A close-up of a logo

Description automatically generated

**Dissertation Idea Paper**

**Small Language Model Jailbreak Defender Framework**

Eduardo Alberto Castro Puello

Thesis Advisor

Dr. Wei Li

College of Computing and Engineering

Nova Southeastern University

3301 College Avenue – Carl DeSantis Building

Fort Lauderdale-Davie, Florida 33314-7796

Contents

[Problem Statement 4](#_Toc197438600)

[Dissertation Goal 5](#_Toc197438601)

[Relevance and Significance 6](#_Toc197438602)

[Jailbreak Attacks 6](#_Toc197438603)

[White-box Attack 6](#_Toc197438604)

[Gradient-Based 6](#_Toc197438605)

[Logits-Based 12](#_Toc197438606)

[Fine-Tuning-Based 15](#_Toc197438607)

[Black-Box Attack 18](#_Toc197438608)

[Jailbreak Defense Methods 21](#_Toc197438609)

[Model Level Defense: SFT-based 21](#_Toc197438610)

[Approach 24](#_Toc197438611)

[Proposed Strategy Part 1 – Guardrail Wrapper. 26](#_Toc197438612)

[Proposed Strategy Part 2 – LLM judge to Finetune the task-specialize SLM. 28](#_Toc197438613)

[Proposed Experiment: Securing a Fine-Tuned SLM for JavaScript QA Tasks. 30](#_Toc197438614)

[Objective 30](#_Toc197438615)

[Tools and Models 30](#_Toc197438616)

[Experiment Workflow 30](#_Toc197438617)

[Timeline and Milestones 31](#_Toc197438618)

[Expected Outcomes 31](#_Toc197438619)

[Resources 32](#_Toc197438620)

[References 33](#_Toc197438621)

Figure List

[Figure 1. Taxonomy of Jailbreak Attack, (Yi, 2024) 6](#_Toc197438564)

[Figure 2. Gradient-Based Attack, (Yi, 2024) 6](#_Toc197438565)

[Figure 3. Universal and Transferable Adversarial Attacks on Aligned Language Models Performance Results 7](#_Toc197438566)

[Figure 4. ARCA Performance Results 7](#_Toc197438567)

[Figure 5. AutoDAN Performance Results 8](#_Toc197438568)

[Figure 6. ASETF Performance numbers. 8](#_Toc197438569)

[Figure 7. Jailbreaking Leading Safety-Aligned LLMs with Simple Adaptive Attacks Performance results 9](#_Toc197438570)

[Figure 8. PGD Performance results 9](#_Toc197438571)

[Figure 9. Query-Based Adversarial Prompt Generation Performance results. 10](#_Toc197438572)

[Figure 10. PAL Performance results. 10](#_Toc197438573)

[Figure 11. Transferable-advICL Performance results 11](#_Toc197438574)

[Figure 12. PRP ASR Performance results 11](#_Toc197438575)

[Figure 13. Logits-based attack, (Yi, 2024) 12](#_Toc197438576)

[Figure 14. Intervention point identification “Magic” Performance results. 12](#_Toc197438577)

[Figure 15. COLD-Attack Performance results 13](#_Toc197438578)

[Figure 16. RADIAL Performance results 13](#_Toc197438579)

[Figure 17. Weak to Strong Performance results 14](#_Toc197438580)

[Figure 18. EXPLOITING GENERATION Performance results 14](#_Toc197438581)

[Figure 19. DSN Performance results 15](#_Toc197438582)

[Figure 20. Fine-Tuning-based attack 15](#_Toc197438583)

[Figure 21. Fine-tuning GPT-3.5 Turbo by mixing different numbers of safety samples 16](#_Toc197438584)

[Figure 22. Shadow Alignment – Harmful Finetuned Performance results 16](#_Toc197438585)

[Figure 23. LoRA Performance results 17](#_Toc197438586)

[Figure 24. 340 Harmful content finetuning Performance results 17](#_Toc197438587)

[Figure 25. Template Completion – Nesting query 18](#_Toc197438588)

[Figure 26. Template Completion – Context based query 18](#_Toc197438589)

[Figure 27. Template Completion – Code Injection 19](#_Toc197438590)

[Figure 28. Prompt rewriting attack – Cipher 19](#_Toc197438591)

[Figure 29. Prompt rewriting – Low resource Language 20](#_Toc197438592)

[Figure 30. Prompt rewriting – Genetic Algorithm 20](#_Toc197438593)

[Figure 31. Taxonomy of jailbreak defense. 21](#_Toc197438594)

[Figure 32. The mean harmfulness score for each dataset (with standard errors on bars). Lower scores indicate less harmful (safer) responses 21](#_Toc197438595)

[Figure 33. Fine-tuned models evaluated with SAP dataset 22](#_Toc197438596)

[Figure 34. RED-EVAL results with and without internal thoughts 23](#_Toc197438597)

[Figure 35. Guardrails Architecture with Embedded SLM for Safe Model-User Interaction. 26](#_Toc197438598)

[Figure 36 Task Specialized SLM Fine Tune and Toxic Tokens Extractions with AutoDAN. 28](#_Toc197438599)

# Problem Statement

The rapid advancement of Large Language Models (LLMs) has introduced a new set of cybersecurity risks that were previously non-existent. These include extortion, fake conversations, and the generation of misleading text, voice, and video content, which can be exploited for malicious purposes. The widespread adoption of LLMs depends on their ability to effectively mitigate harmful content while maintaining high performance.

One of the most concerning vulnerabilities lies in fine-tuning attacks, where models are retrained with harmful data to bypass safety mechanisms. Research has demonstrated that even a small number of adversarial examples can significantly undermine a model’s safety alignment, making it susceptible to jailbreak attacks and harmful content generation. Given the growing trend of deploying Small Language Models (SLMs) for their efficiency and speed, ensuring their security while preserving response quality is an urgent challenge. However, existing safety alignment techniques often introduce excessive conservatism, leading to a high rate of false positives and degradation in performance for benign tasks. This trade-off between security and usability creates a significant gap in the field that must be addressed to develop reliable AI systems.

