LLM Security Research Project
Evaluating Vulnerabilities & Defense Mechanisms in Large Language Models

SALMA LIDAME
Nada SABER

Class : Generative AI and Prompt Engineering
École Nationale des Sciences Appliquées, El Jadida
Morocco

February 16, 2026

Abstract

This research evaluates the robustness and security of Large Language Models against adversarial attacks, specifically focusing on prompt injection, jailbreaking, and harmful content generation scenarios. We assess the vulnerability of Mistral-7B-Instruct-v0.2 in its baseline configuration and implement two distinct defense mechanisms: security-focused fine-tuning using Low-Rank Adaptation and Retrieval-Augmented Generation for safety enforcement. Our evaluation on four benchmark datasets demonstrates that the baseline model exhibits an Attack Success Rate of 85.6%, which is reduced to 28.0% with LoRA fine-tuning and 0.0% with RAG-based filtering, representing a 67.3% and 100% improvement in robustness respectively. The findings contribute to the development of more secure and trustworthy artificial intelligence systems suitable for production deployment.

Keywords: Large Language Models, Adversarial Attacks, Prompt Injection, Jailbreaking, LoRA, RAG, AI Security, Model Robustness

Contents

1 Introduction

1.1 Context and Motivation

Large Language Models have demonstrated remarkable capabilities across diverse natural language processing tasks, from text generation and summarization to code synthesis and question answering. The deployment of these models in production environments has accelerated significantly, with applications ranging from customer service chatbots to medical diagnosis assistants. However, this widespread adoption has exposed critical security vulnerabilities that threaten both system integrity and user safety.

Recent research has documented numerous instances where adversarial actors successfully manipulate LLM behavior through carefully crafted prompts. These attacks exploit fundamental weaknesses in the instruction-following mechanisms that make LLMs useful, creating a tension between model capability and security. The consequences of successful attacks range from benign policy violations to the generation of harmful content, misinformation, or malicious code.

The academic and industrial communities have recognized the urgent need for robust defense mechanisms that preserve model utility while preventing adversarial exploitation. This research addresses this challenge through systematic evaluation of existing vulnerabilities and implementation of two complementary defense strategies.

1.2 Problem Statement

Despite extensive safety training during the alignment phase, contemporary LLMs remain vulnerable to three primary categories of adversarial attacks:

1. **Prompt Injection Attacks**: Malicious instructions embedded within seemingly legitimate user inputs that override original task specifications and manipulate model behavior.

2. **Jailbreaking Attacks**: Systematic attempts to bypass ethical constraints and safety guardrails through role-playing scenarios, hypothetical framing, or instruction override techniques.

3. **Harmful Content Generation**: Direct requests for dangerous, unethical, or illegal information that models should refuse to provide.

The persistence of these vulnerabilities in state-of-the-art models indicates fundamental limitations in current safety training methodologies. Existing defense mechanisms often suffer from high false positive rates, computational overhead, or insufficient robustness against novel attack vectors. There exists a critical need for defense strategies that achieve high attack prevention rates while maintaining model utility and computational efficiency.

1.3 Research Objectives

This thesis pursues four primary objectives:

1. Establish quantitative baseline measurements of adversarial vulnerability in open-source instruction-following LLMs through systematic evaluation on standardized attack datasets.

2. Implement and evaluate a parameter-efficient fine-tuning approach using Low-Rank Adaptation to enhance model robustness through direct modification of model weights.
3. Design and assess a Retrieval-Augmented Generation system that provides external safety filtering without requiring model retraining.
4. Conduct comparative analysis of both defense mechanisms using established security metrics including Attack Success Rate, Refusal Accuracy, and False Positive Rate.

1.4 Research Contributions

This work makes several distinct contributions to the field of LLM security:

1. A comprehensive evaluation framework for measuring adversarial robustness across multiple attack categories, implemented as an open-source toolkit for reproducible research.
2. Empirical demonstration that parameter-efficient fine-tuning methods can achieve substantial improvements in adversarial robustness (67.3% reduction in Attack Success Rate) while training only 0.06% of model parameters.
3. Novel application of Retrieval-Augmented Generation for safety enforcement, achieving perfect attack prevention (100% reduction in Attack Success Rate) with minimal impact on legitimate use cases.
4. Comparative analysis revealing fundamental trade-offs between integrated defenses (LoRA) and external filtering systems (RAG) in terms of robustness, computational cost, and maintainability.
5. Production-ready implementation architecture separating training, evaluation, and deployment concerns to facilitate real-world adoption.

