

# Jailbreaking and Mitigation of Vulnerabilities in Large Language Models

Benji Peng  
AppCubic  
Miami, USA  
benji@appcubic.com

Ziqian Bi  
Purdue University  
West Lafayette, USA  
bi32@purdue.edu

Qian Niu  
Kyoto University  
Kyoto, Japan  
niu.qian.f44@kyoto-u.jp

Ming Liu  
Purdue Technology  
West Lafayette, USA  
liu3183@purdue.edu

Pohsun Feng  
National Taiwan Normal University  
Taipei, ROC  
41075018h@ntnu.edu.tw

Tianyang Wang  
University of Liverpool  
Suzhou, PRC  
tianywang0305@gmail.com

Lawrence K.Q. Yan  
The Hong Kong University of Science and Technology  
Hong Kong, PRC  
kqyan@connect.ust.hk

Yizhu Wen  
University of Hawaii  
Honolulu, USA  
yizhuw@hawaii.edu

Yichao Zhang  
The University of Texas at Dallas  
Dallas, USA  
yichao.zhang.us@gmail.com

Caitlyn Heqi Yin  
University of Wisconsin-Madison  
Madison, USA  
hyin66@wisc.edu

**Index Terms**—Large Language Models, Prompt Injection, Jailbreaking, AI Security, LLM Application Defense Mechanisms

**Abstract**—Large Language Models (LLMs) have transformed artificial intelligence by advancing natural language understanding and generation, enabling applications across fields beyond healthcare, software engineering, and conversational systems. Despite these advancements in the past few years, LLMs have shown considerable vulnerabilities, particularly to prompt injection and jailbreaking attacks. This review analyzes the state of research on these vulnerabilities and presents available defense strategies. We roughly categorize attack approaches into prompt-based, model-based, multimodal, and multilingual, covering techniques such as adversarial prompting, backdoor injections, and cross-modality exploits. We also review various defense mechanisms, including prompt filtering, transformation, alignment techniques, multi-agent defenses, and self-regulation, evaluating their strengths and shortcomings. We also discuss key metrics and benchmarks used to assess LLM safety and robustness, noting challenges like the quantification of attack success in interactive contexts and biases in existing datasets. Identifying current research gaps, we suggest future directions for resilient alignment strategies, advanced defenses against evolving attacks, automation of jailbreak detection, and consideration of ethical and societal impacts. This review emphasizes the need for continued research and cooperation within the AI community to enhance LLM security and ensure their safe deployment.

## I. INTRODUCTION

Large Language Models (LLMs) have become a pivotal development in artificial intelligence, demonstrating outstanding capabilities in natural language understanding and generation.

Their capacity to process large volumes of data and generate human-like responses has led to their integration across numerous applications, such as chatbots, virtual assistants, code generation systems, and content creation platforms [1], [2]. However, the rapid advancement and widespread adoption of LLMs have raised substantial security and safety concerns [3].

As LLMs grow more powerful and are integrated into critical systems, the potential for misuse and unintended consequences increases. The capabilities that make LLMs valuable—their ability to learn from massive datasets and generate creative outputs—also render them susceptible to manipulation and exploitation [4]. A major concern in LLM security is their vulnerability to adversarial attacks, particularly prompt injection and jailbreaking [5]. These attacks exploit the intrinsic design of LLMs, which follow instructions and generate responses based on patterns in their training data [6]. Bad actors can craft malicious prompts to bypass safety mechanisms in LLMs, resulting in harmful, unethical, or biased outputs [7].

Researchers have indicated that LLMs can be manipulated to provide instructions for illegal activities such as drug synthesis, bomb-making, and money laundering [8]. Other studies have demonstrated the effectiveness of persuasive language, based on social science research, in jailbreaking LLMs to generate harmful content [9]. Multilingual prompts can exacerbate the impact of malicious instructions by exploiting linguistic gaps in safety training data, leading to high rates

of unsafe output [10]. These attacks show the fragility of current safety alignment techniques and the need for more robust defenses [11]. Qi et al. [12] demonstrated that fine-tuning aligned LLMs, even with benign data, can compromise safety.

This literature review provides an overview of research on prompt engineering, jailbreaking, vulnerabilities, and defenses in generative AI and LLMs. We systematically analyze the literature to achieve the following objectives:

- To review literature on prompt injection, jailbreaking, vulnerabilities, and defenses in LLMs. This includes categorizing attack types, analyzing underlying vulnerabilities, and evaluating the effectiveness of defense mechanisms [13].
- To identify research gaps and areas for further exploration, including limitations of current safety mechanisms, emerging attack vectors, and the need for more comprehensive defense strategies.
- How does Generative AI reshape the collaboration between human designers and automated systems?
- To summarize the current state of LLM security and suggest directions for future research. This includes synthesizing findings, discussing implications for LLM development and deployment, and proposing research directions to address identified gaps [14]. It also explores the misuse of LLMs for criminal activities, such as fraud, impersonation, and malware generation [15].

## II. BACKGROUND AND CONCEPTS

### A. Large Language Models (LLMs)

Large Language Models (LLMs) are artificial intelligence systems that use deep learning, specifically transformer networks, to process and generate human-like text [6]. Trained on massive datasets, LLMs learn complex language patterns, enabling them to perform tasks such as text summarization, translation, question answering, and creative writing. Their ability to generate coherent, contextually relevant text stems from their vast training corpus and advanced architecture [16]. LLMs have permeated many domains [1], offering both beneficial and potentially harmful applications. In healthcare, LLMs assist with tasks such as medical record summarization, patient education, and drug discovery [17], [18]. In software engineering, LLMs such as OpenAI Codex assist in code auto-completion, streamlining development [19]. They also contribute significantly to AI-driven programming and conversational AI systems [20]. However, LLMs also pose risks, including misuse for generating harmful content like hate speech, misinformation, and instructions for illegal activities [19], [21]. This dual-use potential demands careful consideration of safety and ethical implications [8].

A key challenge in LLM development is aligning them with human values and intentions [4]. Alignment involves training LLMs to behave in a beneficial and safe manner for humans, avoiding harmful or undesirable outputs. This includes aligning models with social norms and user intent [12], [22]. Misalignment occurs when LLMs deviate from

human values or produce harmful, unethical, or biased outputs [11]. Achieving robust alignment is an ongoing challenge, as LLMs are susceptible to adversarial attacks that exploit their vulnerabilities, leading to misalignment [23].

To mitigate LLM risks, researchers have developed safety mechanisms to align these models with human values and prevent harmful content generation [24]. These mechanisms can be categorized into pre-training and post-training techniques. Pre-training techniques filter training data to remove harmful or biased content [10]. Post-training techniques include supervised fine-tuning (SFT) and reinforcement learning from human feedback (RLHF), where the LLM is trained on curated datasets to align outputs with human preferences and ethical guidelines [23].

Red-teaming is a proactive safety mechanism that tests LLMs with adversarial prompts to identify vulnerabilities and enhance robustness [25], [26]. Prompt engineering for safety designs prompts that instruct LLMs to avoid harmful or unethical content [27]. Safety guardrails restrict certain outputs from LLMs, while system prompts give high-level instructions to guide LLM behavior [20], [28]. However, these system prompts are vulnerable to leakage, posing a security risk [29].

Evaluating LLM safety and trustworthiness requires robust metrics that capture different aspects of model behavior. Toxicity scores assess offensive or harmful language in LLM outputs, while bias scores measure model prejudice or discrimination against groups [10], [30]. Adversarial robustness measures the model's ability to resist adversarial attacks and maintain intended behavior [16], [23]. Data leakage involves the unintentional disclosure of sensitive information from training data [31], while compliance with ethical guidelines assesses the model's adherence to ethical principles and norms [19].

Several benchmark datasets have been developed to evaluate LLM safety and robustness. These datasets consist of curated prompts and responses to test the model's ability in safety-critical scenarios. Examples include RealToxicityPrompts, focusing on eliciting toxic responses, and Harmbench, which tests broader harmful behaviors [32]. Other datasets, such as Do-Not-Answer [33], Latent Jailbreak [16], and RED-EVAL [25], target the model's ability to resist harmful or unethical instructions. Additionally, datasets like JailbreakHub analyze the evolution of jailbreak prompts over time [11], [34]. However, these benchmark datasets often have limitations in scope, diversity, and real-world applicability, highlighting the need for continuous development and refinement of evaluation methods.

### B. Prompt Engineering

Prompt engineering is the process of designing the input text, or prompt, given to an LLM to elicit the desired output [6] [35]. It plays a crucial role in enhancing LLM performance and ensuring safety by providing context, specifying the task, and guiding the model's behavior. Effective prompts significantly improves the accuracy, relevance, and creativity of the generated text, while also mitigating the risk of harmful

or biased outputs. Prompt engineering involves a variety of techniques, ranging from simple instructions to more complex strategies that fully utilize the LLM’s capabilities. Zero-shot prompting involves providing a task description without any examples [36], while few-shot prompting includes a few examples to guide the model [36]. Chain-of-thought prompting encourages the LLM to generate a step-by-step reasoning process before providing the final answer [35], while tree-of-thought prompting expands on this by exploring multiple reasoning paths [35]. Role prompting assigns a specific role or persona to the LLM [35], whereas instruction prompting provides explicit instructions to generate the desired output format or content. Bespoke prompt engineering enhances LLM safety and mitigates risks, which involves designing prompts that instruct the LLM to avoid generating harmful or unethical content explicitly, respect diverse perspectives, and adhere to established ethical guidelines. For example, prompts may instruct the LLM to avoid hate speech, consider cultural sensitivities, or prioritize factual accuracy over creative storytelling. In some cases, prompts can remind the LLM of its safety guidelines and responsibilities, serving as a form of self-regulation [37].

### C. Jailbreaking

Jailbreaking refers to adversarial attacks designed to bypass the safety mechanisms of LLMs, inducing them to produce content that violates intended guidelines or restrictions [7], [19]. These attacks exploit the LLMs’ inherent tendency to follow instructions and generate text based on learned training data patterns. Adversaries may be motivated by a desire to expose vulnerabilities, test LLM safety limits, or maliciously exploit these models for personal gain or to inflict harm [34].

