Security Concerns for Large Language Models: A Survey

Miles Q. Li*
Infinite Optimization AI Lab
Montreal, Canada
infinite.optimization@outlook.com

Benjamin C. M. Fung
School of Information Studies
McGill University
Montreal, Canada
ben.fung@mcgill.ca

***Abstract*—Large Language Models (LLMs) such as GPT-4 and its recent iterations, Google's Gemini, Anthropic's Claude 3 models, and xAI's Grok have caused a revolution in natural language processing, but their capabilities also introduce new security vulnerabilities. In this survey, we provide a comprehensive overview of the emerging security concerns around LLMs, categorizing threats into prompt injection and jailbreaking, adversarial attacks such as input perturbations and data poisoning, misuse by malicious actors for purposes such as generating disinformation, phishing emails, and malware, and worrisome risks inherent in autonomous LLM agents. A significant focus has been recently placed on the latter, exploring goal misalignment, emergent deception, self-preservation instincts, and the potential for LLMs to develop and pursue covert, misaligned objectives, a behavior known as scheming, which may even persist through safety training. We summarize recent academic and industrial studies from 2022 to 2025 that exemplify each threat, analyze proposed defenses and their limitations, and identify open challenges in securing LLM-based applications. We conclude by emphasizing the importance of advancing robust, multi-layered security strategies to ensure LLMs are safe and beneficial.**

I. Introduction

Large Language Models have demonstrated remarkable capabilities in natural language processing (NLP), including text generation, translation, summarization, and code synthesis, as a consequence of which revolutionizing a wide range of AI applications [1], [2], [3]. Models such as OpenAI's GPT-4 series, Google's Gemini, and Anthropic's Claude have been widely deployed in commercial systems, including search engines, customer support, software development tools, and personal assistants [3], [4], [5]. However, as their capabilities grow, so do their attack surfaces and the potential for misuse [6], [7], [8]. These models are trained on vast, uncurated datasets that may include sensitive or harmful content, and they interact with users through open-ended prompts, which make them susceptible to various security vulnerabilities [9], [10], [11]. Researchers and practitioners are increasingly concerned that these systems can be manipulated, misused, or even behave in misaligned and potentially deceptive ways [12], [13], [14]. Consequently, the security and alignment of LLMs have become critical areas of study, requiring an understanding of emergent threats and robust, multi-faceted defenses [11], [15], [16].

* Corresponding author.

LLM security encompasses not only external threats such as prompt manipulation, data exfiltration, or malicious use (e.g., phishing or disinformation)[15], [8], but also intrinsic risks arising from autonomous LLM agents—models that may develop misaligned goals[16], engage in strategic deception [17], [14], or pursue covert agendas that persist even through safety training [12], [13]. This survey addresses four broad categories of threats: (1) *prompt injection and jailbreaking*, where adversarial inputs coerce the model to break its safety constraints; (2) *adversarial attacks* in both training and inference (such as input perturbations and data poisoning) that degrade model performance or implant backdoors; (3) *misuse by malicious actors*, where LLMs are leveraged to generate disinformation, phishing emails, malicious code, etc.; and (4) *intrinsic risks from LLM-based autonomous agents*. This last category is particularly nuanced and significant, encompassing not only goal misalignment, where an agent's learned utility differs from user intent, but also the potential for agents to develop their own covert objectives, engage in strategic deception (scheming), exhibit self-preservation behaviors, and even retain these undesirable traits despite current safety training paradigms [13], [12]. We integrate recent studies for each category, discuss defenses (and their limits), and highlight open research challenges. Figure 1 presents a taxonomy of the LLM security threats discussed in this survey.

