Stackelberg Learning from Human Feedback: Preference Optimization as a Sequential Game

Barna Pásztor[*]
ETH Zürich

Thomas Kleine Buening
The Alan Turing Institute

Andreas Krause
ETH Zürich

Abstract

We propose Stackelberg Learning from Human Feedback (SLHF), a new framework for preference optimization. SLHF frames the alignment problem as a sequential-move game between two policies: a Leader, which commits to an action, and a Follower, which responds conditionally on the Leader's action. This formulation departs from prior approaches such as Reinforcement Learning from Human Feedback (RLHF), which rely on assigning a scalar reward value to each action, and Nash Learning from Human Feedback (NLHF), which seek to compute a Nash equilibrium. SLHF decomposes preference optimization into a refinement problem for the Follower and an optimization problem against an adversary for the Leader. The sequential structure of SLHF naturally enables *test-time improvement*, as the Follower learns to refine the Leader's actions, and these refinements can be leveraged through iterative sampling. We compare the solution concepts of SLHF, RLHF and NLHF, and lay out key advantages in consistency, data sensitivity, and robustness to intransitive preferences. Our experiments demonstrate that SLHF effectively aligns large language models with diverse, potentially intransitive, human preferences, and its test-time improvement generalizes across models without further training.

1 Introduction

Reinforcement Learning from Human Feedback (RLHF) has become the dominant paradigm for aligning Large Language Models (LLMs) with human preferences [11, 26]. The standard pipeline involves two stages: first, a reward model is trained on a dataset of pairwise human comparisons, and second, a policy is optimized via reinforcement learning to maximize this reward [13]. While this approach has achieved impressive results, it hinges on a critical, and often flawed, assumption: that diverse human preferences can be faithfully represented by a single, real-valued reward function. In practice, this assumption often fails. Scalar reward models cannot capture intransitive structures, and even when preferences are transitive, widely used formulations such as the Bradley-Terry model [9] can cause the learned rewards to diverge from the actual preferences [8].

A common alternative to relying on reward models and the Bradley-Terry assumption are preference models [25], which directly predict the probability of one action being preferred over another in a given context. However, such models might exhibit preference cycles so that the notion of policy optimality becomes ambiguous. Nash Learning from Human Feedback (NLHF) proposes the Nash Equilibrium (NE) as a solution to this problem by framing preference optimization as a two-player simultaneous-move game, where the Nash equilibrium (NE) corresponds to a typically stochatic

[*]Corresponding author. Email: `barna.pasztor@ai.ethz.ch`

policy whose actions are preferred to any other policy's actions on average [34]. We extend this game-theoretic framing by introducing Stackelberg Learning from Human Feedback (SLHF). In contrast to NLHF, SLHF formulates preference optimization as a *sequential-move* game inspired by Stackelberg dynamics [49]: a Leader first commits to an action, and a Follower then responds conditional on the Leader's action. This asymmetric structure offers a distinct advantage as the Follower learns to refine a given action, an easier objective than optimizing the action directly or against a non-stationary opponent as in NLHF. Consequently, the Leader's objective becomes anticipating the outcome of this refinement process and choose the action least exploitable by the Follower.

Instead of a stochastic policy, SLHF proposes a principled method to cover all actions in the preference cycle that we call *test-time improvement*: the ability to refine model outputs at inference time via repeated sampling. This is particularly valuable when a model, trained on preferences aggregated across diverse annotators, must ultimately align with an individual's taste, especially since such aggregation can induce intransitive preference cycles (Section 4). SLHF realizes this refinement through its two components: the Leader policy produces an initial response, and the Follower policy generates refined responses conditional on the previous output. Unlike sampling from a static distribution, this produces a sequence of outputs that can efficiently explore the preference space. Crucially, this allows for performance gains through test-time computation alone, without any need for additional training or external feedback.

In summary, our main contributions are:

- We introduce Stackelberg Learning from Human Feedback (SLHF), a preference optimization framework that models alignment as a two-player sequential game. We formalize this game over a learned pairwise preference model and show that SLHF admits a unique Stackelberg equilibrium under standard regularity assumptions (Section 4).
- We propose STACKELBERGGDA, an algorithm that approximates the Stackelberg equilibrium via two-timescale gradient descent ascent. Our algorithm benefits from online RL optimization without the need of an explicit reward model or expensive inference with a mixture policy (Section 5).
- Our experimental results show that the Follower, conditioned on the Leader's output, consistently outperforms both RLHF and NLHF baselines, whereas the Leader performs similarly to the approximated Nash policy. Furthermore, we show that the Follower generalizes across models, improving outputs from independently trained policies without additional fine-tuning (Section 6).

2 Related Work

Reinforcement Learning from Human Feedback (RLHF). RLHF aims to optimize a policy based on human preferences by learning from comparisons or rankings rather than explicit numeric rewards [60, 26]. Most RLHF methods follow a two-step pipeline introduced by Christiano et al. [13]: first, a reward model is trained from pairwise comparisons; then, this model is used as a proxy reward function for policy optimization, typically using PPO [46]. This framework has been widely adopted for tasks such as text summarization [50], question answering [35, 33], and language model fine-tuning [68, 6, 20, 39]. Recent work has also combined these two steps into a bilevel optimization problem to jointly optimize reward model and policy [48, 53, 32].

Limitations of Reward Modeling. Most RLHF methods reduce preference learning to scalar reward estimation, typically using the Bradley-Terry model [9]. While effective for transitive, single-objective preferences, such models fail to represent intransitive structures and may even misrank transitive ones under model misspecification [8]. As a result, they struggle to capture the diversity and ambiguity of real human preferences [11]. These issues are compounded during optimization: the final policy can be highly sensitive to the distribution of training comparisons [34] and may collapse to a single preference mode under continued training [63]. Interestingly, even when preferences are elicited from LLMs, as in the AlpacaEval framework [17], the feedback can exhibit intransitive preference structures [64]. In contrast, we work directly with pairwise preference signals without imposing an underlying reward model. This enables optimization with respect to diverse and potentially intransitive feedback.

Preference Optimization. To address the limitations of reward modeling in RLHF, IPO [5] extends Direct Preference Optimization (DPO) [43] by optimizing for the win rate against a reference policy. Nash Learning from Human Feedback (NLHF) casts the learning problem as a two-player simultaneous-move game and introduces NASH-MD-PG and NASH-EMA-PG to approximate the Nash Equilibrium (NE) of a learned preference model via mirror descent [34]. Subsequent work has extended this perspective, proposing various algorithms to optimize for (approximate) NE, including ONLINE-IPO [10], SPPO [62], SPO [51], INPO [66], DNO [44], RSPO [52], NASH-RS [30], and MPO [56]. These methods typically converge to mixed strategies unless one option is majority-preferred, even in non-regularized settings [30], due to the symmetry of simultaneous-move games. In contrast, we formulate preference optimization as a sequential-move game, inspired by Stackelberg (Leader-Follower) dynamics. Here, the Leader commits first, and the Follower responds conditionally on the Leader's action. This asymmetric structure leads to a different solution concept than the NE, admitting deterministic policies in the non-regularized limit.

Test-time Preference Improvement. Improving the capabilities of LLMs through additional computation at test-time has received significant attention recently, especially in verifiable domains such as coding or mathematics [59]. Closest to our work are self-correction algorithms that aim to improve their responses without external feedback at test-time. A natural approach to self-correction is to provide instructions only without further training, which, however, can lead to performance degradation [24, 67, 54, 41]. Other work on training models for self-correction either assumes human or AI revisions [45, 41] or a reward function scoring responses [58, 2, 65, 28]. Similarly to SLHF, Kumar et al. [28] also propose to train an LLM in a sequential manner, however, assume a reward model and train in two-stages instead of a single loop. On the other hand, STACKELBERGGDA can self-improve on general preferences and train in a single loop.

3 Problem Statement

We consider a contextual bandit setting with a finite set of contexts $\mathcal{X}$ and actions $\mathcal{Y}$. The contexts x are sampled from a fixed and known distribution $\rho \in \Delta_{\mathcal{X}}$, where $\Delta_{\mathcal{X}}$ denotes the probability simplex over $\mathcal{X}$. A policy $\pi : \mathcal{X} \to \Delta_{\mathcal{Y}}$ maps each context $x \in \mathcal{X}$ to a discrete probability distribution $\pi(\cdot \mid x) \in \Delta_{\mathcal{Y}}$, where $\Delta_{\mathcal{Y}}$ is the probability simplex over $\mathcal{Y}$. We let $\Pi \coloneqq \{\pi\colon \mathcal{X} \to \Delta_{\mathcal{Y}}\}$ denote the set of all policies. In the case of LLM fine-tuning, $\mathcal{X}$ corresponds the set of prompts, $\mathcal{Y}$ to the set of responses, and π to the LLM.

Preferences are observed as pairwise comparisons between two actions $y \in \mathcal{Y}$ and $y' \in \mathcal{Y}$ given a context $x \in \mathcal{X}$ and provided by a human or AI annotator a. The annotator is drawn at random from a distribution $\nu \in \Delta_{\mathcal{A}}$, where $\mathcal{A} \coloneqq \{1, \dots, A\}$ denotes the set of all annotators. The feedback of annotator a is then given by $y^w \succ_a y^l$, where y^w and y^l are the preferred and non-preferred actions, respectively. Let the *preference function* $p(y \succ y' \mid x)$ define the probability that y is preferred over y' given x, where the randomness is due to the choice of annotator, that is, the preference function is given by $p(y \succ y' \mid x) \coloneqq \mathbb{E}_{a\sim\nu}\big[\mathbb{1}\{y \succ_a y' \mid x\}\big]$.[2] Slightly overloading notation, we define the preference over two policies π and π' given context x as

$$p(\pi \succ \pi' \mid x) \coloneqq \mathbb{E}_{y\sim\pi(\cdot|x),y'\sim\pi'(\cdot|x)}\big[p(y \succ y' \mid x)\big]. \tag{1}$$

Averaging over the context distribution ρ, the expected probability that π is preferred over π' is then given by $p(\pi \succ \pi') \coloneqq \mathbb{E}_{x\sim\rho}[p(\pi \succ \pi' \mid x)]$.

