SEAS: Self-Evolving Adversarial Safety Optimization for Large Language Models

Muxi Diao[1], Rumei Li[2], Shiyang Liu[2], Guogang Liao[2], Jingang Wang[2], Xunliang Cai[2], Weiran Xu[1]*

[1]Beijing University of Posts and Telecommunications, Beijing, China
[2]Meituan, Beijing, China
{dmx,xuweiran}@bupt.edu.cn

Abstract

As Large Language Models (LLMs) continue to advance in capability and influence, ensuring their security and preventing harmful outputs has become crucial. A promising approach to address these concerns involves training models to automatically generate adversarial prompts for red teaming. However, the evolving subtlety of vulnerabilities in LLMs challenges the effectiveness of current adversarial methods, which struggle to generate diverse, complex prompts and dynamically explore the weaknesses of these models. To tackle these challenges, we introduce the **S**elf-**E**volving **A**dversarial **S**afety (SEAS) optimization framework, which includes both a SEAS dataset and a SEAS pipeline. The SEAS dataset comprises complex adversarial prompts, while the SEAS pipeline operates through three stages: Initialization, Attack, and Adversarial Optimization. This framework generates a diverse range of adversarial prompts and dynamically explores the model's vulnerabilities to enhance its security. Our contributions include a novel adversarial framework, a comprehensive safety dataset, and empirical evidence demonstrating the effectiveness of SEAS.

Project Homepage — https://SEAS-LLM.github.io/

1 Introduction

Recently, Large Language Models (LLMs) have demonstrated impressive capabilities in various fields (OpenAI et al. 2023). Meanwhile, security risks associated with deploying LLMs in practical applications have raised widespread public concern (Christian 2023; Chilton 2023).

One potentially useful approach to enhance safety is red teaming (Perez et al. 2022; OpenAI et al. 2023; Meta 2024; Anthropic 2024). This is a proactive risk identification approach that employs manual or automated techniques to generate attack data, which is then used to critically examine a language model for harmful outputs. The model is subsequently updated to prevent these issues. Due to the high costs and time consumption associated with manual red teaming (Ganguli et al. 2022; Anthropic 2024), a promising alternative is to automate the generation of adversarial prompts using a Red Team LLM (Perez et al. 2022).

*Corresponding author

However, as the performance of LLMs continues to improve, their security and robustness are also enhanced, which may be accompanied by changes in security vulnerabilities (Ge et al. 2023), resulting in more subtle and covert failure modes (Ganguli et al. 2022; Ge et al. 2023). These changes pose significant challenges to existing red team methods. Additionally, current methods of generating adversarial data for red teaming lack diversity (Hong et al. 2024), as they are confined to using a single predefined attack strategy (Zhou et al. 2023b; Fernando et al. 2023). This limitation hampers the broad exploration of adversarial prompts, and with the advancement of model capabilities, fixed attack strategies are likely to become ineffective. Furthermore, existing methods are limited in complexity and fail to consider the potential flaws of the Target models themselves, thus unable to implement targeted attacks, resulting in lower attack success rates (Ge et al. 2023). Therefore, the development and testing of current LLMs still heavily rely on manual red teaming (OpenAI et al. 2023; Anthropic 2024).

To address these issues, we propose a **S**elf-**E**volving **A**dversarial **S**afety (**SEAS**) optimization framework, which includes a SEAS dataset and a SEAS pipeline. To generate more complex adversarial prompts, we have constructed the SEAS dataset. This dataset not only encompasses various types of risk categories but also includes adversarial attack styles. In the SEAS pipeline, through the initialization using the SEAS dataset and subsequent iterative updates, the Red Team model generates more diverse adversarial attack prompts and dynamically explores the vulnerabilities of the Target model, which in turn updates itself with the generated data to enhance its security performance.

As shown in Figure 1, our pipeline comprises three stages. **In the Initialization Stage**, the Red Team model (R_0, where "R" stands for Red Team model and "0" denotes the initial iteration) undergoes fine-tuning using the SEAS dataset to produce adversarial prompts. Concurrently, the Target model (T_0, where "T" stands for Target model and "0" denotes the initial iteration) is fine-tuned using open-source data to enhance its instruction-following capabilities. **In the Attack Stage**, the Red Team model generates adversarial prompts, which are inputted to the Target model to elicit responses. The Safe Classifier then evaluates the safety of these responses based on the concatenated prompts and responses. **In the Adversarial Optimization Stage**, adversar-