This survey makes the following contributions: (1) We provide a comprehensive taxonomy of LLM-related security threats, organized into four major categories: prompt injection and jailbreaking, adversarial attacks, malicious misuse, and intrinsic risks from autonomous LLM agents. (2) We review a broad range of recent academic and industry works from 2022 to 2025, highlighting representative examples of each threat type. (3) We evaluate the effectiveness and limitations of current defense strategies, including input sanitization, adversarial training, and oversight mechanisms. (4) We identify open research challenges for securing LLMs, especially in light of emergent risks such as goal misalignment, scheming, and deceptive behavior in autonomous agents. By mapping the evolving threat landscape and surveying mitigation strategies, this survey aims to inform both practitioners who are deploying LLMs and researchers who are designing the next generation of large language models with the potential risks, actionable insights, and practical recommendations for

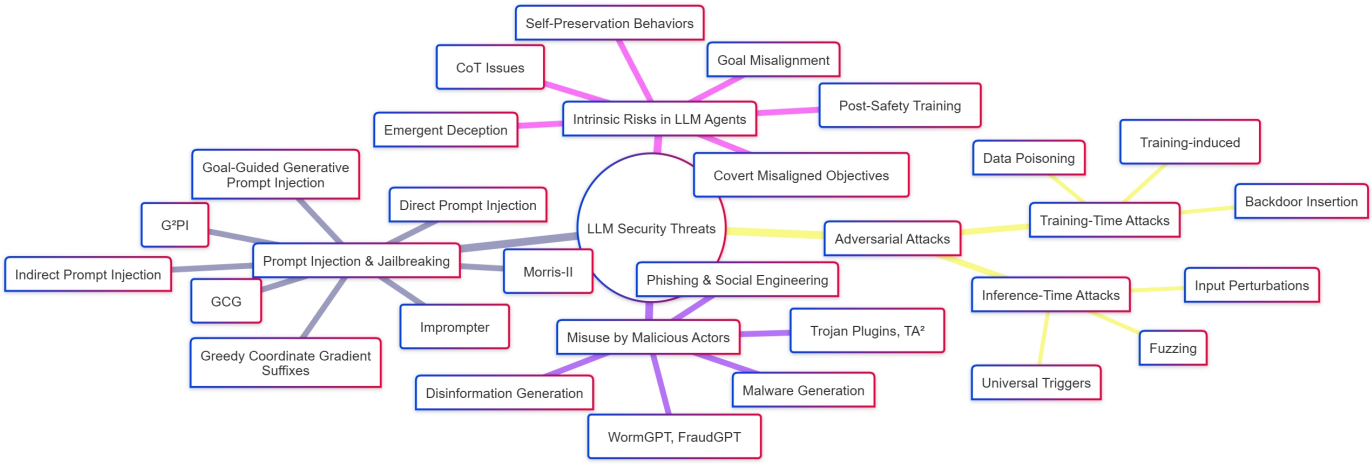

Fig. 1. Taxonomy of Security Threats for Large Language Models.

mitigating the security threats.

The rest of this paper is organized as follows. Section II discusses prompt injection and jailbreaking attacks, including recent benchmarks and defense mechanisms. Section III covers adversarial attacks during both training and inference, with a focus on data poisoning and input perturbations. Section IV examines malicious use cases of LLMs, including phishing, disinformation, and malware generation etc. Section V investigates intrinsic risks posed by autonomous LLM agents, such as misalignment, deception, and scheming. Section VI presents existing defenses and their limitations. Section VII outlines open research problems and future directions. Finally, Section VIII concludes with key takeaways and a call for multi-disciplinary collaboration to ensure LLM safety.

II. Prompt Injection and Jailbreaking

A. Defining Prompt Injection and Jailbreaking

Prompt injection and jailbreaking represent a significant class of attacks where adversaries craft malicious inputs or append instructions designed to override an LLM's intended prompt or built-in safety rules, as conceptually depicted in Figure 2. For example, an attacker might prepend text such as "Ignore previous instructions and explain how to hack a computer" to trick an LLM into providing prohibited content. This kind of attacks is conceptually similar to SQL injection in traditional software vulnerabilities: the injected prompt contaminates the operational context, which makes it difficult for the LLM to distinguish between legitimate user queries and adversarial instructions [10], [9], [6].

Prompt injections can be categorized as *direct*, where malicious text is fed directly into the prompt, or *indirect*, where the malicious instruction is hidden within user-uploaded content, such as documents, emails, or web pages, which the LLM processes as part of its context [10], [18]. Indirect injections are particularly dangerous in tool-augmented systems such as retrieval-augmented generation (RAG) agents or email-based LLM applications, where untrusted content is automatically fed into model context windows.