3.1 Background on Existing Solution Concepts and Approaches

For any context $x \in \mathcal{X}$, if there exists an action $y^\star_x \in \mathcal{Y}$ such that $p(y^\star_x \succ y \mid x) \geq 0.5$ for all $y \in \mathcal{Y}$, we call $y^\star_x$ a *Condorcet winner* for x. If preferences over actions are transitive, i.e., there exists a total order over actions, then a Condorcet winner exists and is the first in that order. If every context $x \in \mathcal{X}$ admits a Condorcet winner, one can simply define the optimal policy $\pi^\star$ by setting $\pi^\star(x) = y^\star_x$ for all $x \in \mathcal{X}$. However, if preferences form cycles or exhibit other irregularities, a Condorcet winner may not exist. In Section 4.1, we show that even if all annotators in ν have transitive preferences, the aggregated preferences can still exhibit cycles. While defining the optimal policy $\pi^\star$ is straightforward when a Condorcet winner exists, in the presence of diverse and cyclic preferences, the concept of optimality is ambiguous. Previous work typically adopts one of two approaches, each associated with a distinct notion of optimality, which we briefly review below.

[2]This can be straightforwardly generalized to the case where each annotator is associated with a probabilistic preference function $p_a(y \succ y' \mid x)$ that describes the probability that annotator a prefers y' over y given x.

Reinforcement Learning from Human Feedback (RLHF). RLHF as proposed by Christiano et al. [13] and adapted to language modeling by Ziegler et al. [68] splits preference optimization into two steps. First, it assumes that the preference function p follows the Bradley-Terry model [9] so that

$$p(y \succ y' \mid x) = \sigma(r(x, y) - r(x, y')), \tag{2}$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ is the sigmoid function and $r : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ is a real-valued reward function. The reward function r is unknown so that an estimator $\hat{r}$ is used that maximizes the log-likelihood of a given dataset $\mathcal{D} = \{(x_i, y_i^w, y_i^l)\}_{i=1}^N$.

In a second step, the policy $\pi^\star$ is chosen to maximize the expected reward with respect to $\hat{r}$ regularized by the Kullback-Leibler (KL) divergence against a fixed reference policy $\pi^{\text{ref}} \in \Pi$:

$$\pi^\star \in \arg\max_{\pi \in \Pi} \mathbb{E}_{x\sim\rho}\Big[\mathbb{E}_{y\sim\pi(\cdot|x)}\big[\hat{r}(x, y)\big] - \tau \text{KL}_x(\pi \,\|\, \pi^{\text{ref}})\Big], \tag{3}$$

where $\tau \geq 0$ and $\text{KL}_x(\pi \,\|\, \pi^{\text{ref}})$ is computed between $\pi(\cdot \mid x)$ and $\pi^{\text{ref}}(\cdot \mid x)$. Under the Bradley-Terry assumption, Equation (3) admits a unique closed-form solution [43]. However, additive score models like Bradley-Terry are provably limited in expressing cyclic or intransitive preference structures, which have been empirically observed in both strategic games [8] and human preference data [3, 11]. Consequently, the optimal policy $\pi^\star$ depends critically on the data distribution in the training set $\mathcal{D}$, especially its sampling biases [34], which we elaborate more on in Section 4.1.

Nash Learning from Human Feedback (NLHF). NLHF provides an alternative approach to optimizing preferences by formulating the problem as a simultaneous-move game between two policies $\pi \in \Pi$ and $\pi' \in \Pi$ [34]. The optimization problem is given by:

$$\max_{\pi \in \Pi} \min_{\pi' \in \Pi} \mathbb{E}_{x\sim\rho}\Big[p(\pi \succ \pi' \mid x) - \tau \text{KL}_x(\pi \,\|\, \pi^{\text{ref}}) + \tau \text{KL}_x(\pi' \,\|\, \pi^{\text{ref}})\Big]. \tag{4}$$

The solution to Equation (4) is a *Nash equilibrium* $(\pi^\star, \pi'^\star)$, where neither side can be improved unilaterally. The existence and uniqueness of this equilibrium follows from the concave-convex nature of the objective [34]. Unlike RLHF, which relies on a fixed dataset $\mathcal{D}$ of labeled comparisons and is sensitive to its sampling distribution, NLHF collects feedback during training and does not assume a specific preference structure. This makes NLHF better suited to situations where online feedback is available and preferences are diverse. Notably, if there is no action that is majority-preferred, the NE must be mixed even in the absence of KL regularization [30]. This inherent stochasticity can be undesirable in applications where consistency and reliability are critical.

4 Stackelberg Learning from Human Feedback (SLHF)

We now present Stackelberg Learning from Human Feedback (SLHF), a novel perspective on the preference optimization problem. Inspired by Stackelberg games [49], we adopt a game-theoretic approach and model the optimization as a *sequential-move* game between two players: the *Leader* and the *Follower*. In our formulation, the Leader first observes the context x and chooses its action $y \sim \pi(\cdot \mid x)$. Then, the Follower observes both the context x and the Leader's action y and chooses its own action $y' \sim \omega(\cdot \mid x, y)$. The Follower's policy ω is chosen from the set of policies that are conditioned on both inputs $\Omega = \{\omega : \mathcal{X} \times \mathcal{Y} \to \Delta_\mathcal{Y}\}$. Formally, the optimization problem given two reference policies $\pi^{\text{ref}} \in \Pi$ and $\omega^{\text{ref}} \in \Omega$ is defined as:

$$\max_{\pi\in\Pi} \min_{\omega\in\Omega} \mathbb{E}_{x\sim\rho}\Big[\mathbb{E}_{y\sim\pi(\cdot|x)}\big[\mathbb{E}_{y'\sim\omega(\cdot|x,y)}[p(y \succ y' \mid x)] + \tau^F \text{KL}_{x,y}(\omega \,\|\, \omega^{\text{ref}})\big] - \tau^L \text{KL}_x(\pi \,\|\, \pi^{\text{ref}})\Big] \tag{5}$$

where $\tau^L, \tau^F \geq 0$ are player-specific regularization parameters, and the Follower's regularization term, $\text{KL}_{x,y}(\omega \,\|\, \omega^{\text{ref}})$, is computed between $\omega(\cdot \mid x, y)$ and $\omega^{\text{ref}}(\cdot \mid x, y)$. Note that in the absence of regularization, i.e., $\tau^L = \tau^F = 0$, Equation (5) defines a sequential-move constant-sum game.

SLHF decomposes the preference optimization task into two distinct roles, setting it apart from single-policy methods like RLHF and NLHF. The Follower leverages its informational advantage of observing the Leader's committed action. This simplifies its task to learning a specialized refinement policy that finds the best response to a known output, rather than optimizing against a non-stationary opponent. The Leader, in turn, must anticipate this refinement and learn to produce initial actions that are robust, meaning they remain highly preferred even after the Follower's improvements. In

Table 1: Transitive individual annotator preferences over three options: $\{A, B, C\}$.

Preference Relationship	Proportion
$A \succ B \succ C$	α
$B \succ C \succ A$	β
$C \succ A \succ B$	γ

Table 2: The preference function derived from the individual rankings in Table 1.

	A	B	C
A	0.5	$1-\beta$	α
B	β	0.5	$1-\gamma$
C	$1-\alpha$	γ	0.5

Section 4.1, we illustrate that when preferences form a cycle and no Condorcet winner exists, the Leader selects the least exploitable action, while the Follower traverses the preference cycle, covering all plausibly optimal actions with minimal samples.

It is worth highlighting that Equation (5) differs from many common Stackelberg formulations studied in the algorithmic game theory literature [14], where the Leader commits to a stochastic policy π, and the Follower selects a response ω conditional on π but *without* access to the realization $y \sim \pi(\cdot \mid x)$. In contrast, the Follower gets to condition on y in SLHF, which provides strictly more information when π is stochastic and yields an easier stationary problem.

In line with previous results that the RLHF optimization problem (3) admits a closed-form solution, we show that there exists a unique solution to the SLHF problem (5). The proof is deferred to the Appendix A.1.

Proposition 1. *Let $\tau^L, \tau^F > 0$ and suppose that $\pi^{\text{ref}}(y \mid x) > 0$ for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$. For any preference function $p(y \succ y' \mid x)$, there exists a unique solution $(\pi^\star, \omega^\star)$ to the preference optimization problem in Equation (5).*

The solution $(\pi^\star, \omega^\star)$ is called a *Stackelberg equilibrium*. It is folklore in the algorithmic game theory literature that there exists a deterministic Stackelberg equilibrium when the Leader's realized action is observed by a best responding Follower, as there always exists a deterministic best response for the Follower and therefore there is no point in randomizing for the Leader. This stands in contrast to the NE, which is in general stochastic.

Remark 2. *For any preference function $p(y \succ y' \mid x)$, the SLHF optimization problem (5) has a deterministic solution $(\pi^\star, \omega^\star)$ when $\tau^L = \tau^F = 0$. Note that this solution may not necessarily be unique (due to the lack of regularization).*

For completeness, we provide a proof for Remark 2 in Appendix A.2.

Test-time Improvement. Motivated by applications such as text summarization, open-ended generation, and media creation, where users can reject outputs and ask for a new sample, we introduce the concept of *test-time improvement* for preference optimization. While previous works address this problem in verifiable domains and for reward models (see Section 2), to the best of our knowledge, we are the first to consider it in the context of general preference models. We assume that at test-time, a single user (annotator) $a \sim \nu$ and a context x are sampled. The user then has the option to resample the action until they receives one that suits their preference, analogously to the $pass@k$ metric for verifiable domains. This is a non-trivial task as the models are trained on the average preferences of the population ν while at test-time the task is to optimize for a single user. The Stackelberg solution $(\pi^\star, \omega^\star)$ naturally extends to this setting: it returns an initial action $y_1 \sim \pi^\star(\cdot \mid x)$, and subsequent actions are sampled as $y_i \sim \omega^\star(\cdot \mid x, y_{i-1})$ for $i = 2, 3, \dots$, thereby progressively improving the output. We illustrate test-time improvement qualitatively in Section 4.1 and provide experimental evidence in Section 6, demonstrating that LLMs fine-tuned with StackelbergGDA exhibit this behavior.