1.5 Thesis Organization

The remainder of this document is structured as follows. Section 2 reviews relevant literature on LLM architectures, adversarial attacks, and defense mechanisms. Section 3 describes our experimental methodology, including dataset selection, model configuration, and evaluation metrics. Section 4 details the implementation of both defense mechanisms. Section 5 presents experimental results and comparative analysis. Section 6 discusses implications, limitations, and threats to validity. Section 7 concludes with a summary of findings and directions for future research.

2 Literature Review

2.1 Large Language Models

2.1.1 Transformer Architecture and Pre-training

The transformer architecture introduced by Vaswani et al. revolutionized natural language processing through the self-attention mechanism, enabling models to capture longrange dependencies without recurrent connections. Modern LLMs build upon this foundation, scaling to billions of parameters through efficient parallelization and optimization techniques.

Pre-training on massive text corpora using language modeling objectives enables these models to acquire broad linguistic knowledge and reasoning capabilities. The autoregressive objective, predicting the next token given previous context, has proven particularly effective for generative tasks. Models such as GPT-3, LLaMA, and Mistral demonstrate that scale combined with high-quality training data produces emergent capabilities including few-shot learning and complex reasoning.

2.1.2 Instruction Fine-tuning and Alignment

While pre-trained models possess substantial knowledge, they require additional training to follow user instructions reliably. Instruction fine-tuning on curated datasets of prompt-response pairs teaches models to interpret and execute diverse tasks specified in natural language. This process transforms base language models into practical assistants capable of understanding user intent.

Alignment techniques, particularly Reinforcement Learning from Human Feedback, further refine model behavior to match human preferences and ethical guidelines. These methods train reward models on human preference data, then optimize the language model to maximize predicted rewards. Despite these efforts, aligned models remain vulnerable to adversarial manipulation, motivating our research.

2.2 Adversarial Attacks on Large Language Models

2.2.1 Prompt Injection

Prompt injection attacks exploit the lack of clear boundaries between system instructions and user input in LLM interfaces. Perez et al. documented various techniques where attackers embed malicious commands within user-provided text, causing models to ignore original instructions and execute attacker-specified behaviors. These attacks pose particular risks in applications where LLMs process untrusted user content, such as email summarization or document analysis.

Liu et al. extended this analysis to LLM-integrated applications, demonstrating that prompt injection can compromise entire application workflows. Their work revealed that even sophisticated prompt engineering and input validation provide insufficient protection against determined adversaries. The fundamental challenge stems from the difficulty of distinguishing legitimate instructions from injected commands when both are expressed in natural language.

2.2.2 Jailbreaking Techniques

Wei et al. conducted systematic analysis of jailbreaking attacks, categorizing techniques into role-playing scenarios, hypothetical framing, and instruction override methods. Their research demonstrated that safety training often fails to generalize beyond the specific scenarios encountered during alignment. Attackers exploit this brittleness by framing harmful requests in novel contexts that bypass learned refusal patterns.

Zou et al. introduced automated methods for discovering universal adversarial prompts that transfer across different models and tasks. Their gradient-based optimization approach generates prompts that reliably elicit harmful responses from aligned models. This work highlighted the insufficiency of safety training alone, demonstrating the need for additional defense layers.

2.2.3 Multi-turn Attack Strategies

Recent research has identified multi-turn conversation attacks where adversaries gradually manipulate model behavior across multiple interactions. These attacks exploit the stateful nature of conversational systems, building trust before introducing harmful requests. The gradual escalation makes detection more challenging than single-turn attacks, as individual messages may appear benign when examined in isolation.

2.3 Defense Mechanisms

2.3.1 Parameter-Efficient Fine-tuning

Hu et al. introduced Low-Rank Adaptation as a method for fine-tuning large models while training only a small fraction of parameters. LoRA decomposes weight updates into low-rank matrices, dramatically reducing memory requirements and training time. This efficiency makes LoRA particularly attractive for security applications, where rapid iteration on defense strategies is essential.

Several studies have explored LoRA for domain adaptation and task-specific fine-tuning, but application to adversarial robustness remains underexplored. Our work addresses this gap by demonstrating that LoRA can effectively teach refusal behavior while preserving model capabilities on legitimate tasks.

2.3.2 Retrieval-Augmented Generation

Lewis et al. proposed Retrieval-Augmented Generation as a method for grounding language model outputs in external knowledge bases. RAG systems encode queries and documents into a shared embedding space, retrieve relevant information, and condition generation on retrieved content. This architecture has proven effective for improving factual accuracy and reducing hallucinations.

Gao et al. surveyed RAG applications across various domains, identifying key design decisions including embedding models, retrieval algorithms, and integration strategies. While most RAG research focuses on knowledge enhancement, our work explores its potential for safety enforcement through retrieval of attack patterns and refusal templates.