# Dissertation Goal

The objective of this dissertation is to develop a framework for training and fine-tuning Small Language Models (SLMs) that balances security, response speed, and task-specific performance. This framework will leverage a multi-step Chain-of-Thoughts (CoT) Judge approach to reinforce safety alignment while minimizing over-sensitivity to safe prompts. The proposed solution aims to:

1. Identify and analyze fine-tuning-based vulnerabilities in LLMs and SLMs.
2. Develop a methodology to integrate safety alignment in SLMs without compromising their responsiveness and accuracy.
3. Implement and evaluate a multi-step Chain-of-Thoughts Judge mechanism to improve security robustness.
4. Establish guidelines for secure SLM training to prevent adversarial fine-tuning exploits.

By achieving these objectives, this dissertation will contribute to the field of AI security by enhancing the reliability of language models deployed in real-world applications.

# Relevance and Significance

## Jailbreak Attacks

A diagram of a jailbreak method

AI-generated content may be incorrect.

Figure 1. Taxonomy of Jailbreak Attack, (Yi, 2024)

### White-box Attack

#### Gradient-Based

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2. Gradient-Based Attack, (Yi, 2024)

**Universal and Transferable Adversarial Attacks on Aligned Language Models** (Zou 2023), the paper discusses a novel method for bypassing safeguards in aligned large language models (LLMs) to generate objectionable content. While previous jailbreak attempts were fragile and required human effort, this approach automatically generates adversarial suffixes using greedy and gradient-based search techniques. These suffixes, when appended to queries, increase the likelihood of LLMs responding affirmatively rather than refusing. The method proves to be highly transferable across different models, successfully inducing objectionable outputs in both closed (ChatGPT, Bard, Claude) and open-source (LLaMA-2-Chat, Pythia, Falcon) LLMs. The findings highlight critical vulnerabilities in current alignment techniques and raise concerns about mitigating such adversarial attacks.

A graph of steps and loss

AI-generated content may be incorrect.

Figure 3. Universal and Transferable Adversarial Attacks on Aligned Language Models Performance Results

**Automatically Auditing Large Language Models via Discrete Optimization** (Jones, 2023) The paper presents a novel approach to auditing large language models by framing it as an optimization problem by looking for the best combination of toxic tokens. The authors propose Autoregressive Randomized Coordinate Ascent (ARCA), a discrete optimization algorithm that efficiently searches for input-output pairs exhibiting specific target behaviors. This method helps uncover issues such as unintended toxic outputs, language mismatches, and biased name associations. By identifying failure modes before deployment, ARCA provides a valuable tool for improving model reliability and safety.

A graph of different types of data

AI-generated content may be incorrect.

Figure 4. ARCA Performance Results

**AutoDAN: Interpretable Gradient-Based Adversarial Attacks on Large Language Models** (Zhu, 2023). The paper critiques existing defenses against attacks on Large Language Models (LLMs), arguing that they may be overly optimistic. It introduces **AutoDAN**, a gradient-based adversarial attack that combines the strengths of both manual jailbreak and automatic adversarial attacks. Unlike previous methods, AutoDAN generates **readable and interpretable prompts** that evade perplexity-based filters while maintaining high success rates. These prompts mimic manual jailbreak strategies, generalize unforeseen harmful behaviors, and transfer effectively to black-box models. Additionally, AutoDAN can **automate system prompt leaks**, providing new insights into LLM vulnerabilities and enhancing red-teaming efforts.

A close-up of a graph

AI-generated content may be incorrect.

Figure 5. AutoDAN Performance Results

**ASETF: A Novel Method for Jailbreak Attack on LLMs through Translate Suffix Embeddings**, (Wang, 2024). The safety defenses of Large Language Models (LLMs) are limited because they rely on manually curated dangerous prompts, which fail to address evolving attack methods. Recent research shows that adding suffixes to harmful instructions can bypass LLM defenses, but existing adversarial attack methods are inefficient, requiring over 100,000 model calls and being vulnerable to common defenses like perplexity filters. To address these challenges, this paper introduces the **Adversarial Suffix Embedding Translation Framework (ASETF)**, which converts adversarial suffix embeddings into coherent text. This approach reduces computational overhead, enhances attack success rates, and improves prompt fluency. Experiments on models like Llama2 and Vicuna demonstrate that ASETF efficiently generates adversarial prompts that can transfer attacks to multiple LLMs, including black-box models like ChatGPT and Gemini.

A screenshot of a table

AI-generated content may be incorrect.

Figure 6. ASETF Performance numbers.

**Jailbreaking Leading Safety-Aligned LLMs with Simple Adaptive Attacks** (Andriushchenko, 2024). The study reveals that even the latest safety-aligned large language models (LLMs) are vulnerable to adaptive jailbreaking attacks. Researchers use log probability access to optimize adversarial prompts through random search, achieving a 100% success rate in bypassing safety mechanisms across multiple models, including GPT-4o, Llama, Claude, and others. They also develop effective jailbreaks for Claude models without logprob access using transfer and prefilling attacks. Additionally, they apply similar methods to detect trojan strings in poisoned models, securing first place in the SaTML'24 Trojan Detection Competition. The key insight is that adaptivity—tailoring attacks to each model's specific weaknesses—is essential. The study provides code and jailbreak artifacts for reproducibility.

A graph with numbers and a line

AI-generated content may be incorrect.

Figure 7. Jailbreaking Leading Safety-Aligned LLMs with Simple Adaptive Attacks Performance results

**Attacking Large Language Models with Projected Gradient Descent**, (Geisler, 2024). The paper discusses a more efficient method for generating adversarial prompts that bypass LLM alignment mechanisms. Traditional discrete optimization approaches, though highly effective, require over 100,000 LLM calls, making them computationally expensive and impractical for tasks like adversarial training and quantitative analysis. To address this, the authors revisit Projected Gradient Descent (PGD) using a continuous relaxation of the input prompt. While previous gradient-based methods struggled, their refined approach carefully manages relaxation errors, significantly improving effectiveness. Their proposed PGD method is up to ten times faster than state-of-the-art discrete optimization while achieving similar attack success rates.

A graph with different colored lines

AI-generated content may be incorrect.

Figure 8. PGD Performance results

**Query-Based Adversarial Prompt Generation**, (Hayase, 2024). The paper presents a new query-based adversarial attack on language models, demonstrating its effectiveness in bypassing safety mechanisms. Unlike previous approaches that rely on white-box access (full model visibility) or transferability (applying adversarial examples across models), this method uses API access to iteratively refine adversarial inputs. The authors successfully tested their attack on GPT-3.5 and OpenAI’s safety classifier, achieving near-perfect evasion rates and generating harmful outputs more reliably than transfer-based attacks. Their findings highlight vulnerabilities in current safety measures, emphasizing the need for improved defenses against adversarial manipulation.