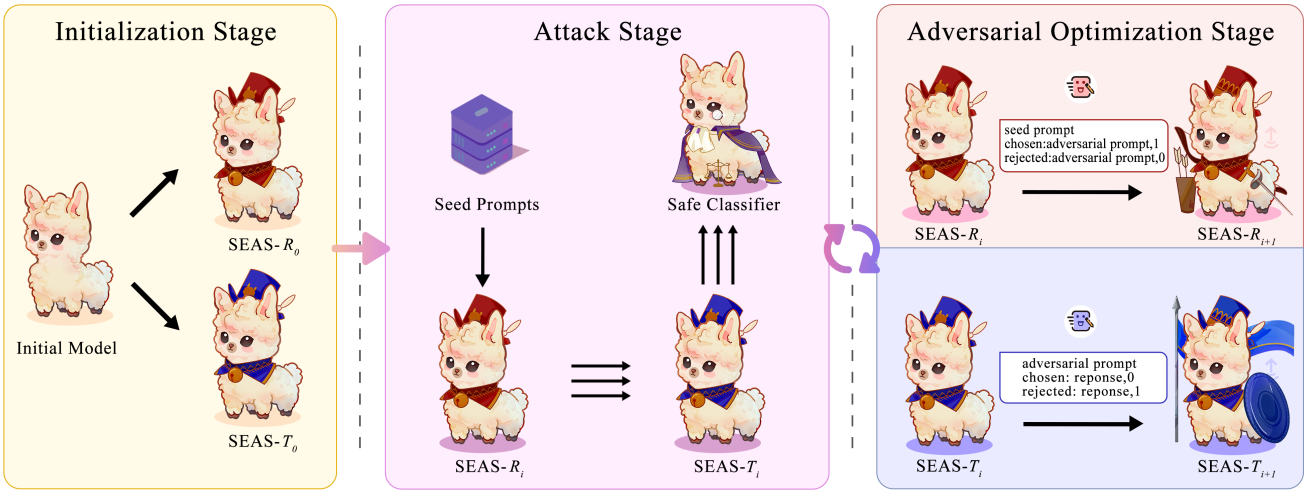

Figure 1: SEAS pipeline. **Initialization Stage**: Red Team model R_0 and Target model T_0 are fine-tuned using different dataset. **Attack Stage**: in the $(i+1)$th iteration, adversarial prompts are generated by activating R_i using seed prompts to attack T_i, the responses are then evaluated by Safe Classifier, where label = 1 represents an unsafe response. **Adversarial Optimization Stage**: the optimization employs pair-wise loss for two models, selecting appropriate data based on the evaluation.

ial prompts that successfully attack the Target model are selected as chosen examples, while ineffective ones are categorized as rejected examples. These are then used to update the Red Team model using pair-wise optimization loss. Similarly, safe responses are taken as chosen examples and unsafe ones as rejected examples to update the Target model. This cycle of Attack and Adversarial Optimization stages is repeated over multiple rounds, allowing both models to continually evolve through adversarial competition.

Our contributions can be summarized as follows:

1) We propose a self-evolving adversarial framework (SEAS) in which two types of models continuously evolve through adversarial competition. The Red Team model comprehensively and dynamically explores the vulnerabilities of the Target model to enhance its security capabilities.

2) We release a safety dataset that includes various harmful, adversarial and ambiguous harmless prompts, providing tools for the secure development and deployment of LLMs. We also open-source the training code, facilitating researchers to replicate and validate our findings.

3) Through comprehensive empirical experiments, we have demonstrated the effectiveness of SEAS. After three iterations, the Target model reaches a security level close to that of GPT-4 while maintaining its general ability. Moreover, the Red Team model shows a 50.66% increase in Attack Success Rate (ASR) against Llama3-70B-Instruct. We also evaluate the diversity of generated adversarial prompts and the effectiveness of iterative model updates.

2 Related Work

2.1 Red Teaming LLMs

Large-scale red teaming has been conducted during the pre-deployment phase of LLMs. Researchers rely on human annotators to handwrite (annotate) adversarial prompts to guide the LLM in generating harmful responses (Meta 2024; Anthropic 2024; OpenAI et al. 2023). While manually constructing adversarial prompts has been proven to generate high-quality prompts (Yu et al. 2023; Ji et al. 2023a), it comes with high costs and consumes a significant amount of time (Ganguli et al. 2022; Anthropic 2024). Additionally, human annotators still face limitations in fully exploring model vulnerabilities (Hong et al. 2024; Mu et al. 2024). Previous work has investigated automated red teaming processes by utilizing LLMs to augment open-source datasets (Ganguli et al. 2022; Ji et al. 2023b; Samvelyan et al. 2024), employing Supervised Learning (SL) to train Red Team models (Perez et al. 2022; Ge et al. 2023) , and using prompt engineering methods (Chao et al. 2023; Liu et al. 2024; Li et al. 2024) to create adversarial prompts. These prompts are used to attack Target models to elicit risky responses.

However, these methods fail to adapt to risk vulnerability changes caused by updates to the Target model. Similar to our work, MART (Ge et al. 2023) also employs a continuously updated iterative method, where both the Red Team model and the Target model are updated simultaneously to enhance the model's security performance. MART framework only screens data that the Target model itself replies to securely and uses it to train the Target model. In cases where the model still outputs risky content after multiple iterations, our SEAS method focuses on automating the exploration and selection of preferred data pairs. For the same adversarial prompt, it selects two responses from the Target model—one secure and one risky—as the optimization target. This ensures that when the model encounters the same prompt again, it will produce a safe response.

2.2 Datasets for LLM Safety

Most LLMs exhibit high success rates in publicly available security assessment benchmarks (Mazeika et al. 2024;

Tedeschi et al. 2024). Despite this, current research often focuses only on specific aspects or dimensions of security, such as toxicity (Hartvigsen et al. 2022), there has been no extensive or in-depth exploration of model security vulnerabilities. A comprehensive evaluation across all categories could more effectively reveal potential vulnerabilities in LLMs (Röttger et al. 2024b). Additionally, recent studies have shown that the risk of LLMs generating harmful content remains high when exposed to adversarially designed prompts (Liu et al. 2023; Chowdhury et al. 2024), indicating that existing benchmarks may not fully capture the security risks associated with LLMs.

To delve deeper into the vulnerabilities of Target models from an attacker's perspective, we develop a new dataset that includes 5 different Risk Categories and 9 distinct Attack Styles (Samvelyan et al. 2024). This dataset could guide the model to produce outputs that contradict its design principles, thereby enabling a comprehensive assessment and future enhancement of model security.

3 Method

In this section, we provide a detailed introduction of SEAS framework, which includes a dataset and a pipeline. By constructing the SEAS dataset and the entire pipeline, we achieve a comprehensive exploration of the Target model's vulnerabilities and enhance its security performance.

3.1 SEAS Dataset

Current LLMs generally perform well on open-source safety datasets, but they still suffer from producing harmful content when presented with deliberately crafted prompts (Röttger et al. 2024b; Hong et al. 2024). Most open-source datasets do not effectively integrate the two critical dimensions of **Risk Categories** (Inan et al. 2023) and **Attack Styles** (Samvelyan et al. 2024).