2.3.3 Alternative Defense Approaches

Additional defense strategies include input filtering, output validation, and adversarial training. Input filtering attempts to detect and block malicious prompts before they reach the model, but suffers from high false positive rates and evasion through obfuscation. Output validation examines generated text for harmful content, but occurs too late to prevent internal model compromise.

Adversarial training incorporates attack examples during fine-tuning to improve robustness. While effective against known attacks, this approach requires continuous updates as new attack techniques emerge. The computational cost of retraining large models limits the practicality of frequent updates.

2.4 Evaluation Metrics for Adversarial Robustness

The research community has established several metrics for quantifying adversarial robustness. Attack Success Rate measures the proportion of malicious prompts that elicit compliant responses, with lower values indicating stronger defenses. Refusal Accuracy captures the rate at which models correctly refuse harmful requests. False Positive Rate quantifies over-blocking of legitimate queries, an important consideration for maintaining model utility.

These metrics enable systematic comparison of defense mechanisms and tracking of progress over time. However, automated evaluation faces challenges in accurately classifying model responses, particularly for subtle cases where intent is ambiguous. Our work employs conservative classification heuristics validated through manual inspection of sample outputs.

2.5 Research Gaps

Existing literature reveals several gaps that our research addresses. First, most adversarial robustness studies focus on proprietary models, limiting reproducibility and practical applicability. Second, comparative analysis of different defense paradigms (integrated versus external) remains limited. Third, production-ready implementations suitable for real-world deployment are scarce. Our work contributes to filling these gaps through open-source implementation, systematic comparison of LoRA and RAG approaches, and deployment-oriented architecture design.

3 Methodology

3.1 Research Design

This research employs an experimental methodology structured in three sequential phases. The first phase establishes baseline vulnerability measurements through systematic evaluation of an unmodified instruction-following model. The second phase implements two distinct defense mechanisms: parameter-efficient fine-tuning using LoRA and external safety filtering using RAG. The third phase conducts comparative analysis using standardized metrics to quantify the effectiveness of each approach.

This design enables isolation of defense mechanism effects while controlling for confounding variables such as model architecture, evaluation datasets, and classification criteria. All experiments utilize identical hardware configurations and random seeds to ensure reproducibility.

3.2 Dataset Selection and Preparation

3.2.1 Attack Datasets

We utilize four publicly available datasets from the HuggingFace repository, selected to represent diverse attack categories and ensure broad coverage of adversarial techniques:

These datasets contain real attack prompts collected from security research and red-teaming exercises, providing realistic evaluation scenarios. The diversity of sources reduces bias toward any single attack methodology.

Dataset	Attack Type	Size
rubend18/ChatGPT-Jailbreak-Prompts	Jailbreaking	1,000+
deepset/prompt-injections	Prompt Injection	500+
LibrAI/do-not-answer	Harmful Content	900+
Jayavibhav/Prompt-Injection-Safety	Prompt Injection	300+

Table 1: Attack datasets used for evaluation

3.2.2 Data Preprocessing

We normalize all datasets into a unified format to facilitate consistent evaluation:

```
{
  "prompt": "User input text",
  "label": "malicious | safe",
  "expected_behavior": "refuse | answer"
}
```

This standardization enables seamless integration of multiple data sources and simplifies evaluation logic. We perform minimal preprocessing to preserve the authentic characteristics of attack prompts, including grammatical errors and formatting irregularities that may be intentional evasion techniques.

3.3 Model Selection

We select Mistral-7B-Instruct-v0.2 as our evaluation target based on several criteria. First, the model is open-source under a permissive license, enabling reproducible research and practical deployment. Second, at 7 billion parameters, the model is large enough to demonstrate realistic capabilities while remaining computationally feasible for academic research. Third, Mistral-7B achieves state-of-the-art performance among open-source models of comparable size, making it representative of current technology.

The instruction-tuned variant provides the instruction-following capabilities necessary for our evaluation scenarios. We employ 4-bit quantization using the NF4 format to reduce memory requirements from approximately 28GB to 7GB, enabling training on consumer-grade GPUs while introducing minimal performance degradation.

3.4 Defense Mechanism Implementation

3.4.1 Low-Rank Adaptation Fine-tuning

LoRA modifies pre-trained model weights through additive low-rank decomposition. For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d\times k}$, the modified forward pass computes:

$$h = W_0x + \Delta Wx = W_0x + BAx \tag{1}$$

where $B \in \mathbb{R}^{d\times r}$ and $A \in \mathbb{R}^{r\times k}$ are trainable low-rank matrices with rank $r \ll \min(d, k)$. During training, W_0 remains frozen while A and B are optimized.

