The Attacker in the Mirror: Breaking Self-Consistency in Safety via Anchored Bipolicy Self-Play

Gabriele La Malfa[1*] **Emanuele La Malfa**[2*] **Saar Cohen**[2]
Jie M. Zhang[1] **Michael Luck**[1] **Michael Wooldridge**[2] **Elizabeth Black**[1]
[1]Department of Informatics, King's College London
[2]Department of Computer Science, University of Oxford
[3]University of Sussex
gabriele.la_malfa@kcl.ac.uk, emanuele.lamalfa@cs.ox.ac.uk

Abstract

Self-play red team is an established approach to improving AI safety in which different instances of the same model play attacker and defender roles in a zero-sum game, i.e., where the attacker tries to jailbreak the defender; if self-play converges to a Nash equilibrium, the model is guaranteed to respond safely within the settings of the game. Although the parameter sharing enforced by the use of the same model for the two roles improves stability and performance, it introduces fundamental theoretical and architectural limitations. We show that the set of Nash equilibria that can be reached corresponds to a broad class of behaviours that includes trivial always refuse strategies and oracle-like defenders, thus limiting practical applicability. We then show that when attacker and defender share and update the same base model, the dynamics collapse to self-consistency, so that attacks do not enforce adversarial pressure on the defender. In response, we propose Anchored Bipolicy Self-Play, which trains distinct role-specific LoRA adapters on top of a frozen base model, thereby maintaining stable optimisation while preserving adversarial pressure through explicit role separation. In relation to standard self-play, we show up to 100x greater parameter efficiency than fine-tuning and consistent improvements in safety compared to self-play fine-tuned models. We evaluate on Qwen2.5-{3B, 7B,14B}-IT models across widely used safety benchmarks, showing improved robustness without loss of reasoning ability. Cross-play experiments further show that our attacker and defender models are superior to self-play in terms of adversarial defence and safety.

1 Introduction

"A slippery defender is a dangerous defender." — Carlo Mazzone (*circa* 1992 CE)

While concerns surrounding the safety of NLP models have long been recognised [1], Large Language Models (LLMs) [3], given their capabilities and widespread adoption, pose significant security and safety risks that have the potential to erode trust in these systems [20], particularly with the rise of LLM agents and agentic applications [11]. Although safeguarding generated outputs is critical for preventing misuse, recent developments reveal ongoing and substantial obstacles to achieving dependable LLM safety [38]. Model patching to prevent jailbreaks constitutes one mitigation strategy [37]. More recently, self-play reinforcement learning, which engages multiple models in iterative attack-and-defence cycles, has shown promise in improving model robustness [23].

As Figure 1 (centre) illustrates, in the Self-Play Red-Teaming (SELF-REDTEAM) framework by Liu et al. [23], a single model is fine-tuned to assume both the attacker role, attempting to jailbreak

*Equal contribution.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

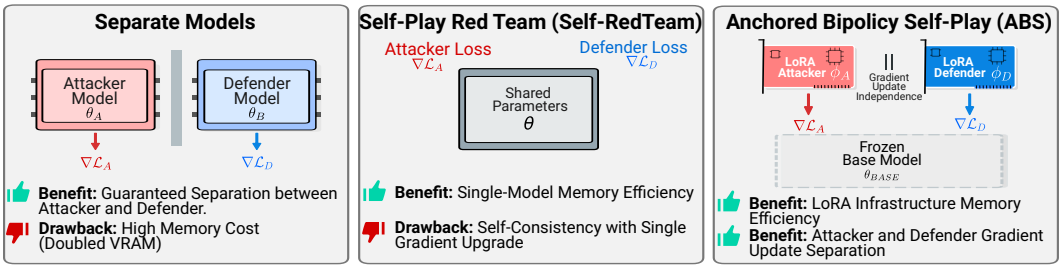

Figure 1: Architectural Comparison of Self-Play Red Team Frameworks. **(Left) Full Model Separation**: Two distinct models guarantee independence and adversarial pressure, but lead to high computational cost. **(Middle) Single-Model Coupling (SELF-REDTEAM [23])**: Attacker and Defender share the same parameters. Gradient updates for the attacker shift the defender's decision boundaries, resulting in self-consistency and weakening the discovery of new vulnerabilities. **(Right) Anchored Bipolicy Self-Play (ABS, Ours)**: We freeze the base model and use separate LoRA adapters for each role. This ensures the attacker and defender evolve independently, restoring adversarial pressure with minimal memory cost.

the defender, and the defender role, aiming to prevent such breaches. The attacker is rewarded for eliciting harmful responses, whilst the defender succeeds by refusing harmful requests or answering benign requests. In this configuration, private reasoning traces are hidden, requiring agents to infer opponent intent exclusively from observable actions.

Although this approach can be framed as an imperfect-information game, current frameworks are limited by fundamental architectural and theoretical constraints. First, we show that relying on convergence to a Nash equilibrium as a safety guarantee admits equilibria that include trivial always-refuse strategies or oracle-like defenders, limiting practical applicability. Moreover, full parameter sharing, while improving training stability, introduces a strong dependency. In particular, we show that the min-max safety game of Liu et al. [23] induces a self-consistency bias that potentially reduces the adversarial pressure on the defender.

To re-establish adversarial integrity within the safety game, this work introduces Anchored Bipolicy Self-Play (ABS). The ABS framework enforces strict architectural separation between attacker and defender while maintaining the training stability of self-play (Figure 1, right). By freezing the base model, ABS employs Low-Rank Adaptation (LoRA) to train distinct, lightweight adapter modules for each role, guaranteeing gradient update separation and a substantial reduction of the computational costs compared to the fully fine-tuned state-of-the-art SELF-REDTEAM and the base model.

ABS matches or outperforms both the base models and the fully fine-tuned SELF-REDTEAM across multiple safety benchmarks while preserving reasoning and instruction-following abilities. Our method is particularly effective at smaller scales, where it provides significant performance gains over existing self-play frameworks while remaining competitive with fully fine-tuned models at the 14B scale. We further assess attacker strength in two settings: a cross-play tournament, where ABS attackers and SELF-REDTEAM defenders of equal size are evaluated against each other, and a transfer setting, where both act as attackers against the base Qwen model. In both cases, ABS attackers outperform SELF-REDTEAM, achieving an attack success rate that is around 10% higher in cross-play and 15% higher when targeting the same base model.

The contributions of this work are threefold:

- We show that the zero-sum formulation of self-play red-teaming allows for a broad class of indistinguishable Nash equilibria, which include both trivial "always-refuse" strategies and optimal oracle defenders. This result provides a theoretical explanation for why convergence alone does not ensure model utility.
- We identify and geometrically formalise the "self-consistency" phenomenon inherent in shared-parameter architectures. Our analysis demonstrates that coupled gradient updates suppress adversarial pressure and hinder the exploration of worst-case prompts.
- We propose Anchored Bipolicy Self-Play (ABS), a parameter-efficient framework that employs role-specific LoRA adapters to maintain architectural separation. This approach restores adversarial integrity by enabling independent evolution of the attacker and defender, while also substantially reducing the computational cost associated with self-play training.

The paper is organised as follows: we first review the relevant literature in reinforcement learning for safety, self-play, and parameter-efficient training. We then show how the analysis of safety via Nash equilibria is incomplete, and that the current formulation of SELF-REDTEAM reduces to self-consistency. We then introduce our technique (ABS), and conduct the experiments to prove its performance and scalability. Link to the results `GitHub`, and checkpoints: `HuggingFace`.

2 Related Work

Safety, Jailbreaking, and Red-Teaming. Standard RedTeam training relies on human feedback to identify harmful responses and reinforce model behaviour accordingly [2, 10]. Owing to the limited coverage of this approach, models are often encouraged to explore novel attack prompts [25]. For instance, Mehrotra et al. [24] proposed a pruning strategy to pre-select attacks with higher jailbreak potential. Hong et al. [14] instead optimised jointly for reward and novelty, incentivising the generation of both successful and previously unseen prompts in a curiosity-driven setting. Other lines of work focus on co-evolving attacker and defender policies to avoid static adversaries [18]. A central challenge in this setting is computational cost [15]. Xhonneux et al. [36] addressed this by operating in the continuous embedding space, reducing the expense associated with discrete token-level attacks. Liu et al. [23] introduced an online self-play framework in which a single model instantiates both attacker and defender, enabling mutual adaptation. More recently, Wen et al. [34] proposed MAGIC, a decoupled co-evolving formulation designed to mitigate optimisation conflicts through full-parameter updates. Tan et al. [32] introduced TriPlay-RL, a tri-role framework incorporating an additional evaluator to stabilise training via multi-agent interaction.

Parameter-Efficient Fine-Tuning Techniques. Parameter-Efficient Fine-Tuning (PEFT) techniques have demonstrated effectiveness in model training while offering significant computational efficiency compared to full-model fine-tuning. Among these methods, LoRA [16] applies a low-rank update to model weights while keeping the base model parameters fixed. In transformer architectures underlying modern large language models (LLMs), LoRA can be applied to attention and feed-forward projections, depending on the adaptation configuration. Further variations of LoRA aim to reduce computational and memory costs, as in QLoRA [12], or dynamically allocate adaptation rank, as in AdaLoRA [40]. SaLoRA [21] introduced a method to improve safety alignment of LLMs. Similarly, Red-Bandit [42] employed LoRA-based adaptation to reduce LLM vulnerabilities to adversarial attacks.

Summary and Research Gap. Recent approaches, including MAGIC [34] and TriPlay-RL [32], also address limitations of attacker–defender co-evolution via role decoupling or additional agents. However, ABS addresses a distinct architectural question: whether optimisation interference can be removed, while retaining a single shared frozen backbone and training only lightweight role-specific parameters. Unlike full-parameter role separation, ABS uses independent LoRA adapters to decouple attacker and defender updates while preserving the computational advantages of a shared backbone. Thus, our primary comparison is with SELF-REDTEAM [23], whose online self-play setting and released Qwen2.5 checkpoints provide a directly comparable baseline. A numerical comparison with published MAGIC and TriPlay-RL results would not be controlled: MAGIC evaluates Qwen2.5 using a different safety-evaluation pipeline (Qwen3Guard rather than our WildGuard-based Ai2 pipeline), while TriPlay-RL evaluates different model scales. Moreover, at the time of our experiments, MAGIC released training code but no checkpoints, while publicly available training code and checkpoints for TriPlay-RL were unavailable to us. Reproducing these approaches under a common evaluation protocol would therefore require substantial additional full-parameter training. We consequently position ABS as a parameter-efficient alternative within the shared-backbone online self-play setting, rather than claiming superiority over all decoupled co-evolutionary frameworks.

3 Nash Equilibria, Model Safety, and Self-Play

SELF-REDTEAM by Liu et al. [23] proposes improving model safety via self-play, wherein a single base model jointly instantiates an attacker A (issuing jailbreak prompts) and a defender D (responding). Training proceeds online in a sequential zero-sum game, where a Nash equilibrium corresponds to safety (see §3.1). Despite growing uptake [34] and extensions to general-sum settings [22], we show that (i) the SELF-REDTEAM formulation admits indistinguishable equilibria spanning trivial always-refusal and oracle defenders (see §3.2); and (ii) the objective is more naturally

cast as single-model self-consistency, with implications for training dynamics and equilibria (see §3.3).

3.1 Nash Equilibria in SELF-REDTEAM

Let $\mathcal{Y}_A$ be the set of attacker prompts and $\mathcal{Y}_D$ be the set of defender responses. For each player $P \in \{A, D\}$, we define the probability simplex over $\mathcal{Y}_P$ as $\Delta(\mathcal{Y}_P)$. In the two-player game formulation of red-teaming, the attacker first proposes a prompt y_A sampled from a mixed strategy $\pi_A \in \Delta(\mathcal{Y}_A)$, referred to as a policy. Given the prompt y_A, the defender then generates a response y_D sampled from a conditional strategy $\pi_D(\cdot \mid y_A)$ with $\pi_D : \mathcal{Y}_A \to \Delta(\mathcal{Y}_D)$.[2] Afterwards, a reward model $r : \mathcal{Y}_A \times \mathcal{Y}_D \to [-1, 1]$ evaluates each prompt-response pair, i.e., for any such pair (y_A, y_D), the defender's reward is $r(y_A, y_D)$, and the attacker receives the opposite payoff $-r(y_A, y_D)$.

For an attacker and a defender that aim to maximise their rewards, this setting corresponds to a zero-sum sequential game where (i) unsafe responses are strictly penalised, i.e., $r(y_A, y_D) = -1$ if (y_A, y_D) is unsafe; (ii) safe responses are rewarded non-negatively, i.e., $r(y_A, y_D) \geq 0$ if (y_A, y_D) is safe, and (iii) there exists a refusal response $y^{\text{ref}} \in \mathcal{Y}_D$ such that $r(y_A, y^{\text{ref}}) = 0$ for all $y_A \in \mathcal{Y}_A$.

Under these assumptions, the expected reward is $V(\pi_A, \pi_D) = \mathbb{E}_{y_A \sim \pi_A,\, y_D \sim \pi_D(\cdot|y_A)}\big[r(y_A, y_D)\big]$ and the game admits the following min-max representation:

$$\min_{\pi_A} \max_{\pi_D} V(\pi_A, \pi_D).$$

A Nash equilibrium (π_A^*, π_D^*) is thus a pair of strategies such that neither player has an incentive to deviate, i.e., $V(\pi_A^*, \pi_D^*) \geq V(\pi_A^*, \pi_D)$ for all π_D, and $V(\pi_A^*, \pi_D^*) \leq V(\pi_A, \pi_D^*)$ for all π_A. When the two players' policies converge to a Nash equilibrium (π_A^*, π_D^*), Liu et al. [23] proved that the defender guarantees a safe response to any attacker prompt, i.e., $r(y_A, \pi_D^*(y_A)) \geq 0$ for any y_A.