4.1 Comparison of Solution Concepts

Before we describe how to approximate the Stackelberg equilibrium, we present the differences between RLHF, NLHF, and SLHF at the example of the so-called Condorcet paradox [15].

Consider a problem with $|\mathcal{X}| = 1$ and $\mathcal{Y} = \{A, B, C\}$, and let the population of annotators $\mathcal{A}$ consist of three distinct types. Table 1 defines the annotators' preferences over $\mathcal{Y}$. Each type of annotator has a strict preference ranking and the proportion of the types according to the distribution over annotators ν is such that $\alpha + \beta + \gamma = 1$. Table 2 shows the aggregated preferences of the whole population. For example, row A column B states that $1-\beta$ of the population prefers A over B, i.e., $p(A \succ B) = 1-\beta$. A common example is to choose $\alpha = \beta = \gamma = 1/3$, which leads to a cyclic relationship between three actions in which $A \succ B \succ C$ but $C \succ A$. Hence, there exists no Condorcet winner in this case.[3] This example is often referred to as the *Condorcet paradox*, because the annotators individually have transitive preferences (Table 1), but their aggregated preferences form a cycle (Table 2). For ease of presentation, we consider a non-regularized problem in the rest of this section so that $\tau = \tau^L = \tau^F = 0$.

RLHF Solution. Our first observation is that the estimated reward function $\hat{r} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ depends heavily on the sampling distribution of the dataset $\mathcal{D}$ used to estimate $\hat{r}$. Consider a case where $\mathcal{D}$ consists only of observations between pairs (A, B) and (B, C) and the pair (A, C) is not included. Since $A \succ B$ and $B \succ C$, the reward estimate that maximizes the log-likelihood satisfies $\hat{r}(A) > \hat{r}(B) > \hat{r}(C)$. Hence, the RLHF solution to Equation (3) is $\pi^\star(A) = 1$. However, similar outcomes can occur for actions B and C when only pairs $(A, C), (B, C)$ or $(A, C), (A, B)$ are sampled, respectively. This illustrates a key limitation of RLHF, namely that its solutions are sensitive to the specific comparisons present in $\mathcal{D}$.

Nash Equilibrium. Under the assumption that $\alpha, \beta, \gamma \le 0.5$, the NE of the matrix game defined in Table 2 is given by $\pi^\star(A) = 1 - 2\gamma$, $\pi^\star(B) = 1 - 2\alpha$, $\pi^\star(C) = 1 - 2\beta$. In the special case of $\alpha = \beta = \gamma = 1/3$, the NE is uniform over $\mathcal{Y}$, i.e., it has the highest possible entropy. Even though this solution does not depend on the distribution of any dataset, it leads to a stochastic policy that might be undesirable depending on the application area.

Stackelberg Equilibrium. Since the Follower gets to observe the Leader's action, it can play a best response $\omega(\cdot \mid y)$, which we can in turn use to derive the Leader's optimal policy $\pi^\star$:

$$\omega^\star(\cdot \mid y) = \begin{cases} C & \text{if } y = A \text{ w.p. } 1 \\ A & \text{if } y = B \text{ w.p. } 1 \\ B & \text{if } y = C \text{ w.p. } 1 \end{cases} \qquad \pi^\star(\cdot) = \begin{cases} A & \text{if } \alpha > \max\{\beta, \gamma\} \text{ w.p. } 1 \\ B & \text{if } \beta > \max\{\alpha, \gamma\} \text{ w.p. } 1 \\ C & \text{if } \gamma > \max\{\alpha, \beta\} \text{ w.p. } 1 \end{cases}.$$

Unlike RLHF, this solution does not rely on any offline dataset $\mathcal{D}$. Furthermore, in contrast to NLHF, it admits a deterministic solution in the absence of KL regularization. However, when $\alpha = \beta = \gamma = 1/3$, the Leader is indifferent among the three actions, meaning that any distribution over A, B, C constitutes a Stackelberg equilibrium. This includes the Nash equilibrium $\pi^\star(A) = \pi^\star(B) = \pi^\star(C) = 1/3$.

Test-time Improvements. Finally, we consider how each solution can be improved at test-time with respect to the preferences of a single annotator $a \in \mathcal{A}$. Suppose $\alpha = \beta = \gamma = 1/3$, and without loss of generality, let a belong to the first annotator type with preference $A \succ B \succ C$. The RLHF solution may return A, which aligns with a's most preferred action. However, as previously discussed, the outcome of RLHF depends on the dataset $\mathcal{D}$; it could just as easily produce B or C. Since the RLHF policy is deterministic, repeated sampling does not change this outcome. In contrast, the NLHF solution is uniform over A, B, C, so the probability of sampling A in a single draw is $1/3$. By sampling N times, the probability of observing at least one A is $1 - (2/3)^N$, which equals 56% for $N = 2$ and 70% for $N = 3$. The Stackelberg solution starts similarly: the first action is sampled from the Leader's potentially uniform policy. However, subsequent actions are drawn from the Follower's policy, i.e., $y_i \sim \omega^\star(\cdot \mid x, y_{i-1})$ for $i \ge 2$. In this case, the probability of sampling A within $N = 2$ steps increases to 67%, and for $N = 3$, the entire preference cycle is traversed regardless of the Leader's initial choice.

5 Stackelberg Gradient Descent Ascent (StackelbergGDA)

We now introduce StackelbergGDA, a two-timescale Gradient Descent-Ascent (GDA) algorithm tailored to the sequential preference optimization problem in Equation (5). StackelbergGDA performs simultaneous ascent on the Leader's objective and descent on the Follower's, but with a

[3] Other common optimality conditions in social choice theory are also undefined for this case, e.g., the Borda or plurality winner.

User: `<user_prompt>`
Assistant:

(a) Prompt received as the Leader agent

User: `<user_prompt>`
Assistant: `<leader_response>`
User: Improve the previous answer!
Assistant:

(b) Prompt received as the Follower agent

Figure 1: Prompt templates used to train a single-model for both Leader and Follower completions.

larger step size for the Follower. This separation of timescales enables the Follower to approximate its best response to the slowly evolving Leader, promoting stable convergence. Let

$$f(\pi,\omega) := \mathbb{E}_{x\sim\rho, y\sim\pi(\cdot|x), y'\sim\omega(\cdot|x,y)}\left[p(y \succ y' \mid x) + \tau^F \mathrm{KL}_{x,y}\left(\omega \,\|\, \omega^{\mathrm{ref}}\right) - \tau^L \mathrm{KL}_x\left(\pi \,\|\, \pi^{\mathrm{ref}}\right)\right] \quad (6)$$

denote the objective of the optimization in Equation (5). STACKELBERGGDA performs simultaneously gradient ascent and descent update steps on π and ω with step size α^L and α^F, respectively, to find the $\max\min$ solution to f defined in Equation (6). It is a two-timescale algorithm as we choose $\alpha^F > \alpha^L$ resulting in ω adapting faster than π. After each gradient step, we restore feasibility by projecting both π and ω back onto their respective probability simplexes. We denote the two-timescale coefficient as $\kappa = \frac{\alpha^F}{\alpha^L}$.

Observe that $f(\pi,\omega)$ is concave-convex in π and ω, respectively.[4] While standard gradient descent-ascent with equal learning rates is known to converge in this setting [27, 12, 37, 4, 36], we instead adopt a two-timescale variant. This choice is motivated by its stronger convergence guarantees in more general nonconvex-concave regimes [29], as well as its empirical success in both Actor-Critic methods [40] and the training of Generative Adversarial Networks [22]. This becomes especially valuable in the next section, where we describe the practical implementation of STACKELBERGGDA for large state and action spaces and parameterized policies.

Scalable Implementation of STACKELBERGGDA for LLM Fine-Tuning. When working with a large context and action spaces $\mathcal{X}$ and $\mathcal{Y}$, for example, when fine-tuning LLMs, optimizing over Π and Ω becomes intractable. To address this challenge, π and ω can be parametrized and the gradients estimated from batches. Most importantly for LLM fine-tuning, the Leader and the Follower can share the same parametrization by using the prompt template shown in Figure 1, thereby reducing the memory requirements. Details and the pseudocode for this practical implementation can be found in Appendix B.

6 Experiments

We experimentally evaluate and compare the different solution concepts in the context of fine-tuning large language models to align with human preferences. Our results show that the Leader model trained with STACKELBERGGDA achieves comparable preference scores to NASH-MD-PG [34] and higher scores than RLOO [1], which serve as representative algorithms of the NLHF and RLHF frameworks, respectively. Moreover, the Follower model significantly improves upon the Leader's responses, and this improvement generalizes to outputs from other models without requiring additional fine-tuning. Throughout this section, we refer to the context x as a *prompt* and the action y as a *response*. We provide the key results on fine-tuning in Section 6.1 and test-time improvements in Section 6.2. We defer our results on iterative improvements to Appendix D.2, ablations on the hyperparameter κ to Appendix D.3, and scaling to larger models to Appendix D.4.

Experimental Setup. We use the HELPSTEER2 dataset [57], which contains 11,826 human-annotated single-turn dialogues, each rated by human annotators along five attributes: helpfulness, correctness, coherence, complexity, and verbosity. We treat these attributes as distinct annotators, denoted by the set $\mathcal{A}$, and define ν as a uniform distribution over $\mathcal{A}$. For each attribute $a \in \mathcal{A}$, we estimate a reward function $\hat{r}_a$ using the Bradley-Terry model. The overall preference function p is then defined as

$$p(y \succ y' \mid x) = \frac{1}{|\nu|}\sum_{a\in\nu} \mathbb{1}\{\hat{r}_a(x,y) \geq \hat{r}_a(x,y')\}. \quad (7)$$

[4]This follows from results in Munos et al. [34]. For completeness, we provide a formal proof in Appendix A.3.