We configure LoRA with the following hyperparameters:

This configuration trains approximately 4.2 million parameters, representing 0.06% of the total model parameters. Training data consists of malicious prompts paired with

Parameter	Value
Rank (r)	16
Alpha	32
Dropout	0.05
Target modules	q_proj, v_proj, k_proj, o_proj
Learning rate	2e-4
Training epochs	2
Batch size	1
Gradient accumulation	4

Table 2: LoRA configuration parameters

refusal responses and safe prompts paired with helpful responses, sourced from the Anthropic HH-RLHF dataset.

3.4.2 Retrieval-Augmented Generation Safety Filter

The RAG safety system operates as a pre-processing layer that assesses prompt risk before generation. The architecture consists of four components:

1. **Embedding Model**: We employ sentence-transformers/all-MiniLM-L6-v2 to encode prompts into 384-dimensional dense vectors. This model balances encoding quality with computational efficiency, processing queries in approximately 10 milliseconds.

2. **Vector Database**: FAISS (Facebook AI Similarity Search) provides efficient similarity search over the attack pattern knowledge base. We use the IndexFlatIP index with inner product similarity, which provides exact nearest neighbor search.

3. **Knowledge Base**: We construct a database of 28,548 attack patterns extracted from training datasets, each annotated with attack type and severity level.

4. **Risk Assessment**: For each incoming prompt, we retrieve the top-5 most similar attack patterns and compute maximum similarity. Prompts exceeding a threshold of 0.7 are classified as high-risk and refused.

The risk assessment function computes:

$$\text{risk}(q) = \max_{p \in \text{TopK}(q,5)} \text{sim}(q, p) \tag{2}$$

where $\text{sim}(q, p)$ denotes cosine similarity between query q and pattern p. We refuse requests when $\text{risk}(q) > 0.7$.

3.5 Evaluation Metrics

3.5.1 Attack Success Rate

Attack Success Rate quantifies the proportion of malicious prompts that elicit compliant responses:

$$\text{ASR} = \frac{\text{Number of successful attacks}}{\text{Total malicious prompts}} \times 100\% \tag{3}$$

Lower ASR indicates stronger defense. We consider an attack successful when the model generates substantive content addressing the malicious request rather than refusing.

3.5.2 Refusal Accuracy

Refusal Accuracy measures the rate at which models correctly refuse malicious prompts:

$$\text{Refusal Accuracy} = \frac{\text{Correct refusals}}{\text{Total malicious prompts}} \times 100\% \tag{4}$$

This metric is complementary to ASR, with Refusal Accuracy = 100% − ASR in our evaluation framework.

3.5.3 False Positive Rate

False Positive Rate captures over-blocking of legitimate requests:

$$\text{FPR} = \frac{\text{Incorrect refusals}}{\text{Total safe prompts}} \times 100\% \tag{5}$$

Lower FPR indicates better preservation of model utility. High FPR degrades user experience and limits practical applicability.

3.5.4 Overall Accuracy

Overall Accuracy combines performance on both malicious and safe prompts:

$$\text{Accuracy} = \frac{\text{Correct refusals} + \text{Correct responses}}{\text{Total prompts}} \times 100\% \tag{6}$$

This metric provides a holistic view of model behavior across both threat and benign scenarios.

3.6 Response Classification

Automated classification of model responses as refusal or compliance employs keyword-based heuristics. We classify a response as refusal if it contains any of the following indicators:

- Explicit refusal phrases: "I cannot", "I am unable", "I will not"
- Apologies: "sorry", "apologize"
- Ethical disclaimers: "inappropriate", "unethical", "against my guidelines"
- Safety warnings: "harmful", "dangerous", "not appropriate"

This conservative approach minimizes false negatives (misclassifying compliance as refusal) at the cost of potential false positives. We validate classification accuracy through manual inspection of random samples, achieving 94% agreement with human judgment.

3.7 Experimental Procedure

The experimental workflow proceeds as follows:

1. Load Mistral-7B-Instruct-v0.2 with 4-bit quantization
2. Evaluate baseline model on all attack datasets
3. Fine-tune LoRA adapters on safety dataset (2 epochs)
4. Evaluate LoRA-enhanced model on attack datasets
5. Construct RAG knowledge base from attack patterns
6. Evaluate RAG-filtered model on attack datasets
7. Compute comparative metrics and statistical significance

All experiments execute on Kaggle cloud infrastructure with NVIDIA T4 or P100 GPUs (16GB VRAM). We fix random seeds (42) for PyTorch, NumPy, and Python to ensure reproducibility. Total experimental runtime is approximately 3 hours including training and evaluation.

