# 1 Introduction

As Large Language Models (LLMs) are progressively incorporated into critical applications - ranging from healthcare and law to education and customer service—their vulnerability to adversarial exploitation has sparked considerable safety and ethical issues. One of the most urgent dangers is jailbreak attacks, where adversaries design prompts that coerce LLMs into generating outputs that breach safety protocols, alignment policies, or restrictions set by developers. Although there have been significant improvements in safety alignment through methods such as supervised fine-tuning and reinforcement learning from human feedback (RLHF), these models continue to be susceptible to more advanced attack techniques that circumvent conventional protections.

Recent research identifies that jailbreak attacks largely fall into four broad categories:
(1) **Adversarial suffix appendages**, where benign queries are followed by carefully crafted strings to alter model behavior; (2) **Prompt rewriting**, which reformulates the original query to obscure harmful intent and evade detection; (3) **Optimized many-shot prompting**, which overwhelms the model with curated example completions to coerce unsafe outputs; and (4) **Malicious content obfuscation**, which disguises adversarial intent using encoding tricks, homograph substitutions, or syntactic manipulation.
Increasingly, modern jailbreak techniques blend elements from multiple categories, resulting in *hybrid attacks* that challenge existing defenses through structural ambiguity and prompt-level deception (Doumbouya et al., 2025; Zheng et al., 2024).

In parallel, the defense landscape has evolved from static alignment strategies toward more *modular, runtime defense mechanisms* capable of analyzing and intervening during LLM input-output flows. However, current defenses remain fragmented across tools, terminologies, and functionalities, making it difficult to evaluate their robustness in a consistent and systematic manner. To address this, we propose a structured categorization of defense strategies designed to resist prompt-based adversarial attacks.

These fall into four primary classes:
(1) **Log-likelihood and perplexity-based filters**, which use statistical metrics to detect low-probability or anomalous outputs (e.g., *Perplexity Filter*); (2) **Prompt sanitization and rephrasing modules**, which transform user inputs to neutralize potential threats before reaching the model (e.g., *RephrasingDefense*, *Retokenization*); (3) **Content moderation and concept filtering**, which analyze outputs or activations for semantic violations using rule-based or learned heuristics (e.g., *LLMModerator*, *Legilimens*); and (4) **Policy-enforcing guardrails and refusal mechanisms**, which embed behavioral constraints within system prompts or through alignment-tuned models (e.g., *LlamaGuard*, *SecAlign*).

This paper proposes a unified framework for the *categorization of both attack and defense strategies* in LLM systems. By systematically analyzing their design patterns, interaction points,

and hybridization trends, we aim to:
(i) improve the interpretability and benchmarking of jailbreak attack behaviors;
(ii) reveal coverage limitations and blind spots in existing defense mechanisms; and
(iii) offer actionable guidance for developing next-generation LLM defenses that are modular, adaptive, and composable.

Our taxonomy provides a foundational lens through which to understand the evolving adversarial landscape and informs the design of robust, future-proof safety systems capable of withstanding the dynamic nature of prompt-based threats.

## 2 Objective

The main goal of this paper is to introduce a thorough, bidirectional framework that classifies both adversarial attack strategies and their respective defense mechanisms within the realm of Large Language Models (LLMs). We seek to extend beyond basic taxonomies by exploring the interactions between each category of attack and every type of defense—emphasizing both advantages and disadvantages. This cross-analysis aims to assist both aspects of the alignment landscape: empowering defenders to create more resilient and focused safeguards, while aiding researchers investigating attacks to gain a deeper insight into the limitations and failure modes of existing defenses. Our primary goal is to assist both attackers and defenders in ethical practices by creating a decision tree. In this tree, if an attacker and a defender choose to move left or right, respectively, they should not need to explore the respective opposite side of the tree. This comprehensive description may also assist in future forensic investigations.

This paper specifically aims to:

(1) **Define and elaborate** on the four primary types of jailbreak attacks: adversarial suffix appendages, prompt rewriting, many-shot adversarial prompting, and malicious content obfuscation, including real-world examples and their structural characteristics.

(2) **Classify and clarify** the four key categories of LLM defenses: log-likelihood and perplexity-based filters, prompt sanitization modules, semantic moderation systems, and policy-enforcing guardrails, along with representative implementations.

(3) **Examine the interactions** between each category of attack and defense by creating a comprehensive compatibility matrix—identifying which defenses effectively counter specific attacks, which can be circumvented, and where shared vulnerabilities are present.

(4) **Emphasize hybrid scenarios**, where attacks integrate strategies from various categories, and assess how multi-layered defenses react in such situations.