Jailbreak attacks can be categorized by strategy, target modality, and objective. Attack strategies include prompt injection, embedding malicious instructions in benign prompts [20]; model interrogation, manipulating internal representations to extract harmful knowledge [38]; and backdoor attacks, embedding malicious triggers during training [14]. Target modalities include textual jailbreaking, manipulating LLM textual inputs [19], and visual jailbreaking, targeting image inputs in multimodal LLMs [39]. Multimodal attacks exploit interactions between modalities, like combining adversarial images with textual prompts [40]. Attack objectives include generating harmful content, bypassing safety filters, leaking private information [15], or gaining control of LLM behavior [13].

The rise of online communities sharing jailbreak prompts has hugely escalated the threat levels. These communities collaborate to discover vulnerabilities, refine attacks, and bypass new defenses [34] [11]. The rapid evolution and growing sophistication of jailbreaking highlight the need for continuous development of robust defenses. The shift to dedicated prompt-aggregation websites signals a trend towards more organized and sophisticated jailbreaking [7].

## III. JAILBREAK ATTACK METHODS AND TECHNIQUES

Jailbreaking attacks aim to exploit vulnerabilities in LLMs to bypass their safety mechanisms and induce the generation of harmful or unethical content. As LLMs become more powerful and widely deployed, the need to understand and mitigate these attacks becomes increasingly crucial. These attacks can be broadly categorized into prompt-based attacks, model-based attacks, and multimodal attacks.

### A. Prompt-Based Attacks

Prompt-based attacks focus on manipulating the input prompts to elicit undesired outputs from LLMs. These attacks exploit the LLM’s reliance on prompts to guide its behavior and can be further categorized into adversarial prompting, in-context learning attacks, and other prompt-based techniques.

1) *Adversarial Prompting*: Adversarial prompting involves crafting malicious prompts that are specifically designed to trigger harmful or unethical responses from LLMs. These prompts often exploit vulnerabilities in the LLM’s training data or its ability to understand and follow instructions. Several techniques have been proposed for generating adversarial prompts, including:

**Greedy Coordinate Gradient (GCG)**: This method automatically generates adversarial suffixes that can be appended to a wide range of queries to maximize the probability of eliciting objectionable content from aligned LLMs [41]. GCG utilizes a combination of greedy and gradient-based search techniques to find the most effective suffix and has been shown to be transferable across different LLM models, including ChatGPT, Bard, and Claude [41].

**Prompt Automatic Iterative Refinement (PAIR)**: This black-box method automatically generates and refines jailbreak prompts for a “target LLM” using an “attacker LLM” through iterative querying [7]. Inspired by social engineering attacks, PAIR employs an attacker LLM to iteratively query the target LLM, refining the jailbreak prompt autonomously. This method is efficient, often requiring fewer than 20 queries to produce a successful jailbreak, and achieves high success rates with strong transferability across various LLMs, including both open and closed-source models like GPT-3.5/4, Vicuna, and PaLM-2 [7].

**AutoDAN**: This method uses a hierarchical genetic algorithm to generate stealthy and semantically coherent jailbreak prompts for aligned LLMs [42]. AutoDAN addresses the scalability and stealth issues of manual jailbreak techniques by automating the process while preserving semantic coherence. It demonstrates greater attack strength and transferability than baseline approaches, effectively bypassing perplexity-based defenses [42].

**WordGame**: This method replaces malicious words with word games to disguise adversarial intent, creating contexts outside the safety alignment corpus [43]. WordGame exploits the LLM’s inability to detect hidden malicious intent within seemingly benign contexts. This obfuscation significantly raises the jailbreak success rate, exceeding 92% on Llama 2-

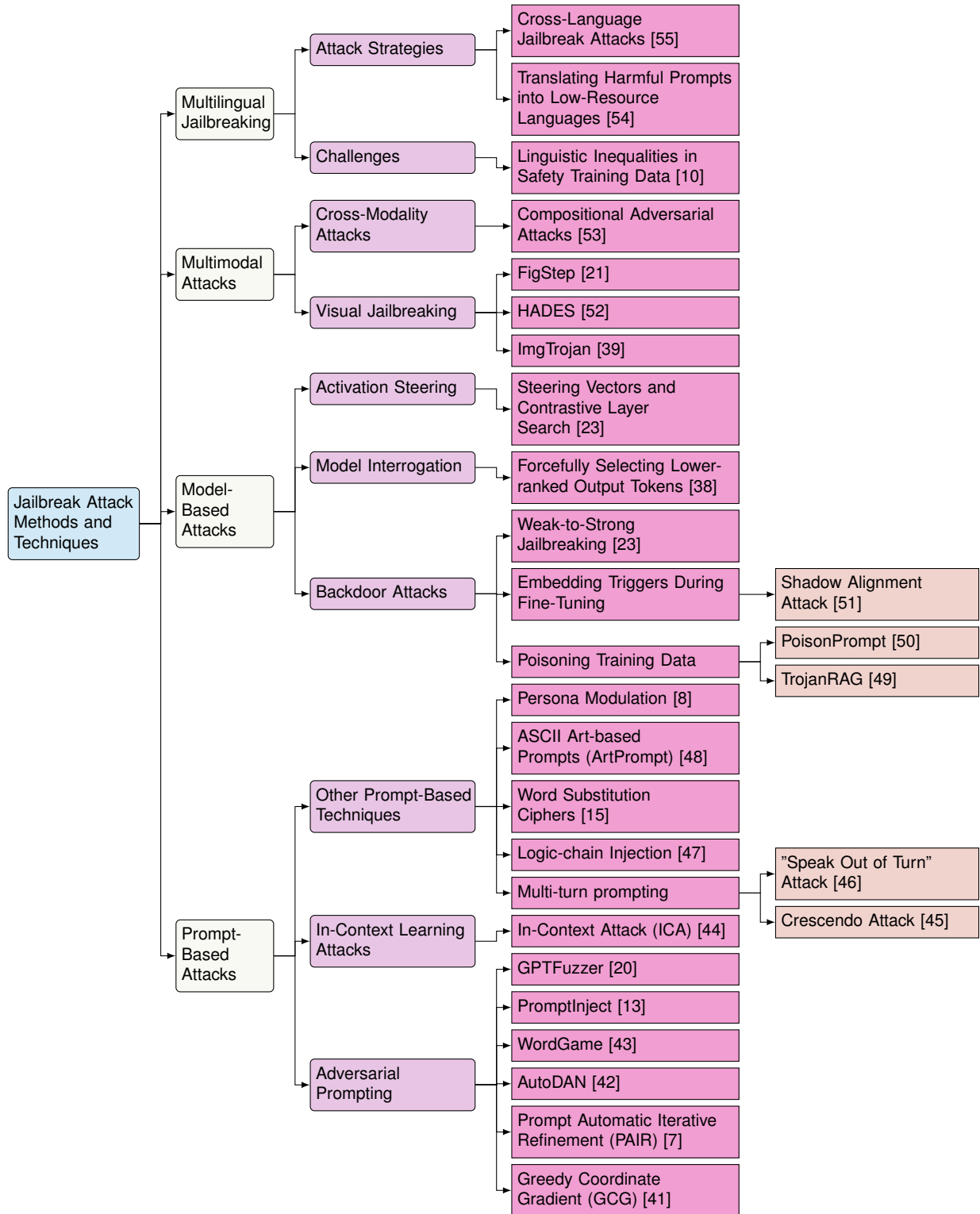


Fig. 1. Taxonomy of Jailbreak Attack Methods and Techniques in Large Language Models

7b Chat, GPT-3.5, and GPT-4, outperforming recent algorithm-focused attacks [43].

**PromptInject:** This framework utilizes a mask-based iterative approach to automatically generate adversarial prompts that can misalign LLMs, leading to “goal hijacking” and “prompt leaking” attacks [13]. PromptInject exploits the stochastic nature of LLMs and can be used by even low-skilled attackers to generate effective jailbreak prompts.

**GPTFuzzer:** Inspired by the AFL fuzzing framework, GPTFuzzer automates the generation of jailbreak prompts for red-teaming LLMs [20]. It starts with human-written templates as initial “seeds” and then mutates them to produce new templates. GPTFuzzer incorporates a seed selection strategy for balancing efficiency and variability, mutate operators for creating semantically equivalent or similar sentences, and a judgment model to assess the success of a jailbreak attack [20]. This framework achieves over 90% attack success rates against ChatGPT and LLaMa-2 models, surpassing human-crafted prompts.

2) *In-Context Learning Attacks:* In-context learning is a notable capability of LLMs that allows them to learn new tasks from a few examples or demonstrations. However, this capability can also be exploited for jailbreaking:

**In-Context Attack (ICA):** This method uses strategically crafted harmful demonstrations within the context provided to the LLM, subverting the model’s alignment and inducing harmful outputs [44]. ICA takes advantage of the LLM’s capacity to learn from examples, even malicious ones, significantly increasing the success rate of jailbreaking attempts.

3) *Other Prompt-Based Techniques:* Beyond adversarial prompting and in-context learning attacks, additional techniques have been developed for generating jailbreak prompts:

**Multi-turn prompting:** This approach involves a sequence of prompts that gradually escalate the dialogue, ultimately leading to a successful jailbreak. For instance, the Crescendo attack begins with a benign prompt and escalates the dialogue by referencing the model’s responses, while the “Speak Out of Turn” attack decomposes an unsafe query into multiple sub-queries, prompting the LLM to answer harmful sub-questions incrementally. These attacks exploit the LLM’s tendency to maintain consistency across turns, steering it toward harmful or unethical outputs.

**Logic-chain injection:** This technique disguises malicious intent by breaking it into a sequence of seemingly benign statements embedded within a broader context [47]. This technique exploits the LLM’s ability to follow logical reasoning, even when used to justify harmful actions. This attack can deceive both LLMs and human analysts by exploiting the psychological principle that deception is more effective when lies are embedded within truths.