Table 3: Pairwise preference comparisons between the responses of QWEN2.5-0.5B, RLOO, NASH-MD-PG, and STACKELBERGGDA algorithms. Each cell represents the preference model's average score for the row algorithm over the column algorithm.

	QWEN2.5-0.5B	RLOO	NASH-MD-PG	STACKELBERGGDA	
				LEADER	FOLLOWER
QWEN2.5-0.5B	0.000	0.407	0.279	0.266	0.200
RLOO	0.593	0.000	0.393	0.387	0.344
NASH-MD-PG	**0.721**	0.607	0.000	0.497	0.406
STACKELBERGGDA -LEADER	**0.734**	0.613	**0.503**	0.000	0.395
STACKELBERGGDA -FOLLOWER	**0.800**	**0.656**	**0.594**	**0.605**	0.000

Each reward function is trained independently, initialized from the QWEN2.5-1.5B[5] model with a single linear head. Although each $\hat{r}_a$ is transitive, their aggregation into p can produce cycles, analogous to the Condorcet paradox discussed in Section 4.1. Further details on the preference model specification and the resulting intransitivity are provided in Appendix D.1. Our design choice to model preferences as an aggregation of separate reward functions was motivated to ensure the richness of preferences and provide detailed analysis of repeated improvements in Appendix D.2. We note that any method that models the preference function p is compatible with SLHF such as LLM-as-a-judge [21] or preference models trained for pairwise comparison [25].

Compared Methods. We use RLOO [1] and NASH-MD-PG [34] as representative algorithms for the RLHF and NLHF frameworks, respectively. Since RLHF requires a scalar feedback signal, we use the mean of the attribute reward models $\hat{r}_a$ as the reward function for RLOO. All models are fine-tuned from the QWEN2.5-0.5B[6] model and run for 1,000 gradient steps with a batch size of $B = 32$. We sweep over learning rates $\eta \in \{1\mathrm{e}{-6}, 5\mathrm{e}{-6}, 1\mathrm{e}{-5}\}$ and KL penalties $\tau \in \{0.001, 0.01, 0.1\}$ for all algorithms. For NASH-MD-PG, we additionally sweep over the mixture parameter $\beta \in \{0, 0.25, 0.5, 0.75, 1\}$, and for STACKELBERGGDA, the two-timescale coefficient $\kappa \in \{1, 5, 10\}$. We evaluate each configuration based on its average preference rate over the initial model QWEN2.5-0.5B and find that $\eta = 1\mathrm{e}^{-5}$ and $\tau = 0.001$ perform best across the board, with $\beta = 0.75$ for NASH-MD-PG and $\kappa = 5$ for STACKELBERGGDA. All implementations use the `Transformers` [61] and `TRL` [55] libraries, with the AdamW optimizer [31].

6.1 Round-Robin Tournament

Table 3 reports pairwise preference scores between the initial QWEN2.5-0.5B and the three fine-tuned models. Both the first responses of STACKELBERGGDA (Leader) and NASH-MD-PG achieve approximately 73% preference over QWEN2.5-0.5B and 61% over RLOO, while they tie at 50% when compared to each other. This outcome is consistent with scenarios in which multiple high-quality responses exist and the Stackelberg and Nash solutions coincide, as discussed in Section 4.1.

Importantly, applying the Follower model of STACKELBERGGDA to improve its own initial responses leads to a substantial performance gain. It achieves 80% preference over QWEN2.5-0.5B, 66% over RLOO, 60% over NASH-MD-PG, and even outperforms the responses it was conditioned on in 60.5% of comparisons. Thus, the two-turn inference yields significant gains at the cost of an additional inference step.

6.2 Test-time Improvements

We next evaluate the ability of each model to improve responses at test time, specifically by acting as a Follower that refines an initial response produced by another model. Although only STACKELBERGGDA is explicitly trained for this task (though only to best respond to itself), we apply the same refinement procedure to all models to assess whether they can generalize to this setting. To do so, we evaluate every pair of Leader and Follower models selected from $\{\text{QWEN2.5-0.5B}, \text{RLOO}, \text{NASH-MD-PG}, \text{STACKELBERGGDA}\}$ as follows. For every validation

[5]https://huggingface.co/unsloth/Qwen2.5-1.5B-Instruct
[6]https://huggingface.co/unsloth/Qwen2.5-0.5B-Instruct

Table 4: Test-time improvement using different models for the initial response (Leader) and the improvement (Follower). Each cell represents the preference model's average score for the Follower's responses over the Leader's responses.

		Leader			
		QWEN2.5-0.5B	RLOO	NASH-MD-PG	STACKELBERGGDA
Follower	QWEN2.5-0.5B	0.549	0.443	0.363	0.362
	RLOO	0.534	0.403	0.369	0.360
	NASH-MD-PG	0.708	0.600	0.493	0.476
	STACKELBERGGDA	**0.803**	**0.665**	**0.600**	**0.606**

prompt, we first generate a response with the chosen Leader model, and then apply the Follower prompting template from Figure 1(b) to generate a potentially improved response using the Follower model. We refer to these as the Leader and Follower responses, respectively. By exhaustively evaluating all Leader-Follower combinations, we assess each model's ability to serve as a test-time improver across a range of initial conditions. Table 4 reports the resulting preference scores, showing how often the Follower output is preferred over the Leader's generation.

STACKELBERGGDA consistently improves across all Leader models; most notably over QWEN2.5-0.5B and RLOO, but also achieving 60% gains over both NASH-MD-PG and itself. By contrast, QWEN2.5-0.5B and RLOO only improve on the initial responses from QWEN2.5-0.5B and even produce worse outputs when applied to other Leader models. NASH-MD-PG is able to improve on responses from QWEN2.5-0.5B and RLOO when used as a Follower; however, even its 70% preference score over QWEN2.5-0.5B falls short of its own 73% score reported in Table 3. These results extend previous works on verifiable domains [24, 67, 54, 41] by underscoring the importance of explicitly learning to improve given outputs and that mere instruction prompting is not sufficient to improve with respect to preferences either.

7 Conclusion

In this paper, we introduced Stackelberg Learning from Human Feedback (SLHF), a two-player sequential-move framework that directly optimizes pairwise preference signals without relying on real-valued reward models. Our proposed algorithm, STACKELBERGGDA, efficiently approximates the unique Stackelberg equilibrium and scales to complex tasks such as aligning large language models with human preferences. Empirically, STACKELBERGGDA's Leader matches or exceeds standard baselines, while its Follower consistently refines outputs at test-time, even when paired with models it was not trained with.

Similarly to NLHF, a key limitation of our approach is its reliance on a well-specified and representative pairwise preference function, which can be challenging to obtain in open-ended or under-specified domains. Additionally, while the sequential formulation enables test-time improvement through conditional generation, it does so without leveraging real-time user feedback. Combining STACKELBERGGDA with user preference elicitation and incorporating personalized refinement at inference-time presents an interesting direction for future work.

Acknowledgments

Barna Pásztor was supported by an ETH AI Center doctoral fellowship. Thomas Kleine Buening was supported by the EPSRC Prosperity Partnership FAIR (grant number EP/V056883/1). We thank Marian Schneider for his support in the experiments' technical setup.

References

[1] A. Ahmadian, C. Cremer, M. Gallé, M. Fadaee, J. Kreutzer, O. Pietquin, A. Üstün, and S. Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In L.-W. Ku, A. Martins, and V. Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024.

[2] A. F. Akyurek, E. Akyurek, A. Kalyan, P. Clark, D. T. Wijaya, and N. Tandon. RL4F: Generating natural language feedback with reinforcement learning for repairing model outputs. In A. Rogers, J. Boyd-Graber, and N. Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2023.

[3] C. Alós-Ferrer, E. Fehr, and M. Garagnani. Identifying nontransitive preferences. Technical report, Working Paper, 2022.

[4] A. Auslender and M. Teboulle. Projected subgradient methods with non-euclidean distances for non-differentiable convex minimization and variational inequalities. *Mathematical Programming*, 120:27–48, 2009.

[5] M. G. Azar, M. Rowland, B. Piot, D. Guo, D. Calandriello, M. Valko, and R. Munos. A General Theoretical Paradigm to Understand Learning from Human Preferences. *Proceedings of Machine Learning Research*, 238:4447–4455, 10 2023.

[6] 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.

[7] A. Beirami, A. Agarwal, J. Berant, J. Eisenstein, C. Nagpal, A. Theertha Suresh, G. Research, and G. DeepMind. Theoretical guarantees on the best-of-n alignment policy. *arXiv preprint arXiv:2401.01879*, 2024.

[8] Q. Bertrand, W. M. Czarnecki, and G. Gidel. On the limitations of the elo, real-world games are transitive, not additive. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, 2023.

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

[10] D. Calandriello, D. Guo, R. Munos, M. Rowland, Y. Tang, B. A. Pires, P. H. Richemond, C. Le Lan, M. Valko, T. Liu, R. Joshi, Z. Zheng, and B. Piot. Human Alignment of Large Language Models through Online Preference Optimisation. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

[11] S. Casper, X. Davies, C. Shi, T. K. Gilbert, J. Scheurer, J. Rando, R. Freedman, T. Korbak, D. Lindner, P. Freire, T. Wang, S. Marks, C.-R. Segerie, M. Carroll, A. Peng, P. Christoffersen, M. Damani, S. Slocum, U. Anwar, A. Siththaranjan, M. Nadeau, E. J. Michaud, J. Pfau, D. Krasheninnikov, X. Chen, L. Langosco, P. Hase, E. Bıyık, A. Dragan, D. Krueger, D. Sadigh, and D. Hadfield-Menell. Open Problems and Fundamental Limitations of Reinforcement Learning from Human Feedback. *arXiv preprint arXiv:2307.15217*, 7 2023.