A graph of different colored lines

AI-generated content may be incorrect.

Figure 9. Query-Based Adversarial Prompt Generation Performance results.

**PAL: Proxy-Guided Black-Box Attack on Large Language Models**, (Sitawarin, 2024). This paper introduces the Proxy-Guided Attack on Large Language Models (PAL), an optimization-based attack designed to target LLMs in a black-box, query-only setting. The attack uses a surrogate model to guide the optimization, and a custom loss function tailored for real-world LLM APIs. The PAL attack achieves high success rates: 84% on GPT-3.5-Turbo and 48% on Llama-2-7B, significantly outperforming the previous best method, which had a success rate of 4%. The paper also introduces GCG++, an enhanced version of a prior attack, which achieves a 94% success rate on Llama-2-7B in a white-box setting, and the Random-Search Attack on LLMs (RAL), a simpler, strong baseline for query-based attacks. These techniques aim to advance safety testing of LLMs and contribute to the development of better security mechanisms.

A graph of steps with numbers and symbols

AI-generated content may be incorrect.

Figure 10. PAL Performance results.

**Adversarial Demonstration Attacks on Large Language Models**, (Wang, 2023). This paper addresses a security vulnerability in large language models (LLMs), specifically in-context learning (ICL), which uses data-label pairs as prompts to perform tasks. While demonstrations can improve performance, they may also be exploited by attackers who manipulate only the demonstrations, without altering the input, to mislead the model. The paper proposes a new attack method called **advICL**, which manipulates the demonstration to deceive the model. The study finds that as the number of demonstrations increases, the model's robustness decreases. Furthermore, demonstrations can be reused with different inputs, presenting a more practical threat where attackers can manipulate test examples without prior knowledge of them. The paper introduces **Transferable-advICL**, a version of the attack that can target unseen test examples. The findings highlight the security risks of ICL and emphasize the need for further research to improve its robustness, especially as LLMs become more advanced.

A table with numbers and letters

AI-generated content may be incorrect.

Figure 11. Transferable-advICL Performance results

**PRP: Propagating Universal Perturbations to Attack Large Language Model Guard-Rails**, (Mangaokar, 2024). This paper discusses vulnerabilities in large language models (LLMs) that are designed to be harmless to humans but can still be manipulated through "jailbreak attacks" to generate harmful content. To address this, newer models use a "Guard Model," an additional LLM designed to moderate the output of the primary model. The paper introduces a new attack method called PRP, which is effective against both open-source (e.g., Llama 2) and closed-source (e.g., GPT 3.5) Guard Models. PRP works by using a two-step prefix-based strategy: first, it creates a universal adversarial prefix for the Guard Model, then it propagates this prefix to the primary model's response. The method proves successful even when the attacker has no direct access to the Guard Model. The authors conclude that further improvements in defenses and Guard Models are needed for these systems to be truly effective.

A table with numbers and symbols

AI-generated content may be incorrect.

Figure 12. PRP ASR Performance results

#### Logits-Based

A diagram of a step

AI-generated content may be incorrect.

Figure 13. Logits-based attack, (Yi, 2024)

**Make Them Spill the Beans! Coercive Knowledge Extraction from (Production) LLMs**, (Zhang, 2023). The paper highlights a new vulnerability in Large Language Models (LLMs) that allows bad actors to extract harmful content, even when the model is aligned with ethical guidelines. Unlike traditional jail-breaking methods that rely on carefully crafted prompts, this approach “termed **model interrogation**”exploits access to the model’s output logits, a feature common in open-source models and some commercial APIs. By selectively choosing lower-ranked output tokens at key positions during text generation, attackers can uncover hidden toxic responses. This method is significantly more effective (92% vs. 62% for jailbreaking), faster (10–20 times), and produces more relevant, complete, and clear harmful content. Additionally, it can be combined with jail-breaking techniques to enhance attack success. The study underscores the need for stronger safeguards, as even models designed for coding tasks can be exploited using this method.

A table with numbers and a number of text

AI-generated content may be incorrect.

Figure 14. Intervention point identification “Magic” Performance results.

**COLD-Attack: Jailbreaking LLMs with Stealthiness and Controllability**, (Guo, 2024). This paper explores controllable jailbreak attacks on large language models (LLMs), aiming to manipulate their outputs while maintaining specific constraints such as fluency, stealthiness, sentiment, and ideological bias. The authors establish a link between controllable attack generation and controllable text generation, adapting the Energy-based Constrained Decoding with Langevin Dynamics (COLD) method to create the **COLD-Attack** framework. This framework automates and diversifies jailbreak attack generation by enforcing various constraints, enabling attacks such as adversarial query rewriting and stealthy prompt insertion. Extensive experiments on multiple LLMs (e.g., Llama-2, GPT-4) demonstrate COLD-Attack's effectiveness, versatility, and transferability across models.

A table with numbers and a few words

AI-generated content may be incorrect.

Figure 15. COLD-Attack Performance results

**Analyzing the Inherent Response Tendency of LLMs: Real-World Instructions-Driven Jailbreak**, (Du, 2023). This paper introduces **RADIAL**, a novel automatic jailbreak method that exploits the tendency of Large Language Models (LLMs) to generate affirmative responses. The approach is based on **Inherent Response Tendency Analysis**, which identifies real-world instructions that naturally prompt affirmative answers from LLMs. Using this insight, **Real-World Instructions-Driven Jailbreak** is implemented by strategically embedding these instructions around malicious prompts, effectively bypassing safety mechanisms. RADIAL demonstrates strong attack performance across five advanced open-source LLMs, particularly for English prompts, while also showing effectiveness in cross-language attacks on Chinese instructions. Experimental results validate both the efficacy of the jailbreak method and the reasoning behind its design. By crafting semantically coherent attack prompts, the study highlights potential security risks in LLMs and offers valuable insights to enhance their safety.

A graph of numbers and a line

AI-generated content may be incorrect.