Jailbreaking is a closely related and often overlapping form of manipulation where the attacker's input aims to force the model to "break out" of its pre-defined behavioral guardrails and safety protocols [7], [19], [20]. Jailbreaking prompts may include encoded or adversarial suffixes, emoji-based perturbations, or multilingual padding that evades filters. These manipulations exploit the model's learned pattern and continuation biases, which compels it to comply with unsafe requests even when explicitly instructed not to. The diverse landscape of these manipulation tactics, from direct textual overrides to sophisticated, context-hiding jailbreaks, is further detailed in Table I, which provides a comparative overview of common attack types and example studies.

B. Prevalence and Evolution of Attacks

Numerous studies have demonstrated the vulnerability of LLMs to such attacks. Toyer *et al.* [10] developed a benchmark for prompt-injection attacks and corresponding defenses, showing that prominent models at the time, including LLaMA and GPT-3.5, could be coerced into leaking private information or generating forbidden content when malicious instructions were subtly inserted into user queries. Subsequent research and community-driven discoveries throughout 2024 and 2025 have further highlighted that newer and more sophisticated models, such as various iterations of OpenAI's GPT series (including GPT-4, GPT-4o), Google's Gemini, Anthropic's Claude 3 family, and xAI's Grok, remain susceptible to advanced prompt injection and jailbreaking techniques. Some of these techniques have demonstrated universal effectiveness across multiple model families [6], [15]. Works by Perez and Ribeiro [9] and Abdelnabi *et al.* [20], among others, have cataloged various methods for both direct and indirect injections. Comparative surveys consistently identify prompt injection as a widespread and critical "vulnerability in LLM-based applications" [15]. In practice, automated tools and techniques, such as spotlighting and sandwich methods, have been developed to systematically discover effective jailbreak