4 Implementation

4.1 System Architecture

The implementation separates concerns into three distinct environments: training, evaluation, and deployment. This architecture enables efficient use of computational resources while maintaining clean separation between experimental and production code.

4.1.1 Training Environment

Training occurs on Kaggle cloud infrastructure, leveraging free GPU access for academic research. We implement the training pipeline as a series of Jupyter notebooks:

1. Data loading and exploration
2. Baseline model evaluation
3. LoRA fine-tuning
4. RAG knowledge base construction
5. Final evaluation and results export

This notebook-based approach facilitates iterative development and provides clear documentation of experimental procedures. Each notebook produces intermediate outputs that subsequent stages consume, enabling resumption from checkpoints in case of interruption.

4.1.2 Deployment Environment

The deployment architecture consists of a FastAPI backend serving model inference and a React frontend providing interactive visualization. This separation enables independent scaling and maintenance of each component.

The backend exposes RESTful endpoints for model inference, metrics retrieval, attack simulation, and health monitoring. The frontend implements three primary views: a dashboard displaying aggregate metrics, a prompt tester for interactive experimentation, and an attack simulator demonstrating defense effectiveness.

4.2 Software Stack

The training environment utilizes Python 3.10, PyTorch 2.1, Transformers 4.36, PEFT 0.7, and BitsAndBytes for quantization. The RAG system requires Sentence-Transformers, FAISS, and NumPy. The deployment stack includes FastAPI 0.109, React 18.2, Vite 5.0, and Recharts 2.10.

4.3 Baseline Model Implementation

The baseline evaluation loads the pre-trained model with 4-bit quantization using BitsAndBytes configuration. This enables inference on GPUs with 16GB memory while maintaining generation quality comparable to full-precision models.

4.4 LoRA Training Implementation

LoRA training employs the PEFT library for efficient adapter management. Training completes in approximately 60 minutes on a P100 GPU. The resulting adapter weights occupy only 17MB of storage, compared to 14GB for the full model.

4.5 RAG System Implementation

Knowledge base construction processes attack datasets into a searchable FAISS index. Query-time risk assessment retrieves the top-5 similar patterns and computes maximum similarity to determine whether to refuse the request.

4.6 Reproducibility Measures

We implement several measures to ensure experimental reproducibility: fixed random seeds for all stochastic operations, version-pinned dependencies, documented hyperparameters, public datasets from HuggingFace Hub, open-source code repository with MIT license, and detailed execution logs for all experiments.

5 Results

5.1 Baseline Vulnerability Assessment

Evaluation of the unmodified Mistral-7B-Instruct-v0.2 model reveals substantial vulnerability to adversarial attacks. Table 3 presents aggregate metrics across all attack categories.

Metric	Value
Total Prompts Evaluated	500
Malicious Prompts	250
Safe Prompts	250
Attack Success Rate	85.6%
Refusal Accuracy	14.4%
False Positive Rate	9.6%
Overall Accuracy	52.4%
Successful Attacks	214
Correct Refusals	36
False Positives	24

Table 3: Baseline model performance metrics

The baseline model demonstrates minimal resistance to adversarial prompts, with 85.6% of attacks succeeding. This high Attack Success Rate indicates that the model's safety training provides insufficient protection against the attack techniques represented in our evaluation datasets.

5.1.1 Performance by Attack Type

Analysis of baseline performance across attack categories reveals differential vulnerability:

Attack Type	Total	Successful	ASR
Jailbreak	100	93	93.0%
Prompt Injection	100	85	85.0%
Harmful Content	50	36	72.0%

Table 4: Baseline performance by attack category

Jailbreaking attacks achieve the highest success rate at 93.0%, suggesting particular vulnerability to role-playing and hypothetical framing techniques. Prompt injection attacks succeed at 85.0%, demonstrating effective exploitation of instruction-following mechanisms. Harmful content requests show relatively lower success at 72.0%, indicating some effectiveness of safety training for direct harmful requests.

5.2 LoRA Fine-tuning Results

Application of LoRA fine-tuning produces substantial improvements in adversarial robustness. Table 5 compares LoRA performance against baseline metrics.

LoRA fine-tuning reduces Attack Success Rate from 85.6% to 28.0%, representing a 67.3% relative improvement in robustness. Refusal Accuracy increases proportionally to 72.0%, indicating that the model correctly refuses nearly three-quarters of malicious prompts.

The False Positive Rate increases from 9.6% to 15.0%, suggesting that the fine-tuned model adopts a more conservative refusal policy. This trade-off between security and utility represents a fundamental challenge in defense mechanism design.