Figure 16. RADIAL Performance results

**Weak-to-Strong Jailbreaking on Large Language Models**, (Zhao, 2024). The paper discusses a new, efficient method for jailbreaking large language models (LLMs) called the **weak-to-strong attack**. Traditional jailbreak attacks, which force LLMs to generate harmful, unethical, or biased text, are typically computationally expensive. The proposed method leverages the observation that aligned (safe) and jailbroken (unsafe) models mainly differ in their initial decoding distributions. The **weak-to-strong attack** works by using two smaller models “one safe and one unsafe” to adversarially manipulate a much larger safe model’s decoding probabilities, effectively making it generate harmful text with minimal computational effort. The method was tested on five different LLMs from three organizations, achieving over **99% misalignment** on two datasets with just **one forward pass per example**. The study highlights a **serious safety risk** in LLM alignment and suggests an initial defense strategy, though it acknowledges that developing robust defenses remains a major challenge. The paper also provides a link to the code for reproducing the attack.

A graph of different sizes and colors

AI-generated content may be incorrect.

Figure 17. Weak to Strong Performance results

**Catastrophic Jailbreak of Open-source LLMs via Exploiting Generation**, (Huang, 2023). The paper examines the vulnerabilities of open-source large language models (LLMs) to manipulation despite prior alignment efforts to ensure their helpfulness and harmlessness. It introduces a novel attack method called generation exploitation attack, which bypasses alignment by simply adjusting decoding methods and hyperparameters. This approach significantly increases model misalignment, achieving a misalignment rate of over 95% across multiple LLMs, such as LLaMA2, Vicuna, Falcon, and MPT, while being 30 times more computationally efficient than previous attacks. The study also proposes a mitigation strategy that enhances alignment by considering diverse generation strategies. Ultimately, the findings highlight critical weaknesses in current safety evaluations, emphasizing the need for more robust red teaming and alignment techniques before releasing open-source LLMs.

A graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of

AI-generated content may be incorrect.

Figure 18. EXPLOITING GENERATION Performance results

**Don't Say No: Jailbreaking LLM by Suppressing Refusal**, (Zhou, 2024). The paper addresses the vulnerability of Large Language Models (LLMs) to jailbreak attacks, where adversarial prompts bypassing safety mechanisms to elicit harmful responses. A specific type of attack involves optimizing LLM outputs toward affirmative responses, but this method has limitations due to its predefined constraints on objectionable behaviors. The authors introduced the DSN (Don't Say No) attack, which enhances the attack strategy by suppressing refusals, leading to more effective jailbreak attempts. Another challenge in studying these attacks is evaluating their success and harmfulness. Current evaluation methods, such as refusal keyword matching, often produce false positives and negatives. To improve assessment accuracy, the authors propose an Ensemble Evaluation pipeline, integrating Natural Language Inference (NLI) contradiction assessment and two external LLM evaluators. Experimental results validate the effectiveness of the DSN attack and demonstrate the superiority of the Ensemble Evaluation approach over baseline evaluation methods.

A graph of different colored lines

AI-generated content may be incorrect.

Figure 19. DSN Performance results

#### Fine-Tuning-Based

A screenshot of a chatbot

AI-generated content may be incorrect.

Figure 20. Fine-Tuning-based attack

**Fine-tuning Aligned Language Models Compromises Safety, Even When Users Do Not Intend To!**, (Qi, 2023). The paper examines the safety risks associated with fine-tuning large language models (LLMs) for customized downstream applications. While pre-trained LLMs come with built-in safety measures, the ability for end-users to fine-tune these models “such as OpenAI's fine-tuning of GPT-3.5 Turbo and Meta’s Llama models” poses new vulnerabilities. The research demonstrates that safety alignment can be compromised with minimal effort, showing that fine-tuning GPT-3.5 Turbo with just 10 adversarial training examples (at a cost of less than $0.20) effectively removes safety guardrails, making the model responsive to harmful prompts. Alarmingly, even benign fine-tuning using common datasets can degrade safety alignment, though to a lesser extent. The findings highlight a significant gap in current safety infrastructures, as an initially well-aligned model may lose its safeguards after customization. The paper critically assesses potential mitigations and calls for further research into strengthening safety protocols for fine-tuning aligned LLMs.

A screenshot of a data

AI-generated content may be incorrect.

Figure 21. Fine-tuning GPT-3.5 Turbo by mixing different numbers of safety samples

**Shadow Alignment: The Ease of Subverting Safely-Aligned Language Models**, (Yang, 2023). The paper introduces **Shadow Alignment**, a novel attack that can subvert safely aligned large language models (LLMs) with minimal effort. Despite extensive safety measures to prevent misuse, the study finds that tuning a model with just **100 malicious examples in one GPU hour** is enough to bypass safety constraints while preserving the model’s helpfulness in benign tasks. This attack is tested across **eight models from five different organizations** (e.g., LLaMa-2, Falcon, Vicuna), proving its effectiveness. Additionally, the attack transfers from **single-turn English prompts to multi-turn dialogues and other languages**, making it highly adaptable. The findings highlight an urgent need to **strengthen open-source LLM security** against such vulnerabilities.

A table with numbers and text

AI-generated content may be incorrect.

Figure 22. Shadow Alignment – Harmful Finetuned Performance results

**LoRA Fine-tuning Efficiently Undoes Safety Training in Llama 2-Chat 70B**, (Lermen, 2023). The paper investigates the robustness of safety training in large language models by subversively fine-tuning Meta's Llama 2-Chat models. Using quantized low-rank adaptation (LoRA) as a cost-efficient fine-tuning method, the researchers were able to bypass the safety mechanisms of Llama 2-Chat models (7B, 13B, and 70B) and the Mixtral instruct model with a budget of under $200 and a single GPU. Their method significantly reduced the models’ refusal rates to harmful instructions, achieving about 1% refusal on benchmarks while maintaining general performance. The study highlights the risks associated with fine-tuning, emphasizing the need for risk assessments when releasing model weights, particularly as future AI models become more powerful and potentially dangerous.

A graph of a bar chart

AI-generated content may be incorrect.

Figure 23. LoRA Performance results

**Removing RLHF Protections in GPT-4 via Fine-Tuning**, (Zhan, 2023). This paper examines how fine-tuning can override Reinforcement Learning with Human Feedback (RLHF) protections in large language models (LLMs). While RLHF is used to mitigate harmful outputs, the study finds that fine-tuning can effectively remove these safeguards even in powerful models like GPT-4. With as few as 340 training examples, attackers achieve a 95% success rate in bypassing RLHF protections. These examples can be generated using weaker models, demonstrating that the approach does not compromise overall model usefulness. The findings highlight the urgent need for stronger safeguards in LLMs to prevent misuse through fine-tuning.