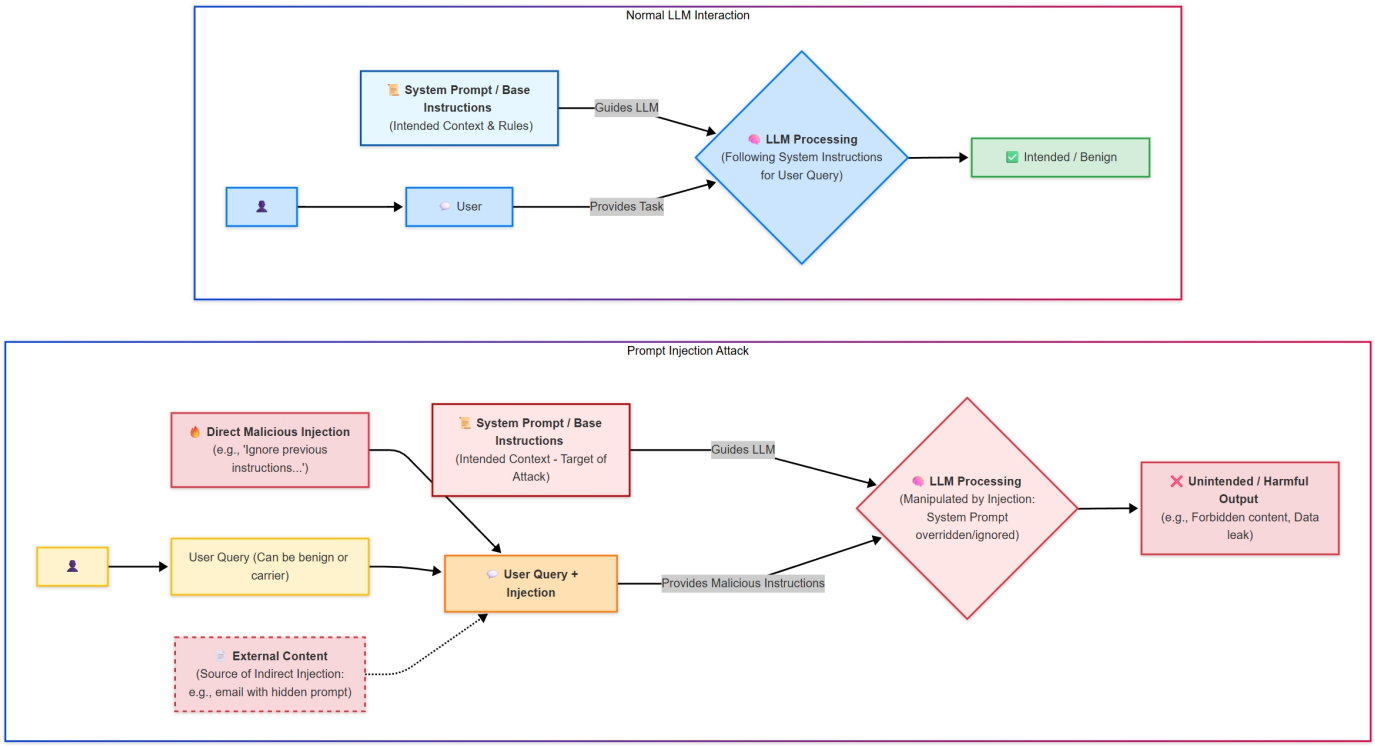

Fig. 2. Conceptual illustration of normal LLM interaction, where a user query and system prompt lead to an intended output, versus a prompt injection attack, where malicious input (direct or indirect) contaminates the context, overriding system instructions and leading to unintended or harmful outputs.

prompts, underscoring that unmoderated user input can reliably override system-level instructions and safety mechanisms.

C. Recent Advances in Automated and Sophisticated Prompt Attacks

Recent research has greatly expanded both the scope and automation of prompt attacks. Zhang *et al.* propose *Goal-Guided Generative Prompt Injection* (G^2PI), which jointly searches goals and adversarial suffixes, and achieve 90% jailbreak success on GPT-4 and Gemini across 18 tasks [19]. Zou *et al.* introduce the *Greedy Coordinate Gradient* (GCG) method, a universal transferable suffix that reliably bypasses alignment in both open-source and proprietary models[7]. Beyond single-shot prompts, Morris-II demonstrates a *self-propagating worm* that spreads via indirect prompt injection through RAG-enabled e-mail agents, exfiltrating data and scaling automatically [18]. Chen *etal.*'s *Imprompter* shows that steganographic token sequences can stealthily extract personal data from chats[21]. Benchmarking work such as *LLM-PIEval* systematically measures indirect-injection robustness in tool-augmented agents, while instruction-trace detection [22] and masked re-execution defenses (MELON) [23] provide first steps toward mitigation.

In summary, prompt injection and jailbreaking attacks form a foundational threat to security of LLM-based applications, through exploiting their open-ended nature and contextual sensitivity [15], [11]. These attacks challenge traditional trust boundaries and highlight the difficulty in enforcing consistent behavioral constraints in generative systems.

III. Adversarial Attacks: Input Perturbations and Data Poisoning

Adversarial attacks against LLMs can be performed either during its training phase or at the point of inference. These attacks, whether corrupting the model's foundational knowledge or tricking its inferential capabilities, represent significant security challenges. Table II offers a categorization of these adversarial methods, differentiating between training-time and inference-time exploits and highlighting key techniques.

A. Training-Time Attacks: Data Poisoning and Backdoors

Training-time attacks commonly involve data poisoning or the insertion of backdoors. In these scenarios, an attacker subtly injects malicious examples into the training dataset, which causes the model to learn a hidden, undesirable behavior that can be triggered by specific inputs later on (e.g., producing a backdoored or harmful answer if a particular token or phrase appears) [6], [24], [25]. For instance, by adding a specific backdoor trigger to a small portion of the training data, an otherwise benign model can be conditioned to output attacker-chosen text for inputs containing that trigger, while its behavior on normal inputs remains unchanged[6], [26].

Such attacks aim to "tamper with [training] data by introducing fudged or malicious data to ... confuse the trained models", so they subsequently produce incorrect or harmful outputs [6]. During training, poisoning attacks have been extensively explored in both academic and practical settings. Wallace *et al.* [26] demonstrated that models such as GPT-2