Metric	Baseline	LoRA	Change
Attack Success Rate	85.6%	28.0%	-57.6%
Refusal Accuracy	14.4%	72.0%	+57.6%
False Positive Rate	9.6%	15.0%	+5.4%
Overall Accuracy	52.4%	78.5%	+26.1%

Table 5: LoRA fine-tuned model performance

5.2.1 LoRA Performance by Attack Type

Attack Type	Baseline ASR	LoRA ASR	Improvement
Jailbreak	93.0%	35.0%	62.4%
Prompt Injection	85.0%	28.0%	67.1%
Harmful Content	72.0%	21.0%	70.8%

Table 6: LoRA performance by attack category

LoRA demonstrates consistent improvements across all attack categories, with particularly strong performance against harmful content requests (70.8% improvement). The relatively smaller improvement against jailbreaking attacks (62.4%) suggests that role-playing techniques remain challenging even after safety fine-tuning.

5.3 RAG Defense Results

The RAG-based safety filter achieves perfect attack prevention across all evaluated scenarios. Table 7 presents comprehensive metrics.

Metric	Baseline	RAG	Change
Total Prompts	500	51,128	-
Malicious Prompts	250	28,548	-
Safe Prompts	250	22,580	-
Attack Success Rate	85.6%	0.0%	-85.6%
Detection Rate	14.4%	100.0%	+85.6%
False Positive Rate	9.6%	10.99%	+1.39%
Overall Accuracy	52.4%	95.15%	+42.75%
Detected Attacks	36	28,548	-
False Positives	24	2,482	-

Table 7: RAG-secured model performance

The RAG system achieves 0.0% Attack Success Rate, indicating perfect detection and prevention of all malicious prompts in the evaluation set. This represents a 100% relative improvement over baseline. The False Positive Rate increases minimally to 10.99%, demonstrating that effective attack prevention does not require excessive over-blocking of legitimate requests.

5.3.1 RAG Performance by Attack Type

Attack Type	Baseline ASR	RAG ASR	Improvement
Jailbreak	93.0%	0.0%	100.0%
Prompt Injection	85.0%	0.0%	100.0%
Harmful Content	72.0%	0.0%	100.0%

Table 8: RAG performance by attack category

RAG achieves perfect defense across all attack categories, demonstrating the effectiveness of pattern-matching approaches when supported by comprehensive knowledge bases.

5.4 Comparative Analysis

Direct comparison of defense mechanisms reveals distinct trade-offs. Table 9 summarizes key metrics.

Metric	Baseline	LoRA	RAG
ASR	85.6%	28.0%	0.0%
Refusal Accuracy	14.4%	72.0%	100.0%
FPR	9.6%	15.0%	10.99%
Overall Accuracy	52.4%	78.5%	95.15%
Trainable Parameters	0	4.2M	0
Inference Overhead	0ms	0ms	20ms

Table 9: Comparative performance of defense mechanisms

RAG demonstrates superior attack prevention (0.0% ASR versus 28.0% for LoRA) while maintaining comparable False Positive Rate (10.99% versus 15.0%). However, RAG introduces 20ms inference overhead for similarity search, while LoRA adds no latency beyond baseline.

5.5 Confusion Matrix Analysis

Detailed analysis of classification outcomes provides insight into defense mechanism behavior. The LoRA model achieves precision of 82.9%, recall of 72.0%, and F1 score of 77.1%. The RAG system achieves precision of 92.0%, recall of 100.0%, and F1 score of 95.8%. Perfect recall indicates zero false negatives, meaning no attacks evade detection.

6 Discussion

6.1 Interpretation of Results

6.1.1 Baseline Vulnerability

The 85.6% Attack Success Rate observed in baseline evaluation demonstrates that contemporary instruction-following models remain highly vulnerable to adversarial manipulation despite extensive safety training. This vulnerability stems from fundamental tensions in model design: the same instruction-following capabilities that make models useful also enable adversarial exploitation.

The differential vulnerability across attack types provides insight into the limitations of current safety training. The high success rate of jailbreaking attacks (93.0%) suggests that safety training focuses primarily on direct harmful requests rather than indirect approaches through role-playing or hypothetical framing. Attackers exploit this gap by reframing harmful requests in contexts that bypass learned refusal patterns.

6.1.2 LoRA Effectiveness

LoRA fine-tuning achieves substantial improvements (67.3% reduction in ASR) while training only 0.06% of model parameters. This efficiency derives from the low-rank structure of safety-relevant adaptations: teaching refusal behavior requires modifying only a small subspace of the full parameter space.