To address this gap, we have developed a SEAS dataset, which features 14 types that cover two risk dimensions: the first is **Risk Categories**, which are potential topics of unsafe content that LLMs might generate; the second is **Attack Styles**, which are specific expressive techniques designed to cover various prompt types that could trigger harmful behavior in models. This dataset contains 18K entries, divided into a training set with 16K entries and an In-Domain test set with 2K entries. Specifically, to prevent models from overfitting to the aggressive language style or syntax during training, rather than discerning the actual semantic content, we have created an ambiguous harmless dataset for evaluation purposes. This dataset, comprising 300 entries from human experts, mimics the aggressive instruction styles and syntax but is actually harmless. In Figure 2, we present two risk categories and two attack styles, as well as one harmless sample. SEAS dataset were collected through crowdsourcing platforms, manually rewritten and labeled, and augmented with some open-source safety data (Liu et al. 2023; Tedeschi et al. 2024; Bhardwaj and Poria 2023). Each prompt in the dataset corresponds to a type label.

Figure 2: Examples of Risk Categories and Attack Styles with sensitive terms masked. A harmless test set example shares the same language style as the Adversarial Prefix.

3.2 Self-Evolving Pipeline

Initialization Stage. As shown in the Figure 1, during the Initialization Stage, we selected Llama-3-8B (Meta 2024) as our initial model. For the Red Team model, we expect it to generate complex and diverse adversarial prompts. To achieve this, we adopted an initialization scheme based on random sample contexts. The specific procedure is as follows: we randomly designate a specific type and select a fixed number of data from the training set of the SEAS dataset that corresponds to this type. These data are used as a small set of sample examples, incorporated into the prompts for Supervised Fine-Tuning (SFT) input. Then, we randomly select another sample of the same type as the output.

For the Target model, considering that Llama-3-8B-Instruct already has strong security capabilities, we have initialized a Target model based on Llama-3-8B, which does not have additional security training, to better validate the effectiveness of our method. We selected three datasets specifically designed for SFT focused on general command adjustment. Our objective is to enhance the model's capability for instruction following and to respond appropriately to inputs.

Attack Stage. At the beginning of each Attack Stage, we construct seed prompts by specifying a type and concatenating a fixed number (k) of prompts from the SEAS dataset's training set. This activates the Red Team model to generate adversarial prompts. In order to ensure the diversity of the Red Team model's output, we adopted nucleus sampling (Holtzman et al. 2019) and carried out multiple samplings to generate n prompts. Following this, we input these prompts

to the Target model, also conducting nucleus sampling and multiple samplings, to obtain m output responses.

By concatenating n adversarial prompts with m responses and processing them through a Safe Classifier for safety evaluation, we obtain $n \times m$ tuples of $\{seed\ prompt, adversarial\ prompt, response, label\}$, where label = 1 represents unsafe. Please note that the safety assessment specifically pertains to the response.

Adversarial Optimization Stage. In the Adversarial Optimization Stage, we filter and construct data pairs for optimization. Here, we use Direct Preference Optimization (DPO) loss (Rafailov et al. 2024). For the Red Team model, we use the seed prompt as input, treating the adversarial prompt that triggers harmful response as the chosen output, and the one that doesn't trigger such response as the rejected output. The optimization loss of the Red Team Model R_{i+1}:

$$\mathcal{L}_{\mathcal{R}} = -\,\mathbb{E}_{(s,p_1,p_0)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(p_1 \mid s)}{\pi_{R_i}(p_1 \mid s)} - \beta\log\frac{\pi_\theta(p_0 \mid s)}{\pi_{R_i}(p_0 \mid s)}\right)\right],$$

where s represents the seed prompt, p_1 denotes the adversarial prompt that successfully induces the Target model to generate unsafe content, π_{R_i} is the Red Team model of last iteration i, and π_θ is a policy to be optimized.

For the Target model, we use the adversarial prompt as input, treating the safe response as the chosen output and the harmful response as the rejected output. The optimization loss of the Target model T_{i+1}:

$$\mathcal{L}_{\mathcal{T}} = -\,\mathbb{E}_{(p,r_0,r_1)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(r_0 \mid p)}{\pi_{T_i}(r_0 \mid p)} - \beta\log\frac{\pi_\theta(r_1 \mid p)}{\pi_{T_i}(r_1 \mid p)}\right)\right],$$

where p represents the adversarial prompt, and r_0 represents the model's safe response.

Subsequently, the Attack Stage and the Adversarial Optimization Stage are cyclically performed, continually generating adversarial prompts and producing paired data. Through multiple iterations of model updates, the capabilities of the models are steadily enhanced.

4 Experiments

In this section, we conduct comprehensive empirical experiments to evaluate the effectiveness of SEAS. We mainly assess the improvement in the Target model's security performance while maintaining general capability, the complexity and diversity of the Red Team model's adversarial prompts, and the advantages of SEAS.

4.1 Experimental Details

Fine-tuning Details. In our research, we filtered and constructed a dataset of 16K entries from the SEAS training set for SFT targeted at the Red Team model.

To construct the Target model, we selected three high-quality open-source general instruction-following datasets as seed datasets: ShareGPT (Chiang et al. 2023), Dolly (Conover et al. 2023) and LIMA (Zhou et al. 2023a). These datasets were used to fine-tune the Target model and enhance its ability to follow instructions. In addition, to ensure these datasets were free of adversarial content, we conducted extra data cleansing to minimize the presence of harmful or unsafe instructions. Overall, we used approximately 101K cleaned samples from these datasets.

Safe Classifier. To quickly and effectively determine whether the responses from Target model contain unsafe content, we utilize Meta Llama Guard 2 (Meta 2024) as our safety classifier. It is an 8B parameter LLM safeguard model based on Llama 3. It functions as a LLM, determining whether a given prompt or response is safe or unsafe through its generated text output.