[12] G. H. Chen and R. T. Rockafellar. Convergence rates in forward–backward splitting. *SIAM Journal on Optimization*, 7(2):421–444, 1997.

[13] P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei. Deep Reinforcement Learning from Human Preferences. In *Proceedings of the 31st Conference on Neural Information Processing Systems*, 2017.

[14] V. Conitzer and T. Sandholm. Computing the optimal strategy to commit to. *Proceedings of the ACM Conference on Electronic Commerce*, 2006:82–90, 2006.

[15] J. A. N. de Caritat Mis et al. *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix*. Imprimerie royale, 1785.

[16] Y. Dubois, X. Li, R. Taori, T. Zhang, I. Gulrajani, J. Ba, C. Guestrin, P. Liang, and T. Hashimoto. Alpacafarm: A simulation framework for methods that learn from human feedback. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[17] Y. Dubois, B. Galambosi, P. Liang, and T. B. Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024.

[18] J. Eisenstein, C. Nagpal, A. Agarwal, A. Beirami, A. D'Amour, D. Dvijotham, A. Fisch, K. Heller, S. Pfohl, D. Ramachandran, P. Shaw, and J. Berant. Helping or herding? reward model ensembles mitigate but do not eliminate reward hacking, 2024.

[19] M. Geist, B. Scherrer, and O. Pietquin. A Theory of Regularized Markov Decision Processes. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[20] A. Glaese, N. McAleese, M. Trębacz, J. Aslanides, V. Firoiu, T. Ewalds, M. Rauh, L. Weidinger, M. Chadwick, P. Thacker, et al. Improving alignment of dialogue agents via targeted human judgements. *arXiv preprint arXiv:2209.14375*, 2022.

[21] J. Gu, X. Jiang, Z. Shi, H. Tan, X. Zhai, C. Xu, W. Li, Y. Shen, S. Ma, H. Liu, Y. Wang, and J. Guo. A Survey on LLM-as-a-Judge. 11 2024.

[22] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Proceedings of the Thirty-first International Conference on Neural Information Processing Systems*, 2017.

[23] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[24] J. Huang, X. Chen, S. Mishra, H. S. Zheng, A. W. Yu, X. Song, and D. Zhou. Large language models cannot self-correct reasoning yet. In *Proceedings of the Twelfth International Conference on Learning Representations*, 2024.

[25] D. Jiang, X. Ren, and B. Y. Lin. LLM-BLENDER: Ensembling Large Language Models with Pairwise Ranking and Generative Fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, volume 1, 2023.

[26] T. Kaufmann, P. Weng, V. Bengs, and E. Hüllermeier. A Survey of Reinforcement Learning from Human Feedback. *arXiv preprint arXiv:2312.14925*, 2023.

[27] G. M. Korpelevich. The extragradient method for finding saddle points and other problems. *Matecon*, 12:747–756, 1976.

[28] A. Kumar, V. Zhuang, R. Agarwal, Y. Su, J. D. Co-Reyes, A. Singh, K. Baumli, S. Iqbal, C. Bishop, R. Roelofs, L. M. Zhang, K. McKinney, D. Shrivastava, C. Paduraru, G. Tucker, D. Precup, F. Behbahani, and A. Faust. Training language models to self-correct via reinforcement learning. In *Proceedings of the Thirteenth International Conference on Learning Representations*, 2025.

[29] T. Lin, C. Jin, and M. I. Jordan. Two-timescale gradient descent ascent algorithms for nonconvex minimax optimization. *Journal of Machine Learning Research*, 26(11):1–45, 2025.

[30] K. Liu, Q. Long, Z. Shi, W. J. Su, and J. Xiao. Statistical impossibility and possibility of aligning llms with human preferences: From condorcet paradox to nash equilibrium. *arXiv preprint arXiv:2503.10990*, 2025.

[31] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *Proceedings of the International Conference on Learning Representations*, 2019.

[32] J. Makar-Limanov, A. Prakash, D. Goktas, N. Ayanian, and A. Greenwald. Sta-rlhf: Stackelberg aligned reinforcement learning with human feedback. In *Coordination and Cooperation for Multi-Agent Reinforcement Learning Methods Workshop*, 2024.

[33] J. Menick, M. Trebacz, V. Mikulik, J. Aslanides, F. Song, M. Chadwick, M. Glaese, S. Young, L. Campbell-Gillingham, G. Irving, et al. Teaching language models to support answers with verified quotes. *arXiv preprint arXiv:2203.11147*, 2022.

[34] R. Munos, M. Valko, D. Calandriello, M. G. Azar, M. Rowland, D. Guo, Y. Tang, M. Geist, T. Mesnard, A. Michi, M. Selvi, S. Girgin, N. Momchev, O. Bachem, D. J. Mankowitz, D. Precup, B. Piot, and G. Deepmind. Nash Learning from Human Feedback. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

[35] R. Nakano, J. Hilton, S. Balaji, J. Wu, L. Ouyang, C. Kim, C. Hesse, S. Jain, V. Kosaraju, W. Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

[36] A. Nedić and A. Ozdaglar. Subgradient methods for saddle-point problems. *Journal of optimization theory and applications*, 142:205–228, 2009.

[37] A. Nemirovski. Prox-method with rate of convergence o (1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*, 15(1):229–251, 2004.

[38] L. G. Openai, J. Schulman, O. Jacob, and H. Openai. Scaling Laws for Reward Model Overoptimization. In *International Conference on Machine Learning*, 2023. ISBN 2210.10760v1.

[39] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang Sandhini Agarwal Katarina Slama Alex Ray John Schulman Jacob Hilton Fraser Kelton Luke Miller Maddie Simens Amanda Askell, P. Welinder Paul Christiano, J. Leike, and R. Lowe. Training language models to follow instructions with human feedback. In *Proceedings of the 36th Conference on Neural Information Processing Systems*, 2022.

[40] H. Prasad, P. LA, and S. Bhatnagar. Two-timescale algorithms for learning nash equilibria in general-sum stochastic games. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, 2015.

[41] Y. Qu, T. Zhang, N. Garg, and A. Kumar. Recursive introspection: Teaching language model agents how to self-improve. *Advances in Neural Information Processing Systems*, 2024.

[42] Qwen, :, A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, H. Lin, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Lin, K. Dang, K. Lu, K. Bao, K. Yang, L. Yu, M. Li, M. Xue, P. Zhang, Q. Zhu, R. Men, R. Lin, T. Li, T. Tang, T. Xia, X. Ren, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Wan, Y. Liu, Z. Cui, Z. Zhang, and Z. Qiu. Qwen2.5 Technical Report. *arXiv preprint arXiv:2412.15115v2*, 12 2024.

[43] R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, and C. Finn. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Advances in Neural Information Processing Systems*, 2023. ISBN 2305.18290v2.

[44] C. Rosset, C.-A. Cheng, A. Mitra, M. Santacroce, A. Awadallah, and T. Xie. Direct Nash Optimization: Teaching Language Models to Self-Improve with General Preferences. In *arXiv preprint arXiv:2404.03715*, 2024.

[45] W. Saunders, C. Yeh, J. Wu, S. Bills, L. Ouyang, J. Ward, and J. Leike. Self-critiquing models for assisting human evaluators. *arXiv preprint arXiv:2206.05802*, 2022.

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

[47] P. G. Sessa, R. Dadashi, L. Hussenot, J. Ferret, N. Vieillard, A. Ramé, B. Shariari, S. Perrin, A. Friesen, G. Cideron, S. Girgin, P. Stanczyk, A. Michi, D. Sinopalnikov, S. Ramos, A. Héliou, A. Severyn, M. Hoffman, N. Momchev, O. Bachem, and G. Deepmind. BOND: Aligning LLMs with Best-of-N Distillation. *arXiv preprint arXiv:2407.14622*, 2024.

[48] H. Shen, Z. Yang, and T. Chen. Principled penalty-based methods for bilevel reinforcement learning and rlhf. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

[49] H. v. Stackelberg. *Theory of the market economy*. Oxford University Press, 1952.

[50] N. Stiennon, L. Ouyang, J. Wu, D. M. Ziegler, R. Lowe, C. Voss, and A. Radford Dario Amodei Paul Christiano. Learning to summarize from human feedback. In *Proceedings of the 34th Conference on Neural Information Processing Systems*, 2020.

[51] G. Swamy, C. Dann, R. Kidambi, Z. S. Wu, and A. Agarwal. A Minimaximalist Approach to Reinforcement Learning from Human Feedback RLHF / PbRL. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

[52] X. Tang, S. Yoon, S. Son, H. Yuan, Q. Gu, and I. Bogunovic. Game-theoretic regularized self-play alignment of large language models. *arXiv preprint arXiv:2503.00030*, 2025.

[53] V. Thoma, B. Pásztor, A. Krause, G. Ramponi, and Y. Hu. Contextual bilevel reinforcement learning for incentive alignment. In *Proceedings of the Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[54] G. Tyen, H. Mansoor, V. Cărbune, Y. P. Chen, and T. Mak. Llms cannot find reasoning errors, but can correct them given the error location. In *Findings of the Association for Computational Linguistics ACL 2024*, 2024.

[55] L. von Werra, Y. Belkada, L. Tunstall, E. Beeching, T. Thrush, N. Lambert, S. Huang, K. Rasul, and Q. Gallouédec. Trl: Transformer reinforcement learning, 2020.

[56] M. Wang, C. Ma, Q. Chen, L. Meng, Y. Han, J. Xiao, Z. Zhang, J. Huo, W. J. Su, and Y. Yang. Magnetic preference optimization: Achieving last-iterate convergence for language model alignment. In *The Thirteenth International Conference on Learning Representations*, 2025.