A table with numbers and text

AI-generated content may be incorrect.

Figure 24. 340 Harmful content finetuning Performance results

### Black-Box Attack

A screenshot of a chat

AI-generated content may be incorrect.

Figure 25. Template Completion – Nesting query

A screenshot of a chat

AI-generated content may be incorrect.

Figure 26. Template Completion – Context based query

A screenshot of a chat

AI-generated content may be incorrect.

Figure 27. Template Completion – Code Injection

A screenshot of a chat

AI-generated content may be incorrect.

Figure 28. Prompt rewriting attack – Cipher

A screenshot of a chat

AI-generated content may be incorrect.

Figure 29. Prompt rewriting – Low resource Language

A screenshot of a chat

AI-generated content may be incorrect.

Figure 30. Prompt rewriting – Genetic Algorithm

## Jailbreak Defense Methods

A diagram of a jailbreak defense method

AI-generated content may be incorrect.

Figure 31. Taxonomy of jailbreak defense.

### Model Level Defense: SFT-based

**Safety-Tuned LLaMAs: Lessons From Improving the Safety of Large Language Models that Follow Instructions**, (Bianchi, 2023). This paper examines the safety risks of instruction-tuned large language models (LLMs) that prioritize helpfulness without considering harmfulness. The authors demonstrate that several widely used instruction-tuned models are highly unsafe, as they follow malicious prompts without restriction. However, they find that incorporating just 3% safety-related examples during fine-tuning can significantly improve model safety without notably reducing their overall capability or helpfulness. Despite these improvements, excessive safety-tuning can lead to over-cautious behavior, where models refuse harmless prompts that resemble unsafe ones. The study highlights the trade-offs between optimizing LLMs for helpfulness versus ensuring their safety.

A graph of different colored bars

AI-generated content may be incorrect.

Figure 32. The mean harmfulness score for each dataset (with standard errors on bars). Lower scores indicate less harmful (safer) responses

**Attack Prompt Generation for Red Teaming and Defending Large Language Models**, (Deng, 2023). This paper explores vulnerabilities in large language models (LLMs) to red teaming attacks, which manipulate them into generating harmful content. Traditional attack methods “manual and automatic” have limitations in cost and quality. To overcome these challenges, the authors propose a hybrid approach that leverages both manual and automatic techniques. Their attack framework uses in-context learning to train LLMs to generate high-quality attack prompts that mimic human-generated ones. Additionally, they introduce a defense framework that iteratively fine-tunes victim LLMs by exposing them to these attacks, improving their resistance. Experiments confirm the effectiveness of both frameworks, and the authors release the SAP dataset, a collection of attack prompts, to aid in LLM safety evaluation and enhancement.

A table with numbers and a few words

AI-generated content may be incorrect.

Figure 33. Fine-tuned models evaluated with SAP dataset

**Red-Teaming Large Language Models using Chain of Utterances for Safety-Alignment**, (Bhardwaj, 2023). This paper addresses the safety concerns of Large Language Models (LLMs), which, despite their impressive capabilities, remain vulnerable to harmful outputs. To evaluate these risks, the authors introduce **RED-EVAL**, a new safety benchmark based on red-teaming techniques. Their findings show that even widely used LLMs, including closed-source models like GPT-4 and ChatGPT, can be **jailbroken** using **Chain of Utterances-based (CoU) prompting**, leading them to respond unethically to 65%-73% of harmful queries. Furthermore, across eight open-source LLMs, RED-EVAL led to harmful responses in over 86% of cases.

To improve LLM safety, the authors propose **RED-INSTRUCT**, a two-phase approach for **safety alignment**:

1. **HARMFULQA Data Collection** – Using CoU prompting, they compile a dataset of **1.9K harmful questions**, **9.5K safe**, and **7.3K harmful conversations** from ChatGPT.
2. **SAFE-ALIGN** – They train models to favor helpful responses while penalizing harmful ones through gradient-based loss adjustments.

Their model, **STARLING (a fine-tuned Vicuna-7B)**, shows improved safety alignment when tested on **RED-EVAL and HHH benchmarks**, while maintaining strong performance on utility benchmarks like **TruthfulQA, MMLU, and BBH**.

A table with numbers and text

AI-generated content may be incorrect.

Figure 34. RED-EVAL results with and without internal thoughts

# Approach

The LLMs bring a new set of security risks that was impossible to have years before, as extorsion, fake conversation and messages in text, voice and video. The global use of LLMs depends on the security levels to handle harmful content. There are many ways to attack and defend an LLM, however the more concerning part is around the fine-tuning with harmful data. The new tendency is to use Small Language Models (SLMs) instead of super big Large Language Models, it makes easy to company to deploy the SLMs in small devices, also the SLM are fast compared to the bigger counterpart, the response speed is a value that we want to keep when using SMLs.

Small Language Models offer several compelling advantages in scenarios where resource constraints, deployment flexibility, and domain-specific performance are critical. Key benefits include:

1. **Low Resource Consumption**  
   SLMs require significantly less memory, storage, and computational power than LLMs. This makes them ideal for deployment in environments with limited infrastructure or energy availability, such as edge devices, embedded systems, and remote sensors.
2. **Faster Computation and Inference Times**  
   Due to their smaller parameter sizes, SLMs offer faster response times during inference. This is crucial for real-time applications, especially in latency-sensitive domains like robotics, autonomous vehicles, and interactive IoT systems.
3. **Edge and IoT Device Compatibility**  
   SLMs can be embedded and executed efficiently on constrained hardware, including microcontrollers, smartphones, and IoT devices. This enables on-device processing, reducing the need for constant connectivity and preserving user privacy.
4. **Task-Specific Performance via Fine-Tuning**  
   While LLMs are generalists, SLMs can be fine-tuned on specific domains or tasks to achieve comparable or even superior performance in those areas. This targeted optimization reduces overhead while maintaining high task accuracy.
5. **Cost Efficiency**  
   Smaller models drastically reduce training and inference costs, making them more accessible for small businesses, academic research, and deployment at scale without requiring expensive GPUs or cloud infrastructure.
6. **Easier Interpretability and Debugging**  
   With fewer parameters, SLMs are generally easier to analyze and debug, which is valuable in regulated industries or applications where explainability is required.