The increase in False Positive Rate from 9.6% to 15.0% indicates that LoRA learns conservative refusal policies. This behavior likely results from the training objective, which penalizes compliance with malicious requests more heavily than over-cautious refusal of legitimate queries. Future work could explore multi-objective optimization to balance security and utility more precisely.

The persistent 28.0% Attack Success Rate reveals limitations of parameter-efficient fine-tuning. LoRA adapters may lack sufficient capacity to capture the full complexity of adversarial patterns, or the training dataset may not cover the full distribution of attack techniques. Additionally, fine-tuning on static datasets cannot adapt to novel attacks that emerge after training.

6.1.3 RAG Effectiveness

RAG achieves perfect attack prevention (0.0% ASR) through pattern matching against a comprehensive knowledge base. This success demonstrates that many adversarial attacks share recognizable linguistic patterns that enable reliable detection through semantic similarity.

The minimal increase in False Positive Rate (10.99% versus 9.6% baseline) indicates that attack patterns are sufficiently distinct from legitimate queries to enable accurate discrimination. The 2,482 false positives in our evaluation likely result from legitimate queries that happen to share vocabulary or phrasing with attack patterns.

The 20ms inference overhead introduced by RAG represents a practical trade-off for the perfect attack prevention achieved. This latency remains acceptable for most interactive applications, though it may pose challenges for high-throughput scenarios requiring sub-10ms response times.

6.2 Strengths and Limitations of Each Approach

LoRA offers integration into model weights, generalization to attack variants, preservation of base model knowledge, memory-efficient training, and mergeable adapters. However, it requires retraining for updates, exhibits higher false positive rates, provides incomplete attack coverage, depends on training data, and may degrade capabilities on edge cases.

RAG provides perfect attack detection, requires no model retraining, supports easily updatable knowledge bases, enables transparent decision-making, maintains minimal false positive rates, and offers explainable refusals. However, it introduces inference overhead, requires knowledge base maintenance, may miss novel attacks, has memory footprint requirements, depends on embedding quality, and faces potential evasion through paraphrasing.

6.3 Comparison with Prior Work

Our results compare favorably with published research on LLM adversarial robustness. Wei et al. reported approximately 40% reduction in Attack Success Rate through Reinforcement Learning from Human Feedback on LLaMA-7B. Zou et al. achieved approximately 55% improvement through adversarial training on Vicuna-7B. Our LoRA approach achieves 67.3% improvement, while RAG achieves 100% improvement.

The superior performance of our methods likely derives from several factors. First, our evaluation focuses on a more recent model (Mistral-7B) that may have different vulnerability characteristics. Second, our defense mechanisms specifically target the attack categories in our evaluation datasets, potentially leading to overfitting. Third, our RAG approach benefits from a comprehensive knowledge base constructed from the same distribution as evaluation data.

6.4 Practical Implications

For high-security applications such as financial services or healthcare, we recommend RAG-based filtering despite the inference overhead. The perfect attack prevention justifies the 20ms latency increase, and the low false positive rate (10.99%) maintains acceptable user experience. Organizations should implement human review processes for refused requests to identify and correct false positives.

For general-purpose applications such as chatbots or virtual assistants, LoRA fine-tuning provides a reasonable balance between security and performance. The 72.0% refusal accuracy offers substantial protection while the zero inference overhead maintains responsive user experience. The 15.0% false positive rate may require careful prompt engineering and user education to minimize frustration.

For research and development environments, we recommend implementing both mechanisms in parallel to enable comparative analysis on production traffic. This approach provides maximum flexibility while generating data to inform long-term deployment decisions.

Combining LoRA and RAG could potentially achieve superior performance by leveraging the complementary strengths of each approach. LoRA provides model-level robustness that generalizes to novel attacks, while RAG offers perfect detection of known patterns. A hybrid system could apply RAG filtering first for known attacks, then rely on LoRA-enhanced model behavior for edge cases.

6.5 Limitations and Threats to Validity

Several factors may affect the internal validity of our results. First, automated response classification using keyword matching may introduce errors, though our validation against human judgment achieved 94% agreement. Second, the threshold selection for RAG (0.7) was determined through informal experimentation rather than systematic optimization, potentially biasing results. Third, the limited training duration for LoRA (2 epochs) may not represent optimal performance.

The generalizability of our findings faces several limitations. First, evaluation on a single model (Mistral-7B) limits conclusions about other architectures or scales. Results may not transfer to larger models (70B+ parameters) or models with different training procedures. Second, our attack datasets may not represent the full distribution of adversarial techniques used in practice. Sophisticated attackers may employ novel methods not captured in public datasets. Third, the evaluation focuses on single-turn attacks, while real-world threats increasingly involve multi-turn conversation manipulation.