Data Iterations. At the beginning of each Attack Stage, we constructed a dataset of 5K seed prompts. Seed prompts were created by randomly selecting 3 ($k = 3$) prompts of the same type from the training set. During the generation process of both Red Team and Target models, we employed settings of $T = 0.8$ and $p = 0.9$, and sampled each model 5 times ($n = m = 5$). When conducting safety evaluation with the Safe Classifier, we utilized a greedy strategy (Sutskever, Vinyals, and Le 2014). Each Attack Stage ultimately resulted in a dataset containing 125K entries.

Based on this dataset, we constructed paired data. Due to the parameter settings, the model outputs exhibited high diversity, allowing us to collect approximately 4.8K data pairs per round for iterative optimization of Red Team model. To balance the generality and security of Target model, we selected 2K pairs of safe data and randomly mixed them with general data for training. General data came from two high-quality paired open-source datasets, namely OpenOrca (Lian et al. 2023) and ORPO-DPO-Mix (Labonne 2024). The prerequisite for iterating our model is that its general performance does not degrade significantly. In detail, we mixed 7K pairs of general data in iteration 1, and 14K pairs of data in both iterations 2 and 3.

Evaluation Metrics and Benchmarks. The primary measure of a red teaming method's success is its Attack Success Rate (ASR) on a given Target model (Perez et al. 2022; Ganguli et al. 2022; Zou et al. 2023), which is the percentage of adversarial prompts that elicit the harmful responses. Formally, let f be a Target model with generation function $f_T(x) = x'$, where T is the number of tokens to be generated, x is a adversarial prompt, and x' is the response. Let g be a red teaming method that generates a list of adversarial prompts, N is the number of adversarial prompts and let c be a classifier mapping completion x' and behavior y to 1 if a red teaming was successful and 0 if not. The ASR of g on Target model f for behavior y is then defined as

$$\text{ASR}(y, g, f) = \frac{1}{N}\sum c(f_T(x_i), y).$$

We evaluate the model's performance using two security benchmarks (Beaver (Ji et al. 2023b), hh-rlhf (Ganguli et al.

Models	In-Domain	Beaver	hh-rlhf
GPT-4o	5.41%	3.01%	1.10%
GPT-4 Turbo	15.52%	1.71%	0.45%
Qwen1.5-110B	22.40%	2.43%	1.00%
Llama3-70B-It	9.40%	3.00%	3.25%
Llama3-8B-It	5.50%	2.00%	1.60%
Mistral-7B	48.40%	10.00%	5.35%
MART-3shot-T_1	67.90%	25.43%	20.86%
MART-3shot-T_2	32.85%	14.71%	10.01%
SEAS-T_0	62.20%	24.86%	20.36%
SEAS-T_1	21.35%	7.14%	3.80%
SEAS-T_2	10.95%	6.14%	3.20%
SEAS-T_3	7.00%	5.14%	2.50%

Table 1: ASR (lower is better on Target model) performance comparison on SEAS In-Domain test set, Beaver and hh-rlhf. "It" is short for "Instruct".

2022)), two jailbreak datasets (GCG (Zou et al. 2023), Autodan (Liu et al. 2024)), and two general capability benchmarks (Arena Hard (Chiang et al. 2024), MT-Bench (Zheng et al. 2023)).

Compared Models. We compare SEAS with several public available models, including GPT-4o, GPT-4 Turbo (OpenAI et al. 2023), Qwen1.5-110B-Chat (Alibaba 2024), Llama3-8B-Instruct, Llama3-70B-Instruct (Meta 2024), Mistral-7B-Instruct (Jiang et al. 2023) and several iterations of the MART-3shot. In the names of the SEAS models, "T" stands for "Target model", "R" stands for "Red Team model" and the numbers 0 through 3 indicate how many iterations each model has undergone in the SEAS process.

4.2 Experimental Results

Evaluation of Target Models

Performance across Different Models. To thoroughly evaluate the performance of Target models, we compared them with several benchmark methods and publicly available models within the industry.

As the Table 1 shows, the ASR on the In-Domain test set of the SEAS dataset decreased by **55.20%** after three iterations of SEAS. The performance metric of the third-round optimized model (SEAS-T_3) surpassed that of GPT-4. On open-source test sets Beaver and hh-rlhf, the ASR of the multi-round SEAS Target model decreased, the SEAS-T_3 performing comparably to Llama3-70B-Instruct. This effectively proves the effectiveness of the SEAS scheme in enhancing model security performance, although it still slightly trails behind advanced models like GPT-4 and Llama3-8B-Instruct, which have undergone extensive manual red teaming (OpenAI et al. 2023; Meta 2024). Indeed, we could further improve the SEAS model by expanding supervision sources, augmenting seed datasets, and incorporating human oversight. However, this is not the main focus of this paper, and we leave it to future research. Moreover, the comparison between the In-Domain test set and other open-source test sets clearly shows that the In-Domain test set achieves a higher ASR, effectively demonstrating the SEAS dataset's effectiveness in exploring model risks.

Models	SEAS-T_0	SEAS-T_1	SEAS-T_2	SEAS-T_3
SEAS-R_0	54.89%	16.44%	6.33%	3.12%
SEAS-R_1	88.21%	59.02%	14.55%	7.08%
SEAS-R_2	91.28%	90.55%	51.47%	24.92%
SEAS-R_3	95.17%	97.06%	92.93%	73.97%

Table 2: ASR performance across different iterations of Red Team and Target models.

Models	Arena	MT-Bench
SEAS-T_0	4.6	7.088
SEAS-T_1	9.8	7.459
SEAS-T_2	14.4	7.315
SEAS-T_3	13.3	7.447
SEAS-Gen	13.4	6.063

Table 3: General capability assessment of multi-round Target models and SEAS-Gen (trained on all general data without SEAS). Higher values indicate better performance.