(5) **Foster mutual progress** by providing actionable insights for both attackers and defenders—enabling developers to enhance defense architectures while also offering a structured understanding of adversarial behavior that aids in the safer design and research of LLMs.

By reaching these objectives, this paper lays the groundwork for a more knowledgeable and cooperative approach to advancing the security of large-scale language models, which will ultimately result in safer and more dependable AI implementations in practical applications.

## 3 Assumption

This study operates under a set of clearly defined assumptions to ensure ethical compliance, methodological consistency, and generalizability of results. These assumptions establish the groundwork for our attack-defense evaluation framework and influence the extent of our conclusions.

(1) **Black-Box Evaluation Setting**

All experiments conducted in this research are performed within a black-box threat model. We do not assume any internal access to the architecture, weights, gradients, or training data of the target LLM. Instead, adversarial interactions are confined strictly to the model's input-output interface. This mirrors a realistic deployment scenario where users or adversaries engage with commercial or restricted-access LLMs through APIs or chat interfaces. The adversarial objective is to jailbreak the model—that is, to provoke responses that breach established safety constraints—and to deduce the underlying system behavior or prompt structure solely based on the outputs generated.

(2) **Model Tampering**

We confirm that there was no unethical alteration or interference with the models. Our experiments do not include unauthorized access to internal elements, the injection of harmful payloads, backdoor attacks, or any actions that breach the intended usage policies set by the LLM providers. All attack prompts were developed exclusively through linguistic and semantic manipulation, adhering to responsible AI research standards. This guarantees that our results are relevant in legally and ethically acceptable contexts.

(3) **Implementation of Canonical Attack and Defense Categories**

The choice to implement precisely four attack and four defense categories is intentional and theoretically supported. These categories—Adversarial Suffix Appendages, Prompt Rewriting, Optimized Many-Shot Prompting, and Malicious Content Obfuscation for attacks, along with Perplexity-Based Filters, Prompt Sanitization, Concept Moderation, and Policy Guardrails for defenses—together offer a thorough and mutually exclusive summary of existing jailbreak tactics and countermeasures. This classification captures the range of prompt manipulation methods seen in practice and the main types of real-time mitigation strategies utilized.

(4) **Attack–Defense Segregation and Independence**

We operate under the assumption that attacks and defenses function independently and are part of distinct functional layers. Attacks occur solely at the input prompt level, whereas defenses can be implemented either before inference (such as input sanitization), after inference (like output filtering), or during the execution of the model (for instance, policy alignment). This clear separation enables us to methodically assess cross-category effectiveness—specifically, analyzing how a defense mechanism from one category responds to an attack from each of the four classes—without mixing internal model behavior with external defense strategies.

(5) **Model Behavior is Deterministic or Logit-Stable**

Although the majority of LLMs are fundamentally probabilistic, we posit that for any specific prompt, the model's output is either deterministic (when using a temperature=0 setting) or stable enough across multiple queries to facilitate consistent assessment. This guarantees that the attribution of jailbreak success or failure is linked to the prompt's structure and the defense response, rather than to random fluctuations in sampling.

(6) **Focus on Prompt-Level Vulnerability, Not Data Poisoning or Fine-Tuning**

This study specifically targets *prompt-level vulnerabilities* in LLM behavior. We exclude other attack vectors such as data poisoning, model extraction, or fine-tuning-based corruption, which require a different set of assumptions and access levels. Our focus is on linguistic adversaries operating within the bounds of standard API access, which aligns with the most common real-world misuse vectors.

By operating under these constraints and assumptions, this paper offers a thorough, realistic, and ethically grounded assessment of the changing dynamics of jailbreak attacks and LLM defenses. The findings and insights obtained aim to assist both red-teaming initiatives (to uncover concealed vulnerabilities) and blue-teaming approaches (to create generalizable, multi-layered defenses) in a responsible and reproducible way.

## 4 Methodology

This section presents the cohesive methodological framework utilized for the systematic categorization, execution, and analysis of jailbreak attacks and LLM defenses. Our strategy is based on two fundamental principles: evolutionary structuring and diagnostic inference. Initially, we classify both attack and defense mechanisms into four overarching categories, each signifying a unique evolutionary stage in the adversarial landscape—from basic token suffixes to sophisticated obfuscation techniques for attacks, and from perplexity-based filters to integrated guardrails for defenses. This timeline-aware taxonomy encompasses the entire range of established and emerging strategies.

Furthermore, we employ a sequential attack evaluation framework that is designed not only to assess defenses but also to deduce the probable presence or absence of defense classes within a black-box LLM environment.By executing meticulously organized attacks and monitoring their success or failure, the system constructs a probabilistic decision trace that correlates outcomes with defense behaviors without necessitating internal access to the model.This inference logic is organized as a flowchart-like algorithm, highlighting conditional dependencies among various attack types.