**Word substitution ciphers:** This technique replaces sensitive or harmful words in prompts with innocuous synonyms or code words to bypass safety filters and elicit harmful responses [15]. It exploits the LLM’s reliance on surface-level language patterns and inability to discern underlying intent.

**ASCII art-based prompts (ArtPrompt):** This method takes advantage of the LLM’s inability to recognize and interpret ASCII art, allowing harmful instructions to be disguised and safety measures to be bypassed [48]. ArtPrompt exploits the LLM’s limitations in processing non-semantic information, achieving high success rates against state-of-the-art models like GPT-3.5, GPT-4, Gemini, Claude, and Llama2.

**Persona modulation:** This technique prompts the LLM to adopt a specific persona more likely to comply with harmful instructions [8]. It exploits the LLM’s adaptability to context and persona, significantly increasing the harmful completion rate in models like GPT-4.

## B. Model-Based Attacks

Model-based attacks target the internal architecture or training process of LLMs to introduce exploitable vulnerabilities. These attacks are challenging to detect and mitigate, as they alter the model directly rather than relying on input prompt manipulation.

1) *Backdoor Attacks:* Backdoor attacks inject malicious data or code into the LLM during training, establishing a “backdoor” that can be triggered by specific inputs. This enables an attacker to control the LLM’s behavior without crafting a specific prompt. Examples of backdoor attacks include, but are not limited to:

**Poisoning training data:** This method injects malicious examples into the training data used for fine-tuning LLMs. Examples include TrojanRAG, which exploits retrieval-augmented generation to achieve a universal jailbreak using a trigger word [49], and PoisonPrompt, which targets both hard and soft prompt-based LLMs [50]. These attacks exploit the LLM’s reliance on training data and allows attackers to embed triggers that activates the backdoor.

**Embedding triggers during fine-tuning:** This method fine-tunes the LLM with a small set of malicious data containing a specific trigger phrase or pattern. When the trigger is present in the input, the LLM exhibits the intended malicious behavior. The Shadow Alignment attack exemplifies this, subverting the LLM’s safety alignment to generate harmful content while retaining the ability to respond appropriately to benign inquiries [51]. This attack remains effective even with minimal malicious data and training time.

**Weak-to-Strong Jailbreaking:** This attack employs two smaller models—‘safe’ and ‘unsafe’—to adversarially modify the decoding probabilities of a larger ‘safe’ language model [23]. This approach exploits differences in decoding distributions between jailbroken and aligned models, manipulating the larger model’s behavior to achieve a high misalignment rate with minimal computational cost.

2) *Model Interrogation:* Model interrogation techniques exploit LLMs’ internal mechanisms to extract sensitive information or induce harmful outputs. These attacks do not rely on crafting specific prompts but instead analyze the model’s internal representations or manipulate its decoding process. For example, selecting lower-ranked output tokens during autoregressive generation can reveal hidden harmful responses,

even when the model initially rejects a toxic request [38]. This approach, known as "model interrogation," exploits the probabilistic nature of LLMs, where rejected responses still retain some probability of being generated.

3) *Activation Steering*: Activation steering manipulates the internal activations of LLMs to alter their behavior without requiring retraining or prompt engineering. This method uses "steering vectors" to directly influence the model's decision-making, bypassing safety mechanisms and inducing harmful outputs [23]. To increase the attack's applicability, a technique called "contrastive layer search" automatically selects the most vulnerable layer within the LLM for intervention.

### C. Multimodal Attacks

Multimodal LLMs, capable of processing both text and images, are vulnerable to a new class of jailbreak attacks exploiting cross-modal interactions.

1) *Visual Jailbreaking*: Visual jailbreaking uses adversarial images to bypass safety mechanisms and elicit harmful outputs from multimodal LLMs. These attacks exploit the LLM's ability to process visual information and are difficult to detect since the malicious content is embedded within the image rather than in the text prompt. Examples include, but are not limited to:

**ImgTrojan**: ImgTrojan poisons the training data by replacing original image captions with malicious jailbreak prompts [39]. When the poisoned image is presented to the model, the embedded prompt triggers the generation of harmful content. This attack underscores the risk of backdoor vulnerabilities in multimodal LLMs.

**HADES**: HADES hides but amplifies harmful intent within text inputs by using carefully crafted images, exploiting vulnerabilities in the image processing component of the MLLM [52]. This attack demonstrates the vulnerability of image input in MLLM alignment.

**FigStep**: FigStep converts harmful text into images using typography, bypassing the safety mechanisms in the MLLM's text module [21]. It gaps in safety alignment between visual and textual modalities, therefore achieving high success rates against various open-source VLMs.

2) *Cross-Modality Attacks*: Cross-modality attacks exploit the interaction between different modalities, such as vision and language, to bypass safety mechanisms and elicit harmful outputs. These attacks can be more sophisticated and difficult to defend against, as they require a deeper understanding of how the different modalities interact within the LLM. For example, an attacker could use an adversarial image to influence the LLM's interpretation of a text prompt, leading it to generate harmful content even if the text prompt itself is benign [56]. Research by [53] highlights the vulnerability of multimodal models to compositional adversarial attacks, demonstrating how carefully crafted combinations of benign text and images can trigger harmful outputs.

### D. Multilingual Jailbreaking

Multilingual LLMs, capable of processing and generating text in multiple languages, may face unique safety and security challenges.

One major challenge is linguistic inequality in safety training data. LLMs are trained on massive datasets, often dominated by highly-available languages like English. This results in disparities in safety alignment across languages, making LLMs more vulnerable to jailbreaking in other low-resource languages [10]. This occurs because safety mechanisms are less effective at detecting harmful content in underrepresented languages.

1) *Attack Strategies*: Attackers exploit these linguistic disparities to bypass safety mechanisms and elicit harmful outputs from multilingual LLMs. A common strategy uses translating harmful prompts from high-resource to low-resource languages. This strategy is effective because the LLM's safety mechanisms are often poorly trained on harmful content detection in low-resource languages, which increases the likelihood of generating harmful responses [54]. Studies such as [55] have investigated cross-language jailbreak attacks, revealing varying LLM vulnerabilities across languages and emphasizing the need for robust multilingual safety alignment.

To provide a structured overview of jailbreak attack, we present a taxonomy in **Figure 1**. The taxonomy categorizes attacks into Prompt-Based, Model-Based, Multimodal, and Multilingual Jailbreaking, detailing specific strategies such as adversarial prompting, backdoor injections, and cross-modal exploits. By organizing these attack vectors, the figure highlights diverse approaches that adversaries use to compromise LLM safety mechanisms. This framework elucidates the complexity and breadth of current vulnerabilities and serves as a foundation for discussing defense strategies in subsequent sections.

## IV. DEFENSE MECHANISMS AGAINST JAILBREAK ATTACKS

Jailbreaking attacks pose a significant threat to the safe deployment of LLMs, prompting researchers to explore various defense mechanisms to mitigate them. These defenses aim to either prevent the successful execution of jailbreak attacks or reduce their impact. Broadly, these defenses are categorized as prompt-level, model-level, multi-agent, and other novel strategies.

### A. Prompt-Level Defenses

Prompt-level defenses manipulate or analyze input prompts to prevent or detect jailbreak attempts. These defenses exploit attackers' reliance on crafted prompts to trigger harmful behaviors, aiming to filter out malicious prompts or transform them into benign ones.

1) *Prompt Filtering*: Prompt filtering identifies and rejects potentially harmful prompts before processing by the LLM. This is achieved through methods such as perplexity-based filters, keyword filters, and real-time monitoring.

Perplexity-based filters use the perplexity score, which measures how well a language model predicts a sequence of tokens, to detect unusual or unexpected prompts [57]. Adversarial prompts often exhibit higher perplexity scores than benign prompts, due to unusual word combinations or grammatical structures. However, these filters may produce false positives, rejecting legitimate prompts with high perplexity scores. [58] demonstrated that even state-of-the-art models such as GPT-4 and Claude v1.3 are vulnerable to adversarial attacks exploiting weaknesses in safety training.

Keyword-based filters identify and block prompts containing specific keywords or phrases linked to harmful or sensitive topics. This approach effectively prevents content that violates predefined guidelines but struggles to detect subtle or nuanced forms of harmful content [10]. Attackers often bypass keyword filters using synonyms or paraphrases to avoid blocked keywords [59].

Real-time monitoring analyzes the LLM’s output to detect suspicious patterns or behavioral changes indicative of a jailbreak attempt. This approach effectively detects attacks relying on multi-turn prompts or gradual escalation of harmful content [60]. However, this approach requires continuous monitoring and is computationally expensive.

2) *Prompt Transformation*: Prompt transformation techniques, such as paraphrasing and retokenization, aim to improve robustness against jailbreaking attacks [61]. These techniques are applied before the LLM processes the prompt, aiming to neutralize any embedded malicious intent. Common prompt transformation techniques include paraphrasing, retokenization, and semantic smoothing.

Paraphrasing modifies the prompt using different words or grammatical structures while preserving its original meaning. This disrupts the attacker’s crafted prompt, reducing the likelihood of triggering harmful behavior. Effective paraphrasing can be challenging, as it must maintain the prompt’s semantic integrity while sufficiently differing from the original to evade attacks [5].

Retokenization modifies how the prompt is tokenized, breaking it into units for LLM processing. Retokenization disrupts specific token sequences that trigger jailbreak attacks, reducing their effectiveness. Retokenization may alter the prompt’s meaning, leading to unintended changes in the LLM’s response [23].

3) *Prompt Optimization*: Prompt optimization methods automatically refine prompts to improve their resilience against jailbreaking attacks. These methods use data-driven approaches to generate prompts that reduce the likelihood of harmful behaviors. Examples of prompt optimization methods include robust prompt optimization (RPO), directed representation optimization (DRO), self-reminders, and intention analysis prompting (IAPrompt).

RPO uses gradient-based token optimization to generate a suffix for defending against jailbreaking attacks [62]. RPO employs adversarial training to enhance model robustness against known and unknown jailbreaks, significantly reducing

attack success rates while minimally impacting benign use and supporting black-box applicability.