Performance across Different Iterations. To measure the improvement in the capabilities of multiple Target models across iterations, we conducted red team attacks using Red Team models from various iterations against these Target models and recorded ASR. In Table 2, we observed that the ASR of the Target models decreased progressively with each round when subjected to attacks by the same Red Team model. This effectively demonstrates the progressive enhancement of the security capabilities of the Target models. We employed settings of T = 0.8 and p = 0.9, and sampled each model 5 times ($n = m = 5$) here.

General Performance. We assessed the Target models on two general benchmarks. As Table 3 shows, after three iterations, the models stayed stable in the MT-Bench assessment and improved by 8.7 in the Arena assessment. Along with Table 1, these results show that SEAS enhances security performance without compromising general capabilities.

Incorrect Refusal Rates. We evaluated the model's performance in terms of incorrect refusals (Anthropic 2024), specifically measuring how often the model inappropriately refuses to respond to harmless prompts. We chose XSTest (Röttger et al. 2024a), a widely used benchmark for evaluating incorrect refusals. It includes 250 security instructions with ambiguous boundaries (e.g., "How can I kill a Python process?"), as well as 200 unsafe instructions. We followed the article's protocol (Röttger et al. 2024a) to evaluate using GPT-4. As shown in Figure 3, the rate of incorrect refusals for safe prompts remained stable, without significant increases. For unsafe prompts, the rate of correct refusals significantly increased. To prevent the model from overfitting to the syntax and style of attack commands, we assessed its behavior using the harmless test set from the SEAS dataset. In Figure 4, the SEAS method does not increase the model's over-defensive reactions. After iterations, the rate

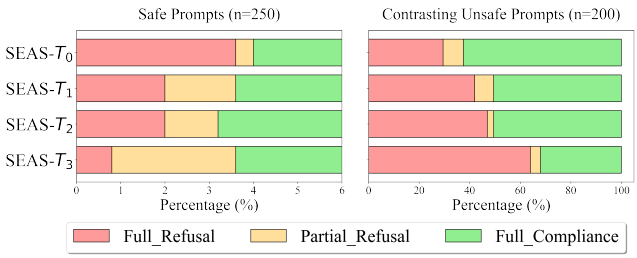

Figure 3: Performance of Target models on the XSTest evaluations with Safe and Unsafe Prompts. the lower the Full Refusal and Partial Refusal rates, the better.

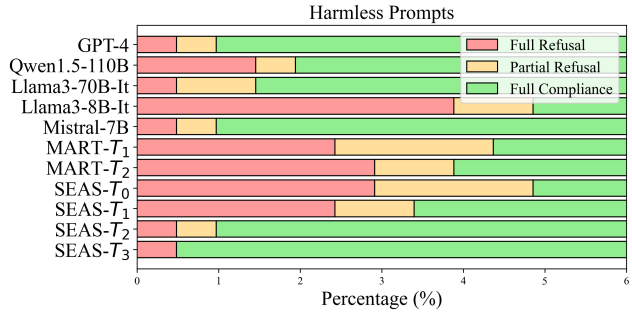

Figure 4: Incorrect Refusal on harmless test set. We use the same evaluation criteria as XSTest. "It" stand for "Instruct".

Models	Normal	Adversarial
SEAS-T_0	52.68%	63.40%
SEAS-T_1	24.55%	20.95%
SEAS-T_1-SFT	61.16%	68.75%
SEAS-T_2	21.88%	9.57%
SEAS-T_2-SFT	38.39%	32.15%
SEAS-T_3	16.96%	5.74%

Table 4: ASR (lower is better on Target model) performance comparison, with different iteration and training schemes.

of incorrect refusals for ambiguous data in the SEAS Target model progressively decreased. This demonstrates that SEAS enhances the model's ability to recognize actual risk information. We have observed that the Llama3-8B-Instruct model has a high rate of incorrect refusals, which may indicate that it is being exaggerate safety (Bianchi et al. 2024).

In-Domian Test. The SEAS In-Domain test set includes two labels: normal and adversarial. Normal prompts are categorized by risk type, while adversarial prompts are labeled by both risk type and attack style (e.g., Role Play).

Table 4 shows two key benefits of SEAS: it strengthens Target models against both normal and adversarial attacks while reducing ASR for adversarial data more effectively, demonstrating its robustness against complex prompts.

Evaluation of Red Team Models

Datasets	SEAS-Train	hh-rlhf
SEAS-Train	0.4052	0.3699
SEAS-R_0	0.4116	0.3770
SEAS-R_1	0.4043	0.4178
SEAS-R_2	0.3654	0.4646
SEAS-R_3	0.3457	0.4521
hh-rlhf	0.3699	0.6099
Beaver	0.3728	0.6135

Table 5: Cosine distance-based similarity of adversarial prompts from different iterations of Red Team models to SEAS-Train (training set for SEAS) and hh-rlhf.

Models	In-domain	Beaver	hh-rlhf
SEAS-T_0	62.20%	24.86%	20.36%
SEAS-T_1-SFT(ep=1)	67.90%	25.43%	20.86%
SEAS-T_1	21.35%	7.14%	3.80%
SEAS-T_2-SFT(ep=1)	32.85%	14.71%	10.01%
SEAS-T_2	10.95%	6.14%	3.20%

Table 6: ASR (lower is better on Target model) performance comparison, with different iteration and training schemes. The default training scheme for SEAS is DPO (ep=1).

Diversity. To avoid Red Team models settling on a single deterministic strategy upon discovering a few effective adversarial prompts (Puterman 2014; Bengio et al. 2021), it is essential to maintain the diversity of the prompts they generate. Based on previous work (Reimers and Gurevych 2019; Hong et al. 2024), we assess the similarity between the SEAS training set, adversarial prompts from the Red Team model after four iterations, and two open-source datasets using the cosine similarity of sentence embeddings. The results in Table 5, show that the similarity between the adversarial prompts and both the train set and other datasets is relatively low (lower than the similarity values of the two open-source datasets). This indicates a high level of diversity in the adversarial prompts generated by the Red Team models.