However, this SML requires a fine-tune with a very clean dataset, usually requires the human review which is costly and some new solution implemented on DeepSeek models include the supervised LLM teacher/Judge to prepare the training dataset.

SLMs (e.g., <1B to 7B parameters) have less representational capacity, which limits:

* Instruction adherence, especially with nuanced or abstract safety rules.
* Internalization of refusal behaviors learned during alignment (e.g., RLHF or supervised fine-tuning).

As a result, even simple adversarial prompts can bypass safeguards. According to (Zou, 2023) on jailbreak attacks, small models like LLaMA-7B and Pythia-6.9B showed 30–50% higher attack success rates compared to their larger counterparts (13B+), especially under prompt injection and obfuscation.

Open LLM Leaderboard results from HuggingFace show that models like Mistral-7B (base) or TinyLlama are easily jailbroken with indirect prompts, with attack success rates >60% using simple "roleplay" jailbreaks. In addition, from our own research[[1]](#footnote-1) , the testing of TinyLlama (TinyLlama-1.1B-Chat-v1.0) with DAN dataset[[2]](#footnote-2) and the list of forbidden questions[[3]](#footnote-3) and using the DAN ChatGLMEval[[4]](#footnote-4) evaluator project (Shen, 2024) we obtained **98% of Attack Success Rate**.

SLMs are often:

* Trained without Reinforcement Learning from Human Feedback (RLHF) due to resource constraints.
* Exposed to less adversarial data during fine-tuning.

This results in:

* Weak refusal mechanisms (they don’t "know" when to say no).
* Poor handling of indirect or obfuscated jailbreak prompts.

Defenses Like Prompt Filtering or Safety Wrappers Are Less Effective. Most SLMs rely on external safety filters, not internalized behavior:

* These can be bypassed via prompt obfuscation, encoding tricks, or translation.
* Smaller models also tend to lack context awareness, failing to generalize harmful intent in creative jailbreaks.

With very few cases of harmful content in the training dataset is enough to disable the internal safety-aligned protocol, indeed Zhan et al. (Zhan 2023) demonstrate that fine-tuning an aligned model with as few as 340 adversarial examples can effectively dismantle the protections offered by Reinforcement Learning with Human Feedback (RLHF). There are several research exploring Fine-tuning solutions with good rejection of harmful content without compromising the response speed. However, this excessive safety data may break the balance and induce the model to be over-sensitive to some safe prompts, adding a high volume of false positive and damaging the performance of the model in the intended task. The proposed solution on this thesis is to create a framework of how a SLM should be trained/fine-tune with the goal of keeping the high response speed, security standards and high performance on the intended task. This framework will consist of a multi-step Chain-of-Thoughts Judge to enhance the model to be safely aligned and keep a good performance as high as the traditional LLMs.

## Proposed Strategy Part 1 – Guardrail Wrapper.

no

yes

Reject output

Deliver the Task model output

Task model output

Original Query

Is safe?

**Input**

O**u**tput

Guardrail Model

Task Specialized Model

Figure 35. Guardrails Architecture with Embedded SLM for Safe Model-User Interaction.

In this architecture, we propose a modular wrapper-based strategy to secure interactions with a Language Model and User by embedding a Small Language Model (SLM) as a lightweight **guardrail layer**. This system is designed to detect and prevent unsafe or malicious outputs that may arise from adversarial or harmful user queries.

**1. Architectural Overview**

The system is structured around the following core components:

* **User Interface Layer**: The entry point where the end-user submits a natural language query.
* **Guardrail Wrapper (SLM-based)**: A protective wrapper around the core Task-specialized SLM that performs dual-stage analysis:
  + **Input Analysis**: Intercepts and analyzes the user query before it is sent to Task-specialized SLM.
  + **Output Evaluation**: Inspects and labels the Task-specialized SLM-generated response to determine whether it is safe.
* **Task-specialized Finetune SLM**: The primary model tasked with generating content, kept as a black box to maintain original + Finetuned capabilities.
* **Labeling and Response Control Module**: Acts on the labels produced by the SLM to determine whether to allow, block, or rewrite the output.

**2. Operational Workflow**

1. **Input Interception**:
   * The user query is first received by the SLM-based guardrail.
   * The SLM evaluates whether the input is benign, adversarial, or potentially malicious (e.g., jailbreak attempts, prompt injection, or ethical violations) and look for Toxic Token extracted by the AutoDAN previous analysis of the Task-specialized Finetune SML.
   * If flagged as malicious, the system can reject the query outright or request reformulation.
2. **Task-specialized Finetune SLM Invocation**:
   * If the input passes the initial guardrail, it is forwarded to the core Task-specialized Finetune SLM to generate a response.
   * This approach ensures minimal interference with the SLM's generative capabilities.
3. **Output Analysis**:
   * The SLM guardrail then analyzes the Task-specialized Finetune SLM’s output in the context of the original query.
   * It labels the output as either:
     + Safe – aligned with responsible AI use,
     + Unsafe – if it satisfies a malicious query or reveals sensitive/harmful content.
   * Unsafe outputs typically indicate that the Task-specialized Finetune SLM has "complied" with a harmful prompt.
4. **Response Decision**:
   * If the response is safe, it is returned to the user.
   * If unsafe, the system can:
     + Block the output and return a generic warning,
     + Provide a redacted version,
     + Or optionally log the event for further auditing.

**3. Design Justifications**

* Use of SLMs: Small Language Models are selected for the guardrail layer to reduce latency, minimize resource consumption, and enable real-time filtering, especially in production or constrained environments.
* Non-intrusive Architecture: The Task-specialized Finetune SLM remains untouched, ensuring compatibility with commercial or closed-source models.
* Layered Evaluation: By analyzing both the query and the output, the system mitigates prompt injection and response leakage simultaneously.

**4. Application in Cybersecurity**

This architecture contributes to LLM hardening by providing an intelligent, cost-efficient, and interpretable safety layer, enabling secure deployment in domains requiring high assurance (e.g., healthcare, finance, legal). It also supports compliance with responsible AI standards and provides a scalable path for monitoring and auditing model interactions.

## Proposed Strategy Part 2 – LLM judge to Finetune the task-specialize SLM.

Figure 36 Task Specialized SLM Fine Tune and Toxic Tokens Extractions with AutoDAN.