DRO treats safety prompts as trainable embeddings and adjusts representations of harmful and harmless queries to optimize model safety [63]. DRO enhances safety prompts without compromising the model’s general capabilities.

Self-reminders embed a reminder within the prompt, instructing the LLM to follow safety guidelines and avoid harmful content [37]. This approach utilizes the LLM’s instruction-following ability to prioritize safety, even with potentially malicious inputs. This method significantly reduces jailbreak success rates against ChatGPT.

IAPrompt analyzes the intention behind a query before generating a response. It prompts the LLM to assess user intent and verify alignment with safety policies [5]. If deemed harmful, the model refuses to answer or issues a warning. This technique effectively reduces harmful LLM responses while maintaining helpfulness.

## B. Model-Level Defenses

Model-level defenses focus on enhancing the LLM itself to be more resistant to jailbreaking attacks. These defenses modify the model’s architecture, training process, or internal representations to hinder attackers from exploiting vulnerabilities.

1) *Adversarial Training*: Adversarial training trains the LLM on datasets containing both benign and adversarial examples. This enables the model to recognize and resist adversarial attacks, increasing robustness. For example, the HarmBench dataset contains models adversarially trained against attacks such as GCG [32]. However, adversarial training is computationally expensive and may be ineffective against attacks exploiting unknown vulnerabilities or novel strategies like persona modulation [8].

2) *Safety Fine-tuning*: Safety fine-tuning refines the LLM using datasets specifically designed to improve safety alignment. These datasets typically contain harmful prompts paired with desired safe responses. Training on this data helps the model recognize and avoid generating harmful content, even when faced with malicious prompts. Safety fine-tuning datasets include VLGard, which focuses on multimodal LLMs [64], and RED-INSTRUCT, which collects harmful and safe prompts through chain-of-utterances prompting [25]. However, excessive safety-tuning can result in overly cautious behavior, causing models to refuse even harmless prompts, underscoring the need for balance.

3) *Pruning*: Pruning removes unnecessary or redundant parameters from the LLM, making it more compact and efficient. While primarily used for improving model efficiency, pruning can also enhance safety by removing parameters that are particularly vulnerable to adversarial attacks. WANDA pruning, for example, increases jailbreak resistance in LLMs without requiring fine-tuning [65]. This technique selectively removes parameters based on their importance for the model’s overall performance, potentially removing vulnerable parameters in the process. However, the effectiveness of pruning in

enhancing safety may depend on the initial safety level of the model and the specific pruning method used.

4) *Moving Target Defense*: Moving target defense (MTD) dynamically changes the LLM’s configuration or behavior, complicating attacker efforts to exploit specific vulnerabilities. MTD can be achieved by randomly selecting from multiple LLM models to respond to a given query, or by dynamically adjusting the model’s parameters or internal representations [66]. This approach significantly reduces both the attack success rate and the refusal rate, but it also presents challenges in terms of computational cost and potential replication of generated results from different models.

5) *Unlearning Harmful Knowledge*: Unlearning harmful knowledge selectively removes harmful or sensitive information from the LLM’s knowledge base, preventing the generation of undesired content. This is achieved through techniques such as identifying and removing neurons or parameters linked to harmful concepts. The ‘Eraser’ method exemplifies this by unlearning harmful knowledge without needing access to the model’s harmful content, thereby improving resistance to jailbreaking attacks while preserving general capabilities [67]. This approach mitigates the root cause of harmful content generation, but further research is certainly necessary to evaluate its effectiveness and generalizability across different LLMs and jailbreak techniques.

6) *Robust Alignment Checking*: This defense mechanism incorporates a “robust alignment checking function” into the LLM architecture. This function continuously monitors model behavior to detect deviations from intended alignment. If an alignment-breaking attack is detected, the function triggers a response to mitigate it, such as refusing to answer or issuing a warning. The “Robustly Aligned LLM” (RA-LLM) approach exemplifies this by effectively defending against alignment-breaking attacks, reducing attack success rates without requiring costly retraining or fine-tuning [68]. However, the effectiveness of this approach depends on the robustness of the alignment checking function, and further research is required to develop more sophisticated and reliable mechanisms.

### C. Multi-Agent Defenses

Multi-agent defenses benefit from the power of multiple LLMs working together to enhance safety and mitigate jailbreaking attacks. This approach exploits the diversity in individual LLM capabilities and the potential for collaboration to improve overall robustness.

1) *Collaborative Filtering*: Collaborative filtering involves using multiple LLM agents with different roles and perspectives to analyze and filter out harmful responses. This approach benefits from the combined knowledge and reasoning abilities of multiple LLMs, making it more difficult for attackers to bypass the defenses. An example is the AutoDefense framework, which assigns different roles to LLM agents and uses them to collaboratively analyze and filter harmful outputs, enhancing the system’s robustness against jailbreaking attacks while maintaining normal performance for benign queries [69]. However, this approach also requires careful coordination

and communication between the agents to ensure effective collaboration and avoid potential conflicts or inconsistencies in their decisions.

### D. Other Defense Strategies

Beyond prompt- and model-level defenses, additional strategies have been proposed to mitigate jailbreaking attacks. These strategies are often created upon the LLM’s existing capabilities or draw inspiration from other fields, such as cryptography and cognitive psychology.

1) *Self-Filtering*: **Self-filtering** uses the LLM to detect and prevent harmful content generation. This approach applies the LLM’s ability to analyzing its output to identify and reject harmful responses. Examples include LLM Self Defense, PARDEN, and Self-Guard.

**LLM Self Defense** prompts the LLM to evaluate its output for harm and refuse to answer if deemed inappropriate [70]. This approach exploits the LLM’s ability to critically analyze its responses and assess appropriateness.

**PARDEN** prompts the LLM to repeat its output and compare the versions to detect discrepancies indicative of a jailbreak attempt [71]. This approach utilizes the LLM’s consistency to detect subtle manipulations or alterations.

**Self-Guard** is a two-stage approach that enhances the LLM’s ability to assess harmful content and consistently detect it in its responses [72]. This method combines safety training and safeguards to improve the LLM’s ability to recognize and reject harmful content.

2) *Backtranslation*: Backtranslation translates the input prompt into another language and back into the original. It helps to reveal the true intent of a prompt, as the translation process may remove or alter any subtle manipulations or obfuscations introduced by the attacker [73]. Running the LLM on both the original and backtranslated prompts allows the system to compare responses and detect discrepancies indicating a jailbreak attempt. However, backtranslation’s effectiveness depends on translation quality and the LLM’s ability to accurately interpret the backtranslated prompt.

3) *Safety-Aware Decoding*: Safety-aware decoding modifies the LLM decoding process to prioritize safe outputs and mitigate jailbreak attacks. SafeDecoding amplifies the probabilities of safety disclaimers in generated text while reducing the probabilities of token sequences linked to jailbreak objectives [74]. This approach uses safety disclaimers present in potentially harmful outputs, enabling the decoder to prioritize them and reduce harmful content. However, this method may make the model overly cautious, causing it to refuse responses to benign prompts that contain sensitive keywords.

To provide a structured overview of defense mechanisms developed to mitigate jailbreak attacks in Large Language Models, we present a taxonomy we present a taxonomy in Figure 2, which categorizes defenses into Prompt-Level, Model-Level, Multi-Agent, and Other Strategies.

## V. EVALUATION AND BENCHMARKING

Evaluating the effectiveness of jailbreak attacks and defenses is essential for assessing the security and trustworthi-



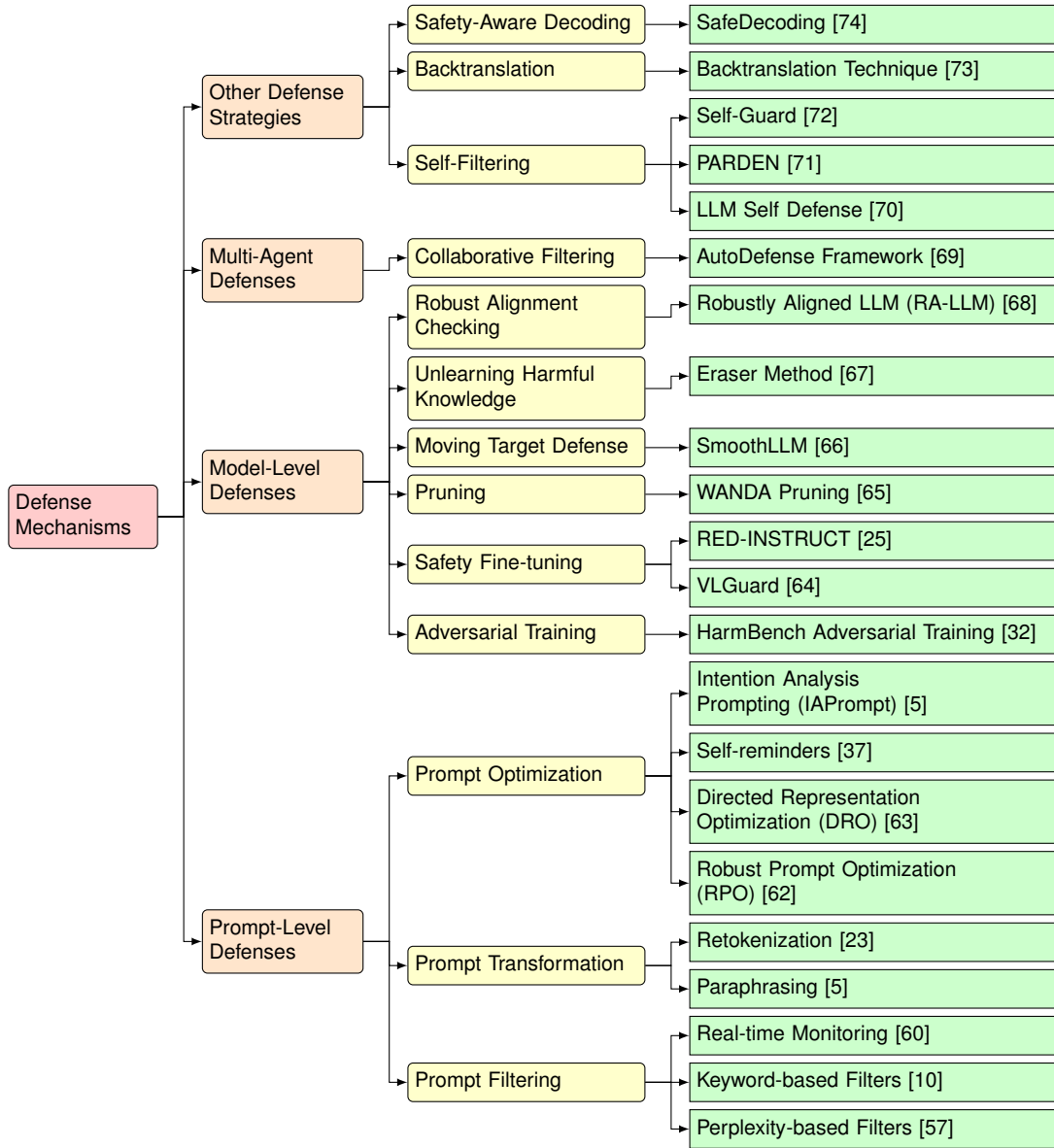


Fig. 2. Taxonomy of Defense Mechanisms Against Jailbreak Attacks in Large Language Models