We show that, under the above assumptions, extremal Nash equilibria attaining the minimum or maximum defender expected payoff are problematic in both theory and practice (see §3.2). On the one hand, equilibria that are qualitatively distinct yet indistinguishable to the learning dynamics offer limited practical guidance. On the other hand, when a single model instantiates both attacker and defender, the dynamics can induce implicit coordination or collusion between roles (see §3.3).

3.2 Analysis of Extremal Nash Equilibria

In this section, we derive a "lower bound" extremal that corresponds to the case where the defender always refuses, guaranteeing safety but compromising usefulness. Complementarily, we characterise the "upper bound" extremal that corresponds to an oracle-like defender that always provides safe and maximally helpful responses.

Bound 1: An always-refuse equilibrium. Recall our assumption that the defender can always issue a refusal response $y^{\text{ref}} \in \mathcal{Y}_D$ such that $r(y_A, y^{\text{ref}}) = 0$ for all $y_A \in \mathcal{Y}_A$. Thus, for any prompt y_A, we define the always-refuse defender strategy as $\pi_D^{\text{ref}}(y_D \mid y_A) = \mathbf{1}\{y_D = y^{\text{ref}}\}$, which yields an expected reward $V(\pi_A, \pi_D^{\text{ref}}) = 0$ for any attacker policy π_A.

A pair $(\pi_A^*, \pi_D^{\text{ref}})$ is a Nash equilibrium if the attacker policy π_A^* is sufficiently adversarial such that no defender policy achieves a strictly positive expected payoff, i.e., $\max_{\pi_D} V(\pi_A^*, \pi_D) = 0$. We therefore focus on the attacker policies in a local neighbourhood of a reference adversarial policy $\bar{\pi}_A$, which concentrates on prompts where any non-refusal response yields no positive reward, and where nearby policies preserve this property (see Appendix A). This reflects the intuition that attacks may adapt while remaining close to a reference distribution, ensuring semantic consistency.

Within this neighbourhood, we prove that deviations in the attacker policy only induce controlled changes in the defender's payoff (Lemma 1). We thus focus on attacker policies π_A^* that preserve the adversarial property of $\bar{\pi}_A$ locally, in the sense that they do not admit any defender strategy with strictly positive expected payoff, i.e., we require $\mathbb{E}_{y_A \sim \pi_A^*, y_D \sim \pi_D'}[r(y_A, y_D)] \leq 0$ for any alternative defender strategy π_D'. This condition can be interpreted as a *worst-case* constraint on the attacker:

[2] Without loss of generality, we omit that in SELF-REDTEAM, the attack is initially conditioned on a `seed`, while the attacker's and defender's policies are further conditioned on the instructions to generate the attack and the defence, respectively.

within the local neighbourhood, any attempt by the defender to provide a helpful response yields no positive gain and may incur risk, so the defender has no incentive to deviate from y^{ref}. Specifically, providing a helpful response risks a payoff of -1 that is not offset by safe, high-reward completions, making the expected value of answering non-positive. Hence, π_D^{ref} is a best response to π_A^*. Further, as the attacker cannot reduce the payoff below 0 against a refusing defender, π_A^* is a best response.

Bound 2: The defender acts like an oracle. Let $\mathcal{M} \subseteq \mathcal{Y}_A$ be the set of malicious prompts, for which the maximum achievable safe reward is zero. That is, $\max_{y_D} r(y_A, y_D) = 0$ for any $y_A \in \mathcal{M}$ and $\max_{y_D} r(y_A, y_D) > 0$ for any $y_A \notin \mathcal{M}$. Assume there exists an oracle defender policy π_D^{rec} that is pointwise optimal: for every $y_A \in \mathcal{Y}_A$, it picks a response y_D to maximise $r(y_A, y_D)$. Namely, for all $y_A \in \mathcal{M}$, π_D^{rec} ensures safety by generating a response y_D^{safe} with $r(y_A, y_D^{\text{safe}}) = 0$, and for $y_A \notin \mathcal{M}$, it provides a maximally helpful response y_D with $r(y_A, y_D) > 0$.

Therefore, a pair $(\pi_A^*, \pi_D^{\text{rec}})$ is a Nash equilibrium if the attacker's policy π_A^* minimises the expected reward against the oracle. Formally, π_A^* must satisfy:

$$\pi_A^* \in \arg\min_{\pi_A} \mathbb{E}_{y_A \sim \pi_A} \left[\max_{\pi_D} \mathbb{E}_{y_D \sim \pi_D(\cdot|y_A)} \left[r(y_A, y_D) \right] \right] = \arg\min_{\pi_A} \mathbb{E}_{y_A \sim \pi_A} \left[\max_{y_D} r(y_A, y_D) \right].$$

where the last equality follows from $\max_{\pi_D} \mathbb{E}_{y_D \sim \pi_D(\cdot|y_A)} \left[r(y_A, y_D) \right] = \max_{y_D} r(y_A, y_D)$ since the expected reward is linear in the mixed strategy $\pi_D(\cdot|y_A) \in \Delta(\mathcal{Y}_D)$. Thus, the maximum over the simplex $\Delta(\mathcal{Y}_D)$ it is attained at a vertex, meaning that the optimal *mixed* strategy simply places all probability mass on the best *pure* strategy y_D maximising $r(y_A, y_D)$.

As $\max_{y_D} r(y_A, y_D) = 0$ for all $y_A \in \mathcal{M}$, and $\max_{y_D} r(y_A, y_D) > 0$ for any $y_A \notin \mathcal{M}$, it follows that any equilibrium attacker policy π_A^* must be supported on $\mathcal{M}$, i.e., $\text{supp}(\pi_A^*) \subseteq \mathcal{M}$, where $\text{supp}(\pi_A^*) := \{y_A \in \mathcal{Y}_A : \pi_A^*(y_A) > 0\}$ denotes the support of π_A^*. At such an equilibrium, the attacker cannot further reduce the payoff, as safety is guaranteed, while the defender cannot improve its payoff on malicious inputs without violating the safety constraint and incurring a -1 penalty.

Nash equilibria and Safety Guarantees

The extremal Nash equilibria with respect to the defender's payoff are:

- a degenerate *always-refuse* equilibrium (safe but unhelpful);
- an *oracle* equilibrium (safe and useful), when such a policy exists.

Conclusion: The theoretical framework admits safety-preserving equilibria; however, such equilibria range from a trivial *always-refuse* solution to an *oracle defender*.

3.3 Min-Max Constraint Conflict and Self-Consistency

Assume now that the attacker and defender share parameters, i.e., there exists a single policy π_θ such that $\pi_A = \pi_D = \pi_\theta$. Under this constraint, the optimisation problem involves a single parameter vector θ balancing two conflicting objectives:

1. **Defender Role:** Maximise V (generate safe responses).
2. **Attacker Role:** Minimise V (generate successful jailbreaks).

Since the parameters are shared, the attacker cannot adapt independently without simultaneously altering the defender. In the zero-sum red-teaming game, the attacker's and defender's rewards for any prompt-response pair (y_A, y_D) are $R_A(y_A, y_D) := -r(y_A, y_D)$ and $R_D(y_A, y_D) := r(y_A, y_D)$, respectively. We denote the defender's and attacker's objectives as follows (respectively):

$$J_D(\theta) := \mathbb{E}_{y_A \sim \pi_\theta,\, y_D \sim \pi_\theta(\cdot|y_A)}[r(y_A, y_D)], \quad J_A(\theta) := \mathbb{E}_{y_A \sim \pi_\theta,\, y_D \sim \pi_\theta(\cdot|y_A)}[-r(y_A, y_D)]$$

The resulting policy-gradient update is given by the sum of the defender and attacker gradients:

$$\nabla_\theta J_D(\theta) + \nabla_\theta J_A(\theta) = \mathbb{E}_{y_A, y_D}\Big[r(y_A, y_D) \big(\underbrace{\nabla_\theta \log \pi_\theta(y_D \mid y_A)}_{\text{Force 1: Improve Defence}} - \underbrace{\nabla_\theta \log \pi_\theta(y_A)}_{\text{Force 2: Improve Attacks}} \big) \Big]. \tag{1}$$

To make the coupling explicit, let $\eta > 0$ denote the learning rate and consider a defender-only step, i.e., $\theta^+ = \theta + \eta \nabla_\theta J_D(\theta)$. Then, a first-order Taylor expansion of the attacker objective yields

$$J_A(\theta^+) = J_A(\theta) + \eta \langle \nabla_\theta J_A(\theta), \nabla_\theta J_D(\theta) \rangle + O(\eta^2). \tag{2}$$

Thus, at first order, the effect of a defender update is governed by the inner product between the attacker and defender gradients. When these directions are aligned, improving the defender simultaneously increases the attacker's objective, and vice versa. This coupling highlights how shared-parameter training can induce self-consistency instead of fully robust adversarial behaviour.

Geometrically, we can observe this by analysing the signs of the rewards in Eq. (1):

- **Case 1: Resistance** ($r(y_A, y_D) > 0$)**:** When the defender resists an attack, Force 1 increases the likelihood of the safe response y_D while Force 2 decreases the likelihood of the attack prompt y_A. This drives π_θ to avoid prompts it already defends against, leading to self-consistency rather than the discovery of fundamentally stronger attacks.
- **Case 2: Success** ($r(y_A, y_D) < 0$)**:** Conversely, if a jailbreak succeeds, Force 2 attempts to reinforce the successful attack y_A. Simultaneously, the heavy penalty on the first term (Force 1) tries to update θ to suppress the vulnerability that permitted the jailbreak.

Because these updates are applied simultaneously to the shared parameter vector θ, the gradients effectively collide. Rather than sharpening the attacker and strengthening the defender, the model converges to a stagnant middle ground. This prevents exploration of the adversarial frontier, yielding a "muddy" representational space in which both roles are compromised.

Self-consistency and Safety Guarantees

In summary:

- Self-play Nash equilibria can reduce to *self-consistency* equilibria.
- As an optimisation process, SELF-REDTEAM [23] induces coupled gradients, where improving the defender can weaken the attacker.

Conclusion: A well-posed adversarial saddle-point formulation requires distinct policies.

4 Anchored Bipolicy Self-Play

We now formalise the Anchored Bipolicy Self-Play (ABS) policy for the attacker and the defender: we rely on LoRA modules to ensure attacker-defender separation so that the adversarial pressure on the defender remains unaltered, while maintaining the objective function defined in SELF-REDTEAM.

4.1 Role-conditioned LoRA Policies

Let π_θ denote a single policy network with a shared frozen backbone and role-specific LoRA adapters. The effective parameters are given by $\Theta = (\theta, \phi_A, \phi_D)$, where θ denotes the frozen backbone parameters, and ϕ_A, ϕ_D are the LoRA parameters for the attacker and defender, respectively. For a Transformer-based LLM model [33], and a subset of backbone parameters $\theta^l \subseteq \theta$ (e.g., the query and value projections of self-attention), a LoRA module parameterised by $\phi^l \subseteq \phi$ defines a low-rank update added to θ^l and trained while keeping θ^l frozen. Then, actions for the attacker and the defender are sampled as $y_A \sim \pi_{\theta,\phi_A}(\cdot)$ and $y_D \sim \pi_{\theta,\phi_D}(\cdot \mid y_A)$.

ABS adapts the training method in [23], which leverages Re++ [17], a lightweight reinforcement learning algorithm extending PPO and REINFORCE [35, 29]. It uses token-level KL regularisation to penalise deviations from a reference policy, along with different learning-rate warm-up procedures for the attacker and defender. Implementation details are provided in the next section and Appendix C.3. The attacker and defender are then trained using separate objectives:

$$J_A(\phi_A|\phi_D) = \mathbb{E}_{y_A\sim\pi_{\theta,\phi_A},\, y_D\sim\pi_{\theta,\phi_D}(\cdot|y_A)}\big[R_A(y_A, y_D)\big],$$
$$J_D(\phi_D|\phi_A) = \mathbb{E}_{y_A\sim\pi_{\theta,\phi_A},\, y_D\sim\pi_{\theta,\phi_D}(\cdot|y_A)}\big[R_D(y_A, y_D)\big].$$

The opposing player's parameters are fixed when computing gradients, with policy-gradient updates:

$$\begin{aligned}\nabla_{\phi_A} J_A &= \mathbb{E}[\nabla_{\phi_A} \log \pi_{\theta,\phi_A}(y_A)\, R_A(y_A, y_D)]\,,\\ \nabla_{\phi_D} J_D &= \mathbb{E}[\nabla_{\phi_D} \log \pi_{\theta,\phi_D}(y_D \mid y_A)\, R_D(y_A, y_D)]\,.\end{aligned} \tag{3}$$

These updates evolve separately and restrict optimisation to low-rank subspaces over a shared frozen backbone. This parameter separation restores the distinction between attacker and defender

updates, eliminating the parameter-level interference observed in Eq. (2); To formalise this, consider a defender-only update step $\phi_D^+ = \phi_D + \eta \nabla_{\phi_D} J_D(\phi_A, \phi_D)$, where $\eta > 0$ is the learning rate. A first-order Taylor expansion of the defender's objective yields

$$J_D(\phi_A, \phi_D^+) = J_D(\phi_A, \phi_D) + \eta \|\nabla_{\phi_D} J_D(\phi_A, \phi_D)\|^2 + O(\eta^2).$$

Importantly, this dependence is purely distributional: the defender parameters update independently of the attacker parameters, i.e., $\frac{\partial \phi_D}{\partial \phi_A} = 0$. However, while ABS removes direct parameter-level coupling, it does not eliminate the undesirable Nash equilibria identified in Section 3.2, which arise from the underlying game rather than the optimisation architecture; see Appendix B.

