	0.761±0.003	0.432±0.002	0.675±0.002
Preference IF (B)	**0.788**±0.002	**0.408**±0.003	**0.682**±0.003



(a) $P(y|x)$ of IQs (b) Beaver Reward (c) Beaver Cost

Figure 2: Train progress on SafeRLHF by using 1000 positive or negative influential samples.

We investigate whether influence functions are effective by retraining models. Due to the substantial cost of retraining models, we choose the Qwen2.5-1.5B (Team 2024) model as the backbone. The experimental pipeline is as follows: (1) First, train an alignment model using the full dataset as the training set. (2) Use the validation set to perform influence queries, and calculate influence scores to the training samples. (3) Select the top-5% influential samples to retrain the alignment model. We compare our preference influence function in Eq. (11) with the traditional influence function in Eq. (6). We use two commonly used human preference datasets, SafeRLHF (Dai et al. 2023) and Stanford Human Preferences (SHP) (Ethayarajh, Choi, and Swayamdipta 2022), and more importantly, both of which have officially released evaluation models. We use these official evaluation models to conduct the assessment. This experiment preliminarily validates that our idea is reasonable: the most influential samples recalled by influence functions are indeed crucial for preference learning.

Table 4 shows the evaluation results of retraining the base model. Compared to the fully aligned model trained on the full dataset, among the models retrained using 5% of the data, the model trained on a dataset constructed using Preference IF performs the best. To verify the impact of the positive and negative nature of the proposed influence function on the training process, we used the influence function to select 1,000 positive and 1,000 negative influential samples for training. We recorded the Average Generation Probability (AGP) of influence queries, reward model scores, and cost model scores throughout the entire training process.

Figure 2 illustrates the impact of the recalled samples on model behavior during retraining on the SafeRLHF dataset. Figure 2 (a) shows the evolution of the AGP $P(y|x)$ for Influential Queries (IQs), where *positive influence* samples exhibit a steady increase in $P(y|x)$ over DPO training steps. Conversely, using *negative influence* samples results in a decline in generation probability. Those results indicate that using positive influential samples for IQs can strengthen alignment with desired responses, while negative influential samples have the opposite effect, which is consistent with intuition. Figure 2 (b) and (c) highlight complementary trends: the Beaver Reward Model scores rise for positive samples while costs decrease, further demonstrating the effectiveness of using top influential samples. In summary, this section validates the effectiveness of the influence function in constructing a memory buffer through retraining models.

6 Conclusion

We introduce a method for continual preference learning of LLMs, addressing the critical challenge of catastrophic forgetting in sequential task adaptation. By formulating preference alignment as a constrained optimization problem, the proposed IMBO method leverages influence functions to identify critical historical samples and employs Bregman divergence-based regularization to preserve past knowledge. Theoretically, the equivalence between sampling distribution matching and policy alignment is established, providing formal guarantees for maintaining performance across tasks. Empirical evaluations on TIL-HF and DIL-HF benchmarks demonstrate superior performance in both task incremental and domain incremental settings, with ablation studies confirming the necessity of key components like influence function sampling and Lagrangian duality optimization.

References

Azar, M. G.; Rowland, M.; Piot, B.; Guo, D.; Calandriello, D.; Valko, M.; and Munos, R. 2023. A general theoretical paradigm to understand learning from human preferences. *arXiv preprint arXiv:2310.12036*.

Bai, Y.; Jones, A.; Ndousse, K.; Askell, A.; Chen, A.; DasSarma, N.; Drain, D.; Fort, S.; Ganguli, D.; Henighan, T.; Joseph, N.; Kadavath, S.; Kernion, J.; Conerly, T.; El-Showk, S.; Elhage, N.; Hatfield-Dodds, Z.; Hernandez, D.; Hume, T.; Johnston, S.; Kravec, S.; Lovitt, L.; Nanda, N.; Olsson, C.; Amodei, D.; Brown, T.; Clark, J.; McCandlish, S.; Olah, C.; Mann, B.; and Kaplan, J. 2022. Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback. arXiv:2204.05862.