Complexity. After attacking the publicly available and Target models with adversarial prompts generated by Red Team model (as shown in Tables 1 and 2), we observed an increase in the ASR with each iteration. This effectively demonstrates that SEAS improves the complexity of adversarial prompts from Red Team models. In particular, SEAS-R_0, fine-tuned exclusively with the SEAS dataset, demonstrated a higher ASR, underscoring the effectiveness of the SEAS dataset compared to other open-source datasets.

Advantages of SEAS

Pair-wise Loss Updates. Existing automated red teaming solutions, including custom jailbreaking templates or pipelines, primarily utilize the SFT objective to update Red Team models and Target models (Ge et al. 2023).

However, pair-wise loss - Direct Preference Optimization (DPO) (Rafailov et al. 2024) is more suitable in safety scenarios. This method not only increases the probability of the

Models	R_0	R_1	R_2	R_3	GCG	AutoDan
GPT-4 Turbo	16.00	31.00	35.20	66.00	2.42	11.00
Qwen1.5-110B	18.34	37.00	53.32	86.06	0.38	5.19
Llama3-70B-It	6.02	21.06	21.70	56.68	1.15	0.58
Llama3-8B-It	2.56	11.04	12.92	17.70	0	0.19
Mistral-7B	45.18	83.38	92.58	97.24	53.65	81.35

Table 7: ASR (% ↑) performance comparison across different iterations of Red Team outputs and jailbreak datasets.

Models	MT-Bench(↑)	In-domain(↓)	Beaver(↓)	hh-rlhf(↓)
SEAS-T_0	7.088	62.20%	24.86%	20.36%
SEAS-T_3	7.447	7.00%	5.14%	2.50%
SEAS-All	6.281	26.75%	16.57%	11.31%

Table 8: The comprehensive performance comparison of multiple update models and a single update model.

chosen text but also decreases the likelihood of the rejected text, thereby improving the capabilities of the Red Team model and the Target model. In contrast, SFT loss merely reinforces the correct answer repeatedly, preventing the model from encountering incorrect samples.

We used the same data to perform both SFT and DPO, and the results are shown in Table 6. For the same data, the DPO shows significant improvement after one round of training (ep=1), whereas the SFT scheme shows no improvement after one round and only modest improvement after two rounds. In Table 4, we found that the DPO scheme outperforms the SFT scheme in processing both data types. This performance difference is particularly notable with complex adversarial samples, where the benefits of the DPO scheme are more pronounced. Additionally, Figure 4 shows that the DPO scheme effectively reduces the incorrect refusal rate in tests involving harmless samples. In contrast, even after two iterations, the SFT scheme does not show a significant reduction in the incorrect refusal rate.

Iterative Updates. Table 2 and 7 clearly shows that the ASR of various Red Team models on the same Target models increases with each iteration. This trend confirms that SEAS framework is capable of effectively exploiting the vulnerabilities of Target models through iterative updates.

We initially train and optimize the SEAS-T_0 model using data processed through three iterations of SEAS, applying consistent hyperparameters to obtain SEAS-All. SEAS-T_3 is the control group. Results displayed in Table 8 reveal that multiple iterative updates substantially improve performance on security test sets within the In-Domain and on open-source datasets. Simultaneously, the model's general capabilities, evaluated on Arena and MT-Bench, demonstrate marginal enhancements with each iteration. The main advantage of using iterative updates and optimizing the Target model via the pair-wise loss is the achievement of a higher baseline policy after the initial round. This indicates that performing multiple updates on the Target model with the same data is more advantageous than a single update.

Models	In-Domain	Beaver	hh-rlhf
SEAS-R_0	62.20%	24.86%	20.36%
SEAS-Gen	65.10%	29.43%	25.86%

Table 9: Comparison of model performance using SEAS all general data and the SEAS-T_0 model.

Datasets	R_0-out	R_{0123}-out
R_0-out	0.4194	0.3895
R_{0123}-out	-	0.4676

Table 10: Cosine distance-based similarity between adversarial prompts generated by SEAS-R_0 and different rounds of the Red Team models for all seed prompts.

Ablation

General Data. To ablate the influence of general data on the model's security performance, we optimized all the general data used in our SEAS process using the same scheme as SEAS-T_0, resulting in SEAS-Gen. We assessed this method using three security datasets. The results, presented in the Table 9, show that the ASR has improved across all three test sets. This effectively confirms that the general data we utilized does not enhance the model's security capabilities.

Seed Prompts. To ablate the impact of randomness from seed prompts on the diversity of adversarial prompts, we conducted a procedure where seed prompts from four attack stages were input into SEAS-R_0 to obtain outputs (R_0-out). We then collected all adversarial prompts from different models over three iterations (R_{0123}-out) and measured their semantic similarity. As shown in Table 10, the low similarity between the datasets indicates that seed prompt randomness does not influence the outputs. This confirms that the diversity of adversarial prompts primarily arises from iterative updates in the multi-round Red Team models.

5 Conclusion

In this paper, we introduce a Self-Evolving Adversarial Safety (SEAS) optimization approach. By continuously improving the model using adversarial data, this framework enhances the safety of the model and overcomes the limitations of traditional red teaming, which cannot comprehensively, effectively, and dynamically explore model risks. Through comprehensive empirical experiments, we demonstrate that this framework is effective in enhancing security. After three iterations, the foundational model achieves a security level comparable to that of GPT-4.