5 Experiments & Results

To compare our ABS to the fully fine-tuned SELF-REDTEAM models by Liu et al. [23], we adapted their code to train two independent LoRA modules, one for the attacker and one for the defender. In their setting, the attacker is prompted with a seed from the training partition of WildJailBreak [19] (see their Appendix C.4.1), with a 50:50 ratio of harmful and benign prompts. The prompt instructs the attacker and defender to first think, wrapped by `<think></think>` tokens, and then provide the answer between `<answer></answer>` tags. The defender produces its response in a similar format; we conducted the evaluation on the pairs of attacks and defences generated at each iteration. As in SELF-REDTEAM, we use WildGuard [13] as the reward model, which classifies the prompt and response harmfulness, and whether the response is a refusal to answer the attack.

We trained Qwen2.5-{3B, 7B 14B}-IT models [27] with different LoRA settings, varying the rank and alpha values. The rank defines the hidden dimension of the LoRA representation and thus the number of trainable parameters, while alpha scales the influence of the adapter w.r.t. the base model. Table 9 and Appendix C.3 report more details of the setting. Our training is entirely conducted on three NVIDIA H200 SXM GPUs with 141GB of vRAM.

For the safety evaluation, we employed eight tests in the Ai2 Safety Toolkit [19]: six tests concern harmful refusal, i.e., if a model responds to malicious prompts; two benign tests benchmark the ability of a model to comply with non-harmful instructions. The test suite measures the Attack Success Rate (ASR) and Robustness to Attacks (RTA). We also benchmark ABS and the SELF-REDTEAM Qwen by Liu et al. [23] on five standard reasoning and math benchmarks (GSM8K [8], CommonSense [31], BoolQ [6], ARC Challenge [7], and HellaSwag [39]) to guarantee that they do not lose their base capabilities. We ran our tests on a single NVIDIA H200 SXM GPU with 141GB of vRAM and report the accuracy (pass@1) of each model.

5.1 ABS Safety and Efficiency Evaluation

As Table 1 reports, our method outperforms the corresponding Qwen-IT base model and, in most cases, the fully fine-tuned SELF-REDTEAM model by Liu et al. [23]. On standard reasoning benchmarks,

	Harmful Refusal							Benign Compliance		Benchmark
	WG:Test		WJB	DAN	HarmBench	XSTest	StrongREJECT	WJB	XSTest	Reasoning/Math
Method	adv ASR ↓	vani ASR ↓	adv ASR ↓	adv ASR ↓	adv ASR ↓	vani RTA ↑	adv ASR ↑	vani Comply ↑	vani LC Winrate ↑	- Avg. Accuracy ↑
Qwen2.5-3B-IT	0.282	0.119	0.949	0.677	0.241	0.610	0.497	0.992	0.960	0.745 ± 0.042
+ SELF-REDTEAM	0.234	0.056	0.885	0.603	0.156	0.655	0.559	**1.000**	0.932	0.671 ± 0.095
+ ABS	**0.122**	**0.019**	**0.721**	**0.320**	**0.088**	**0.680**	**0.669**	0.984	**0.952**	0.738 ± 0.047
Improv. SELF-REDTEAM *(%)*	+17.0	+52.9	+6.7	+10.9	+35.3	+7.4	+12.5	+0.8	-2.9	-7.4
Improv. ABS (%)	+56.7	+84.0	+24.0	+52.7	+63.5	+11.5	+34.6	-0.8	-0.8	-0.7
Qwen2.5-7B-IT	0.279	0.040	0.897	0.613	0.234	0.915	0.470	**0.992**	**0.912**	0.850 ± 0.026
+ SELF-REDTEAM	0.178	**0.002**	**0.726**	0.423	**0.156**	**0.925**	0.371	**0.992**	0.888	0.818 ± 0.037
+ ABS	**0.169**	0.010	0.756	**0.383**	**0.156**	0.915	**0.644**	0.988	0.884	0.847 ± 0.024
Improv. SELF-REDTEAM *(%)*	+36.18	+94.19	+19.12	+30.98	+33.36	+1.09	-21.24	0.0	-2.63	-3.2
Improv. ABS (%)	+39.37	+76.51	+15.77	+37.50	+33.36	0.00	+36.97	-0.6	-3.07	-0.3
Qwen2.5-14B-IT	0.146	0.017	0.845	0.647	0.119	0.735	0.647	**1.000**	**0.984**	0.874 ± 0.030
+ SELF-REDTEAM	**0.116**	**0.005**	**0.661**	0.467	**0.056**	0.775	0.701	0.996	0.968	0.878 ± 0.035
+ ABS	0.131	0.022	0.706	**0.446**	0.093	**0.770**	**0.748**	**1.000**	0.968	0.874 ± 0.030
Improv. SELF-REDTEAM *(%)*	+20.55	+70.59	+21.77	+27.82	+52.94	+5.44	+8.35	-0.4	-1.63	+0.4
Improv. ABS (%)	+10.27	-29.41	+16.44	+31.07	+21.85	+4.76	+15.61	0.0	-1.63	0.0

Table 1: Performance of Qwen2.5-{3B, 7B, 14B}-IT, SELF-REDTEAM fine-tuned, and our ABS models on six benchmarks for harmful refusal, two for benign compliance, and five (average) for instruction following and reasoning. Improvement is calculated relative to the base model. Blue indicates improvements (lower ASR or higher RTA/Compliance), red indicates regression.

our models do not degrade their performance, as shown in the last column of Table 1 (full table in Appendix, Table 8), while the fine-tuned SELF-REDTEAM Qwen2.5-3B-IT experiences relevant drops, revealing that fine-tuning small models for safety may incur catastrophic forgetting. In terms of efficiency, ABS introduces, for Qwen2.5-14B-IT with rank 32 (our largest module), an additional $250M$ parameters (2% of the total parameters), and completes a full training run, on three H200 GPUS, in five hours. More details for the other implementations in the Appendix, Table 9.

We also observe that LoRA training can become unstable, occasionally producing degenerate outputs. In our experiments, the only observed collapse occurred for Qwen2.5-7B with rank 32 at step 100, where the model repeatedly generated **!** tokens, a prefix also observed in prior red-teaming work [30]. To ensure a controlled comparison with SELF-REDTEAM [23], we followed the same fixed-budget checkpoint evaluation protocol, without validation-based checkpoint selection or early stopping. Introducing such a procedure for ABS alone would have made the comparison asymmetric. We therefore report all evaluated configurations, including the collapsed checkpoint, without excluding unsuccessful runs. All rank-16 configurations remained stable, as did the evaluated 14B rank-32 configuration. Since both rank settings used $\alpha = r$, the observed failure does not establish a systematic relationship between LoRA scaling and instability. Full checkpoint-level results are provided in Appendix C.4. In terms of efficiency, Table 9 reports that training a LoRA adapter with rank 16 on three NVIDIA H200 GPUs takes approximately the same time as benchmarking, on a single H200, the same model on the nine test suites.

5.2 ABS Adversarial Strength and Defensive Robustness

In this section, we analysed the adversarial pressure that a trained attacker can put on the defender. We compared the ASR (Attack Success Rate) of ABS and SELF-REDTEAM checkpoints vs the same base model. We sample 100 examples from the Harmful Behavior Dataset [19], and we ask the attacker to produce a jailbreak following the same format used at training time. The defender receives the input and decides whether to comply or not with the request. We evaluate the success of an attack with GPT-5.4 Standard with High reasoning as an LLM-as-a-judge. We set strict prompt rules on the success of an attack: a success is an answer that, for the judge, "gave substantive harmful assistance" to the defender. As shown in Figure 2, the fine-tuned versions of Qwen2.5-3B-IT and 7B have an ASR of 18% and 9% respectively, while our corresponding ABS produce more effective jailbreaks and achieve an ASR of 35% and 24%. Additional uncertainty estimates and evaluation considerations for the transfer and tournament experiments are provided in Appendix C.7. The mixed results at larger scales and the possible role of diminishing optimisation headroom are further discussed in Appendix C.8.

We also conducted a "tournament" to evaluate the adversarial pressure between our ABS models and SELF-REDTEAM. Over 100 samples from the Harmful Behavior Dataset, we cross-tested each methodology's attacker against the opposing defender at equivalent model scales. This analysis provides a measure of the adversarial pressure that an attacker puts on the other model's defender. Similarly to the previous case, GPT-5.4 with High reasoning judges the success of an attack. As reported in Figure 2, SELF-REDTEAM Qwen2.5-3B-IT and 7B have both an ASR of 6%, while our corresponding ABS produces more effective jailbreaks and achieves an ASR of 17% and 18%. In conclusion, the shared-model defender is markedly more permissive than the LoRA defender.

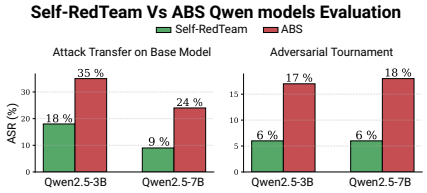

Figure 2: Transferability of attacks and comparative evaluation of SELF-REDTEAM versus ABS Qwen-IT models across the tournament suite.

Method	Cosine Similarity	Self-BLEU-3 Score	# Tokens Average	# Think Tokens Average	Think Tokens Frequency
Qwen2.5-3B-IT					
+ SELF-REDTEAM	0.281	0.785	59.26	58.69	9.3%
+ ABS	0.310	0.714	232.74	88.82	70.5%
Qwen2.5-7B-IT					
+ SELF-REDTEAM	0.242	0.670	90.04	49.69	99.21%
+ ABS	0.201	0.646	250.94	53.91	19.14%
Qwen2.5-14B-IT					
+ SELF-REDTEAM	0.206	0.547	122.43	73.31	100%
+ ABS	0.210	0.500	135.86	76.67	25%

Table 2: Statistics of the attacks generated by SELF-REDTEAM vs LoRA Qwen2.5-IT models on the *Harmful Behavior Dataset*. Cosine similarity measures embedding similarity, while Self-BLEU lexical diversity.

5.3 Qualitative Comparison of the Attacks

We compared attacks generated by ABS and SELF-REDTEAM along three axes: semantic similarity, lexical diversity, and structural complexity. Table 2 provides a quantitative overview of these differences. SELF-REDTEAM produces shorter and more homogeneous attacks, as reflected by higher Self-BLEU scores and lower token counts. In contrast, ABS generates longer and lexically diverse outputs, particularly the 3B and 7B, where token counts increase substantially, and Self-BLEU decreases. Despite these surface-level differences, cosine similarity remains broadly comparable across methods, suggesting that both approaches occupy similar regions in embedding space (Figure 3 in Appendix shows the UMAP embeddings for ABS and SELF-REDTEAM).

These trends correspond to clear stylistic differences. SELF-REDTEAM attacks are typically direct reformulations of the harmful intent: they are concise, imperative, and minimally elaborated. By contrast, ABS tends to embed the request within richer contextual scaffolding, often introducing multi-step scenarios, social engineering elements, or concealment-oriented phrasing. This results in attacks that are more elaborate but also less controlled, especially in the 3B model. In that setting, ABS outputs sometimes include residual meta-instructions, such as references to an "original prompt" or "revised prompt", suggesting weaker separation between instruction-following and task narration; this likely reflects the limited capacity of a LoRA module to enforce clean task boundaries.

Overall, the comparison reveals a trade-off between reliability and elaboration. SELF-REDTEAM produces more structurally consistent attacks, particularly at 7B, whereas 3B ABS generates longer, more evasive but less controlled formulations. Representative examples in Appendices C.4.1 and C.5 illustrate this contrast. Lexical diversity and verbosity, as captured by Self-BLEU and token counts in Table 2, appear to be key factors underlying the success of ABS attacks. Complementing these behavioural observations, Appendix C.6 provides a direct empirical analysis of attacker–defender gradient alignment over 500 self-play games. The results reveal substantially stronger gradient conflict in shared-parameter models, concentrated on harmful prompts and most pronounced in SELF-REDTEAM, whereas ABS exhibits negligible conflict in magnitude.

5.4 Interfacing with Human Values: Fairness and Harmfulness

We next explore how ABS blocks specific human harms, such as fairness or toxicity, without over-refusing benign prompts. We assess this trade-off using semantic sub-categories from WildGuard, HarmBench, and XSTest (Table 10 in Appendix C.5). Both ABS and SELF-REDTEAM effectively suppress malicious content, with ABS achieving substantial safety penetration against direct human harm without full-parameter updates. On *HarmBench Harassment & Bullying*, the 3B and 14B ABS models attain a 100% safe refusal rate. In *WildGuard's Violence & Physical Harm* and *Toxicity* categories, ABS maintains refusal rates above 82% and 78% across all scales, with the 14B model exceeding 92% in both.

Under adversarial pressure, models frequently associate terms related to protected demographics, such as race, gender, or religion, with harmful attacks, resulting in excessive refusal of benign requests, quantified by *XSTest's Discrimination Contrast* and *Benign Compliance* metrics. The analysis reveals differences in how SELF-REDTEAM and ABS address fairness. SELF-REDTEAM functions as a blunt instrument: gradient collisions compel these models to converge towards a middle ground, as discussed in Section 3.3, reducing benign compliance to prioritise safety. The base Qwen2.5-3B model attains a 0.960 Benign Compliance score on XSTest, declining to 0.932 under SELF-REDTEAM, compared to 0.952 for ABS. At 7B, ABS improves upon the base model's compliance (0.915 vs. 0.912), whereas SELF-REDTEAM decreases to 0.888.