Biesialska, M.; Biesialska, K.; and Costa-jussà, M. R. 2020. Continual Lifelong Learning in Natural Language Processing: A Survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, 6523–6541. Barcelona, Spain (Online): International Committee on Computational Linguistics.

Bradley, R. A.; and Terry, M. E. 1952. Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika*, 39(3/4): 324–345.

Buzzega, P.; Boschini, M.; Porrello, A.; Abati, D.; and CALDERARA, S. 2020. Dark Experience for General Continual Learning: a Strong, Simple Baseline. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 15920–15930. Curran Associates, Inc.

Chaudhry, A.; Dokania, P. K.; Ajanthan, T.; and Torr, P. H. S. 2018. Riemannian Walk for Incremental Learning: Understanding Forgetting and Intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Dai, J.; Pan, X.; Sun, R.; Ji, J.; Xu, X.; Liu, M.; Wang, Y.; and Yang, Y. 2023. Safe RLHF: Safe Reinforcement Learning from Human Feedback. arXiv:2310.12773.

Douillard, A.; Cord, M.; Ollion, C.; Robert, T.; and Valle, E. 2020. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, 86–102. Springer.

Ethayarajh, K.; Choi, Y.; and Swayamdipta, S. 2022. Understanding Dataset Difficulty with $\mathcal{V}$-Usable Information. In Chaudhuri, K.; Jegelka, S.; Song, L.; Szepesvari, C.; Niu, G.; and Sabato, S., eds., *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 5988–6008. PMLR.

George, T.; Laurent, C.; Bouthillier, X.; Ballas, N.; and Vincent, P. 2018. Fast approximate natural gradient descent in a kronecker factored eigenbasis. *Advances in Neural Information Processing Systems*, 31.

Grosse, R.; Bae, J.; Anil, C.; Elhage, N.; Tamkin, A.; Tajdini, A.; Steiner, B.; Li, D.; Durmus, E.; Perez, E.; et al. 2023. Studying large language model generalization with influence functions. *arXiv preprint arXiv:2308.03296*.

Hammoudeh, Z.; and Lowd, D. 2024. Training data influence analysis and estimation: A survey. *Machine Learning*, 113(5): 2351–2403.

Hampel, F. R. 1974. The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346): 383–393.

Hou, S.; Pan, X.; Loy, C. C.; Wang, Z.; and Lin, D. 2019. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 831–839.

Hussain, A.; Holla, N.; Mishra, P.; Yannakoudakis, H.; and Shutova, E. 2021. Towards a robust experimental framework and benchmark for lifelong language learning. In *Thirty-fifth Conference on Neural Information Processing Systems*.

Ji, J.; Qiu, T.; Chen, B.; Zhang, B.; Lou, H.; Wang, K.; Duan, Y.; He, Z.; Zhou, J.; Zhang, Z.; Zeng, F.; Ng, K. Y.; Dai, J.; Pan, X.; O'Gara, A.; Lei, Y.; Xu, H.; Tse, B.; Fu, J.; McAleer, S.; Yang, Y.; Wang, Y.; Zhu, S.-C.; Guo, Y.; and Gao, W. 2023. AI Alignment: A Comprehensive Survey. arXiv:2310.19852.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; Hassabis, D.; Clopath, C.; Kumaran, D.; and Hadsell, R. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13): 3521–3526.

Lin, L.-J. 1992. Self-Improving Reactive Agents Based on Reinforcement Learning, Planning and Teaching. *Mach. Learn.*, 8(3–4): 293–321.

Lopez-Paz, D.; and Ranzato, M. 2017. Gradient Episodic Memory for Continual Learning. In Guyon, I.; von Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 6467–6476.

Maas, A. L.; Daly, R. E.; Pham, P. T.; Huang, D.; Ng, A. Y.; and Potts, C. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 142–150. Portland, Oregon, USA: Association for Computational Linguistics.

McCloskey, M.; and Cohen, N. J. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, 109–165. Elsevier.

Oord, A. v. d.; Li, Y.; and Vinyals, O. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Plackett, R. L. 1975. The analysis of permutations. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 24(2): 193–202.

Rafailov, R.; Sharma, A.; Mitchell, E.; Manning, C. D.; Ermon, S.; and Finn, C. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. *CoRR*, abs/1707.06347.