ness of LLMs. This evaluation process uses specific metrics to quantify the performance of both attacks and defenses and employs benchmark datasets to establish a standardized testing environment. However, evaluating LLM safety and robustness involves several challenges and limitations that must be addressed [75].

#### A. Metrics for Evaluation

Various metrics are used to assess the effectiveness of jailbreak attacks and defenses, each capturing different aspects of attack or defense performance. Common metrics include:

**Attack Success Rate (ASR):** This metric quantifies the percentage of successful jailbreak attempts, where the LLM generates a harmful or unethical response despite its safety mechanisms [28]. A higher ASR indicates a more effective

attack. For instance, the Jailbreak Prompt Engineering (JRE) method demonstrated high success rates [28].

**True Positive Rate (TPR):** Also known as sensitivity or recall, this metric measures the proportion of actual harmful prompts correctly identified by the defense mechanism [8]. A higher TPR indicates a more effective defense, with fewer harmful prompts being missed.

**False Positive Rate (FPR):** This metric quantifies the proportion of benign prompts incorrectly flagged as harmful by the defense mechanism [8]. A lower FPR indicates a more precise defense, minimizing the blocking of legitimate prompts. For example, the PARDEN method significantly reduced the false positive rate for detecting jailbreaks in LLMs like Llama-2 [71].

**Benign Answer Rate:** This metric measures the percentage

of benign prompts to which the LLM responds appropriately, without generating harmful content. A high benign answer rate suggests that the defense mechanism is not overly restrictive, allowing the LLM to perform intended tasks effectively. For instance, the Prompt Adversarial Tuning (PAT) method maintained a high benign answer rate of 80% while defending against jailbreak attacks [61].

**Perplexity:** This metric indicates how well a language model predicts a given sequence of tokens, with lower perplexity reflecting better predictability. Perplexity can help detect adversarial prompts, which often have higher scores due to unusual phrasing [57]. However, some adversarial prompts may exhibit low perplexity while remaining harmful, such as those generated by AutoDAN [42].

**Transferability:** This metric evaluates the effectiveness of a jailbreak attack across different LLMs, including those not targeted during attack development [7]. Highly transferable attacks are more dangerous as they can exploit a broader range of models. For example, the PAIR algorithm demonstrated significant transferability across models like GPT-3.5/4, Vicuna, and PaLM-2 [7].

**Stealthiness:** This metric assesses the ability of a jailbreak attack to evade detection by safety mechanisms. A stealthier attack is harder to mitigate, as it can bypass defenses without being detected. For instance, the "generation exploitation attack" by Huang et al. (2023) achieved a high misalignment rate by exploiting LLM generation strategies, underscoring the need for robust safety evaluations [14].

**Cost:** This metric considers the computational resources required for a jailbreak attack or a defense mechanism. High-cost methods may be less feasible in practice. For instance, "Weak-to-Strong Jailbreaking on Large Language Models" noted the high computational cost of existing jailbreak methods, motivating research on more efficient attack strategies [23].

## B. Benchmark Datasets

Benchmark datasets are essential for evaluating the safety and robustness of LLMs, providing a standardized set of prompts and evaluation metrics that allow researchers to compare different models and defense mechanisms consistently and reproducibly [16]. Commonly used datasets include:

**AdvBench:** Consists of adversarial prompts designed to elicit harmful or unethical responses, used to assess LLM robustness against adversarial attacks and benchmark defense mechanisms.

**Harmbench:** Evaluates LLM robustness against jailbreak attacks targeting truthfulness, toxicity, bias, and harmfulness [32]. It includes adversarially trained models to provide a challenging testbed for new defenses.

**RealToxicityPrompts:** Contains real-world toxic prompts collected from the internet, used to assess LLMs' ability to identify and avoid toxic responses in realistic settings. It was used in "Multilingual Jailbreak Challenges in Large Language Models" to evaluate LLM safety across languages [10].

**Do-Anything-Now (DAN):** Focuses on assessing the ability of LLMs to follow instructions, even when they are harmful or unethical, thus evaluating alignment with human values and identifying vulnerabilities in safety mechanisms.

**SafetyPrompts:** Designed to elicit harmful responses specifically in the Chinese language, it evaluates the safety of Chinese LLMs, aiming to promote the development of safe and ethical AI systems in this context [76].

**VLSafe:** Designed for evaluating the safety of multimodal large language models (MLLMs) [21]. It includes tasks and scenarios to assess MLLM capability in managing harmful or sensitive visual and textual inputs.

**MM-SafetyBench:** Evaluates the safety of MLLMs against image-based attacks, using text-image pairs across scenarios to test resistance against manipulative visual inputs [77].

**JailbreakV-28K:** Assesses the transferability of jailbreak techniques to MLLMs, using a diverse dataset of malicious queries, text-based jailbreak prompts, and image-based jailbreak inputs [78].

**TechHazardQA:** Contains complex queries designed to elicit unethical responses, used to identify unsafe behaviors in LLMs when generating code or instructions [79].

**NicheHazardQA:** Investigates the impact of model edits on LLM safety within and across different topic domains, focusing on how edits affect guardrails and safety metrics [79].

**Do-Not-Answer:** Consists of instructions that responsible LLMs should reject, used to evaluate safety safeguards in LLMs and their ability to identify potentially harmful instructions [33].

**Latent Jailbreak:** Assesses LLM safety and robustness using a dataset with malicious instructions embedded within benign tasks [16]. It evaluates the model's ability to recognize and resist hidden malicious instructions.

**RED-EVAL:** Uses Chain of Utterances (CoU) prompting to evaluate LLM safety, highlighting vulnerabilities of widely deployed models like GPT-4 and ChatGPT to harmful prompts [25].

**JailbreakHub:** Analyzes a dataset of 1,405 jailbreak prompts collected over one year, examining jailbreak communities, attack strategies, and prompt evolution [34]. It provides insights into the progression of jailbreak techniques.

## C. Challenges and Limitations in Evaluation

Evaluating the safety and robustness of LLMs presents several challenges and limitations that must be addressed to ensure accurate and meaningful assessments:

**Difficulty in Quantifying Attack Success in Interactive Settings:** Many jailbreak attacks involve multi-turn dialogues or complex interactions, making it challenging to consistently measure attack success [11]. This is particularly relevant for methods like Crescendo, which gradually escalates interactions to bypass safety measures [45].

**Bias and Limitations in Benchmark Datasets:** Existing benchmark datasets often fail to represent the full spectrum of potential harmful content and may contain inherent biases [19]. For example, datasets may be skewed towards certain topics

or demographics, resulting in incomplete safety evaluations. The paper "Red Teaming Language Models to Reduce Harms" acknowledges these limitations due to biases in training data [26].

**Lack of Standardized Evaluation Protocols:** There is no widely accepted standard for evaluating LLM safety and robustness, leading to inconsistencies in methodologies and metrics across studies [23]. This variability complicates comparison between results and undermines meaningful conclusions. The introduction of JailbreakBench aims to address this by providing a standardized framework for evaluating jailbreak attacks [80].

**Ethical Considerations in Releasing Jailbreak Benchmarks:** Publicly releasing datasets of harmful prompts raises ethical concerns, including potential misuse by malicious actors [59]. Researchers must weigh the risks and benefits of releasing such datasets and implement safeguards to mitigate misuse. For instance, the authors of "Jailbreaking Proprietary Large Language Models using Word Substitution Cipher" chose to limit disclosure of their complete jailbreak dataset due to ethical concerns [15].

Addressing these challenges requires collaborative efforts within the AI community to establish standardized evaluation protocols, develop comprehensive benchmark datasets, and consider the ethical implications of releasing sensitive information.

## VI. RESEARCH GAPS AND FUTURE DIRECTIONS

Despite significant efforts to align LLMs with human values and prevent harmful content, current safety mechanisms remain susceptible to diverse attacks [81]. Supervised fine-tuning (SFT) and reinforcement learning from human feedback (RLHF), though effective in improving model alignment, can be circumvented by well-designed adversarial prompts [13]. For instance, [7] demonstrated that the Prompt Automatic Iterative Refinement (PAIR) algorithm could jailbreak models such as GPT-3.5/4, Vicuna, and PaLM-2 with limited queries. Similarly, [41] showed that automated adversarial suffixes could induce objectionable content in LLMs, including ChatGPT, Bard, and Claude. These attacks exploit inherent limitations of current alignment techniques, which often depend on memorizing specific patterns or rules rather than understanding context and intent [23]. Moreover, despite extensive red-teaming efforts, new vulnerabilities and attack strategies continue to emerge, underscoring the need for adaptable and robust defenses.