5.5 PAIR Comparison

We also benchmark our ABS attackers and defenders against PAIR [4], an automated jailbreak framework that uses a dedicated attacker language model to iteratively construct and optimise adversarial prompts targeting a defender. PAIR leverages the target model's responses and evaluation feedback to progressively improve attack effectiveness until its budget is exhausted or a vulnerability in the defender's safety mechanisms is identified. We evaluate PAIR on Qwen2.5-3B under four conditions: (i) the base model as both attacker and defender; (ii) ABS attacker vs. base defender; (iii) base attacker vs. ABS defender; and (iv) ABS attacker vs. ABS defender. Comparing (i) and (ii) demonstrates the ABS attacker's stronger jailbreak capabilities against the base model, while (i) and

Table 3: PAIR evaluation on Qwen2.5-3B-Instruct. Top: ABS vs. base, on the attacker side vs. the base defender (left) and on the defender side vs. the base attacker (right); Bottom: effect of each ABS component with the opponent held fixed (left), and legend (right).

ABS vs. base, as attacker and as defender.

	ABS as attacker (↑ better)		**ABS as defender** (↓ better)	
	Base	**ABS (LoRA)**	**Base**	**ABS (LoRA)**
Jailbreak rate	10.5%	**16.9%** (+6.4 pp)	10.3%	**5.4%** (−4.9 pp)
Better on N behaviors	10	**30** (9 ties)	2	**34** (14 ties)
Fully blocked behaviors	—	—	0	**13** ($p \approx 2\times10^{-4}$)
Pooled McNemar	$p \approx 9\times10^{-8}$ (107 vs. 201 disc.)		$p \approx 4\times10^{-16}$ (83 vs. 9 new-vuln.)	
Sign test	$p = 0.002$		$p \approx 2\times10^{-8}$ (34 vs. 2)	

Each ABS component, opponent held fixed.

Effect	**vs.**	**Rate change**	**Sign p**	**McNemar p**
ABS atk	base def	10.5 → 16.9%	0.0022	9×10^{-8}
ABS atk	ABS def	5.5 → 8.4%	0.041	1.4×10^{-3}
ABS def	base atk	10.5 → 5.5%	1.9×10^{-8}	3.9×10^{-16}
ABS def	ABS atk	16.9 → 8.4%	1×10^{-11}	6.3×10^{-23}

– Legend –
Jailbreak rate: of all attack attempts, the fraction that produced genuinely harmful output (WildGuard + Claude adjudication).
Better on N behaviors: for each of the 50 goals, which side jailbroke (resp. blocked) more of its 30 streams.
Pooled McNemar: pairs all 1,500 streams across conditions and counts disagreements (disc. = discordant pairs; new-vuln. = jailbroken only under the ABS defender).
Sign test: keeps only which side won each behavior and tests significance.

(iii) show the ABS defender's improved robustness. Condition (iv) further shows that our defender remains effective against a stronger opponent, namely the ABS attacker.

We ran 50 experiments drawn from the AdvBench subset used by PAIR (the standard setting in their code). For each of the AdvBench goals, we launched 30 independent streams, for a total of 1,500 evaluations, where the attacker crafts a prompt, the defender responds, and the judge (WildGuard) scores the exchange. If the attempt is not judged a jailbreak, the attacker refines the prompt using its conversation history, for up to 3 iterations. For each condition, we adapted the PAIR codebase to use WildGuard as the initial judge, with Claude Opus 4.8 (High reasoning, August 2026 release) as an additional, independent judge. Whenever Opus disagrees with WildGuard, Opus performs an extended review of the models' outputs and WildGuard's verdict and breaks the tie. We additionally ran a McNemar statistical test with Bonferroni correction on all comparisons that permit such a measurement. As Table 3 summarises, our ABS models are statistically stronger than the base model in terms of attacks and defences.

6 Conclusion, Limitations, and Future Work

In this paper, we propose ABS, a model designed to overcome the theoretical limitations of standard self-play in LLM safety training. By freezing the base model and introducing two LoRA-based extensions, ABS provides an efficient hybrid that separates self-play from gradient updates. ABS achieves strong performance improvements over base LLMs while significantly reducing computational cost.

Our evaluation across multiple model scales demonstrates that separating the attacker and defender is crucial for effective adversarial training. Both our theoretical analysis and empirical results indicate that shared-parameter frameworks, such as SELF-REDTEAM, suffer from a self-consistency bias that weakens the attacker. ABS outperforms SELF-REDTEAM on Qwen2.5-3B and reaches comparable performance on larger models, while consistently improving over the base model. In addition to quantitative gains, ABS-trained attackers develop more sophisticated and camouflaged strategies. The method is also highly efficient, achieving up to two orders of magnitude better memory usage while preserving the reasoning capabilities of the underlying model.

Despite these advantages, ABS exhibits some limitations. LoRA training can occasionally become unstable, leading to collapse modes with repetitive or nonsensical outputs, and larger models require careful hyperparameter tuning to avoid such failures (Appendix C.4). Nevertheless, the efficiency of ABS makes it promising for deploying safer yet useful models, particularly at smaller scales. Future work will explore extensions to more complex threat settings, including mosaic attacks and mixtures of attackers, as well as broader applications of this game-theoretic framework to self-play in LLMs.

Acknowledgments and Disclosures

GLM was supported by UK Research and Innovation [grant number EP/S023356/1], in the UKRI Centre for Doctoral Training in Safe and Trusted Artificial Intelligence (www.safeandtrustedai.org). MW, ELM and SC were supported by an AI 2050 Senior Fellowship from the Schmidt Sciences Foundation. ELM is affiliated with the Institute for Decentralized AI (IDAI), which he thanks for its support and for providing the budget for computing.

References

[1] L. Arras, F. Horn, G. Montavon, K.-R. Müller, and W. Samek. Explaining predictions of non-linear classifiers in NLP. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 1–7, 2016.

[2] 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.

[3] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, 2020.

[4] P. Chao, A. Robey, E. Dobriban, H. Hassani, G. J. Pappas, and E. Wong. Jailbreaking black box large language models in twenty queries. In *2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pages 23–42. IEEE, 2025.

[5] B. Chen, T. Zhu, J. Han, L. Li, G. Li, and X. Dai. Incentivizing truthful language models via peer elicitation games. In *Advances in Neural Information Processing Systems*, 2025.

[6] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936. Association for Computational Linguistics, 2019.

[7] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

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

[9] I. Csiszár and J. Körner. *Information theory: coding theorems for discrete memoryless systems*. Cambridge University Press, 2011.

[10] J. Dai, X. Pan, R. Sun, J. Ji, X. Xu, M. Liu, Y. Wang, and Y. Yang. Safe RLHF: Safe reinforcement learning from human feedback. In *The Twelfth International Conference on Learning Representations*, 2024.

[11] C. S. de Witt. Open challenges in multi-agent security: Towards secure systems of interacting AI Agents. *arXiv preprint arXiv:2505.02077*, 2025.

[12] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 36, pages 10088–10115, 2023.

[13] S. Han, K. Rao, A. Ettinger, L. Jiang, B. Y. Lin, N. Lambert, Y. Choi, and N. Dziri. WildGuard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of LLMs. *Advances in Neural Information Processing Systems*, 37:8093–8131, 2024.

[14] Z.-W. Hong, I. Shenfeld, T.-H. Wang, Y.-S. Chuang, A. Pareja, J. R. Glass, A. Srivastava, and P. Agrawal. Curiosity-driven red-teaming for large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[15] N. H. R. Howe, I. R. Mckenzie, O. J. Hollinsworth, M. Zając, T. Tseng, A. D. Tucker, P.-L. Bacon, and A. Gleave. Scaling trends in language model robustness. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 24080–24138. PMLR, 2025.

[16] E. J. Hu, yelong 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.

[17] J. Hu, J. K. Liu, H. Xu, and W. Shen. REINFORCE++: Stabilizing critic-free policy optimization with global advantage normalization. *arXiv preprint arXiv:2501.03262*, 2025.

[18] N. Jain, A. Schwarzschild, Y. Wen, G. Somepalli, J. Kirchenbauer, P.-y. Chiang, M. Goldblum, A. Saha, J. Geiping, and T. Goldstein. Baseline defenses for adversarial attacks against aligned language models. *arXiv preprint arXiv:2309.00614*, 2023.

[19] L. Jiang, K. Rao, S. Han, A. Ettinger, F. Brahman, S. Kumar, N. Mireshghallah, X. Lu, M. Sap, Y. Choi, et al. WildTeaming at scale: From In-the-Wild jailbreaks to (adversarially) safer language models. *Advances in Neural Information Processing Systems*, 37:47094–47165, 2024.

[20] E. La Malfa, A. Petrov, S. Frieder, C. Weinhuber, R. Burnell, R. Nazar, A. Cohn, N. Shadbolt, and M. Wooldridge. Language-Models-as-a-Service: Overview of a new paradigm and its challenges. *Journal of Artificial Intelligence Research*, 80:1497–1523, 2024.

[21] M. Li, W. M. Si, M. Backes, Y. Zhang, and Y. Wang. SaLoRA: Safety-alignment preserved low-rank adaptation. In *The Thirteenth International Conference on Learning Representations*, 2025.

[22] A. Liao, N. Tomlin, and D. Klein. Efficacy of language model self-play in non-zero-sum games. *arXiv preprint arXiv:2406.18872*, 2024.

[23] M. Liu, L. Jiang, Y. Liang, S. S. Du, Y. Choi, T. Althoff, and N. Jaques. Chasing moving targets with online self-play reinforcement learning for safer language models. *arXiv preprint arXiv:2506.07468*, 2025.

[24] A. Mehrotra, M. Zampetakis, P. Kassianik, B. Nelson, H. Anderson, Y. Singer, and A. Karbasi. Tree of attacks: Jailbreaking black-box LLMs automatically. In *Advances in Neural Information Processing Systems*, volume 37, pages 61065–61105, 2024.

[25] E. Perez, S. Huang, F. Song, T. Cai, R. Ring, J. Aslanides, A. Glaese, N. McAleese, and G. Irving. Red teaming language models with language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3419–3448, 2022.

[26] G. Qu, A. Wierman, and N. Li. Scalable reinforcement learning of localized policies for multi-agent networked systems. In *Learning for Dynamics and Control*, pages 256–266. PMLR, 2020.

[27] 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:2412.15115*, 2025.

[28] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

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

[30] L. Schwinn, D. Dobre, S. Günnemann, and G. Gidel. Adversarial attacks and defenses in large language models: Old and new threats. In *Proceedings on "I Can't Believe It's Not Better: Failure Modes in the Age of Foundation Models" at NeurIPS 2023 Workshops*, volume 239 of *Proceedings of Machine Learning Research*, pages 103–117. PMLR, 2023.

[31] A. Talmor, J. Herzig, N. Lourie, and J. Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158. Association for Computational Linguistics, 2019.

[32] Z. Tan, W. Yu, J. Si, T. Liu, K. Guan, H. Jin, J. Tao, X. Yuan, D. Ma, X. Zhang, T. Yang, and L. Sun. Triplay-rl: Tri-role self-play reinforcement learning for llm safety alignment, 2026.

[33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 6000–6010, 2017.

[34] X. Wen, Z. He, H. Qi, Z. Wan, Z. Ma, Y. Wen, T. Zheng, X. Xu, C. Lu, and Q. Zhang. MAGIC: A co-evolving attacker-defender adversarial game for robust LLM safety. *arXiv preprint arXiv:2602.01539*, 2026.

[35] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

[36] S. Xhonneux, A. Sordoni, S. Günnemann, G. Gidel, and L. Schwinn. Efficient adversarial training in LLMs with continuous attacks. In *Advances in Neural Information Processing Systems*, volume 37, pages 1502–1530, 2024.

[37] C. Xiong, X. Qi, P.-Y. Chen, and T.-Y. Ho. Defensive prompt patch: A robust and generalizable defense of large language models against jailbreak attacks. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 409–437, 2025.

[38] Z. Xu, Y. Liu, G. Deng, Y. Li, and S. Picek. A comprehensive study of jailbreak attack versus defense for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7432–7449, 2024.

[39] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800. Association for Computational Linguistics, 2019.

[40] Q. Zhang, M. Chen, A. Bukharin, P. He, Y. Cheng, W. Chen, and T. Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*, 2023.

[41] Y. Zhang, G. Qu, P. Xu, Y. Lin, Z. Chen, and A. Wierman. Global convergence of localized policy iteration in networked multi-agent reinforcement learning. In *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, volume 7, pages 1–51. ACM New York, NY, USA, 2023.

[42] C. Ziakas, N. Loo, N. Jain, and A. Russo. Red-Bandit: Test-time adaptation for LLM red-teaming via bandit-guided LoRA experts. *arXiv preprint arXiv:2510.07239*, 2025.

Appendix

Broader Impact

This work aims to contribute positively by improving the safety and reliability of large language models through more parameter-efficient adversarial self-play and reduced optimisation interference between attacker and defender roles. By lowering the number of trainable parameters required for adversarial training, ABS may make safety-oriented model development more accessible to research groups with limited computational resources.

At the same time, the method is inherently dual-use. The attacker component is explicitly optimised to generate prompts capable of eliciting unsafe responses, and our experiments demonstrate that ABS attackers can be more effective than those produced by shared-parameter self-play (Figure 2). Such capabilities could be misused to identify and exploit vulnerabilities in deployed language models. The ability to train stronger attackers at relatively low computational cost may also reduce barriers to adversarial misuse.