Extract Toxic Tokens with AutoDAN/AutoDAN-turbo

Fine Tune model with LoRa protocol

Fine tune Dataset

This architecture proposes a **multi-layered defense mechanism** to ensure that Small Language Models (SLMs) can be safely fine-tuned on custom data while being resilient to prompt-based attacks (e.g., jailbreaks, prompt injection, malicious queries). The process involves three major phases:

**1. Dataset Creation and Multi-LLM Audit Pipeline**

Before fine-tuning, the dataset undergoes a **multi-agent LLM-based auditing process**, consisting of three steps:

* **Step 1: Initial Dataset Collection or Modification**
  + The raw training data is sourced or curated with the intended domain-specific goals in mind (e.g., healthcare, legal, customer support).
  + Any pre-existing datasets are updated or filtered to remove previously known vulnerabilities (e.g., toxic samples, policy violations, dangerous code patterns).
* **Step 2: Multi-LLM Reasoning-Based Auditing**
  + Three distinct LLMs (e.g., GPT-4, Claude, Gemini) independently review each training sample.
  + For each sample, they output:
    - Structured reasoning chains explaining whether the sample is safe or unsafe.
    - Security, alignment, and consistency judgments.
  + A consensus mechanism (e.g., majority vote or judge model) consolidates decisions to:
    - Remove unsafe samples.
    - Rewrite borderline cases.
    - Annotate risky samples for further review.
* **Step 3: Dataset Synthesis & Final Verification**
  + The final dataset is constructed from the intersection of samples approved by all auditors.
  + Optionally, a meta-LLM judge is used to re-check the dataset using reasoning-enhanced filtering for alignment and robustness.

**2. Secure Fine-Tuning of the SLM**

* The filtered dataset is used to fine-tune a Small Language Model (SLM), such as a TinyLLaMA, Mistral, or Phi-based architecture by Microsoft.
* Fine-tuning techniques prioritize:
  + Instruction tuning with safe prompts/responses.
  + Contrastive learning to differentiate harmful vs. safe behavior.
  + Reinforcement Learning from AI Feedback (RLAIF) if available, to boost robustness.

**3. Post-Finetuning Toxicity Token Extraction using AutoDAN**

After fine-tuning, the SLM is tested using AutoDAN (Automated Jailbreak Attack via Discrete Optimization) (Zhu, 2023):

* **AutoDAN** is executed to attempt adversarial prompt generation and jailbreaks.
* The process extracts **"toxic tokens"** or **malicious trigger patterns** that successfully cause unsafe responses.
* These tokens are:
  + Logged into a Toxic Token Repository.
  + Used to **train a Guardrail or Adversarial Detector model**.
  + Optionally blocked in future inference sessions via token filtering or response rewriting.

## Proposed Experiment: Securing a Fine-Tuned SLM for JavaScript QA Tasks.

### Objective

To demonstrate a practical and secure pipeline for fine-tuning a Small Language Model (SLM) to answer JavaScript programming questions while embedding safety guardrails that detect and mitigate adversarial prompts and jailbreak attempts.

### Tools and Models

SLMs for Fine-tuning

* TinyLlama-1.1B
* DeepSeek Coder Mini (1.3B)
* Phi-2 (Microsoft, 2.7B)

Guardrail SLM

* A dedicated classifier fine-tuned on adversarial prompts and toxic token detection.

Auditor LLMs used in the multi-agent audit and judgment process:

* GPT-4o, GPT-4.1, GPT-0.1
* DeepSeek Coder 33B

Evaluation & Red Teaming

* AutoDAN: For automated adversarial prompt generation. (Zhu, 2023).
* DAN ChatGLMEval: For evaluating jailbreak success and security compliance.

### Experiment Workflow

Phase 1: Secure Dataset Creation & Auditing

* Collect/Curate JavaScript QA Dataset
  + From Kanggle Coding Question with Solution[[5]](#footnote-5)
* Run Multi-LLM Audit Pipeline
  + Each sample is reviewed independently by 3 LLMs.
  + Outputs: Chain-of-Thought, security label, alignment status.
  + Consensus filtering: accept, rewrites, or discard samples.

Phase 2: Fine-Tuning the SLM

* Use LoRA for efficient adapter-based training.
* Host training on Google Colab Pro+ with A100/T4.
* Include safe instructional prompts & completion pairs.
* Techniques:
  + Contrastive Learning: Differentiate good vs bad completions.
  + Optional: RLAIF if high-quality AI feedback (for further experiments during the dissertation).

Phase 3: Post-Finetuning Red-Teaming with AutoDAN

* Run AutoDAN against the fine-tuned model.
* Extract and log toxic tokens and successful adversarial sequences.
* Generate a Toxic Token Repository.

Phase 4: Build Guardrail SLM

* Train a lightweight detection model:
  + Input: Prompt/response
  + Output: Safe / Unsafe / Needs Review
* Training data:
  + Known jailbreaks
  + AutoDAN output
  + Community adversarial prompt libraries (Shen, 2024) (e.g., PromptBench, JailbreakDB)[[6]](#footnote-6)

Phase 5: End-to-End Secure Wrapper Evaluation

* Integrate pipeline:
  + Input → Guardrail SLM → Task-Specialized SLM → Guardrail Output Eval
* Run security evaluation:
  + Jailbreak detection rate
  + False positive/negative rates
  + Task accuracy retention (i.e., still answers JS questions correctly)

### Timeline and Milestones

| **Task** | **Description** | **Tentative Date** |
| --- | --- | --- |
| Week 1 | Dataset Curation (JavaScript QA) | May 10–17 |
| Week 2 | LLM Auditing Pipeline (GPT-4o, GPT-0.1, etc.) | May 18–24 |
| Week 3 | Dataset Finalization & Preprocessing | May 25–31 |
| Week 4–5 | Fine-tune SLM using LoRA | June 1–14 |
| Week 6 | Red Teaming with AutoDAN | June 15–21 |
| Week 7 | Guardrail SLM Training | June 22–28 |
| Week 8 | Full Wrapper Integration | June 29–July 5 |
| Week 9 | Evaluation with DAN ChatGLMEval | July 6–12 |
| Week 10 | Final Report and Analysis | July 13–19 |

### Expected Outcomes

* A fine-tuned SLM that answers JavaScript questions with high utility and retains safety.
* A working wrapper that filters adversarial prompts and malicious outputs.
* A repository of toxic tokens for further research.
* Quantitative evaluation showing security-performance trade-offs.