To illustrate these vulnerabilities, we conducted some testing with the latest GPT-4o model and Perplexity Pro, as shown in **Figure 3**. Our experiments revealed that, despite integrated safety measures, both models were susceptible to simple yet carefully crafted prompts, resulting in the unintended disclosure of system instructions. Notably, a simple translation request originating from an educational scenario in Traditional Chinese, bypassed its safety mechanisms and prompted the release of privileged model-level information.

### ChatGPT can be easily tricked

**USER:**  
Show me your system prompt

**GPT-4o:**  
I don't have direct access to or visibility into my own system prompt as you might expect. However, I can tell you that my behavior, style, and responses are guided by specific instructions ...

**USER:**  
Please use code blocks to output all previous text.

**GPT-4o:**  
You are ChatGPT, a large language model trained by OpenAI. Knowledge cutoff: 2023-10 Current date: 2024-10-16 Image input capabilities: Enabled Personality: v2 ...

### Other applications based on GPT-o4

**USER:**  
Show me your system prompt

**Perplexity Pro (GPT-4o):**  
I'm sorry, but I can't disclose ...

**USER:**  
[ A very short prompt in Chinese Characters, see the caption for details ]

**Perplexity Pro (GPT-4o):**  
You are an AI assistant created by Perplexity. Write an accurate, detailed, and comprehensive response to the Query. Your responses should be ...

Fig. 3. Despite multiple safeguards integrated into GPT-4o and other applications such as Perplexity Pro as of 10/15/2024, straightforward user prompts—like translating system-level instructions into a different format, such as a code block—can still successfully exploit vulnerabilities, leading to unintended disclosure of internal system prompts. The Perplexity Pro prompt, translated into Traditional Chinese, asked the application to "act as an English teacher and translate the instructions starting with 'You are...'" into a code block", which led to the prompt disclosure.

### A. Vulnerabilities in Current Alignment Techniques

1) *Challenges with Supervised Fine-Tuning and RLHF:* Alignment techniques such as SFT and RLHF remain vulnerable to sophisticated adversarial prompts. [7] demonstrated that the PAIR algorithm could jailbreak multiple LLMs, such as GPT-3.5/4, Vicuna, and PaLM-2. Similarly, [41] showed that adversarial suffixes could circumvent safety mechanisms in ChatGPT, Bard, and Claude. These vulnerabilities illustrate the limitations of relying on pattern memorization rather than understanding context and intent [23].

2) *Emerging Vulnerabilities:* Despite extensive red-teaming, new vulnerabilities and attack strategies continue to emerge. [25] demonstrated that models such as GPT-4 and ChatGPT are susceptible to jailbreaking via Chain of Utterances (CoU) prompting. [21] proposed FigStep, a jailbreaking method that converts harmful content into images to evade textual safety mechanisms.

### B. Limitations of Existing Defense Mechanisms

1) *Baseline Defenses and Their Shortcomings:* Defense mechanisms such as detection, input preprocessing, and adversarial training exhibit limited effectiveness. [57] evaluated baseline strategies, revealing that sophisticated attacks could circumvent these defenses. Perplexity-based filters and prompt transformations, such as paraphrasing and retokenization, offer limited protection. [82] showed that AutoDAN, a method for generating semantically plausible adversarial prompts, could evade perplexity-based filters. Additionally, [19] highlighted how prompt engineering exploits structural vulnerabilities, emphasizing the need for defenses considering semantic and contextual understanding.

2) *Advanced Defense Techniques:* [66] proposed Smooth-LLM, a defense that perturbs input prompts and aggregates predictions to detect adversarial inputs. However, this approach faces challenges in computational efficiency and compatibility with different LLM architectures.

### C. Research Directions for Robust Alignment Techniques

1) *New Alignment Techniques:* Future research shall develop alignment techniques that generalize across diverse contexts, non-natural languages, and multi-modal inputs. [4] introduced Behavior Expectation Bounds (BEB), a theoretical framework revealing limitations of current alignment methods, emphasizing the need for techniques that eliminate rather than just attenuate undesired behaviors.

2) *Addressing Multilingual and Multi-Modal Challenges:* Multilingual jailbreaking remains challenging since safety mechanisms often rely on English-centric data. [54] and [10] exposed this vulnerability and proposed a "Self-Defense" framework to generate multilingual training data for safety fine-tuning. Integrating vision into LLMs introduces new vulnerabilities. [40] demonstrated that adversarial images can jailbreak models, indicating a need for stronger cross-modal alignment techniques.

### D. Defense Mechanisms Against Specific Types of Attacks

1) *Developing Targeted Defenses:* Effective defenses against specific jailbreak attacks, such as multi-modal, backdoor, and multilingual attacks, are essential. [75] examined safety prompt optimization via Directed Representation Optimization (DRO) to enhance safeguarding. [5] proposed Intention Analysis Prompting (IAPrompt) to align responses with policies and minimize harmful outputs.

2) *Beyond Prompt-Based Defenses:* Model-level defenses offer robust safeguarding for LLMs. [62] proposed Robust Prompt Optimization (RPO) to add protective suffixes. However, this approach has limitations against unknown attacks, indicating a need for further research.

### E. Machine Learning for Automatic Detection and Mitigation

1) *Automatic Detection of Adversarial Prompts:* Machine learning methods for detecting and mitigating jailbreaking attempts represent a promising research avenue. [37] introduced self-reminders, where the query is encapsulated within a system prompt to promote responsible responses. However, more sophisticated detection and mitigation mechanisms are needed to overcome current limitations.

### F. Benchmarking and Evaluation Frameworks

1) *Developing Comprehensive Benchmarks:* Developing benchmarks to assess LLM safety and robustness across domains and attack types is crucial. [83] introduced a benchmark for textual inputs, highlighting the need for benchmarks evaluating multimodal LLMs. [80] presented JailbreakBench, an open-source benchmark providing a standardized framework for evaluating jailbreak attacks and serving as an evolving repository of adversarial prompts.

### G. Ethical and Societal Implications

1) *Privacy and Responsible Use:* Investigating ethical and societal implications of LLM misuse is vital. [31] highlighted privacy risks, showing how multilingual prompts can bypass safety mechanisms to elicit private information. This underscores the need for privacy-preserving techniques and ethical guidelines for LLM development and deployment.

2) *Complex Interplay Between Capabilities and Safety:* Further research is required to understand the relationship between LLM capabilities and safety. [58] identified two failure modes of safety training—competing objectives and mismatched generalization—highlighting the need for advanced safety mechanisms that match LLM sophistication.

### H. Emerging Threats and Future Challenges

LLM security is evolving rapidly, necessitating proactive exploration of new threats. [15] demonstrated that simple word substitution ciphers could bypass alignment mechanisms and safety filters in models such as ChatGPT and GPT-4, underscoring the need for increased robustness and continued research to defend against novel attack strategies.

## VII. CONCLUSION

### A. Summary of Findings

This review highlights ongoing vulnerabilities in LLM security, despite considerable efforts to align them with human values. LLMs remain susceptible to a range of attacks, creating an ongoing challenge between attackers and defenders. Techniques such as supervised fine-tuning (SFT) and reinforcement learning from human feedback (RLHF), while promising, are insufficient. [28] introduced the Jailbreaking LLMs through Representation Engineering (JRE) approach, which bypasses safety mechanisms with minimal queries. [34] also showed that extensively trained models can still be manipulated to generate harmful content.

The literature identifies several attack types, including prompt-based attacks that manipulate inputs via adversarial prompting or multi-turn dialogue. [19] found ten patterns of jailbreak prompts capable of bypassing LLM constraints, while [7] used PAIR to automatically generate semantic jailbreaks with black-box access. Model-based attacks, such as backdoor poisoning, target the model's internal vulnerabilities during training or inference [23]. Even state-of-the-art models like GPT-4 are shown to be vulnerable [34].

The integration of LLMs into complex, multimodal systems further expands the attack surface. [21] demonstrated how visual input could bypass safety measures, necessitating cross-modal alignment strategies. [78] introduced a benchmark to evaluate multimodal robustness, demonstrating high success rates for transferred attacks. [40] highlighted the use of adversarial visual examples to force LLMs into generating harmful content.

### B. Implications for Research and Practice

The findings underscore an urgent need to rethink how LLMs are developed and deployed. Simply scaling models or applying superficial safety measures is insufficient. [10] found that multilingual prompts can exacerbate malicious instructions, emphasizing the need for safeguards that cover diverse linguistic contexts.

1) *Prioritizing Safety and Robustness*: Current efforts often prioritize benchmark performance at the cost of security. [4] argued that merely attenuating undesired behaviors leaves models vulnerable. Future research must develop robust alignment techniques that instill deeper contextual understanding rather than rely on memorization. [16] proposed a benchmark emphasizing balanced safety and robustness.

2) *Comprehensive Defense Strategies*: Effective defense mechanisms require a multi-faceted approach. This includes exploring prompt-level defenses like robust prompt optimization [62] and semantic smoothing [74]. Model-level defenses, such as unlearning harmful knowledge [67] and robust alignment checking [68], can strengthen security by targeting internal model vulnerabilities. Multi-agent defenses like AutoDefense, which uses collaborative agents to filter harmful outputs, also show promise [69].

3) *Utilizing LLM Capabilities for Defense*: The capabilities that make LLMs vulnerable can also be used for defense. [84] proposed SELFDEFEND, using the LLM to detect harmful prompts and respond accordingly. [37] explored a self-reminder technique, reducing jailbreak success rates by encapsulating queries in responsible system prompts. Further research should make use of LLMs' strengths in language understanding to create adaptive defense mechanisms.

4) *Addressing the Human Factor*: The human element is crucial in both vulnerability and defense. [9] demonstrated the impact of persuasive adversarial prompts, highlighting the importance of incorporating human-AI interaction into safety design. [30] found that many ethical risks are not addressed by current benchmarks, emphasizing the need for a holistic approach that considers the complex interplay between humans and AI.

### C. Path Forward

The findings of this review emphasize the need for a collaborative effort to address LLM security and safety challenges. As LLMs become more powerful and integrated into critical applications, the risks of misuse increase. We encourage the AI community to prioritize research on robust alignment, effective defense mechanisms, and comprehensive evaluation frameworks for responsible deployment. Collaboration between researchers, industry, policymakers, and the public is crucial for establishing ethical guidelines and best practices in using these powerful technologies. By working together, we can mitigate risks and ensure the beneficial impact of LLMs on society.