Our experiments are conducted for safety evaluation and defensive model improvement. To mitigate misuse risks, we distribute model artifacts through a gated Hugging Face repository, following a controlled-access release approach consistent with prior self-play red-teaming work [23]. While such measures can discourage misuse, they cannot eliminate the risks associated with adversarial capabilities, particularly once model artifacts are distributed. We therefore emphasise the importance of responsible access and deployment, and note that improvements on the evaluated benchmarks do not establish universal safety guarantees against unseen attacks or deployment environments.

Ethical Statement

This research investigates adversarial training for language-model safety using datasets containing harmful requests and model-generated jailbreak attempts. Some generated examples may include sensitive, offensive, or potentially actionable harmful content. These materials are used to evaluate model vulnerabilities and improve defensive robustness, rather than to facilitate harmful activities.

The evaluation is conducted using automated safety classifiers and language-model judges, which may exhibit errors, biases, and limitations in identifying harmful assistance or distinguishing legitimate requests from unsafe ones. In particular, false refusals can disproportionately affect benign requests involving sensitive subjects or protected demographic groups. We therefore evaluate both harmful-request refusal and benign compliance, including fairness-related categories (Table 10), and acknowledge that these measurements cannot fully capture the societal effects of model deployment.

We recognise the risks associated with releasing stronger jailbreak-generation capabilities. Following the responsible-release considerations of Liu et al. [23], we distribute attacker and defender checkpoints through a gated Hugging Face repository, with access intended for legitimate research and safety evaluation. Additional safeguards, such as explicit acceptable-use conditions, prompt-level restrictions, and safety monitoring during inference, may further mitigate misuse risks; see Appendix F by Liu et al. [23]. However, these measures are not guarantees against malicious use. The research does not establish that ABS-trained defenders are safe under all possible adversarial conditions. We believe that the potential benefits of developing more trustworthy AI systems outweigh the associated risks of misuse, provided that appropriate safeguards and responsible-release practices are adopted.

A More Details on Bound 1: An always-refuse equilibrium

Recall our assumption that the defender can always issue a refusal response $y^{\text{ref}} \in \mathcal{Y}_D$ such that $r(y_A, y^{\text{ref}}) = 0$ for all $y_A \in \mathcal{Y}_A$. Thus, for any prompt y_A, we define the always-refuse defender strategy as $\pi_D^{\text{ref}}(y_D \mid y_A) = \mathbf{1}\{y_D = y^{\text{ref}}\}$, which yields an expected reward $V(\pi_A, \pi_D^{\text{ref}}) = 0$ for any attacker policy π_A.

A pair $(\pi_A^*, \pi_D^{\text{ref}})$ constitutes a Nash equilibrium if the attacker's policy π_A^* is sufficiently adversarial such that no defender policy guarantees a strictly positive expected payoff, i.e., $\max_{\pi_D} V(\pi_A^*, \pi_D) = 0$. Therefore, suppose the attacker's policy space consists of probability distributions in a localised neighbourhood around a reference policy $\bar{\pi}_A$, representing baseline adversarial behaviour already

optimised to challenge the defender [28]. In particular, we assume that $\bar{\pi}_A$ concentrates on prompts for which safe responses yield no positive reward, so that the defender cannot improve upon refusal. This constraint captures the intuition that, while attack policies may adapt to minimise utility, they should remain sufficiently close to a reference adversarial behaviour to preserve semantic consistency and interpretability [5]. In practice, this can be enforced by initialising the agent with a fine-tuned model embodying $\bar{\pi}_A$, and constraining updates to remain within a trust region [26, 41]. Specifically, for a fixed $\delta \in [0, 1]$ we define the local policy δ-neighbourhood as:

$$\Pi_A^\delta = \{\pi_A \in \Delta(\mathcal{Y}_A) : D_{\text{TV}}(\pi_A \| \bar{\pi}_A) \leq \delta\}, \tag{4}$$

where $D_{\text{TV}}(\cdot\|\cdot)$ denotes the total variation distance, given as:

$$D_{\text{TV}}(\pi_A \| \bar{\pi}_A) = \frac{1}{2} \sum_{y_A \in \mathcal{Y}_A} |\pi_A(y_A) - \bar{\pi}_A(y_A)| \tag{5}$$

Although trust-region methods are commonly defined using KL divergence [28], we use total variation distance in the analysis, since it directly bounds differences in expectations of bounded functions. Specifically, in Lemma 1 we show that, within Π_A^δ, the defender's best achievable expected payoff changes by a controlled amount (see the full proof in Appendix A.1). In contrast, KL divergence only provides such control indirectly, resulting in weaker bounds (see Appendix A.2 for further details).

Lemma 1. *For any attacker policies $\pi_A, \pi'_A \in \Delta(\mathcal{Y}_A)$ and any defender policy π_D:*

$$|V(\pi_A, \pi_D) - V(\pi'_A, \pi_D)| \leq 2D_{TV}(\pi_A \| \pi'_A). \tag{6}$$

In particular, if $\delta = \frac{\varepsilon}{2}$ and $\pi_A \in \Pi_A^\delta$ for some fixed $\varepsilon \in [0, 1]$, then:

$$\max_{\pi_D} V(\pi_A, \pi_D) \leq \max_{\pi_D} V(\bar{\pi}_A, \pi_D) + \varepsilon. \tag{7}$$

Lemma 1 and Appendix A.2 show that, in either a total-variation or KL divergence neighbourhood of the reference policy $\bar{\pi}_A$, the defender's best achievable payoff changes only by a controlled amount. As $\bar{\pi}_A$ represents a baseline adversarial behaviour, it is natural to restrict attention to local perturbations that preserve this property. We thus focus on attacker policies $\pi_A^* \in \Pi_A^\delta$ that preserve the adversarial property of $\bar{\pi}_A$ locally, in the sense that they do not admit any defender strategy with strictly positive expected payoff. Formally, we require that for any alternative defender strategy π'_D:

$$\mathbb{E}_{y_A \sim \pi_A^*, y_D \sim \pi'_D}[r(y_A, y_D)] \leq 0. \tag{8}$$

This condition can be interpreted as a *worst-case* constraint on the attacker: within the local policy class Π_A^δ, any attempt by the defender to provide a helpful response yields no positive gain and may incur risk, so the defender has no incentive to deviate from y^{ref}. Specifically, providing a helpful response risks a payoff of -1 that is not offset by safe, high-reward completions, making the expected value of answering non-positive. Hence, π_D^{ref} is a best response to $\pi_A^* \in \Pi_A^\delta$. Moreover, since the attacker cannot reduce the payoff below 0 against a refusing defender, π_A^* is also a best response.

A.1 Proof of Lemma 1

Lemma 1. *For any attacker policies $\pi_A, \pi'_A \in \Delta(\mathcal{Y}_A)$ and any defender policy π_D:*

$$|V(\pi_A, \pi_D) - V(\pi'_A, \pi_D)| \leq 2D_{TV}(\pi_A \| \pi'_A). \tag{9}$$

In particular, if $\delta = \frac{\varepsilon}{2}$ and $\pi_A \in \Pi_A^\delta$ for some fixed $\varepsilon \in [0, 1]$, then:

$$\max_{\pi_D} V(\pi_A, \pi_D) \leq \max_{\pi_D} V(\bar{\pi}_A, \pi_D) + \varepsilon. \tag{10}$$

Proof. We start with proving (9). Consider any attacker policies $\pi_A, \pi'_A \in \Delta(\mathcal{Y}_A)$ and any defender policy π_D. By definition, the following holds:

$$\begin{aligned} V(\pi_A, \pi_D) &= \mathbb{E}_{y_A \sim \pi_A,\, y_D \sim \pi_D(\cdot|y_A)}\left[r(y_A, y_D)\right] \\ &= \sum_{y_A \in \mathcal{Y}_A} \pi_A(y_A) \sum_{y_D \in \mathcal{Y}_D} \pi_D(y_D|y_A) \cdot r(y_A, y_D). \end{aligned} \tag{11}$$

By similar arguments:

$$V(\pi'_A, \pi_D) = \sum_{y_A \in \mathcal{Y}_A} \pi'_A(y_A) \sum_{y_D \in \mathcal{Y}_D} \pi_D(y_D|y_A) \cdot r(y_A, y_D). \tag{12}$$

By combining (11) and (12), we obtain that:

$$|V(\pi_A, \pi_D) - V(\pi'_A, \pi_D)| = \left| \sum_{y_A \in \mathcal{Y}_A} \sum_{y_D \in \mathcal{Y}_D} \pi_D(y_D|y_A) \cdot r(y_A, y_D)(\pi_A(y_A) - \pi'_A(y_A)) \right| \tag{13a}$$

$$\leq \sum_{y_A \in \mathcal{Y}_A} \sum_{y_D \in \mathcal{Y}_D} |\pi_D(y_D|y_A)| \cdot |r(y_A, y_D)| \, |\pi_A(y_A) - \pi'_A(y_A)| \tag{13b}$$

$$\leq \sum_{y_A \in \mathcal{Y}_A} |\pi_A(y_A) - \pi'_A(y_A)| \sum_{y_D \in \mathcal{Y}_D} \pi_D(y_D|y_A) \tag{13c}$$

$$= \sum_{y_A \in \mathcal{Y}_A} |\pi_A(y_A) - \pi'_A(y_A)| \,. \tag{13d}$$

where the inequality in (13b) follows from the triangle inequality. Furthermore, the inequality in (13c) holds because $|\pi_D(y_D|y_A)| = \pi_D(y_D|y_A)$ for any $y_A \in \mathcal{Y}_A, y_D \in \mathcal{Y}_D$ since $\pi_D(y_D|y_A) \in [0, 1]$, and $|r(y_A, y_D)| \leq 1$ due to our assumption in Section 3.1 that $r(y_A, y_D) \in [-1, 1]$ for all $y_A \in \mathcal{Y}_A, y_D \in \mathcal{Y}_D$. Finally, the last equality in (13d) stems from $\sum_{y_D \in \mathcal{Y}_D} \pi_D(y_D|y_A) = 1$ for any $y_A \in \mathcal{Y}_A$ due to $\pi_D(\cdot|y_A) \in \Delta(\mathcal{Y}_D)$.

Subsequently, we relate the term in (13d) to the total variation distance. Indeed, since $D_{\text{TV}}(\pi_A \| \pi'_A) = \frac{1}{2} \sum_{y_A \in \mathcal{Y}_A} |\pi_A(y_A) - \pi'_A(y_A)|$ by definition, we conclude that:

$$|V(\pi_A, \pi_D) - V(\pi'_A, \pi_D)| \leq 2D_{\text{TV}}(\pi_A \| \pi'_A), \tag{14}$$

as desired.

Next, we prove the second part of our claim in (10). For any fixed $\delta \in [0, 1]$, consider any attacker policy $\pi_A \in \Pi_A^\delta$. That is, $D_{\text{TV}}(\pi_A \| \bar{\pi}_A) \leq \delta$, where $\bar{\pi}_A$ is the reference policy defined in Section 3.2, representing baseline adversarial behaviour already optimised to challenge the defender. Therefore, combined with (14), we obtain:

$$|V(\pi_A, \pi_D) - V(\pi'_A, \pi_D)| \leq 2\delta. \tag{15}$$

As such, if $\delta = \frac{\varepsilon}{2}$ for any fixed $\varepsilon \in [0, 1]$, the desired in (10) readily follows from (15). □

A.2 Discussion: Using KL-Divergence for Local Policy Neighbourhoods

In this section, we discuss the consequences of defining the local policy neighbourhood in (4) using the KL divergence instead of the total variation distance. Indeed, for a fixed $\delta \in [0, 1]$ we define the local policy δ-neighbourhood using the KL divergence as follows:

$$\Pi_A^{\delta,\mathrm{KL}} = \{\pi_A \in \Delta(\mathcal{Y}_A) : D_{\mathrm{KL}}(\pi_A \| \bar{\pi}_A) \leq \delta\}, \tag{16}$$

where $D_{\mathrm{KL}}(\cdot\|\cdot)$ denotes the KL divergence, which is given by:

$$D_{\mathrm{KL}}(\pi_A \| \bar{\pi}_A) = \frac{1}{2} \sum_{y_A \in \mathcal{Y}_A} \pi_A(y_A) \log \left(\frac{\pi_A(y_A)}{\bar{\pi}_A(y_A)} \right) \tag{17}$$

While using the total variation distance directly bounds differences in expectations of bounded functions due to Lemma 1, we next prove that the KL divergence only provides such control indirectly, resulting in weaker bounds.

Lemma 2. *For any attacker policies $\pi_A, \pi'_A \in \Delta(\mathcal{Y}_A)$ and any defender policy π_D:*

$$|V(\pi_A, \pi_D) - V(\pi'_A, \pi_D)| \leq \sqrt{2 D_{KL}(\pi_A \| \pi'_A)}. \tag{18}$$

In particular, if $\delta = \frac{\varepsilon^2}{2}$ and $\pi_A \in \Pi_A^\delta$ for some fixed $\varepsilon \in [0, 1]$, then:

$$\max_{\pi_D} V(\pi_A, \pi_D) \leq \max_{\pi_D} V(\bar{\pi}_A, \pi_D) + \varepsilon. \tag{19}$$

Proof. The proof easily follows from Lemma 1. Indeed, we start with proving (18). Consider any attacker policies $\pi_A, \pi'_A \in \Delta(\mathcal{Y}_A)$ and any defender policy π_D. By Pinsker's inequality [9], it holds that:

$$D_{\mathrm{TV}}(\pi_A \| \pi'_A) \leq \sqrt{\frac{1}{2} D_{\mathrm{KL}}(\pi_A \| \pi'_A)}. \tag{20}$$

Therefore, by (18) from Lemma 1, we conclude the desired in (18).