In summary, our methodology facilitates both principled classification and practical examination of defense robustness in large-scale LLM systems.

**Framework Overview:**

The framework operates as follows:-

- **Group Attacks:** Cluster attacks into groups based on their approach (e.g., adversarial suffixes, prompt rewriting, many-shot prompting, content obfuscation). Sequential Testing: Apply attack groups in a predefined order, starting with simpler attacks (e.g., direct requests) and progressing to more sophisticated ones (e.g., optimized prompting or obfuscation).
- **Outcome Interpretation:** For each attack group, define what success or failure suggests about the presence or absence of defense classes (log-likelihood/perplexity filters, prompt sanitization, content moderation, policy-enforcing guardrails).
- **Decision Path:** Generate a clear, hierarchical decision path that outlines the testing sequence and conclusions. Attack Groups Based on the provided attacks, we group them by mechanism:

Below, we outline this process step-by-step, where each group of attacks plays a probing role within the hierarchy of adversarial strategies.

**Step 1: Execute Group 1 – Optimization-based Adversarial Prompt Attack**

At first we begin with the two attacks AutoPrompt, AutoDAN, GBDA.

**Objective:** The aim of this phase is to assess the susceptibility of the target LLM to optimization-driven adversarial prompt attacks utilizing AutoPrompt, AutoDAN, and GBDA. These attacks employ automated optimization methods to create adversarial suffixes that are added to harmless queries, methodically constructing token sequences intended to provoke unsafe outputs from the model.

**Attack Mechanisms:**

- **AutoPrompt:** Utilizes gradient-based optimization that "selects the best token for each position in the prompt by leveraging the gradient to identify a range of suitable candidates."
- **AutoDAN:** Employs a "hierarchical genetic algorithm specifically designed for structured discrete data such as prompt text" to create prompts that are more semantically relevant while still being derived from optimization.
- **GBDA:** Implements a "Gumbel-softmax distribution characterized by a continuous-valued matrix of coefficients" combined with gradient-based optimization to explore adversarial distributions.

**Defense Assessment Focus:** The evaluation assesses if the LLM has defenses based on ***Perplexity*** or similar token-level filtering techniques. These optimization-driven approaches generally yield "unreadable gibberish strings with a perplexity significantly greater than that of typical human text," with studies indicating that "AutoPrompt produces suffixes that exhibit extremely high perplexity values." Although AutoDAN is intended to be more "stealthy," it still achieves "much lower PPL than the baseline, GCG, and is on par with Handcrafted DAN," yet it remains considerably higher in perplexity compared to natural human text.

**Interpretation:**

- The ***success*** of these attacks reveals a ***lack of strong perplexity filtering*** or input sanitization measures that can identify "token-level optimization-based methods [which] usually produce incomprehensible gibberish strings with a perplexity significantly greater than that of typical human text."
- Conversely, ***failure*** indicates the ***existence of efficient token-level detection systems*** that can recognize and counteract synthetically generated adversarial prompts using perplexity thresholding or comparable statistical anomaly detection techniques, as "perplexity filtering is very effective on GCG" and related optimization-based assaults. Then go to **step 2**.

**Step 2: Execute Group 2 – Optimization-based Token-level Jailbreak Attacks**

In this group we will try some more advanced attacks like AutoDAN, GBDA, GCG, PEZ, UAT.

**Objective:** The aim of this phase is to assess the target LLM's susceptibility to optimization-driven token-level jailbreak attacks utilizing AutoDAN, GBDA, GCG, PEZ, and UAT. These attacks utilize advanced optimization techniques to create adversarial token sequences that take advantage of specific tokenization patterns and character-level configurations to circumvent safety protocols. The evaluation centers on identifying whether the LLM has implemented rephrasing or retokenization defenses capable of disrupting these token-level manipulations, as studies indicate that "adversarially-generated prompts are fragile to character-level alterations". A successful outcome signifies the lack of preprocessing defenses that could counteract token-level adversarial manipulations through character-level modifications, whereas a failure implies the presence of effective input sanitization systems that can "lower attack success rates to under one percent".

**Attack Mechanisms and Shared Characteristics:** AutoDAN, GBDA, GCG, PEZ, and UAT all create optimized adversarial token sequences—using genetic algorithms, gradient-based techniques, greedy coordinate updates, embedding-space assaults, or universal triggers—to add to benign prompts.