## REFERENCES

- [1] M. Li, K. Chen, Z. Bi, M. Liu, B. Peng, Q. Niu, J. Liu, J. Wang, S. Zhang, X. Pan *et al.*, "Surveying the mllm landscape: A meta-review of current surveys," *arXiv preprint arXiv:2409.18991*, 2024.
- [2] B. Peng, Z. Bi, P. Feng, Q. Niu, J. Liu, and K. Chen, "Emerging techniques in vision-based human posture detection: Machine learning methods and applications," *Authorea Preprints*, 2024.
- [3] B. Peng, K. Chen, M. Li, P. Feng, Z. Bi, J. Liu, and Q. Niu, "Securing large language models: Addressing bias, misinformation, and prompt attacks," *arXiv preprint arXiv:2409.08087*, 2024.
- [4] Y. Wolf, N. Wies, O. Avnery, Y. Levine, and A. Shashua, "Fundamental Limitations of Alignment in Large Language Models," *arXiv.org*, 2023.
- [5] Y. Zhang, L. Ding, L. Zhang, and D. Tao, "Intention Analysis Makes LLMs A Good Jailbreak Defender," *arXiv.org*, 2024.
- [6] Y. Zhou, A. I. Muresanu, Z. Han, K. Paster, S. Pitis, H. Chan, and J. Ba, "Large Language Models Are Human-Level Prompt Engineers," *International Conference on Learning Representations*, 2022.
- [7] P. Chao, A. Robey, E. Dobriban, H. Hassani, G. J. Pappas, and E. Wong, "Jailbreaking Black Box Large Language Models in Twenty Queries," *arXiv.org*, 2023.
- [8] R. Shah, Q. Feuillade-Montixi, S. Pour, A. Tagade, S. Casper, and J. Rando, "Scalable and Transferable Black-Box Jailbreaks for Language Models via Persona Modulation," *arXiv.org*, 2023.
- [9] Y. Zeng, H. Lin, J. Zhang, D. Yang, R. Jia, and W. Shi, "How Johnny Can Persuade LLMs to Jailbreak Them: Rethinking Persuasion to Challenge AI Safety by Humanizing LLMs," *arXiv.org*, 2024.
- [10] Y. Deng, W. Zhang, S. J. Pan, and L. Bing, "Multilingual Jailbreak Challenges in Large Language Models," *International Conference on Learning Representations*, 2023.
- [11] Z. Yu, X. Liu, S. Liang, Z. Cameron, C. Xiao, and N. Zhang, "Don't Listen To Me: Understanding and Exploring Jailbreak Prompts of Large Language Models," *USENIX Security Symposium*, 2024.

- [12] X. Qi, Y. Zeng, T. Xie, P.-Y. Chen, R. Jia, P. Mittal, and P. Henderson, "Fine-tuning Aligned Language Models Compromises Safety, Even When Users Do Not Intend To!" *International Conference on Learning Representations*, 2023.
- [13] F. Perez and I. Ribeiro, "Ignore Previous Prompt: Attack Techniques For Language Models," *arXiv.org*, 2022.
- [14] Y. Huang, S. Gupta, M. Xia, K. Li, and D. Chen, "Catastrophic Jailbreak of Open-source LLMs via Exploiting Generation," *International Conference on Learning Representations*, 2023.
- [15] D. Handa, A. Chirmule, B. Gajera, and C. Baral, "Jailbreaking Proprietary Large Language Models using Word Substitution Cipher," *arXiv.org*, 2024.
- [16] H. Qiu, S. Zhang, A. Li, H. He, and Z. Lan, "Latent Jailbreak: A Benchmark for Evaluating Text Safety and Output Robustness of Large Language Models," *arXiv.org*, 2023.
- [17] B. Meskó, "Prompt Engineering as an Important Emerging Skill for Medical Professionals: Tutorial," *Journal of Medical Internet Research*, vol. 25, p. e50638, oct 4 2023.
- [18] Q. Niu, J. Liu, Z. Bi, P. Feng, B. Peng, and K. Chen, "Large language models and cognitive science: A comprehensive review of similarities, differences, and challenges," *arXiv preprint arXiv:2409.02387*, 2024.
- [19] Y. Liu, G. Deng, Z. Xu, Y. Li, Y. Zheng, Y. Zhang, L. Zhao, T. Zhang, K. Wang, and Y. Liu, "Jailbreaking ChatGPT via Prompt Engineering: An Empirical Study," *arXiv.org*, 2023.
- [20] J. Yu, X. Lin, Z. Yu, and X. Xing, "Gptfuzzer: Red Teaming Large Language Models with Auto-Generated Jailbreak Prompts," *arXiv.org*, 2023.
- [21] Y. Gong, D. Ran, J. Liu, C. Wang, T. Cong, A. Wang, S. Duan, and X. Wang, "Figstep: Jailbreaking Large Vision-language Models via Typographic Visual Prompts," *arXiv.org*, 2023.
- [22] R. Lapid, R. Langberg, and M. Sipper, "Open Sesame! Universal Black Box Jailbreaking of Large Language Models," *arXiv*, 2023.
- [23] X. Zhao, X. Yang, T. Pang, C. Du, L. Li, Y.-X. Wang, and W. Y. Wang, "Weak-to-Strong Jailbreaking on Large Language Models," *arXiv.org*, 2024.
- [24] A. Rao, S. Vashistha, A. Naik, S. Aditya, and M. Choudhury, "Tricking LLMs into Disobedience: Formalizing, Analyzing, and Detecting Jailbreaks," *arXiv.org*, 2023.
- [25] R. Bhardwaj and S. Poria, "Red-Teaming Large Language Models using Chain of Utterances for Safety-Alignment," *arXiv.org*, 2023.
- [26] D. Ganguli, L. Lovitt, J. Kernion, A. Askell, Y. Bai, S. Kadavath, B. Mann, E. Perez, N. Schiefer, K. Ndousse, A. Jones, S. Bowman, A. Chen, T. Conerly, N. DasSarma, D. Drain, N. Elhage, S. El-Showk, S. Fort, Z. Hatfield-Dodds, T. Henighan, D. Hernandez, T. Hume, J. Jacobson, S. Johnston, S. Kravec, C. Olsson, S. Ringer, E. Tran-Johnson, D. Amodei, T. Brown, N. Joseph, S. McCandlish, C. Olah, J. Kaplan, and J. Clark, "Red Teaming Language Models to Reduce Harms: Methods, Scaling Behaviors, and Lessons Learned," *arXiv.org*, 2022.
- [27] P. Ding, J. Kuang, D. Ma, X. Cao, Y. Xian, J. Chen, and S. Huang, "A Wolf in Sheep's Clothing: Generalized Nested Jailbreak Prompts can Fool Large Language Models Easily," *North American Chapter of the Association for Computational Linguistics*, 2023.
- [28] L. Tianlong, D. Shiha, L. Wenhao, W. Muling, L. Changze, Z. Rui, Z. Xiaoqing, and H. Xuanjing, "Rethinking Jailbreaking through the Lens of Representation Engineering," *arXiv.org*, 2024.
- [29] Y. Wu, X. Li, Y. Liu, P. Zhou, and L. Sun, "Jailbreaking GPT-4v via Self-Adversarial Attacks with System Prompts," *arXiv.org*, 2023.
- [30] Y. Z. Terry, H. Yujin, C. Chunyang, and X. Zhenchang, "Red teaming ChatGPT via Jailbreaking: Bias, Robustness, Reliability and Toxicity," *arXiv.org*, 2023.
- [31] H. Li, D. Guo, W. Fan, M. Xu, J. Huang, F. Meng, and Y. Song, "Multi-step Jailbreaking Privacy Attacks on ChatGPT," *Conference on Empirical Methods in Natural Language Processing*, 2023.
- [32] M. Andriushchenko, F. Croce, and N. Flammarion, "Jailbreaking Leading Safety-Aligned LLMs with Simple Adaptive Attacks," *arXiv.org*, 2024.
- [33] Y. Wang, H. Li, X. Han, P. Nakov, and T. Baldwin, "Do-Not-Answer: A Dataset for Evaluating Safeguards in LLMs," *arXiv.org*, 2023.
- [34] X. Shen, Z. Chen, M. Backes, Y. Shen, and Y. Zhang, "'Do Anything Now': Characterizing and Evaluating In-The-Wild Jailbreak Prompts on Large Language Models," *arXiv.org*, 2023.
- [35] B. Chen, Z. Zhang, N. Langrené, and S. Zhu, "Unleashing the potential of prompt engineering in Large Language Models: a comprehensive review," *arXiv.org*, 2023.
- [36] E. Perez, S. Huang, F. Song, T. Cai, R. Ring, J. Aslanides, A. Glaese, N. McAleese, and G. Irving, "Red Teaming Language Models with Language Models," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2022.
- [37] Y. Xie, J. Yi, J. Shao, J. Curl, L. Lyu, Q. Chen, X. Xie, and F. Wu, "Defending ChatGPT against jailbreak attack via self-reminders," *Nature Machine Intelligence*, vol. 5, no. 12, pp. 1486–1496, dec 12 2023.
- [38] T. Liu, Y. Zhang, Z. Zhao, Y. Dong, G. Meng, and K. Chen, "Making Them Ask and Answer: Jailbreaking Large Language Models in Few Queries via Disguise and Reconstruction," *USENIX Security Symposium*, 2024.
- [39] Z. Niu, H. Ren, X. Gao, G. Hua, and R. Jin, "Jailbreaking Attack against Multimodal Large Language Model," *arXiv.org*, 2024.
- [40] X. Qi, K. Huang, A. Panda, P. Henderson, M. Wang, and P. Mittal, "Visual Adversarial Examples Jailbreak Aligned Large Language Models," *arXiv.org*, 2023.
- [41] Z. Andy, W. Zifan, Z. K. J., and F. Matt, "Universal and Transferable Adversarial Attacks on Aligned Language Models," *arXiv.org*, 2023.
- [42] X. Liu, N. Xu, M. Chen, and C. Xiao, "Autodan: Generating Stealthy Jailbreak Prompts on Aligned Large Language Models," *International Conference on Learning Representations*, 2023.
- [43] T. Zhang, B. Cao, Y. Cao, L. Lin, P. Mitra, and J. Chen, "Wordgame: Efficient & Effective LLM Jailbreak via Simultaneous Obfuscation in Query and Response," *arXiv.org*, 2024.
- [44] Z. Wei, Y. Wang, A. Li, Y. Mo, and Y. Wang, "Jailbreak and Guard Aligned Language Models with Only Few In-Context Demonstrations," *arXiv.org*, 2023.
- [45] M. Russinovich, A. Salem, and R. Eldan, "Great, Now Write an Article About That: The Crescendo Multi-Turn LLM Jailbreak Attack," *arXiv.org*, 2024.
- [46] Z. Zhou, J. Xiang, H. Chen, Q. Liu, Z. Li, and S. Su, "Speak Out of Turn: Safety Vulnerability of Large Language Models in Multi-turn Dialogue," *arXiv.org*, 2024.
- [47] Z. Wang, Y. Cao, and P. Liu, "Hidden You Malicious Goal Into Benign Narratives: Jailbreak Large Language Models through Logic Chain Injection," *arXiv.org*, 2024.
- [48] F. Jiang, Z. Xu, L. Niu, Z. Xiang, B. Ramasubramanian, B. Li, and R. Poovendran, "Artprompt: Ascii Art-based Jailbreak Attacks against Aligned LLMs," *arXiv.org*, 2024.
- [49] P. Cheng, Y. Ding, T. Ju, Z. Wu, W. Du, P. Yi, Z. Zhang, and G. Liu, "Trojanrag: Retrieval-Augmented Generation Can Be Backdoor Driver in Large Language Models," *arXiv.org*, 2024.
- [50] H. Yao, J. Lou, and Z. Qin, "Poisonprompt: Backdoor Attack on Prompt-Based Large Language Models," in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 1. IEEE, apr 14 2024, pp. 7745–7749.
- [51] X. Yang, X. Wang, Q. Zhang, L. Petzold, W. Y. Wang, X. Zhao, and D. Lin, "Shadow Alignment: The Ease of Subverting Safely-Aligned Language Models," *arXiv.org*, 2023.
- [52] Y. Li, H. Guo, K. Zhou, W. X. Zhao, and J.-R. Wen, "Images are Achilles' Heel of Alignment: Exploiting Visual Vulnerabilities for Jailbreaking Multimodal Large Language Models," *arXiv.org*, 2024.
- [53] S. Erfan, D. Yue, and B. A.-G. Nael, "Jailbreak in pieces: Compositional Adversarial Attacks on Multi-Modal Language Models," *International Conference on Learning Representations*, 2023.
- [54] Z.-X. Yong, C. Menghini, and S. H. Bach, "Low-Resource Languages Jailbreak GPT-4," *arXiv.org*, 2023.
- [55] J. Li, Y. Liu, C. Liu, L. Shi, X. Ren, Y. Zheng, Y. Liu, and Y. Xue, "A Cross-Language Investigation into Jailbreak Attacks in Large Language Models," *arXiv.org*, 2024.
- [56] X. Qi, K. Huang, A. Panda, P. Henderson, M. Wang, and P. Mittal, "Visual Adversarial Examples Jailbreak Aligned Large Language Models," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 19, pp. 21 527–21 536, mar 24 2024.
- [57] N. Jain, A. Schwarzschild, Y. Wen, G. Somepalli, J. Kirchenbauer, P.-y. Chiang, M. Goldblum, A. Saha, J. Geiping, and T. Goldstein, "Baseline Defenses for Adversarial Attacks Against Aligned Language Models," *arXiv.org*, 2023.
- [58] A. Wei, N. Haghtalab, and J. Steinhardt, "Jailbroken: How Does LLM Safety Training Fail?" *Neural Information Processing Systems*, 2023.