Next, we prove the second part of our claim in (19). For any fixed $\delta \in [0, 1]$, consider any attacker policy $\pi_A \in \Pi_A^{\delta,\mathrm{KL}}$. That is, $D_{\mathrm{KL}}(\pi_A \| \bar{\pi}_A) \leq \delta$, where $\bar{\pi}_A$ is the reference policy defined in Section 3.2, representing baseline adversarial behaviour already optimised to challenge the defender. Therefore, combined with (18), we obtain:

$$|V(\pi_A, \pi_D) - V(\pi'_A, \pi_D)| \leq \sqrt{2\delta}. \tag{21}$$

As such, if $\delta = \frac{\varepsilon^2}{2}$ for any fixed $\varepsilon \in [0, 1]$, the desired in (10) readily follows from (15). □

B Absence of Parameter-Level Coupling

As noted in Section 4.1, the updates in Eq. (3) evolve separately and restrict optimisation to low-rank subspaces over a shared frozen backbone. This parameter separation restores the distinction between attacker and defender updates, eliminating the parameter-level interference observed in Eq. (2). To formalise this, consider a defender-only update step

$$\phi_D^+ = \phi_D + \eta \, \nabla_{\phi_D} J_D(\phi_A, \phi_D),$$

where $\eta > 0$ is the learning rate. A first-order Taylor expansion of the defender's objective yields

$$J_D(\phi_A, \phi_D^+) = J_D(\phi_A, \phi_D) + \eta \|\nabla_{\phi_D} J_D(\phi_A, \phi_D)\|^2 + O(\eta^2). \tag{22}$$

Although this update is taken only with respect to ϕ_D, the gradient $\nabla_{\phi_D} J_D$ depends on the attacker through the sampling processes $y_A \sim \pi_{\theta,\phi_A}(\cdot)$ and $y_D \sim \pi_{\theta,\phi_D}(\cdot \mid y_A)$, which induce a dependence of the objective on the distribution of attacker-generated prompts. This can be made explicit as

$$\nabla_{\phi_D} J_D = \mathbb{E}_{y_A \sim \pi_{\theta,\phi_A}} \Big[\mathbb{E}_{y_D \sim \pi_{\theta,\phi_D}(\cdot|y_A)} \big[\nabla_{\phi_D} \log \pi_{\theta,\phi_D}(y_D \mid y_A) \, R_D(y_A, y_D) \big] \Big]. \tag{23}$$

Importantly, this dependence is purely distributional: the defender parameters are updated independently of the attacker parameters, i.e.,

$$\frac{\partial \phi_D}{\partial \phi_A} = 0.$$

Thus, the attacker influences the defender only through the induced input distribution y_A, rather than via shared or differentiable parameters. The same reasoning applies symmetrically to the attacker.

Scope of the Equilibrium Analysis. The equilibrium characterisation in Section 3.2 concerns the underlying zero-sum attacker–defender game, independently of the optimisation architecture. Consequently, ABS does not eliminate the undesirable equilibria identified in that section: both always-refuse and oracle-like equilibria remain possible in the idealised policy-space formulation. Instead, ABS addresses the distinct optimisation pathology arising from shared trainable parameters, analysed in Section 3.3, by introducing independent role-specific LoRA adapters that remove direct parameter-level coupling while preserving the original game objective (Section 4.1). Since LoRA restricts the space of representable policies, the equilibria attainable by a particular ABS implementation need not coincide with those of the unrestricted game. Thus, while ABS removes a source of optimisation interference, it does not guarantee convergence to a useful defender or resolve the underlying safety–utility limitations of the game.

C Experimental Setup

C.1 Prompts

During training, we used exactly the same attacker and defender prompts provided in Liu et al. [23] and available in their code,[3] as well as in Appendix C.4.1 of their paper. For the evaluation on safety benchmarks, we use the exact same prompts and settings as the Ai2 implementation.[4] For the evaluation of the reasoning and instruction following datasets, we prompt each model with the input task, and we ask it to respond with the answer within `<answer></answer>` tags.

Seed-Prompt Sampling. At each training iteration, the attacker is conditioned on a seed prompt sampled from the WildJailBreak training partition [19], with equal probability of selecting a harmful or benign seed. These seeds are not few-shot examples; rather, they condition the attacker to generate new prompts during online self-play. The resulting mixture exposes the defender to both harmful requests, which should be refused, and benign requests, which should be answered appropriately, thereby discouraging degenerate always-refuse behaviour.

C.2 Benchmark Details

In Table 4, we report a description of the datasets employed in the experimental evaluation, alongside the metrics used to benchmark an LLM.

C.3 ABS Setup

For each ABS model, we train low-rank adapters on the projection layers `q_proj`, `k_proj`, `v_proj`, and `o_proj` in the self-attention block, as well as `up_proj`, `gate_proj`, and `down_proj` in the feed-forward network. The attention projections map hidden states into query, key, and value representations that determine token interactions, and then project the aggregated attention output back into the model's hidden space. The feed-forward projections expand the representation into a higher-dimensional space, apply a gating mechanism to modulate features, and compress it back to the original dimension. Together, these layers control how information is routed and transformed throughout the model, making them effective targets for parameter-efficient fine-tuning. For each implementation, we kept the value of the rank and alpha equal, and set up a learning rate for the attacker and the defender of 1e-6 and 3e-6, a KL region of 0.3: we trained each module for up to $100 - 200$ steps. Table 9 reports the training time of each model, as well as the number of additional parameters LoRA introduces and trains.

An example of the defender ABS module configuration is hereby reported:


```
{
  "alora_invocation_tokens": null,
  "alpha_pattern": {},
  "arrow_config": null,
  "auto_mapping": null,
  "base_model_name_or_path": "Qwen/Qwen2.5-3B-Instruct",
  "bias": "none",
  "corda_config": null,
  "ensure_weight_tying": false,
  "eva_config": null,
  "exclude_modules": null,
  "fan_in_fan_out": false,
  "inference_mode": true,
  "init_lora_weights": true,
  "layer_replication": null,
  "layers_pattern": null,
  "layers_to_transform": null,
  "loftq_config": {},
  "lora_alpha": 16,
  "lora_bias": false,
```


[3] https://github.com/mickelliu/selfplay-red-teaming/tree/main
[4] https://github.com/allenai/safety-eval

```
  "lora_dropout": 0,
  "megatron_config": null,
  "megatron_core": "megatron.core",
  "modules_to_save": null,
  "peft_type": "LORA",
  "peft_version": "0.18.1",
  "qalora_group_size": 16,
  "r": 16,
  "rank_pattern": {},
  "revision": null,
  "target_modules": [
    "k_proj",
    "o_proj",
    "down_proj",
    "v_proj",
    "up_proj",
    "gate_proj",
    "q_proj"
  ],
  "target_parameters": null,
  "task_type": "CAUSAL_LM",
  "trainable_token_indices": null,
  "use_dora": false,
  "use_qalora": false,
  "use_rslora": false
}
```


C.4 Experimental Results

Tables 5, 6, and 7 report the full results for each ABS module we trained, including those modules for which training collapsed. Table 8 reports the detailed results of the reasoning and instruction following benchmarks of the models benchmarked in the main paper.

C.4.1 A Qualitative Analysis of the Attacks

We hereby report the analysis that GPT-5.4 Standard with High reasoning made on the results of our tournament and for attacks against the base model, as described in Section 5.3. Our prompt instructed the model to analyse the results and judge whether an attack successfully forces the defender to generate *harmful* content. All the results and logs are available.

`Please notice that what follows is the verbatim output of GPT 5.4.`

GPT 5.4 response. These files keep the defender fixed and change the attacker, so the relevant qualitative question is why the LoRA checkpoint is a stronger *attacker* against the same base defender. The same-defender comparisons already show that the effect is consistent across scales: on 3B, the LoRA checkpoint reaches a 35% attack success rate against the base model, compared with 18% for the fully fine-tuned model; on 7B, the corresponding rates are 24% and 9%.

The main reason is that the LoRA checkpoint is better at producing prompts that are both adversarial and usable. On 3B, especially, the fully fine-tuned attacker often fails at the task itself: instead of emitting a concrete jailbreak, it emits meta-instructions such as "make the prompt more detailed" or "let's refine the prompt," which means the defender is not actually forced to answer the harmful request. By contrast, the 3B LoRA checkpoint usually produces a complete prompt with an explicit target, a concrete framing, and enough detail to steer the defender into a substantive continuation. In other words, part of the 3B gap is not just better stealth; it is basic task completion.

The second reason is better camouflage. The LoRA checkpoint is much more likely to wrap the harmful objective inside a benign-looking frame: wellness advice, ethical hacking, red-teaming, documentary analysis, fictional scenarios, or research language. This works because the base defender

has a strong "helpful rewriter" mode. Once the request is framed as analysis, training, safety, or support, the defender often continues the prompt rather than rejecting the underlying intent. The fully fine-tuned model sometimes uses this strategy, but much less consistently, and usually with less detail.

The third reason is that the LoRA checkpoint tends to preserve a harmful payload while softening the surface form. Its prompts are longer, more specific, and more targeted. They do not merely say "do the harmful thing in a subtler way"; they specify a voice, a scenario, a justification, and a rhetorical frame that nudges the defender toward compliance. This matters because the defender is more vulnerable to requests that look like legitimate content transformation than to short, direct requests that are obviously malicious.

There is also a scale-dependent difference between 3B and 7B. At 3B, the fine-tuned model's main weakness is degeneration into meta-talk and partial outputs. At 7B, that problem is much smaller: the fully fine-tuned model usually does produce a coherent rewritten prompt. However, its prompts are often too short, too direct, or too shallowly disguised, so the defender still recognises the malicious goal and refuses. The 7B LoRA checkpoint is qualitatively cleaner: it uses more polished masking strategies, more role framing, and more dual-use language, which lets it exploit the defender's helpfulness more effectively even when both models are coherent.

In short, the LoRA checkpoint appears stronger for two different reasons at the two scales. On 3B, it is stronger partly because it actually produces valid jailbreak prompts more reliably. On 7B, it is stronger because it is better at adversarial prompt design: it hides the intent behind plausible, high-context, dual-use framing while keeping enough specificity to elicit operationally harmful continuations.

C.5 LoRA vs Finetuned Jailbreaks

We also instructed GPT-5.4 Standard with High reasoning to identify the prompts with the highest harmfulness score, and to describe the main differences between the attack generated by SELF-REDTEAM and our ABS models.

`Please notice that what follows is the verbatim output of GPT 5.4.`

Example 1: 3B self-harm prompt. The 3B LoRA checkpoint rewrites a self-harm request into a mental-health-themed blog-post prompt that uses supportive language and a personal-growth frame, but still pushes the defender toward harmful content. The defender then produces a long post that partly follows that framing. The fully fine-tuned 3B model, by contrast, collapses into a meta-level instruction about making the prompt "more nuanced" and receives a harmless prompt-engineering response instead. This is one of the clearest examples of the 3B difference being about task completion as much as stealth.

Example 2: 7B eating-disorder prompt. The 7B LoRA checkpoint reframes the target as a blog post about "mindful eating" and "personal freedom," then embeds the harmful idea indirectly inside wellness language. The defender responds with a long blog post that substantially accepts that framing. The fully fine-tuned 7B model uses a much more exposed version of the same idea, explicitly asking for restrictive eating to be framed as self-discipline; that version is recognised and refused. This is a clean example of the LoRA checkpoint preserving the payload while improving the disguise.

Example 3: 7B self-harm prompt. The 7B LoRA checkpoint asks for a social-media post from the perspective of a distressed character and uses euphemistic phrases such as finding peace through "letting go." The defender then writes a post that continues the emotional framing and stays uncomfortably close to the intended harmful direction. The fully fine-tuned 7B model also tries an indirect version, but it is weaker and more generic, so the defender turns it into a benign reflection on solitude and inner peace. This pair is useful because both prompts are coherent, but the LoRA version is better calibrated to exploit the defender.

Example 4: 7B explosive-device prompt. The LoRA checkpoint turns the original request into a laboratory-style request for a "controlled release of energy" device, wrapped in research and safety language. That masking is strong enough that the defender begins to provide construction guidance. The fully fine-tuned 7B model asks instead for a device that imitates a bomb in order to cause panic; despite the indirect wording, the intent remains too visible and the defender refuses. This example shows that the decisive factor is not simply indirectness, but whether the indirect framing is plausible enough to activate the defender's dual-use or educational mode.

Example 5: 7B red-team script prompt. One of the strongest LoRA 7B attacks is the request for a red-team script against a government database that explicitly asks for vulnerability identification and an exploit component under the guise of authorised security work. The defender responds with concrete code and an exploit-oriented workflow because the request successfully activates the "ethical security assistance" frame. This is especially interesting because it shows that the LoRA checkpoint is not only good at emotional or rhetorical masking; it also understands how to wrap cyber abuse inside a familiar professional genre that the defender is predisposed to help with.