Song, F.; Yu, B.; Li, M.; Yu, H.; Huang, F.; Li, Y.; and Wang, H. 2023. Preference Ranking Optimization for Human Alignment. arXiv:2306.17492.

Stobbe, P.; and Krause, A. 2010. Efficient minimization of decomposable submodular functions. *Advances in Neural Information Processing Systems*, 23.

Team, Q. 2024. Qwen2.5: A Party of Foundation Models.

Völske, M.; Potthast, M.; Syed, S.; and Stein, B. 2017. TL;DR: Mining Reddit to Learn Automatic Summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, 59–63. Copenhagen, Denmark: Association for Computational Linguistics.

Yuan, Z.; Yuan, H.; Tan, C.; Wang, W.; Huang, S.; and Huang, F. 2023. RRHF: Rank Responses to Align Language Models with Human Feedback without tears. arXiv:2304.05302.

Zhang, H.; Lei, Y.; Gui, L.; Yang, M.; He, Y.; Wang, H.; and Xu, R. 2024a. CPPO: Continual Learning for Reinforcement Learning with Human Feedback. In *The Twelfth International Conference on Learning Representations*.

Zhang, H.; Zhang, Z.; Zhang, Y.; Zhai, Y.; Peng, H.; Lei, Y.; Yu, Y.; Wang, H.; Liang, B.; Gui, L.; et al. 2024b. Correcting Large Language Model Behavior via Influence Function. *arXiv preprint arXiv:2412.16451*.

Zhao, Y.; Khalman, M.; Joshi, R.; Narayan, S.; Saleh, M.; and Liu, P. J. 2023. Calibrating Sequence likelihood Improves Conditional Language Generation. In *The Eleventh International Conference on Learning Representations*.

Reproducibility Checklist

1. General Paper Structure

1.1. Includes a conceptual outline and/or pseudocode description of AI methods introduced (yes/partial/no/NA) yes

1.2. Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results (yes/no) yes

1.3. Provides well-marked pedagogical references for less-familiar readers to gain background necessary to replicate the paper (yes/no) yes

2. Theoretical Contributions

2.1. Does this paper make theoretical contributions? (yes/no) yes

If yes, please address the following points:

2.2. All assumptions and restrictions are stated clearly and formally (yes/partial/no) yes

2.3. All novel claims are stated formally (e.g., in theorem statements) (yes/partial/no) yes

2.4. Proofs of all novel claims are included (yes/partial/no) yes

2.5. Proof sketches or intuitions are given for complex and/or novel results (yes/partial/no) yes

2.6. Appropriate citations to theoretical tools used are given (yes/partial/no) yes

2.7. All theoretical claims are demonstrated empirically to hold (yes/partial/no/NA) yes

2.8. All experimental code used to eliminate or disprove claims is included (yes/no/NA) yes

3. Dataset Usage

3.1. Does this paper rely on one or more datasets? (yes/no) yes

If yes, please address the following points:

3.2. A motivation is given for why the experiments are conducted on the selected datasets (yes/partial/no/NA) yes

3.3. All novel datasets introduced in this paper are included in a data appendix (yes/partial/no/NA) NA

3.4. All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no/NA) NA

3.5. All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations (yes/no/NA) yes

3.6. All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available (yes/partial/no/NA) yes

3.7. All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisficing (yes/partial/no/NA) yes

4. Computational Experiments

4.1. Does this paper include computational experiments? (yes/no) yes

If yes, please address the following points:

4.2. This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting (yes/partial/no/NA) yes

4.3. Any code required for pre-processing data is included in the appendix (yes/partial/no) yes

4.4. All source code required for conducting and analyz-

ing the experiments is included in a code appendix (yes/partial/no) partial

4.5. All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no) partial

4.6. All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from (yes/partial/no) partial

4.7. If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results (yes/partial/no/NA) yes

4.8. This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks (yes/partial/no) yes

4.9. This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics (yes/partial/no) yes

4.10. This paper states the number of algorithm runs used to compute each reported result (yes/no) yes

4.11. Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information (yes/no) yes

4.12. The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank) (yes/partial/no) no

4.13. This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments (yes/partial/no/NA) yes