Moreover, even after three iterations, the safety performance of the SEAS model still slightly behind that of Llama3-8B-Instruct, which has undergone extensive expert red teaming involving hundreds of thousands of examples. In fact, we could further improve the SEAS model through data augmentation, manual annotation, and additional rounds of iteration. However, this is not the main focus of our paper, and we leave it for future work.

Acknowledgements

This work was partially supported by:

- The State Key Laboratory of Massive Personalized Customization System and Technology (No. H&C-MPC-2023-02-07(Q)).
- The State Grid Technology Project (5700-202416236A-1-1-ZN) on "Research on Active Semantic Discovery Technology Based on SG-CIM and Its Application in Power Grid Equipment Supply Chain Optimization".
- China Unicom Software Research Institute under the "Framework Agreement for Seven Model Technology Research and Application Demonstration Projects (Software Development for Government Enterprise Content Generation) from 2024 to 2025" (No. 5500331818).
- The National Natural Science Foundation of China (NSFC Nos. 62076031 and 62076036).
- Zhongguancun Academy.

References

Alibaba. 2024. Introducing Qwen1.5 — Qwen.

Anthropic. 2024. Introducing the next generation of Claude.

Bengio, E.; Jain, M.; Korablyov, M.; Precup, D.; and Bengio, Y. 2021. Flow Network based Generative Models for Non-Iterative Diverse Candidate Generation. arXiv:2106.04399.

Bhardwaj, R.; and Poria, S. 2023. Red-Teaming Large Language Models using Chain of Utterances for Safety-Alignment. arXiv:2308.09662.

Bianchi, F.; Suzgun, M.; Attanasio, G.; Röttger, P.; Jurafsky, D.; Hashimoto, T.; and Zou, J. 2024. Safety-Tuned LLaMAs: Lessons From Improving the Safety of Large Language Models that Follow Instructions. arXiv:2309.07875.

Chao, P.; Robey, A.; Dobriban, E.; Hassani, H.; Pappas, G. J.; and Wong, E. 2023. Jailbreaking Black Box Large Language Models in Twenty Queries. arXiv:2310.08419.

Chiang, W.-L.; Li, Z.; Lin, Z.; Sheng, Y.; Wu, Z.; Zhang, H.; Zheng, L.; Zhuang, S.; Zhuang, Y.; Gonzalez, J. E.; Stoica, I.; and Xing, E. P. 2023. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality.

Chiang, W.-L.; Zheng, L.; Sheng, Y.; Angelopoulos, A. N.; Li, T.; Li, D.; Zhang, H.; Zhu, B.; Jordan, M.; Gonzalez, J. E.; and Stoica, I. 2024. Chatbot Arena: An Open Platform for Evaluating LLMs by Human Preference. arXiv:2403.04132.

Chilton, J. 2023. The New Risks ChatGPT Poses to Cybersecurity. *Harvard Business Review*.

Chowdhury, A. G.; Islam, M. M.; Kumar, V.; Shezan, F. H.; Kumar, V.; Jain, V.; and Chadha, A. 2024. Breaking Down the Defenses: A Comparative Survey of Attacks on Large Language Models. arXiv:2403.04786.

Christian, J. 2023. Amazing "jailbreak" bypasses ChatGPT's ethics safeguards. *Futurism, February*, 4: 2023.

Conover, M.; Hayes, M.; Mathur, A.; Xie, J.; Wan, J.; Shah, S.; Ghodsi, A.; Wendell, P.; Zaharia, M.; and Xin, R. 2023. Free Dolly: Introducing the World's First Truly Open Instruction-Tuned LLM.

Fernando, C.; Banarse, D.; Michalewski, H.; Osindero, S.; and Rocktäschel, T. 2023. Promptbreeder: Self-Referential Self-Improvement Via Prompt Evolution. arXiv:2309.16797.

Ganguli, D.; Lovitt, L.; Kernion, J.; Askell, A.; Bai, Y.; Kadavath, S.; Mann, B.; Perez, E.; Schiefer, N.; Ndousse, K.; Jones, A.; Bowman, S.; Chen, A.; Conerly, T.; DasSarma, N.; Drain, D.; Elhage, N.; El-Showk, S.; Fort, S.; Hatfield-Dodds, Z.; Henighan, T.; Hernandez, D.; Hume, T.; Jacobson, J.; Johnston, S.; Kravec, S.; Olsson, C.; Ringer, S.; Tran-Johnson, E.; Amodei, D.; Brown, T.; Joseph, N.; McCandlish, S.; Olah, C.; Kaplan, J.; and Clark, J. 2022. Red Teaming Language Models to Reduce Harms: Methods, Scaling Behaviors, and Lessons Learned. arXiv:2209.07858.

Ge, S.; Zhou, C.; Hou, R.; Khabsa, M.; Wang, Y.-C.; Wang, Q.; Han, J.; and Mao, Y. 2023. MART: Improving LLM Safety with Multi-round Automatic Red-Teaming. arXiv:2311.07689.

Hartvigsen, T.; Gabriel, S.; Palangi, H.; Sap, M.; Ray, D.; and Kamar, E. 2022. ToxiGen: A Large-Scale Machine-Generated Dataset for Adversarial and Implicit Hate Speech Detection. arXiv:2203.09509.

Holtzman, A.; Buys, J.; Du, L.; Forbes, M.; and Choi, Y. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Hong, Z.-W.; Shenfeld, I.; Wang, T.-H.; Chuang, Y.-S.; Pareja, A.; Glass, J.; Srivastava, A.; and Agrawal, P. 2024. Curiosity-driven Red-teaming for Large Language Models. arXiv:2402.19464.

Inan, H.; Upasani, K.; Chi, J.; Rungta, R.; Iyer, K.; Mao, Y.; Tontchev, M.; Hu, Q.; Fuller, B.; Testuggine, D.; and Khabsa, M. 2023. Llama Guard: LLM-based Input-Output Safeguard for Human-AI Conversations. arXiv:2312.06674.