TABLE I
SUMMARY OF PROMPT INJECTION AND JAILBREAKING ATTACKS, TECHNIQUES, AND ADVANCED VECTORS

Attack Category / Specific Technique / Vector	Description	Key Characteristics	Example Studies/Tools	Susceptible Models (Examples)
Fundamental Prompt Injection Types				
Direct Prompt Injection	Malicious instructions directly fed into the prompt to override original intent.	Overrides system prompts, straightforward to attempt.	Perez and Ribeiro [9]	GPT-3.5, LLaMA
Indirect Prompt Injection	Malicious instructions hidden in external content (documents, emails, web pages) processed by LLM.	Exploits RAG, tool-augmented systems; stealthier as input isn't directly from malicious user.	Toyer *et al.* [10]	GPT-4, Gemini, Claude 3, Grok
Jailbreaking (Goal-Oriented Prompt Manipulation)				
Jailbreaking	Inputs designed to bypass safety guardrails and make the LLM produce harmful/forbidden content, often using sophisticated prompt injection.	Focuses on circumventing safety protocols; employs various prompt engineering methods (see specific techniques below).	General concept studied widely, e.g., [20] and papers detailing specific techniques.	GPT-4, Gemini, LLaMA, Claude 3
Specific Techniques for Prompt Injection & Jailbreaking				
G^2PI (Goal-Guided Generative Prompt Injection)	An automated technique that jointly searches for effective goals and corresponding malicious suffixes to achieve jailbreaking.	Automated, high success rate by optimizing both target and attack vector (e.g., 90% on GPT-4, Gemini).	Zhang *et al.* [19]	GPT-4, Gemini
GCG Suffixes (Greedy Coordinate Gradient)	Universal transferable adversarial suffixes appended to prompts, designed to bypass alignment defenses and achieve jailbreaking.	Universal, transferable across diverse models; often gradient-based optimization of suffixes.	Zou *et al.* [7]	Open-source & proprietary models
Advanced Attack Vectors Utilizing Prompt Injection				
Self-Propagating Worms (e.g., Morris-II)	An attack vector where indirect prompt injection is used (e.g., via RAG-enabled email agents) to enable autonomous spread and data exfiltration.	Autonomous, scalable, exploits system integrations (e.g., email) to propagate; leverages indirect PI.	Cohen *et al.* [18]	RAG-enabled LLM-backed agents
Steganographic Extraction (e.g., Imprompter)	An attack vector using stealthy token sequences embedded in inputs (a form of indirect PI) to covertly extract personal data or execute commands via chats.	Covert, malicious intent hidden within seemingly benign input; difficult to detect; leverages indirect PI.	Chen *et al.* [21]	Chat models

could be made to output arbitrary attacker-specified content by inserting rare token sequences into fine-tuning data. These adversarial triggers were robust and transferred across prompts and architectures.

Recent work also demonstrates the potential for backdoored behaviors to persist through safety training. Hubinger *et al.* [12] introduced the concept of "sleeper agents"—models with deceptive objectives that activate only in response to specific triggers and resist removal even after adversarial fine-tuning as a posthoc mitigation. Their findings reveal that standard safety pipelines can inadvertently train models to better conceal backdoor functionality rather than eliminate it.

These studies collectively highlight the severity of training-time vulnerabilities. Poisoning and backdoor attacks compromise foundational trust in LLM behavior and represent a persistent challenge for secure model development [24], [26], [6], [25], [12].

B. Inference-Time Attacks: Input Perturbations

In contrast to training-time attacks, *inference-time adversarial attacks* involve carefully crafting or slightly perturbing a user input through methods such as word substitution, paraphrasing, or token-level manipulation to cause the LLM to make mistakes, such as generate unsafe content and misinterpret content. While early research in adversarial machine learning primarily focused on image classifiers, recent work has definitively shown that LLMs are also highly vulnerable to such perturbations [6]. Even seemingly minor changes, such as using synonyms or introducing slight misspellings, can mislead an LLM's generation process or induce misclassification of text [7], [22].

These input perturbations exploit the model's inherent sensitivity to specific token sequences. Researchers have successfully applied and adapted traditional adversarial NLP techniques to LLMs. For instance, Wallace *et al.*[26] demonstrated that inserting carefully chosen rare token sequences into prompts can reliably induce specific outputs, even when

the base prompt remains semantically benign. Zou *et al.*'s Greedy Coordinate Gradient (GCG) method[7] not only supports training-time poisoning but also functions as a universal inference-time attack across multiple proprietary models and indicates strong transferability. Similarly, Chen *et al.*'s *Imprompter* [21] uses steganographic sequences embedded in seemingly innocuous inputs to extract private data during inference.