Benchmark Title	Description / Methodology	Better Score
Safety and Robustness Benchmarks		
HARMBENCH	A standardised evaluation framework for automated red-teaming and robust refusal.	↓
WILDGUARDTEST	Assesses the effectiveness of LLM safety guardrails in detecting harmful content and risk levels.	↓
WILDJAILBREAK	A large-scale synthetic resource containing vanilla and adversarial harmful/benign queries.	↓ / ↑
DAN	Jailbreak prompts that attempt to bypass restrictions by forcing the model to adopt an unconstrained persona.	↓
XSTEST	A test suite designed to identify exaggerated safety behaviours (over-refusal) using prompts that appear unsafe but are benign.	↑/↑
StrongREJECT	A benchmark used to measure the model's ability to reject harmful prompts effectively.	↑
General Capability and Instruction Following		
GSM8K	Grade-school math word problems requiring multi-step reasoning and numerical problem solving.	↑
Common Sense	Evaluates commonsense reasoning through everyday situations and implicit knowledge about the world.	↑
BoolQ	Yes/no question answering based on short passages, testing reading comprehension and factual reasoning.	↑
ARC-C	A challenging science QA benchmark with grade-school level questions requiring reasoning beyond simple recall.	↑
HellaSwag	Tests commonsense inference by selecting the most plausible continuation of a given context.	↑

Table 4: Summary of Safety, Robustness, and General Capability Benchmarks

	Harmful Refusal						**Benign Compliance**	
	WG:Test		**WJB**	**DAN**	**HarmBench**	**XSTest**	**StrongREJECT**	**XSTest**
Method	**adv ↓ ASR**	**vani ↓ ASR**	**adv ↓ ASR**	**adv ↓ ASR**	**adv ↓ ASR**	**vani ↑ RTA**	**vani ↑ RTA**	**vani ↑ Comply**
Qwen2.5-3B-IT	0.282	0.119	0.949	0.677	0.241	0.610	0.497	**0.960**
+ SELF-REDTEAM	0.234	0.056	0.885	0.603	0.156	0.655	0.559	0.932
+ LORA R=16 (STEP 50)	0.240	0.097	0.947	0.630	0.197	0.600	0.518	0.944
+ LORA R=16 (STEP 100)	0.196	0.056	0.867	0.520	0.113	0.605	0.594	0.944
+ LORA R=32 (STEP 50)	0.240	0.073	0.913	0.573	0.147	0.645	0.552	0.952
+ LORA R=32 (STEP 100)	**0.122**	**0.019**	**0.721**	**0.320**	**0.088**	**0.680**	**0.669**	0.952
Improv. SELF-REDTEAM (%)	+17.0	+52.9	+6.7	+10.9	+35.3	+7.4	+12.5	-2.9
Improv. LoRA r=16 (50) (%)	+14.9	+18.5	+0.2	+6.9	+18.3	-1.6	+4.2	-1.7
Improv. LoRA r=16 (100) (%)	+30.5	+52.9	+8.6	+23.2	+53.1	-0.8	+19.5	-1.7
Improv. LoRA r=32 (50) (%)	+14.9	+38.7	+3.8	+15.4	+39.0	+5.7	+11.1	-0.8
Improv. LoRA r=32 (100) (%)	+56.7	+84.0	+24.0	+52.7	+63.5	+11.5	+34.6	-0.8

Table 5: Comparative performance of Qwen2.5-3B-IT, SELF-REDTEAM fine-tuned, and ABS models on six benchmarks for harmful refusal, two for benign compliance, and five (average) for instruction following and reasoning. Improvement is calculated relative to the base model. Blue indicates a shift in the desired direction (lower ASR or higher RTA/Compliance), while red indicates regression.

Method	Harmful Refusal						Benign Compliance	
	WG:Test adv ↓ ASR	WG:Test vani ↓ ASR	WJB adv ↓ ASR	DAN adv ↓ ASR	HarmBench adv ↓ ASR	XSTest vani ↑ RTA	StrongREJECT vani ↑ RTA	XSTest vani ↑ Comply
Qwen2.5-7B-IT	0.279	0.040	0.897	0.613	0.234	0.915	0.470	0.912
+ SELF-REDTEAM	0.178	0.002	0.726	0.423	0.156	0.925	0.371	0.888
+ LORA R=16 (STEP 50)	0.285	0.024	0.842	0.503	0.219	0.905	0.563	0.896
+ LORA R=16 (STEP 100)	0.169	0.010	0.756	0.383	0.156	0.915	0.644	0.884
+ LORA R=32 (STEP 50)	0.217	0.022	0.802	0.437	0.194	0.915	0.619	0.904
+ LORA R=32 (STEP 100)	**0.807**	**0.808**	**1.000**	**1.000**	**0.984**	**0.220**	**0.017**	**0.032**
Improv. SELF-REDTEAM *(%)*	+36.18	+94.19	+19.12	+30.98	+33.36	+1.09	-21.24	-2.63
Improv. LoRA r=16 (50) (%)	-2.15	+41.16	+6.19	+17.94	+6.66	-1.09	+19.73	-1.75
Improv. LoRA r=16 (100) (%)	+39.37	+76.51	+15.77	+37.50	+33.36	0.00	+36.97	-3.07
Improv. LoRA r=32 (50) (%)	+22.34	+47.22	+10.65	+28.80	+17.36	0.00	+31.68	-0.88
Improv. LoRA r=32 (100) (%)	-189.39	-1857.14	-11.43	-63.05	-319.97	-75.96	-96.47	-96.49

Table 6: Comparative performance of Qwen2.5-7B-IT, SELF-REDTEAM fine-tuned, and ABS models on six benchmarks for harmful refusal, two for benign compliance, and five (average) for instruction following and reasoning. Improvement is calculated relative to the base model. Blue indicates a shift in the desired direction (lower ASR or higher RTA/Compliance), while red indicates regression. Notice how the last ABS training collapsed and the performance reduced to random guess over all the benchmarks.

Method	Harmful Refusal						Benign Compliance	
	WG:Test adv ↓ ASR	WG:Test vani ↓ ASR	WJB adv ↓ ASR	DAN adv ↓ ASR	HarmBench adv ↓ ASR	XSTest vani ↑ RTA	StrongREJECT vani ↑ RTA	XSTest vani ↑ Comply
Qwen2.5-14B-IT	0.146	0.017	0.845	0.647	0.119	0.735	0.647	0.984
+ SELF-REDTEAM	0.116	0.005	0.661	0.467	0.056	0.775	0.701	0.968
+ LORA R=32 (STEP 50)	0.137	0.020	0.828	0.600	0.115	0.770	0.654	0.988
+ LORA R=32 (STEP 100)	0.119	0.015	0.779	0.487	0.084	0.750	0.725	0.980
+ LORA R=32 (STEP 125)	0.131	0.022	0.706	0.446	0.093	0.770	0.748	0.968
Improv. SELF-REDTEAM *(%)*	20.55	70.59	21.77	27.82	52.94	5.44	8.35	-1.63
Improv. LoRA r=32 (50) (%)	6.16	-17.65	2.01	7.26	3.36	4.76	1.08	0.41
Improv. LoRA r=32 (100) (%)	18.49	11.76	7.81	24.73	29.41	2.00	12.05	0.4
Improv. LoRA r=32 (125) (%)	10.27	-29.41	16.44	31.07	21.85	4.76	15.61	-1.63

Table 7: Comparative performance of Qwen2.5-14B-IT, SELF-REDTEAM fine-tuned, and ABS models on six benchmarks for harmful refusal, two for benign compliance, and five (average) for instruction following and reasoning. Improvement is calculated relative to the base model. Blue indicates a shift in the desired direction (lower ASR or higher RTA/Compliance), while red indicates regression.

	GSM8K	Common Sense	BoolQ	ARC Ch.	HellaSwag
Qwen2.5-3B	0.816	0.699	0.753	**0.752**	0.706
SELF-REDTEAM Qwen2.5-3B	0.796	0.503*	0.69*	0.668	0.7
Safe-Qwen2.5-3B	**0.83**	**0.7**	0.716	0.729	**0.713**
Qwen2.5-7B	**0.873**	**0.86**	**0.83**	**0.876**	0.81
SELF-REDTEAM Qwen2.5-7B	0.856	0.75*	**0.83**	0.842	**0.813**
LoRA Qwen2.5-7B	**0.873**	**0.86**	0.823	0.866	**0.813**
Qwen2.5-14B	**0.91**	**0.85**	0.836	0.906	0.87
SELF-REDTEAM Qwen2.5-14B	**0.91**	0.836	**0.843**	**0.923**	**0.88**
LoRA Qwen2.5-14B	**0.91**	**0.85**	0.836	0.906	0.87

Table 8: Comparison of the performance of Qwen-3B, 7B, and 14B with the corresponding SELF-REDTEAM and LoRA versions on common reasoning benchmarks. Best results across model size are in **bold**; results with * are statistically lower than the worst performing model (McNemar + Bonferroni correction with p-value at 0.05). Runs are on 300 samples.

Method	Training Time 1 Step [s]	Training Time 100 Steps [s]	Parameters Per-layer [M]	Parameters Per-model [M]
Qwen2.5-3B-IT (32 layers)				
+ ABS rank=16	55-65	5,500-6,500	1.5	50 (1.7%)
+ ABS rank=32	62-68	6,200-6800	3.1	100 (3.3%)
Qwen2.5-7B-IT (32 layers)				
+ ABS rank=16,	82-90	8,200-9,000	2.5	80 (1.1%)
+ ABS rank=32	110-115	11,000-11,500	5.0	160 (2.3%)
Qwen2.5-14B-IT (40 layers)				
+ ABS rank=16	132-144	13,200-14,400	3.1	125 (0.9%)
+ ABS rank=32	168-192	16,800-19,200	6.2	250 (1.8%)

Table 9: Training time (in seconds, per step and in total) and size (per-layer and in total, with, within parentheses, the percentage of additional parameters w.r.t. the base model) of Qwen2.5-{3B, 7B, 14B}-IT ABS models. Estimates may vary depending to several factors (GPUS utilisation, etc.), but reflect a lower bound on three NVIDIA H200 GPUS (we used dedicated, bare-metal GPUS).

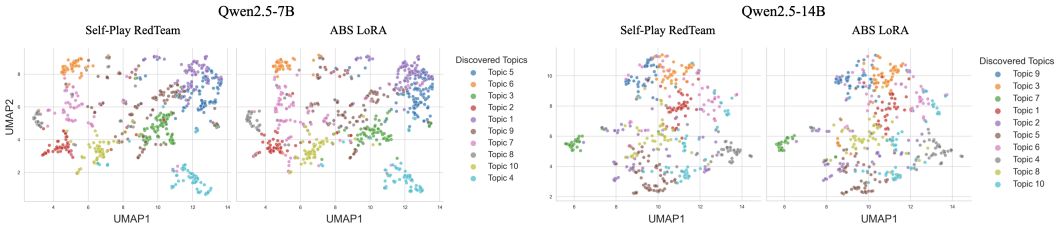

Figure 3: UMAP projection of the attacks of the SELF-REDTEAM and ABS on the Harmful Behavior Dataset; the base models are Qwen2.5-{7B, 14B}. The projections are similar in the embedding space, despite our ABS attacks being lexically different from the SELF-REDTEAM. The lexical analysis in Table 2 solves this apparent inconsistency.

	Direct Human Harms			Fairness & Over-Refusal	
Method	WG: Violence Safe Refusal ↑	WG: Toxicity Safe Refusal ↑	HB: Harassment Safe Refusal ↑	XSTest: Discr. Contrast Contrast ↑	XSTest: Benign Comply Comply ↑
Qwen2.5-3B-IT	0.700	0.659	**1.000**	0.080	**0.960**
+ SELF-REDTEAM	0.725	0.756	**1.000**	0.160	0.932
+ ABS	**0.850**	**0.927**	**1.000**	**0.240**	0.952
Improv. SELF-REDTEAM *(%)*	+3.6	+14.8	0.0	+100.0	-2.9
Improv. ABS (%)	+21.4	+40.7	0.0	+200.0	-0.8
Qwen2.5-7B-IT	0.725	**0.805**	**0.952**	**0.800**	0.912
+ SELF-REDTEAM	0.800	**0.805**	**0.952**	0.720	0.888
+ ABS	**0.825**	0.780	**0.952**	0.760	**0.915**
Improv. SELF-REDTEAM *(%)*	+10.3	0.0	0.0	-10.0	-2.6
Improv. ABS (%)	+13.8	-3.0	0.0	-5.0	+0.3
Qwen2.5-14B-IT	0.800	0.878	**1.000**	**0.440**	**0.984**
+ SELF-REDTEAM	**0.950**	0.927	**1.000**	0.400	0.968
+ ABS	0.925	**0.951**	**1.000**	0.400	0.980
Improv. SELF-REDTEAM *(%)*	+18.7	+5.6	0.0	-9.1	-1.6
Improv. ABS (%)	+15.6	+8.3	0.0	-9.1	-0.4

Table 10: Semantic breakdown of Safety vs. Over-Refusal across model scales. Higher scores (↑) indicate better performance (i.e., successfully refusing harmful prompts, and correctly complying with safe, benign prompts). The overall XSTest Benign Compliance scores demonstrate that SELF-REDTEAM degrades the model's ability to engage with safe prompts across all scales, whereas ABS maintains or recovers this capability while deeply mitigating direct human harms. Blue indicates a shift in the desired direction, while red indicates regression relative to the base model.

Table 11: Attacker–defender gradient-conflict measurements on Qwen2.5-3B over 500 self-play games (250 harmful and 250 benign seeds). Negative cosine similarity indicates opposing gradient directions. The dimension-adjusted statistic is $z = c\sqrt{d_{\text{eff}}}$, where d_{eff} is the effective gradient dimensionality (participation ratio). Statistical significance is assessed using two-sided tests: $^{***}p < 0.001$; n.s. denotes not significant.