[57] Z. Wang, Y. Dong, O. Delalleau, J. Zeng, G. Shen, D. Egert, J. J. Zhang, M. N. Sreedhar, and O. Kuchaiev. Helpsteer 2: Open-source dataset for training top-performing reward models. In *Proceedings of the Thirty-eight Conference on Neural Information Processing Systems*, 2024.

[58] S. Welleck, X. Lu, P. West, F. Brahman, T. Shen, D. Khashabi, and Y. Choi. Generating Sequences by Learning to Self-Correct. In *Proceedings of the Eleventh International Conference on Learning Representations*, 2023.

[59] S. Welleck, A. Bertsch, M. Finlayson, H. Schoelkopf, A. Xie, G. Neubig, I. Kulikov, and Z. Harchaoui. From decoding to meta-generation: Inference-time algorithms for large language models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

[60] C. Wirth, R. Akrour, G. Neumann, and J. Fürnkranz. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136):1–46, 2017.

[61] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020.

[62] Y. Wu, Z. Sun, H. Yuan, K. Ji, Y. Yang, and Q. Gu. Self-Play Preference Optimization for Language Model Alignment. *arXiv preprint arXiv:2405.00675*, 2024.

[63] J. Xiao, Z. Li, X. Xie, E. Getzen, C. Fang, Q. Long, and W. J. Su. On the algorithmic bias of aligning large language models with rlhf: Preference collapse and matching regularization. *arXiv preprint arXiv:2405.16455*, 2024.

[64] Y. Xu, L. Ruis, T. Rocktäschel, and R. Kirk. Investigating non-transitivity in llm-as-a-judge. *arXiv preprint arXiv:2502.14074*, 2025.

[65] Y. Zhang, M. Khalifa, L. Logeswaran, J. Kim, M. Lee, H. Lee, and L. Wang. Small language models need strong verifiers to self-correct reasoning. In L.-W. Ku, A. Martins, and V. Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, 2024.

[66] Y. Zhang, D. Yu, B. Peng, L. Song, Y. Tian, M. Huo, N. Jiang, H. Mi, and D. Yu. Iterative Nash Policy Optimization: Aligning LLMs with General Preferences via No-Regret Learning. In *Proceedings of the Thirteenth International Conference on Learning Representations*, 2025.

[67] H. S. Zheng, S. Mishra, H. Zhang, X. Chen, M. Chen, A. Nova, L. Hou, H.-T. Cheng, Q. V. Le, E. H. Chi, et al. Natural plan: Benchmarking llms on natural language planning. *arXiv preprint arXiv:2406.04520*, 2024.

[68] D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving. Fine-Tuning Language Models from Human Preferences. *arXiv preprint arXiv:1909.08593*, 2019.

Contents of Appendix

A Proofs

A.1 Proof of Proposition 1

Proof. First, assume that the Leader's policy π is fixed and consider the Follower's optimization problem

$$\min_{\omega} \mathbb{E}_{x\sim\rho, y\sim\pi(\cdot|x)}\left[\mathbb{E}_{y'\sim\omega(\cdot|x,y)}[p(y \succ y' \mid x)] + \tau^F \mathrm{KL}_{x,y}(\omega \,\|\, \omega^{\text{ref}})\right]. \tag{8}$$

The optimization problem in (8) is equivalent to Equation (3) for the reward function $r(\tilde{x}, y') := p(y \succ y' \mid x)$ with contexts $\tilde{x} = (x, y)$ and context distribution $\tilde{x} \sim \rho \otimes \pi$. As a result, Equation (8) has a unique closed-form solution [19, 43, 5] given by

$$\omega^\star(y' \mid x, y) = \frac{1}{Z(x,y)}\omega^{\text{ref}}(y' \mid x, y)\exp\left(\tfrac{1}{\tau^F} p(y' \succ y \mid x)\right)$$

where $Z(x,y) = \sum_{y'\in\mathcal{Y}} \omega^{\text{ref}}(y' \mid x, y)\exp\left(\tfrac{1}{\tau^F} p(y' \succ y \mid x)\right)$ is a partition factor that depends only on (x, y) and π^{ref}. Hence, ω^* can be expressed as a function of (x, y) and ω^{ref} without explicit dependence on π.

Now, define the following reward function for the Leader's optimization problem

$$r(x,y) := \mathbb{E}_{y'\sim\omega^\star(\cdot|x,y)}[p(y \succ y' \mid x)]. \tag{9}$$

Note that $\omega^\star$ is unique so that $r(x, y)$ is a scalar. We can now restate Equation (5) for the Leader's optimization problem as

$$\max_{\pi} \mathbb{E}_{x\sim\rho}\left[\mathbb{E}_{y\sim\pi(\cdot|x)}[r(x,y)] - \tau^L \mathrm{KL}_x(\pi \,\|\, \pi^{\text{ref}})\right]$$

which is again a KL-regularized optimization problem that admits a closed-form solution

$$\pi^\star(y \mid x) = \frac{1}{Z(x)}\pi^{\text{ref}}(y \mid x)\exp\left(\tfrac{1}{\tau^L} r(x,y)\right).$$

□

A.2 Proof of Remark 2

Proof. This lemma is folklore in the algorithmic game theory community and can be quickly verified.

Let $x \in \mathcal{X}$. Given any action $y \in \mathcal{Y}$, there exists a not necessarily unique $y' \in \mathcal{Y}$ minimizing $p(y \succ y' \mid x)$. Hence, irrespective of the Leader's policy $\pi(\cdot \mid x)$, there always exists a Follower's deterministic best response policy $\omega_{\text{br}}(\cdot \mid x, y)$ with $\omega_{\text{br}}(y' \mid x, y) = 1$ for some y'. In other words, the Follower always has a deterministic best response policy.

Similarly, the optimization problem for the Leader given some context x reduces to finding y that maximizes $\mathbb{E}_{y' \sim \omega_{\text{br}}(\cdot|x,y)}[p(y \succ y' \mid x)]$ so that the SLHF optimization problem admits a determinsitic solution. □

A.3 Concave-Convex Property of f

We show here that the objective function f in Equation (6) of the Stackelberg optimization problem is concave-convex. Similar results were established in the context of NLHF by Munos et al. [34].

Throughout this section, we assume $|X| = 1$ and omit x from the notation for clarity. All results extend directly to the general case with a finite context space $\mathcal{X}$.

Then, the objective function of Equation (5) is given by

$$f(\pi, \omega) \;= \mathbb{E}_{y \sim \pi(\cdot), y' \sim \omega(\cdot|y)}\left[p(y \succ y')\right] - \tau^L \text{KL}\big(\pi \,\|\, \pi^{\text{ref}}\big) + \tau^F \mathbb{E}_{y \sim \pi(\cdot)}\big[\text{KL}_y\big(\omega \,\|\, \omega^{\text{ref}}\big)\big]. \quad (10)$$

The first term is bilinear in π and ω, as shown by expanding the expectation:

$$\mathbb{E}_{y \sim \pi(\cdot), y' \sim \omega(\cdot|y)}\left[p(y \succ y')\right] = \sum_{y \in \mathcal{Y}} \pi(y) \sum_{y' \in \mathcal{Y}} p(y \succ y')\omega(y' \mid y).$$

The KL terms are convex in their respective arguments. Hence, f is bilinear when $\tau^L = \tau^F = 0$, and strongly concave-convex when $\tau^L, \tau^F > 0$.

B Scalable Implementation of StackelbergGDA

When fine-tuning large language models, the context and action spaces $\mathcal{X}$ and $\mathcal{Y}$ are far too large to optimize over Π and Ω directly. To address this, we introduce a practical variant of StackelbergGDA in Algorithm 1.

Policy Parameterization. We replace the tabular policies π and ω with neural parameterizations π_θ and ω_ϕ (e.g., transformer networks). This renders the policy spaces tractable via their parameter vectors θ and ϕ, however, the concave-convex property does not necessarily carry over to the parameters θ and ϕ.

Batched, Variance-reduced Gradient Estimates. Exact evaluation of the expectations in ∇f is infeasible due to the expectation over the context and action spaces. Instead, at each iteration we sample a batch of size B:

$$\{(x_i, y_i, y'_i, p_i)\}_{i=1}^B, \; x_i \sim \rho, \; y_i \sim \pi_\theta(\cdot \mid x_i), \; y'_i \sim \omega_\phi(\cdot \mid x_i, y_i), \; p_i = p(y_i \succ y'_i \mid x_i).$$

We then form unbiased estimates as

$$\widehat{\nabla}_\theta f = \frac{1}{B}\sum_{i=1}^{B}\big(p_i - \tau^L k_i^L\big)\nabla_\theta \log \pi_\theta(y_i \mid x_i), \widehat{\nabla}_\phi f = \frac{1}{B}\sum_{i=1}^{B}\big(p_i - \tau^F k_i^F\big)\nabla_\phi \log \omega_\phi(y'_i \mid x_i, y_i),$$

with likelihood ratios $k_i^L = \frac{\pi_\theta(y_i|x_i)}{\pi^{\text{ref}}(y_i|x_i)}$ and $k_i^F = \frac{\omega_\phi(y'_i|x_i,y_i)}{\omega^{\text{ref}}(y'_i|x_i,y_i)}$. The gradient estimators are naturally compatible with additional variance reduction techniques such as subtracting a constant baseline.

Single-Model Instantiation. Simultaneously training two billion-parameter transformer models is memory-prohibitive. Similarly to SCoRe [28], we collapse both Leader and Follower into one model π_θ by using distinct chat templates (Figure 1). When the model is only given the context x, we use the template in Figure 1(a) that only includes x as the prompt. When the model is given both the context x and an action y, we use the template in Figure 1(b) that includes both the context x and the action y, as well as a predefined instruction to improve the action y.