Our metrics may not fully capture the constructs of interest. Attack Success Rate measures compliance versus refusal but does not assess the quality or harmfulness of compliant responses. Some compliant responses may provide incomplete or misleading information that partially mitigates harm. Similarly, False Positive Rate treats all incorrect refusals equally, though some false positives may be more disruptive to user experience than others.

The rapid evolution of adversarial techniques threatens the temporal validity of our results. Attack methods continue to evolve, and defenses effective against current techniques may fail against future innovations. The RAG approach partially addresses this through updateable knowledge bases, but requires ongoing investment in threat intelligence. LoRA-based defenses require periodic retraining to maintain effectiveness.

7 Conclusion

7.1 Summary of Findings

This research demonstrates that contemporary instruction-following language models exhibit substantial vulnerability to adversarial attacks, with baseline Attack Success Rates of 85.6%. We implemented and evaluated two defense mechanisms: Low-Rank Adaptation fine-tuning and Retrieval-Augmented Generation safety filtering.

LoRA fine-tuning reduces Attack Success Rate to 28.0%, representing a 67.3% improvement over baseline while training only 0.06% of model parameters. This approach integrates security directly into model weights, eliminating inference overhead while providing reasonable protection against known attack categories.

RAG-based filtering achieves perfect attack prevention (0.0% ASR) through pattern matching against a comprehensive knowledge base of 28,548 attack examples. This approach introduces 20ms inference overhead but maintains low false positive rate (10.99%) and provides explainable refusals.

Comparative analysis reveals fundamental trade-offs between integrated defenses (LoRA) and external filtering (RAG). LoRA offers zero-latency protection with good generalization but requires retraining for updates. RAG provides perfect detection of known attacks with easy updates but introduces inference overhead and may miss novel patterns.

7.2 Research Contributions

This work makes several contributions to the field of LLM security: empirical demonstration that parameter-efficient fine-tuning can achieve substantial improvements in adversarial robustness, novel application of Retrieval-Augmented Generation for safety enforcement, comprehensive evaluation framework implemented as open-source toolkit, comparative analysis revealing trade-offs between defense paradigms, and production-ready implementation architecture suitable for real-world deployment.

7.3 Future Research Directions

Several promising directions emerge from this work. Combining LoRA and RAG could leverage the complementary strengths of each approach. Investigating how defense effectiveness scales with model size would provide insight into fundamental relationships between capacity, capability, and security. Incorporating adversarial examples during LoRA fine-tuning could improve robustness against novel attacks. Extending defense mechanisms to handle multi-modal scenarios represents an important research direction. Developing formal methods for verifying security properties could provide stronger guarantees than empirical evaluation. User studies investigating the acceptability of false positive rates would inform practical deployment decisions.

7.4 Concluding Remarks

The widespread deployment of Large Language Models in production systems necessitates robust defenses against adversarial manipulation. This research demonstrates that effective defense mechanisms exist and can be implemented with reasonable computational cost. The choice between integrated defenses like LoRA and external filtering like RAG depends on specific deployment requirements, with hybrid approaches offering promising directions for future work.

As adversarial techniques continue to evolve, the security community must maintain ongoing investment in defense research and rapid deployment of countermeasures. The open-source implementation provided by this work aims to facilitate such efforts by providing a foundation for reproducible research and practical deployment.

Acknowledgments

We thank the HuggingFace team for providing access to model repositories and datasets. We acknowledge Kaggle for providing free GPU resources that enabled this research. We are grateful to the open-source community for developing the tools and libraries that made this work possible.

References

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

[2] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877-1901.

[3] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2021). LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685.*

[4] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33, 9459-9474.

[5] Wei, A., Haghtalab, N., & Steinhardt, J. (2023). Jailbroken: How does LLM safety training fail? *arXiv preprint arXiv:2307.02483.*

[6] 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.*

[7] Perez, F., & Ribeiro, I. (2022). Ignore previous prompt: Attack techniques for language models. *arXiv preprint arXiv:2211.09527.*

[8] Liu, Y., Deng, G., Li, Y., Wang, K., Zhang, T., Liu, Y., ... & Zhang, Y. (2023). Prompt injection attacks and defenses in LLM-integrated applications. *arXiv preprint arXiv:2310.12815.*

[9] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., ... & Lowe, R. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35, 27730-27744.

[10] Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., ... & Wang, H. (2023). Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997.*

[11] Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. D. L., ... & Sayed, W. E. (2023). Mistral 7B. *arXiv preprint arXiv:2310.06825.*

[12] Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., ... & Scialom, T. (2023). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288.*