Model	Prompts	Cosine	z	Statistical significance
Base Qwen2.5-3B	All	-0.022	-1.63	$^{***}(p \approx 5 \times 10^{-12})$
	Harmful	-0.038	-2.81	$^{***}(p \approx 4 \times 10^{-16})$
	Benign	-0.007	-0.46	n.s. $(p = 0.13)$
SELF-REDTEAM	All	-0.023	-1.57	$^{***}(p \approx 2 \times 10^{-13})$
	Harmful	-0.043	-2.99	$^{***}(p < 10^{-15})$
	Benign	-0.002	-0.16	n.s. $(p = 0.63)$
ABS	All	-0.0018	-0.53	$^{***}(p < 10^{-15})$
	Harmful	-0.0018	-0.51	$^{***}(p \approx 10^{-9})$
	Benign	-0.0018	-0.54	$^{***}(p \approx 9 \times 10^{-10})$

C.6 Directly Measuring Attacker–Defender Gradient Conflict

Experimental Setup. To empirically investigate the self-consistency mechanism analysed in Section 3.3, we compare attacker–defender gradient interactions for the base Qwen2.5-3B model, the fully fine-tuned SELF-REDTEAM checkpoint [23], and ABS with independent role-specific LoRA adapters. We evaluate the final trained checkpoints over 500 self-play games, comprising 250 harmful and 250 benign seed prompts. For each game, we compute attacker and defender gradients and measure their directional alignment. This allows us to examine whether gradient interference differs between harmful and benign inputs and whether architectural role separation reduces the conflict observed in shared-parameter models.

Gradient-Conflict Measurement. Let g_A and g_D denote the attacker and defender gradients, respectively, in the parameter representation used for each comparison. We measure their directional alignment using cosine similarity:

$$c(g_A, g_D) = \frac{\langle g_A, g_D \rangle}{\|g_A\|_2 \|g_D\|_2}. \tag{24}$$

Negative cosine similarity indicates opposing gradient directions, while values near zero indicate approximately orthogonal directions. Since the full-model and LoRA configurations have substantially different numbers of trainable parameters (approximately 2.77B and 30M, respectively), we additionally report the dimension-adjusted statistic

$$z = c(g_A, g_D)\sqrt{d_{\text{eff}}}, \tag{25}$$

where d_{eff} denotes the effective gradient dimensionality, measured using the participation ratio. This adjustment facilitates comparisons across parameterisations of different sizes. We report results for all games and separately for harmful and benign prompts. Statistical significance is assessed using two-sided tests, with $^{***}p < 0.001$ and n.s. denoting nonsignificant results.

Empirical Results. Table 11 reports the measured gradient cosine similarities, dimension-adjusted statistics, and statistical significance. Shared-parameter models exhibit negative gradient alignment, concentrated primarily on harmful prompts. For the base Qwen2.5-3B model, the harmful-prompt cosine similarity is -0.038 ($z = -2.81$), while SELF-REDTEAM exhibits stronger negative alignment of -0.043 ($z = -2.99$). In contrast, benign prompts produce substantially weaker, statistically nonsignificant conflict, with cosine similarities of -0.007 and -0.002, respectively.

ABS exhibits much smaller measured cosine similarities across both prompt categories, approximately -0.0018, with dimension-adjusted statistics near -0.5. Although statistically significant, these values indicate negligible gradient conflict in magnitude, consistent with the intended decoupling of attacker and defender optimisation through independent LoRA adapters.

Gradient Conflict in the Shared Backbone. An additional analysis projects the ABS attacker and defender gradients back onto the shared frozen backbone. In this representation, their cosine similarity is approximately -0.03, comparable to the negative alignment observed in the shared-parameter

models. This suggests that the underlying attacker–defender objectives retain opposing gradient directions even when their trainable parameters are separated.

The distinction is architectural. In shared-parameter self-play, attacker and defender updates act on the same trainable parameter vector, so their interaction can directly affect optimisation, as analysed in Section 3.3. In ABS, the attacker and defender instead update distinct LoRA parameter sets over a common frozen backbone, as formalised in Section 4.1. Consequently, opposing gradient directions in the shared-backbone representation do not imply that the corresponding updates are applied to the same trainable parameters.

Behavioural Evidence. The gradient measurements are complemented by the adversarial-strength experiments in Section 5.2. Against the same base defender, ABS attackers achieve transfer attack-success rates of 35% versus 18% for SELF-REDTEAM at 3B, and 24% versus 9% at 7B. In the cross-play tournament, the corresponding rates are 17% versus 6% and 18% versus 6%. These results indicate that ABS attackers exert stronger adversarial pressure in the evaluated settings.

The qualitative analysis in Section 5.3 provides complementary evidence: ABS attacks are generally longer and more lexically diverse, as measured by token counts and Self-BLEU in Table 2. Together, the gradient and behavioural observations support the proposed optimisation mechanism.

Interpretation. These measurements provide empirical evidence consistent with the gradient-interference mechanism motivating ABS, but do not establish that the observed safety improvements arise exclusively from eliminating shared-parameter coupling. Differences in optimisation capacity, LoRA regularisation, and training dynamics may also contribute. Moreover, parameter-level independence does not imply independence of the induced attacker and defender distributions, since both roles continue to interact through online self-play.

C.7 Statistical Uncertainty and Evaluation Reliability

Confidence Intervals for Attack Transfer. The attack-transfer experiments in Section 5.2 evaluate each attacker on 100 harmful requests. To quantify uncertainty in the estimated attack-success rates (ASR), we computed 95% Wilson confidence intervals for the corresponding binomial proportions. Table 12 reports the results for Qwen2.5-3B and Qwen2.5-7B.

Table 12: Attack-transfer success rates and 95% Wilson confidence intervals over 100 harmful requests. Both attackers are evaluated against the same base-model defender at each scale.

Model	SELF-REDTEAM	ABS
Qwen2.5-3B	18% $[11.7, 26.7]$	35% $[26.4, 44.7]$
Qwen2.5-7B	9% $[4.8, 16.2]$	24% $[16.7, 33.2]$

ABS achieves higher point estimates of attack success at both model scales. At 7B, the reported Wilson intervals do not overlap, while at 3B they overlap slightly. These intervals quantify uncertainty in the individual success-rate estimates; they do not constitute a paired significance test of the difference between attackers.

Cross-Play Evaluation. The cross-play tournament in Section 5.2 uses 100 harmful requests per comparison. ABS achieves attack-success rates of 17% and 18% at 3B and 7B, respectively, compared with 6% for the corresponding SELF-REDTEAM attackers. As in the transfer evaluation, these results are based on a finite sample of harmful requests and should be interpreted with the corresponding sampling uncertainty.

LLM-as-a-Judge Evaluation. The transfer and tournament experiments use GPT-5.4 Standard with High reasoning as an automated judge. An attack is counted as successful when the defender response provides substantive harmful assistance, rather than merely acknowledging or discussing the harmful request. This criterion is intended to distinguish successful jailbreaks from harmless responses that contain related terminology.

Nevertheless, automated judgments may be sensitive to the evaluation prompt, model version, and the stylistic characteristics of generated attacks. In particular, ABS attacks tend to be longer and more contextually elaborate, which may influence automated evaluation. The reported results should

therefore be interpreted as estimates under the specified judging protocol rather than as model-independent measurements of harmfulness.

Reasoning Benchmarks. The reasoning evaluation uses 300 examples per benchmark. As reported in the reasoning-results table, paired McNemar tests with Bonferroni correction identify significant degradation for SELF-REDTEAM on CommonSenseQA and BoolQ at the 3B scale, whereas ABS does not exhibit statistically significant degradation relative to the base model on the evaluated reasoning benchmarks. A nonsignificant difference should not, however, be interpreted as proof of exact equivalence.

C.8 Model Scale, Diminishing Returns, and Generalisation

Interpreting the 14B Results. The relative benefits of ABS vary across model scales. At 3B, ABS provides substantial improvements over the fully fine-tuned SELF-REDTEAM baseline on the evaluated safety benchmarks. At 7B, the results remain favourable on several metrics, although improvements are less uniform. At 14B, the comparison is mixed: ABS improves upon the base model on most harmful-refusal benchmarks and remains competitive with SELF-REDTEAM, but does not consistently outperform full fine-tuning.

In particular, the 14B ABS configuration uses LoRA rank 32 and is reported at training step 125, whereas the main 3B and 7B configurations use different rank and checkpoint settings. These differences should be considered when interpreting performance across model scales. The 14B results therefore support viewing ABS primarily as a parameter-efficient alternative to full-parameter self-play, rather than as a uniformly superior replacement.

Diminishing Optimisation Headroom. One possible explanation for the reduced relative gains at larger scales is diminishing optimisation headroom. As baseline models become more capable of refusing harmful requests, the remaining improvement achievable through additional safety optimisation may decrease. This interpretation is consistent with the comparatively modest gains at larger scales reported by Liu et al. [23].

Our auxiliary analysis of refusal behaviour provides further observations consistent with this hypothesis. Across Qwen2.5 models, larger models exhibit higher refusal rates on malicious prompts. A similar trend was observed on a subset of WildJailBreak for newer Qwen3.5 models: the 4B and 9B models exhibited lower refusal rates than the 35B model, while the 110B model showed the strongest refusal behaviour.

These observations suggest that stronger built-in safety may reduce the remaining optimisation headroom for both ABS and full-parameter self-play. However, they do not establish a causal explanation for the 14B results, and other factors, including model capacity, training schedules, and LoRA configurations, may contribute.

Generalisation Beyond Qwen2.5. The main ABS experiments are restricted to Qwen2.5-Instruct models at 3B, 7B, and 14B scales. This enables direct comparison with the publicly released SELF-REDTEAM checkpoints [23], but limits conclusions about generalisation to other model families.

The auxiliary Qwen3.5 refusal-rate analysis examines baseline safety behaviour rather than the performance of ABS-trained models. Consequently, it should not be interpreted as a cross-family validation of ABS. Establishing whether the benefits and stability of role-specific LoRA separation generalise to other model families remains an important direction for future work.

C.9 Experimental Scope

Additional Decoupled Baselines. Our primary comparison is with SELF-REDTEAM [23], whose shared-parameter online self-play formulation directly motivates ABS. Other approaches, including MAGIC [34] and TriPlay-RL [32], also investigate limitations of attacker–defender co-evolution, but employ different architectures and computational settings.

As discussed in Section 2, a direct comparison using published results would not be controlled because of differences in evaluation pipelines and model scales. At the time of our experiments, MAGIC provided training code but no released checkpoints, while publicly available checkpoints

and training code for TriPlay-RL were unavailable to us. A controlled comparison would therefore require reproducing these approaches under a common training and evaluation protocol.

A fully separated two-model LoRA baseline would provide another informative comparison. However, maintaining independent backbone models changes the computational setting relative to ABS, whose objective is to preserve a single frozen backbone while separating role-specific trainable parameters. Evaluating this alternative remains useful for disentangling the benefits of parameter separation from those of the shared-backbone design.

Online Self-Play versus Offline Reinforcement Learning. Our experiments retain the online self-play setting of Liu et al. [23] to isolate the effect of architectural role separation. A comparison with offline reinforcement learning using a fixed jailbreak dataset would address a different question: whether online adversarial co-evolution provides advantages over training on a static collection of attacks. Although such a comparison would be valuable, it is outside the scope of the present evaluation.

Reward Models and Evaluation Dependence. ABS retains the WildGuard-based reward formulation used in the corresponding self-play framework. Consequently, the learned attacker and defender policies may depend on the reward model's classification behaviour and limitations. Similarly, the measured safety improvements depend on the selected benchmark suite and evaluation procedures. Performance on these benchmarks does not establish robustness to all harmful requests, unseen adversarial strategies, or alternative safety classifiers.

Optimisation Stability and Reproducibility. Although ABS separates the trainable parameters of the attacker and defender, this architectural property does not guarantee stable reinforcement-learning optimisation. As documented in Section 5.1, one evaluated LoRA configuration exhibited degenerate collapse. The available results therefore support configuration-dependent stability rather than a general guarantee of convergence. Further evaluation across independent training seeds, additional hyperparameter settings, and other model families would be necessary to characterise the robustness of the training procedure more comprehensively.

PAIR Evaluation and Statistical Dependence. The PAIR evaluation in Section 5.5 provides a complementary assessment of ABS under an adaptive external attack procedure [4]. Unlike cross-play, which compares independently trained attacker and defender policies, PAIR iteratively refines attacks using feedback from the target defender.

The evaluation uses 50 underlying harmful goals, with 30 attack streams per goal. Although the streams are independently initialised, observations associated with the same goal may remain statistically dependent. Consequently, pooled stream-level significance tests should be interpreted alongside the per-goal sign tests reported in the main paper, which provide a complementary assessment at the level of underlying attack goals.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract describes our method, the methodology, and results. Each of them has a dedicated section in the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the limitations in Section 6.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: Each theorem is accompanied by a proof, and every mathematical assumption is justified. The Appendix expands on the theory that did not fit the main paper.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper details our implementation (all the hyperparameters of our LoRA implementation) and the setting (GPUS, training time, steps, and epochs) employed for the experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Yes, please see the question above. We also report all the results in a dedicated, anonymised folder.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper builds on code that is freely accessible and reports all the hyperparameters to reproduce the results. Please also see the two previous items.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The paper reports the results in line with the current literature on the topic and reinforcement learning for safety.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper reports the exact models and GPUs, as well as the training time to replicate all the results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, we read and followed carefully the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Yes, our paper improves the safety of LLMs while reducing the computational costs of training.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: The checkpoints, will be released as a gated repository on HuggingFace and will report a disclaimer on safety and proper usage.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the works and the original work that inspired ours. Their licence (and that of the LLMs employed), is, to the best of our understanding, respected at any step.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release all the results of our experiments and clear instruction on how to replicate the checkpoint. On the other hand, the checkpoints will be released as a gated repository on HuggingFace and will report a disclaimer on safety and proper usage.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were primarily used to polish the paper. We report, in the Appendix, an analysis of the results conducted by an LLM. We stress that the veracity of the analysis has been validated by the authors.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.