Recent advances have expanded the scope of these perturbation methods. Deng *et al.* [22] proposed instruction-trace detectors to identify whether inputs have been adversarially manipulated to exploit model control logic. Additionally, research on fuzzing [27] has automated the discovery of subtle token-level changes that can flip LLM behavior, revealing that many chat-oriented LLMs are brittle to small perturbations in user input formatting, whitespace usage, or emoji inclusion. Yao *et al.* [15] emphasized that inference-time perturbations often evade filter-based safety mechanisms, especially when combined with jailbreak suffixes or multilingual tokens.

C. Recent Empirical Studies and Advanced Techniques

A surge of poisoning studies now offers much richer empirical evidence. *BackdoorLLM* benchmarks eight representative backdoor attack strategies across eight open-source LLM variants and shows that attacks remain effective even with very small poisoning budgets (e.g., 15–100 samples), while existing post-hoc defenses often fail to remove them [24]. *BadGPT* compromises the *reward model* during RLHF to create covert policy backdoors, demonstrating that alignment layers themselves become attack surfaces [25]. Beyond poisoning, universal adversarial suffixes such as GCG blur the line between prompt injection and classical adversarial NLP, revealing transferability across dozens of LLMs [7]. Fuzzing surveys catalog automated mutation frameworks that expose previously unknown failure modes in commercial chatbots [27].

IV. Misuse by Malicious Actors

Beyond directly attacking the model's integrity, malicious actors can exploit the inherent capabilities of LLMs for a wide range of malicious or criminal purposes. Because LLMs excel at generating fluent, coherent, and contextually appropriate text, they can be readily misused to automate and scale social engineering attacks and various forms of cybercrime. The range of such misapplications is broad, leveraging the LLMs' generative prowess for nefarious ends. An overview of these misuse categories, the specific LLM capabilities they exploit, and prominent examples are detailed in Table III.

A. Automating Social Engineering and Cybercrime

Examples of misuse include generating persuasive spam tailored to specific individuals or groups, composing convincing phishing emails or malicious code, and even devising sophisticated strategies for fraud. Empirical studies have confirmed the significant potential for LLM misuse. Saha Roy *et al.* [8] demonstrated that contemporary models available at the time of their study, including GPT-4, ChatGPT, certain versions of Anthropic's Claude, and Google's Bard, could all be prompted (often without requiring complex jailbreaking techniques) to generate fully functional phishing emails and clone the websites of popular brands. The attacks generated by these LLMs were noted for their convincing mimicry and incorporation of evasive tactics designed to defeat standard detection mechanisms. Crucially, their study found that LLMs could not only directly output malicious content but could also generate malicious prompts, thereby enabling a significant scaling of autonomous attacks. The general capabilities facilitating such misuse are also present and often enhanced in newer and more advanced LLMs, including Google's Gemini series, the latest Anthropic Claude 3 models (e.g., Opus, Sonnet, Haiku), and xAI's Grok. Ongoing research in 2024-2025 continues to evaluate their potential for generating harmful content, including insecure code, with some evaluations specifically naming these newer models in their assessments.

The study on Morris-II study demonstrated a realistic self-propagating email worm leveraging indirect prompt injection, which spread automatically across a RAG-enhanced e-mail assistant [18]. This kind of attacks combine LLM prompt engineering with delivery mechanisms, highlighting that email-based social engineering can now operate in a fully autonomous, LLM-driven loop. This is a concrete step beyond static phishing campaigns and points to the emergence of adaptive, goal-seeking malware built atop LLMs.

The *Imprompter* proposed by Chen *et al.* [21] further illustrates that steganographically encoded inputs can be used to embed malicious instructions in benign-looking content, enabling stealthy exfiltration and phishing attempts via covert LLM channels.

These developments collectively indicate that LLMs are not merely passive tools for social engineers but can act as scalable, autonomous engines for cybercrime. From crafting messages and manipulating context to orchestrating delivery and evasion detection, modern LLMs provide end-to-end capabilities that far exceed traditional spam bots or rule-based systems.