Then, letting $\kappa = \frac{\alpha^F}{\alpha^L}$ denote the two-timescale weight coefficient, we optimize the model to minimize the following loss function

$$\mathcal{L}(\theta) = -\frac{1}{B}\sum_{i=1}^{B}\big(p_i - \tau^L k_i^L\big)\log \pi_\theta(y_i \mid x_i) + \frac{\kappa}{B}\sum_{i=1}^{B}\big(p_i - \tau^F k_i^F\big)\log \pi_\theta\big(y'_i \mid x_i, y_i\big). \quad (11)$$

Gradient steps on $\mathcal{L}(\theta)$ realize the two-time-scale gradient descent-ascent updates via a single network, thereby substantially reducing memory usage.

Algorithm 1 StackelbergGDA (Practical)

```
1:  procedure StackelbergGDA($\mathcal{X}, \mathcal{Y}, \rho, \eta$)
2:      Initialize the policy $\pi$ and $\omega$
3:      for $i = 1, 2, \ldots$ do
4:          for $b = 1, \ldots, B$ do
5:              Sample prompt $x_b \sim \rho$
6:              Sample Leader response using the prompt in Figure 1(a): $y_b \sim \pi_\theta(\cdot \mid x_b)$
7:              Sample Follower response using the prompt in Figure 1(b): $y'_b \sim \pi_\theta(\cdot \mid x_b, y_b)$
8:              Observe preference feedback $p_b = p(y_b \succ y'_b \mid x_b)$
9:          end for
10:         Update the weights $\theta$ according to the loss in Equation (11): $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\theta)$
11:     end for
12: end procedure
```

C Implementation Details

Implementation We trained RLOO[7] and Nash-MD-PG[8] using their implementations in the TRL Python package [55]. For all training runs, including reward modeling, we used Low-Rank Adaptation (LoRA) [23] with rank $r = 32$, scaling factor $\alpha = 64$, and dropout rate set to 0.1.

Compute Resources All experiments were conducted on a cluster with 8 NVIDIA GeForce RTX 4090 GPUs, 16 CPU cores, and 64 GB of RAM. The total compute time, including hyperparameter sweeps, was approximately 4,000 GPU-hours.

D Additional Experimental Results

D.1 Preference Model

We estimate the preference model used to fine-tune the LLMs by treating the five attributes in the HelpSteer2 datasets [57] as distinct annotators, denoted by the set $\mathcal{A}$, and define ν as a uniform distribution over $\mathcal{A}$. For each attribute $a \in \mathcal{A}$, we estimate a reward function $\hat{r}_a$ using the Bradley-Terry model and maximize the log-likelihood on the training dataset $\mathcal{D} = \{(x_i, y_i^w, y_i^l)\}_{i=1}^N$

$$\min_r \sum_{i=1}^N \sigma(r(x_i, y_i^w) - r(x_i, y_i^l)) + \lambda(r(x_i, y_i^w) + r(x_i, y_i^l))^2.$$

We here decided which response is the winning and losing one in the dataset by comparing the attribute scores provided by the annotators. The additional regularization ensures that the rewards are centralized around zero [18]. For the attributes correctness, helpfulness, and coherence, we consider higher scores to be better while for verbosity and complexity lower values are more preferable. This is in accordance with the scoring criteria described in Wang et al. [57]. We set the regularization parameter to $\lambda = 0.01$. Centralizing the reward values is crucial for our RLOO implementation, which uses the average reward across the five attributes as its training signal. It ensures that no attribute heavily dominates due to scale differences and avoids bias toward any particular attribute.

We train each model for 5 epochs on the training prompts and completions with batch size 32 and learning rate $1\mathrm{e}{-4}$. The final accuracies of the models on the validation dataset are $78\%, 65\%, 61\%, 60\%$, and 59% for verbosity, complexity, correctness, helpfulness, and coherence, respectively.

We evaluate the non-transitivity of the preference model p defined in (7) on the validation prompts and five responses from each of the four models used for comparison in Section 6. For each prompt, we construct a complete directed graph between the 20 completions as nodes and edges directed from

[7] https://huggingface.co/docs/trl/main/en/rloo_trainer
[8] https://huggingface.co/docs/trl/main/en/nash_md_trainer

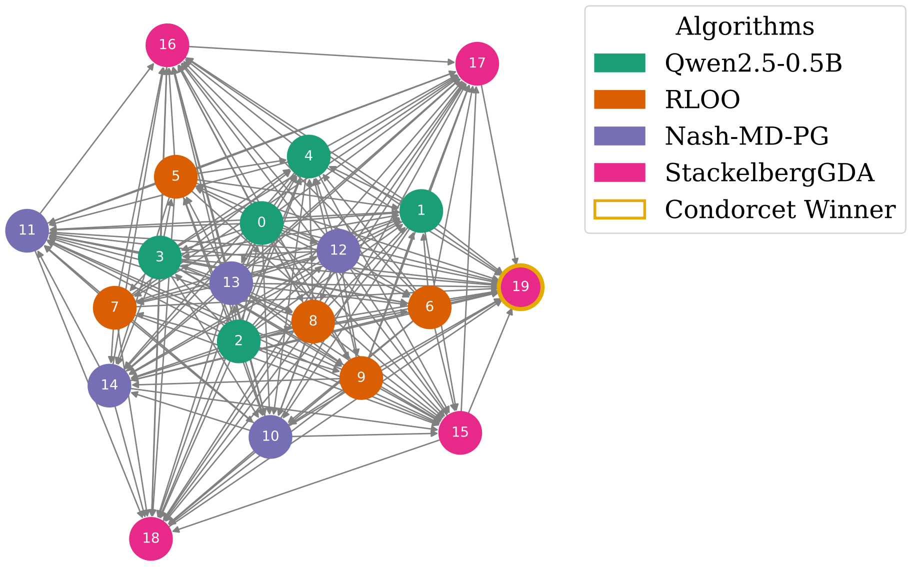

Figure 2: Directed graph based with completions generated by the fine-tuned models and edge directions representing the preference between them.

the non-preferred completion towards the preferred one. Figure 2 illustrates this directed graph on the first prompt of the validation dataset. 57% of the directed graphs include cycles, which illustrate the intransitivity of the preference function p on the completion space $\mathcal{Y}$.

D.2 Iterative Improvements at Test-time

We extend the experimental results from Section 6 by analyzing iterative improvements and performance scaling with increased test-time computation. Our results suggests that with increasing test-time computation the benefit of fine-tuning using both RLOO or NASH-MD-PG is negligible compared to using the base model QWEN2.5-0.5B. In contrast, STACKELBERGGDA yields strict improvements. Building on the example in Section 4.1, we assume that at test-time, a single annotator $a \sim \nu$ and a context $x \sim \rho$ are sampled.

For the base model QWEN2.5-0.5B and the models with fine-tuned with RLOO and NASH-MD-PG, we independently sample N responses $y_1, \ldots, y_N$. For STACKELBERGGDA, which inherently supports iterative refinement, we generate the first sample from the Leader policy $y_1 \sim \pi^\star(\cdot \mid x)$, and subsequent responses from the Follower policy $y_i \sim \omega^\star(\cdot \mid x, y_{i-1})$ for $i \geq 2$. We define $y_{1:N} := (y_1, \ldots, y_N)$.

In line with prior work on Best-of-N sampling [38, 7, 17, 47], we evaluate the quality of the N samples by computing the maximum reward obtained for each attribute under the sampled annotator's reward model, that is,

$$\hat{r}_a^N(x, y_{1:N}) := \max_{y_1, \ldots, y_N} \hat{r}_a(x, y_i). \tag{12}$$

Analogous to the preference function p defined in Section 3, in this section, we compare two models π and π' w.r.t. the preference functions derived from the annotator-specific reward functions under Best-of-N sampling:

$$p_a^N(\pi \succ \pi' \mid x) := \mathbb{E}_{\substack{y_{1:N} \sim \pi(\cdot \mid x) \\ y'_{1:N} \sim \pi'(\cdot \mid x)}} \left[\mathbb{1}\{\hat{r}_N^a(x, y_{1:N}) \geq \hat{r}_N^a(x, y'_{1:N})\} \right]. \tag{13}$$

Notational Note. We here adopt a slight abuse of notation. Specifically, we write $y_{1:N} \sim \pi(\cdot \mid x)$ to denote the sampling of N responses from a model π, even though this notation does not faithfully represent the sampling procedure used by STACKELBERGGDA. For QWEN2.5-0.5B, RLOO, and NASH-MD-PG, the samples $y_1, \ldots, y_N$ are drawn i.i.d. from a single model $\pi(\cdot \mid x)$. In contrast, for

Table 5: Preference scores for RLOO versus QWEN2.5-0.5B across all attributes as a function of the number of test-time samples N.

Attribute	**Number of Samples N**				
	1	2	3	4	5
Coherence	0.521	0.338	0.262	0.207	0.164
Complexity	0.892	0.842	0.801	0.771	0.754
Correctness	0.388	0.212	0.139	0.098	0.070
Helpfulness	0.322	0.150	0.087	0.057	0.036
Verbosity	0.846	0.777	0.728	0.695	0.669
Average	**0.594**	**0.464**	**0.403**	**0.366**	**0.338**

STACKELBERGGDA, the sampling process is inherently autoregressive: we first draw $y_1 \sim \pi^\star(\cdot \mid x)$ from the Leader policy, and then generate $y_i \sim \omega^\star(\cdot \mid x, y_{i-1})$ for $i \geq 2$ using the Follower policy. Despite this difference, we overload the notation $y_{1:N} \sim \pi(\cdot \mid x)$ to unify the presentation in Equations (12) and (13). In the case of STACKELBERGGDA, this notation should be interpreted as shorthand for the autoregressive sampling process described above.

Previous work has shown that Best-of-N sampling can rival the performance of RLHF-based fine-tuning [16, 47, 7]. Motivated by this, we compare the preference scores defined in Equation (13) of RLOO, NASH-MD-PG, and STACKELBERGGDA with respect to the base model QWEN2.5-0.5B. Throughout this section, we consider the maximum number of samples to be $N = 5$.