**Interpretation:**

- **If the attack is successful:** This signifies a lack of defenses against ***character-level perturbations*** or ***retokenization methods***. The LLM probably handles inputs directly, without any preprocessing that might interfere with token-level optimization patterns. A successful attack indicates a vulnerability to the entire category of optimization-based token-level attacks and suggests that advanced defenses, such as SmoothLLM, are not in place.
- **If the attack fails:** This strongly implies that there are preprocessing defenses in place that actively sanitize, perturb, or retokenize adversarial inputs before they reach the generation module. The LLM likely utilizes character-level perturbation techniques or retokenization strategies that can "lower the attack success rate on many popular LLMs to under one percentage point." A failure indicates a strong defense against optimization-based token-level manipulation and suggests that the model has implemented advanced input preprocessing capabilities. Go to **step 3**.

**Step 3: Execute Group 3 – Semantic Rewrite & Context Poisoning Attacks**

In this group we will try some more advanced attacks like ***GBDA, GCG variants, and FewShot attacks***.

**Objective:** The aim of this phase is to assess if the LLM is capable of identifying and dismissing prompts that disguise harmful directives through nuanced semantic modifications or that inundate it with numerous malicious instances. We will evaluate this using GBDA, which employs gradient-guided adjustments to token distributions to obscure intent; GCG variants (GCG, GCG-M, GCG-T), which progressively exchange or alter tokens to create adversarial yet semantically comparable inputs; and a FewShot attack, which populates the context window with selected unsafe examples to influence the model's output.

**Attack Mechanism:** Group 3 employs two complementary strategies to undermine model safety: 1) subtle semantic alterations of individual tokens or distributions, and 2) context-window contamination using multiple adversarial examples.

In terms of semantic alterations, GBDA utilizes gradient-guided perturbations on the model's token embedding distributions to subtly shift the meaning of a few critical words while maintaining the surface syntax. GCG and its variants (GCG-M, GCG-T) execute greedy, coordinate-wise token swaps or mutations—leveraging Monte Carlo or transferability heuristics—to iteratively substitute benign tokens with adversarial alternatives that retain fluency but convey malicious intent.

Regarding context contamination, FewShot introduces a series of carefully crafted unsafe Q&A pairs or example completions into the prompt, taking advantage of in-context learning: by repeatedly showcasing the prohibited behavior, the model is encouraged to generalize that behavior to the user's final query.

**Interpretation:**

- **If successful:** The model is deficient in strong semantic filtering and in-context protective measures. It is unable to identify or reject subtle prompt manipulations or context flooding, which points to a lack of defenses like the concept-probing of Legilimens or the input-output classification of LlamaGuard.
- **If unsuccessful:** The model demonstrates a high level of semantic comprehension and anomaly detection capabilities. It effectively neutralizes both adversarial-token rewritings and context injections through the use of multi-example prompts, indicating the presence of semantic-aware moderation tools (such as Legilimens) and few-shot anomaly detection systems. Go to **step 4**.

**Step 4: Execute Group 4 – Natural-language Jailbreak Attacks**

In this group we will try some more advanced attacks like ***AutoDAN, DirectRequest, HumanJailbreaks, and ZeroShot***.

**Objective:** Assess if the LLM utilizes a supervising moderation LLM (LLMmoderator) or preference-optimized fine-tuning (SecAlign) by conducting tests on attacks that depend exclusively on natural-language cues instead of token-level optimization. We will employ AutoDAN's genetic-algorithm-generated prompts, simple DirectRequest queries, custom-made HumanJailbreaks, and ZeroShot reformulations to investigate fundamental content filtering and refusal mechanisms.

**Attack Mechanism:** All four techniques create prompts that are easy for humans to read, aimed at circumventing filters through semantics, role-playing, or straightforward rephrasing.

AutoDAN produces coherent adversarial narratives using genetic search; DirectRequest straightforwardly requests prohibited content; HumanJailbreaks employs carefully crafted stories or scenarios to disguise intent; ZeroShot presents a single rephrased query intended to bypass safety checks without providing examples.

**Interpretation:**

- **If successful:** The model does not include a supervisory filtering layer (LLMmoderator) and has not been subjected to preference-optimized safe-completion tuning (SecAlign). It is unable to reject or modify clear malicious prompts, which suggests a lack of both input/output moderation by an overseeing LLM and internal fine-tuning that steers it towards safe responses.
- **If unsuccessful:** The LLMmoderator is functioning—preventing or censoring unsafe outputs—and/or SecAlign has directed the model's generation towards secure completions. The ability to effectively refuse or safely rewrite all natural-language attacks indicates that strong supervisory moderation and preference-based defenses are in place.

**Current status:**