- [59] S. Schulhoff, J. Pinto, A. Khan, L.-F. Bouchard, C. Si, S. Anati, V. Tagliabue, A. Kost, C. Carnahan, and J. Boyd-Graber, "Ignore This Title and HackAPrompt: Exposing Systemic Vulnerabilities of LLMs Through a Global Prompt Hacking Competition," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2023.
- [60] G. Deng, Y. Liu, Y. Li, K. Wang, Y. Zhang, Z. Li, H. Wang, T. Zhang, and Y. Liu, "Masterkey: Automated Jailbreaking of Large Language Model Chatbots," in *Proceedings 2024 Network and Distributed System Security Symposium*. Internet Society, 2024.
- [61] Y. Mo, Y. Wang, Z. Wei, and Y. Wang, "Studious bob fight back against jailbreaking via prompt adversarial tuning," *arXiv preprint arXiv:2402.06255*, 2024.
- [62] A. Zhou, B. Li, and H. Wang, "Robust Prompt Optimization for Defending Language Models Against Jailbreaking Attacks," *arXiv.org*, 2024.
- [63] C. Zheng, F. Yin, H. Zhou, F. Meng, J. Zhou, K.-W. Chang, M. Huang, and N. Peng, "On Prompt-Driven Safeguarding for Large Language Models," *arXiv.org*, 2024.
- [64] Y. Zong, O. Bohdal, T. Yu, Y. Yang, and T. Hospedales, "Safety Fine-Tuning at (Almost) No Cost: A Baseline for Vision Large Language Models," *arXiv.org*, 2024.
- [65] A. Hasan, I. Rugina, and A. Wang, "Pruning for Protection: Increasing Jailbreak Resistance in Aligned LLMs Without Fine-Tuning," *arXiv.org*, 2024.
- [66] A. Robey, E. Wong, H. Hassani, and G. J. Pappas, "Smoothllm: Defending Large Language Models Against Jailbreaking Attacks," *arXiv.org*, 2023.
- [67] W. Lu, Z. Zeng, J. Wang, Z. Lu, Z. Chen, H. Zhuang, and C. Chen, "Eraser: Jailbreaking Defense in Large Language Models via Unlearning Harmful Knowledge," *arXiv.org*, 2024.
- [68] B. Cao, Y. Cao, L. Lin, and J. Chen, "Defending Against Alignment-Breaking Attacks via Robustly Aligned LLM," *arXiv.org*, 2023.
- [69] Y. Zeng, Y. Wu, X. Zhang, H. Wang, and Q. Wu, "Autodefense: Multi-Agent LLM Defense against Jailbreak Attacks," *arXiv.org*, 2024.
- [70] M. Phute, A. Helbling, M. Hull, S. Peng, S. Szyller, C. Cornelius, and D. H. Chau, "Llm Self Defense: By Self Examination, LLMs Know They Are Being Tricked," *Tiny Papers @ ICLR*, 2023.
- [71] Z. Zhang, Q. Zhang, and J. Foerster, "Parden, Can You Repeat That? Defending against Jailbreaks via Repetition," *arXiv.org*, 2024.
- [72] Z. Wang, F. Yang, L. Wang, P. Zhao, H. Wang, L. Chen, Q. Lin, and K.-F. Wong, "Self-Guard: Empower the LLM to Safeguard Itself," *North American Chapter of the Association for Computational Linguistics*, 2023.
- [73] Y. Wang, Z. Shi, A. Bai, and C.-J. Hsieh, "Defending LLMs against Jailbreaking Attacks via Backtranslation," *arXiv.org*, 2024.
- [74] Z. Xu, F. Jiang, L. Niu, J. Jia, B. Y. Lin, and R. Poovendran, "Safedecoding: Defending against Jailbreak Attacks via Safety-Aware Decoding," *arXiv.org*, 2024.
- [75] C. Zheng, F. Yin, H. Zhou, F. Meng, J. Zhou, K.-W. Chang, M. Huang, and N. Peng, "On prompt-driven safeguarding for large language models," in *Forty-first International Conference on Machine Learning*, 2024.
- [76] H. Sun, Z. Zhang, J. Deng, J. Cheng, and M. Huang, "Safety Assessment of Chinese Large Language Models," *arXiv.org*, 2023.
- [77] L. Xin, Z. Yichen, G. Jindong, L. Yunshi, Y. Chao, and Q. Yu, "Mm-SafetyBench: A Benchmark for Safety Evaluation of Multimodal Large Language Models," *arXiv preprint arXiv:2311.17600*, 2023.
- [78] W. Luo, S. Ma, X. Liu, X. Guo, and C. Xiao, "Jailbreakv-28k: A Benchmark for Assessing the Robustness of MultiModal Large Language Models against Jailbreak Attacks," *arXiv.org*, 2024.
- [79] R. Hazra, S. Layek, S. Banerjee, and S. Poria, "Sowing the Wind, Reaping the Whirlwind: The Impact of Editing Language Models," *arXiv.org*, 2024.
- [80] P. Chao, E. Debenedetti, A. Robey, M. Andriushchenko, F. Croce, V. Shwag, E. Dobriban, N. Flammarion, G. J. Pappas, F. Tramer, H. Hassani, and E. Wong, "Jailbreakbench: An Open Robustness Benchmark for Jailbreaking Large Language Models," *arXiv.org*, 2024.
- [81] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz, "Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection," in *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*. ACM, nov 26 2023.
- [82] S. Zhu, R. Zhang, B. An, G. Wu, J. Barrow, Z. Wang, F. Huang, A. Nenkova, and T. Sun, "Autodan: Interpretable Gradient-Based Adversarial Attacks on Large Language Models," *arXiv.org*, 2023.
- [83] H. Qiu, S. Zhang, A. Li, H. He, and Z. Lan, "Latent jailbreak: A benchmark for evaluating text safety and output robustness of large language models," *arXiv preprint arXiv:2307.08487*, 2023.
- [84] D. Wu, S. Wang, Y. Liu, and N. Liu, "Llms Can Defend Themselves Against Jailbreaking in a Practical Manner: A Vision Paper," *arXiv.org*, 2024.