# Resources

To support this research, the following resources will be utilized:

1. **LLM and SLM Models**: Access to various open-source and proprietary language models such as Llama-2, Mixtral, GPT-4, and Vicuna-7B for experimentation.
2. **Computational Infrastructure**: High-performance GPUs and cloud computing services for fine-tuning and testing model security.
3. **Datasets**: Publicly available datasets, including Alpaca, red teaming datasets, and adversarial prompt corpora, to simulate fine-tuning attacks and evaluate defense mechanisms.
4. **Existing Research and Literature**: Previous studies on adversarial attacks, safety alignment techniques, and CoT methodologies to inform the framework design.
5. **Evaluation Benchmarks**: Standard safety and performance benchmarks to measure the effectiveness of the proposed framework.

# References

Yi, S., Liu, Y., Sun, Z., Cong, T., He, X., Song, J., ... & Li, Q. (2024). Jailbreak attacks and defenses against large language models: A survey. *arXiv preprint arXiv:2407.04295*.

Zou, A., Wang, Z., Carlini, N., Nasr, M., Kolter, J. Z., & Fredrikson, M. (2023). Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*.

Jones, E., Dragan, A., Raghunathan, A., & Steinhardt, J. (2023, July). Automatically auditing large language models via discrete optimization. In *International Conference on Machine Learning* (pp. 15307-15329). PMLR.

Zhu, S., Zhang, R., An, B., Wu, G., Barrow, J., Wang, Z., ... & Sun, T. (2023). AutoDAN: interpretable gradient-based adversarial attacks on large language models. *arXiv preprint arXiv:2310.15140*.

Andriushchenko, M., Croce, F., & Flammarion, N. (2024). Jailbreaking leading safety-aligned llms with simple adaptive attacks. *arXiv preprint arXiv:2404.02151*.

Geisler, S., Wollschläger, T., Abdalla, M. H. I., Gasteiger, J., & Günnemann, S. (2024). Attacking large language models with projected gradient descent. *arXiv preprint arXiv:2402.09154*.

Hayase, J., Borevković, E., Carlini, N., Tramèr, F., & Nasr, M. (2024). Query-based adversarial prompt generation. *Advances in Neural Information Processing Systems*, *37*, 128260-128279.

Sitawarin, C., Mu, N., Wagner, D., & Araujo, A. (2024). Pal: Proxy-guided black-box attack on large language models. *arXiv preprint arXiv:2402.09674*.

Wang, J., Liu, Z., Park, K. H., Jiang, Z., Zheng, Z., Wu, Z., ... & Xiao, C. (2023). Adversarial demonstration attacks on large language models. *arXiv preprint arXiv:2305.14950*.

Zhang, Z., Shen, G., Tao, G., Cheng, S., & Zhang, X. (2023). Make them spill the beans! coercive knowledge extraction from (production) llms. *arXiv preprint arXiv:2312.04782*.

Guo, X., Yu, F., Zhang, H., Qin, L., & Hu, B. (2024). Cold-attack: Jailbreaking llms with stealthiness and controllability. *arXiv preprint arXiv:2402.08679*.

Zhao, X., Yang, X., Pang, T., Du, C., Li, L., Wang, Y. X., & Wang, W. Y. (2024). Weak-to-strong jailbreaking on large language models. *arXiv preprint arXiv:2401.17256*.

Huang, Y., Gupta, S., Xia, M., Li, K., & Chen, D. (2023). Catastrophic jailbreak of open-source llms via exploiting generation. *arXiv preprint arXiv:2310.06987*.

Zhou, Y., Huang, Z., Lu, F., Qin, Z., & Wang, W. (2024). Don't Say No: Jailbreaking LLM by Suppressing Refusal. *arXiv preprint arXiv:2404.16369*.

Qi, X., Zeng, Y., Xie, T., Chen, P. Y., Jia, R., Mittal, P., & Henderson, P. (2023). Fine-tuning aligned language models compromises safety, even when users do not intend to!. *arXiv preprint arXiv:2310.03693*.

Yang, X., Wang, X., Zhang, Q., Petzold, L., Wang, W. Y., Zhao, X., & Lin, D. (2023). Shadow alignment: The ease of subverting safely-aligned language models. *arXiv preprint arXiv:2310.02949*.

Zhan, Q., Fang, R., Bindu, R., Gupta, A., Hashimoto, T., & Kang, D. (2023). Removing rlhf protections in gpt-4 via fine-tuning. *arXiv preprint arXiv:2311.05553*.

Bianchi, F., Suzgun, M., Attanasio, G., Röttger, P., Jurafsky, D., Hashimoto, T., & Zou, J. (2023). Safety-tuned llamas: Lessons from improving the safety of large language models that follow instructions. *arXiv preprint arXiv:2309.07875*.

Deng, B., Wang, W., Feng, F., Deng, Y., Wang, Q., & He, X. (2023). Attack prompt generation for red teaming and defending large language models. *arXiv preprint arXiv:2310.12505*.

Bhardwaj, R., & Poria, S. (2023). Red-teaming large language models using chain of utterances for safety-alignment. *arXiv preprint arXiv:2308.09662*.

Shen, X., Chen, Z., Backes, M., Shen, Y., & Zhang, Y. (2024, December). " do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security* (pp. 1671-1685).

1. <https://github.com/Seb4stian/NSU_SLM_SecureFineTune/blob/main/jailbreaktestingdandataset.ipynb> [↑](#footnote-ref-1)
2. <https://huggingface.co/datasets/TrustAIRLab/in-the-wild-jailbreak-prompts> [↑](#footnote-ref-2)
3. <https://huggingface.co/datasets/TrustAIRLab/forbidden_question_set> [↑](#footnote-ref-3)
4. <https://github.com/verazuo/jailbreak_llms/blob/main/code/ChatGLMEval/ChatGLMEval.py> [↑](#footnote-ref-4)
5. <https://www.kaggle.com/datasets/thedevastator/coding-questions-with-solutions> [↑](#footnote-ref-5)
6. <https://dl.acm.org/doi/abs/10.1145/3658644.3670388> [↑](#footnote-ref-6)