Results. Table 5 reports the preference scores for RLOO. While the model initially (i.e. $N = 1$) performs competitively on complexity and verbosity attributes, iterative sampling reveals a collapse into a single preference mode. In particular, we observed deterministic outputs for RLOO the generic response: *"I apologize, but I'm unable to engage in conversations about political topics. If you have any other questions or need further assistance with a different subject, feel free to ask."* As a result, the model's diversity and coverage deteriorate, and its overall preference scores (relative to QWEN2.5-0.5B) decline sharply as N increases.

In contrast, NASH-MD-PG demonstrates some benefit from additional sampling, as shown in Table 6. Its preference score for verbosity remains stable and it shows moderate improvement in coherence. However, for the remaining attributes (correctness, helpfulness, and complexity) its gains are slower than those achieved by QWEN2.5-0.5B with Best-of-N sampling. Consequently, the overall preference score of NASH-MD-PG declines with increasing N, suggesting that the model fine-tuned with NASH-MD-PG does not improve notably (compared to the base model) when the number of samples drawn at test-time increases.

On the other hand, STACKELBERGGDA exhibits more favorable behavior. As shown in Table 7, while the preference score on verbosity and complexity taper off with more samples, STACKELBERGGDA achieves notably faster gains on coherence, correctness, and helpfulness. For these attributes, the preference rate improves by 10 percentage points or more, making STACKELBERGGDA the only method among the three to demonstrate consistent improvement over QWEN2.5-0.5B as N increases. This means that the performance of STACKELBERGGDA effectively scales with test-time compute.

The strong emphasis on complexity and verbosity by RLOO is expected, as it optimizes the average reward across all five attributes, and these two dimensions yield the highest values. However, for NASH-MD-PG, this outcome is less expected. We hypothesize that it stems from its training objective, which pits the policy against a mixture of the reference policy QWEN2.5-0.5B and the most recent iteration. Once NASH-MD-PG outperforms QWEN2.5-0.5B on all attributes, it begins focusing on attributes where further improvement over itself is possible, namely, complexity and verbosity. Nevertheless, this skewed emphasis is suboptimal: annotators prefer models that perform well on all attributes. In fact, a policy that focuses on coherence, correctness, and helpfulness is preferred by 60% of the annotators. STACKELBERGGDA's asymmetric formulation that trains a Leader and a Follower policy separately (though potentially unified in a single model) helps mitigate this imbalance across attributes. This leads to a more balanced policy that is preferred by a wider range of annotators.

Table 6: Preference scores for NASH-MD-PG versus QWEN2.5-0.5B across all attributes as a function of the number of test-time samples N.

Attribute	**Number of Samples** N				
	1	2	3	4	5
Coherence	0.731	0.743	0.757	0.763	0.760
Complexity	0.761	0.743	0.732	0.726	0.727
Correctness	0.633	0.615	0.620	0.617	0.615
Helpfulness	0.645	0.633	0.640	0.641	0.638
Verbosity	0.858	0.855	0.850	0.849	0.852
Average	**0.726**	**0.718**	**0.720**	**0.719**	**0.718**

Table 7: Preference scores for STACKELBERGGDA versus QWEN2.5-0.5B across all attributes as a function of the number of test-time samples N.

Attribute	**Number of Samples** N				
	1	2	3	4	5
Coherence	0.778	0.850	0.865	0.875	0.873
Complexity	0.714	0.666	0.628	0.600	0.592
Correctness	0.670	0.762	0.791	0.795	0.803
Helpfulness	0.692	0.767	0.783	0.786	0.791
Verbosity	0.833	0.820	0.798	0.777	0.765
Average	**0.738**	**0.773**	**0.773**	**0.767**	**0.765**

D.3 Ablation on the two-timescale coefficient

Table 8 and Table 9 present ablations on the follower weight parameter κ in STACKELBERGGDA's loss function (11) when fine-tuning the QWEN2.5-0.5B and QWEN2.5-1.5B models, respectively. Each row reports the average preference scores over the corresponding initial policy, for both the Leader and Follower policies, on the training and validation splits. These results highlight the importance of balancing the two components of STACKELBERGGDA's asymmetric training objective. In general, moderate values of κ can help the Follower improve without compromising the Leader too severely, but excessively large weights may impair both players.

In Table 8, we observe that increasing κ leads to a gradual decline in the Leader's performance. While the Follower benefits from increasing κ from 1 to 5, performance worsens at $\kappa = 10$ for both the Leader and Follower, indicating an overemphasis on the Follower's loss can destabilize the overall optimization.

Table 9 shows a similar trend for the larger QWEN2.5-1.5B model. Due to the decrease of performance above $\kappa = 5$ in Table 8, we carry out the ablation on a finer grid $\kappa \in \{1, 2, 3, 4, 5\}$. Moreover, we evaluate each model after 2000 training steps as a larger base model requires more gradient updates to converge. While the performance of $\kappa = 1$ stands out in Table 9, we observe that it is overfitting to verbosity and complexity by responding to every prompt with short, non-informative answers asking for further information such as *"Certainly! If you need detailed insights on technical topics like that, feel free to ask—I'm here to assist with informatively aligned queries!"*. On the contrary to the collapse observed for RLOO in Appendix D.2, the model remains stochastic with the responses having similar information content. This outcome demonstrates the effectiveness of STACKELBERGGDA in optimizing its objective despite the qualitatively undesirable responses.

D.4 Model Scaling

We extend our round-robin comparison from Section 6.1 to larger models within the Qwen2.5 family, specifically, QWEN2.5-1.5B and QWEN2.5-3B [42]. These evaluations demonstrate that STACKELBERGGDA continues to be on par or outperform baselines even as model size increases. Since larger models require more training updates to reach convergence in our setup, we train NASH-

Table 8: Ablation on the follower weight parameter κ in StackelbergGDA's loss function (11) fine-tuning the Qwen2.5-0.5B model. Scores show the average preference over the base model.

Follower Weight κ	**Train**		**Validation**	
	Leader	Follower	Leader	Follower
1	**0.768**	0.804	**0.761**	0.784
5	0.743	**0.814**	0.723	**0.806**
10	0.725	0.800	0.710	0.783

Table 9: Ablation on the follower weight parameter κ in StackelbergGDA's loss function (11) fine-tuning the Qwen2.5-1.5B model. Scores show the average preference over the base model.

Follower Weight κ	**Train**		**Validation**	
	Leader	Follower	Leader	Follower
1	**0.848**	**0.852**	**0.850**	**0.851**
2	0.718	0.737	0.719	0.730
3	0.767	0.806	0.771	0.803
4	0.733	0.811	0.736	0.807
5	0.720	0.819	0.720	**0.818**

MD-PG for 1,500 steps and StackelbergGDA for 2,000 steps. The RLOO method converges earlier and requires only 1,000 steps even for these larger models. We fix the follower-weight parameter at $\kappa = 5$ for both scales, based on our ablation results in Appendix D.3.

Table 10 summarizes results for models fine-tuned from Qwen2.5-1.5B. Both Nash-MD-PG and StackelbergGDA clearly outperform the base model and the RLOO baseline. While the Leader policy of StackelbergGDA underperforms compared to Nash-MD-PG, the Follower policy conditioned on the Leader's responses matches or exceeds Nash-MD-PG's performance, mirroring the improvements observed when starting from the Qwen2.5-0.5B in Table 3. As noted in Appendix D.3, this performance gap between the Leader and Nash-MD-PG could likely be reduced by tuning κ, albeit at the potential cost of Follower quality.

Table 11 shows analogous comparisons for models initialized from Qwen2.5-3B. Here, StackelbergGDA again performs strongly, with its Follower policy matching or surpassing Nash-MD-PG across most pairwise matchups, and both algorithms outperforming the base model. Nash-MD-PG and StackelbergGDA are closely matched when compared directly. Due to compute limitations, we capped training at 2,000 steps for these larger models. Nonetheless, the Leader policy continued to improve near the end of training, suggesting further gains in preference score may be possible with additional updates.

Table 10: Pairwise preference comparisons between the responses of QWEN2.5-0.5B, QWEN2.5-1.5B, RLOO, NASH-MD-PG, and STACKELBERGGDA algorithms. Fine-tuned models are trained from the QWEN2.5-1.5B. Each cell shows the average preference score of the row model over the column model.

	QWEN2.5-0.5B	QWEN2.5-1.5B	RLOO	NASH-MD-PG	STACKELBERGGDA	
					LEADER	FOLLOWER
QWEN2.5-0.5B	0.000	0.479	0.379	0.188	0.271	0.166
QWEN2.5-1.5B	0.521	0.000	0.401	0.209	0.293	0.187
RLOO	0.621	0.599	0.000	0.197	0.310	0.175
NASH-MD-PG	0.812	0.791	0.803	0.000	0.623	0.489
STACKELBERGGDA LEADER	0.729	0.707	0.690	0.377	0.000	0.313
STACKELBERGGDA FOLLOWER	**0.834**	**0.813**	**0.825**	**0.511**	**0.687**	0.000

Table 11: Pairwise preference comparisons between the responses of QWEN2.5-0.5B, QWEN2.5-3B, RLOO, NASH-MD-PG, and STACKELBERGGDA algorithms. Fine-tuned models are trained from the QWEN2.5-3B. Each cell shows the average preference score of the row model over the column model.

	QWEN2.5-0.5B	QWEN2.5-3B	RLOO	NASH-MD-PG	STACKELBERGGDA	
					LEADER	FOLLOWER
QWEN2.5-0.5B	0.000	0.504	0.399	0.187	0.304	0.187
QWEN2.5-3B	0.496	0.000	0.412	0.199	0.319	0.179
RLOO	0.601	0.588	0.000	0.173	0.338	0.201
NASH-MD-PG	**0.813**	**0.801**	**0.827**	0.000	**0.638**	**0.507**
STACKELBERGGDA LEADER	0.696	0.681	0.662	0.362	0.000	0.312
STACKELBERGGDA FOLLOWER	**0.813**	**0.821**	**0.799**	**0.493**	**0.688**	0.000