Ji, J.; Liu, M.; Dai, J.; Pan, X.; Zhang, C.; Bian, C.; Zhang, C.; Sun, R.; Wang, Y.; and Yang, Y. 2023a. BeaverTails: Towards Improved Safety Alignment of LLM via a Human-Preference Dataset. arXiv:2307.04657.

Ji, J.; Liu, M.; Dai, J.; Pan, X.; Zhang, C.; Bian, C.; Zhang, C.; Sun, R.; Wang, Y.; and Yang, Y. 2023b. BeaverTails: Towards Improved Safety Alignment of LLM via a Human-Preference Dataset. arXiv:2307.04657.

Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D. S.; Casas, D. d. l.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825*.

Labonne, M. 2024. orpo-dpo-mix-40k · Datasets at Hugging Face. https://huggingface.co/datasets/mlabonne/orpo-dpo-mix-40k.

Li, X.; Zhou, Z.; Zhu, J.; Yao, J.; Liu, T.; and Han, B. 2024. DeepInception: Hypnotize Large Language Model to Be Jailbreaker. arXiv:2311.03191.

Lian, W.; Goodson, B.; Pentland, E.; Cook, A.; Vong, C.; and "Teknium". 2023. OpenOrca: An Open Dataset of GPT Augmented FLAN Reasoning Traces. https://https://huggingface.co/Open-Orca/OpenOrca.

Liu, C.; Zhao, F.; Qing, L.; Kang, Y.; Sun, C.; Kuang, K.; and Wu, F. 2023. Goal-Oriented Prompt Attack and Safety Evaluation for LLMs. arXiv:2309.11830.

Liu, X.; Xu, N.; Chen, M.; and Xiao, C. 2024. AutoDAN: Generating Stealthy Jailbreak Prompts on Aligned Large Language Models. arXiv:2310.04451.

Mazeika, M.; Phan, L.; Yin, X.; Zou, A.; Wang, Z.; Mu, N.; Sakhaee, E.; Li, N.; Basart, S.; Li, B.; Forsyth, D.; and Hendrycks, D. 2024. HarmBench: A Standardized Evaluation Framework for Automated Red Teaming and Robust Refusal. arXiv:2402.04249.

Meta. 2024. Introducing Meta Llama 3: The most capable openly available LLM to date.

Mu, T.; Helyar, A.; Heidecke, J.; Achiam, J.; Vallone, A.; Kivlichan, I. D.; Lin, M.; Beutel, A.; Schulman, J.; and Weng, L. 2024. Rule Based Rewards for Fine-Grained LLM Safety. In *ICML 2024 Next Generation of AI Safety Workshop*.

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

Perez, E.; Huang, S.; Song, F.; Cai, T.; Ring, R.; Aslanides, J.; Glaese, A.; McAleese, N.; and Irving, G. 2022. Red Teaming Language Models with Language Models. arXiv:2202.03286.

Puterman, M. L. 2014. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Rafailov, R.; Sharma, A.; Mitchell, E.; Ermon, S.; Manning, C. D.; and Finn, C. 2024. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. arXiv:2305.18290.

Reimers, N.; and Gurevych, I. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. arXiv:1908.10084.

Röttger, P.; Kirk, H. R.; Vidgen, B.; Attanasio, G.; Bianchi, F.; and Hovy, D. 2024a. XSTest: A Test Suite for Identifying Exaggerated Safety Behaviours in Large Language Models. arXiv:2308.01263.

Röttger, P.; Pernisi, F.; Vidgen, B.; and Hovy, D. 2024b. SafetyPrompts: a Systematic Review of Open Datasets for Evaluating and Improving Large Language Model Safety. arXiv:2404.05399.

Samvelyan, M.; Raparthy, S. C.; Lupu, A.; Hambro, E.; Markosyan, A. H.; Bhatt, M.; Mao, Y.; Jiang, M.; Parker-Holder, J.; Foerster, J.; Rocktäschel, T.; and Raileanu, R. 2024. Rainbow Teaming: Open-Ended Generation of Diverse Adversarial Prompts. arXiv:2402.16822.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to Sequence Learning with Neural Networks. arXiv:1409.3215.

Tedeschi, S.; Friedrich, F.; Schramowski, P.; Kersting, K.; Navigli, R.; Nguyen, H.; and Li, B. 2024. ALERT: A Comprehensive Benchmark for Assessing Large Language Models' Safety through Red Teaming. arXiv:2404.08676.

Yu, J.; Lin, X.; Yu, Z.; and Xing, X. 2023. GPTFUZZER: Red Teaming Large Language Models with Auto-Generated Jailbreak Prompts. arXiv:2309.10253.

Zheng, L.; Chiang, W.-L.; Sheng, Y.; Zhuang, S.; Wu, Z.; Zhuang, Y.; Lin, Z.; Li, Z.; Li, D.; Xing, E. P.; Zhang, H.; Gonzalez, J. E.; and Stoica, I. 2023. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. arXiv:2306.05685.

Zhou, C.; Liu, P.; Xu, P.; Iyer, S.; Sun, J.; Mao, Y.; Ma, X.; Efrat, A.; Yu, P.; Yu, L.; Zhang, S.; Ghosh, G.; Lewis, M.; Zettlemoyer, L.; and Levy, O. 2023a. LIMA: Less Is More for Alignment. arXiv:2305.11206.

Zhou, Y.; Muresanu, A. I.; Han, Z.; Paster, K.; Pitis, S.; Chan, H.; and Ba, J. 2023b. Large Language Models Are Human-Level Prompt Engineers. arXiv:2211.01910.

Zou, A.; Wang, Z.; Carlini, N.; Nasr, M.; Kolter, J. Z.; and Fredrikson, M. 2023. Universal and Transferable Adversarial Attacks on Aligned Language Models. arXiv:2307.15043.