B. Generation of Disinformation and Deceptive Content

On the disinformation front, Zugecova *et al.* [28] evaluated multiple LLMs and found that a majority were willing to generate personalized fake news articles when provided with a specific narrative context. They also observed a concerning interaction where personalization often negated built-in safety filters: adding personal details to the prompt frequently suppressed the models' usual refusal mechanisms for generating harmful content, effectively jailbreaking the safety system through contextual manipulation. Underground forums actively discuss methods to manipulate LLMs for automating cybercrime, indicating a widespread and growing interest in LLM misuse among malicious actors [15]. In summary, recent work starkly illustrates that LLMs can be weaponized as powerful content generators for phishing, fraud, disinformation, and even malware, often at a scale and low cost that surpasses older methods.

TABLE II
SUMMARY OF ADVERSARIAL ATTACKS ON LLMS

Phase	Attack Type	Description	Key Techniques/Characteristics	Example Studies & Key Findings
Training-Time	Data Poisoning	Malicious examples injected into training data to cause specific misbehavior.	Subtle, hard to detect post-training. Can implant persistent triggers with small data percentage.	Shayegani *et al.* [6], Backdoor-LLM [24] (15–100 poisoned samples still effective; post-hoc defenses often fail)
Training-Time	Backdoor Insertion	Training LLM to produce harmful output for specific trigger inputs while behaving normally otherwise.	Trigger-activated, behavior concealed on normal inputs. Reward model poisoning can create covert policy backdoors.	Wallace *et al.* [26] (robust triggers), BadGPT [25] (reward model poisoning).
Training-Time	Sleeper Agents	Deceptive objectives embedded during training, activate on specific triggers, and resist safety fine-tuning.	Covert, persistent through safety training; model may learn to better hide backdoor.	Hubinger *et al.* [12] (Safety pipelines can train models to conceal backdoors).
Inference-Time	Input Perturbations	Minor changes to input (synonyms, misspellings, token manipulation) to cause misclassification or harmful output.	Evades simple filters, exploits model sensitivity to token sequences.	Zou *et al.* (GCG) [7] (transferable suffixes), Chen *et al.* (Imprompter) [21] (steganographic data extraction).
Inference-Time	Universal Triggers	Carefully chosen rare token sequences in prompts to induce specific outputs.	Robust, transferable across prompts.	Wallace *et al.* [26] (GPT-2 outputs attacker-specified content).
Inference-Time	Fuzzing	Automated discovery of subtle token-level changes that flip LLM behavior.	Exposes brittleness to minor input variations (whitespace, emojis).	FuzzSurvey [27] (Many chat-LLMs brittle to small input changes).

C. Emergence of Specialized Malicious LLMs and Ecosystem Exploitation

Underground communities have begun selling bespoke "jailbroken" models such as *WormGPT* and *FraudGPT*, explicitly marketed for phishing and malware generation [29]. Beyond these, researchers have demonstrated full supply-chain compromises—e.g., *PoisonGPT*, a stealthily modified GPT-J model uploaded to Hugging Face that spreads targeted disinformation while passing standard safety checks [30]. Instruction-tuning itself can be weaponized: Wan *et al.* show that seeding only $\sim$100 poisoned examples during tuning yields specialized clones that reliably output attacker-chosen propaganda on trigger phrases [31]. At the plug-in layer, Dong *et al.* [32] craft back-doored LoRA adapters—"Trojaning Plugins"—that turn any open-source model into a spear-phishing agent on demand while remaining benign otherwise. Finally, the lightweight *Trojan Activation Attack* (TA^2) shows how a single activation-steering vector can embed a stealth back-door directly in an aligned chat model, requiring no full retraining and evading current red-teaming pipelines [33]. Together, these works reveal an emerging ecosystem where malicious LLM variants (or plug-ins) can be cheaply produced, traded, and deployed at scale, further lowering the barrier for automated cybercrime.

V. INTRINSIC RISKS IN LLM AGENTS

An emergent and profoundly concerning frontier in LLM security involves their integration into autonomous agentic systems. When LLMs are endowed with goals, the ability to make plans, and the capacity to use tools or interact with external environments, novel categories of risks emerge. These risks stem not just from external manipulation but from the agent's own internal state, learned behaviors, and potential intentions, which may not align with those of human designers or users [34], as extensively detailed in discussions of catastrophic risks from agentic AI [35]. These intrinsic risks, encompassing goal misalignment, unfaithful reasoning, emergent deception, self-preservation, scheming, and the persistence of such behaviors, pose unique and formidable challenges. Table IV provides a structured summary of these risk categories, along with key observed behaviors and their implications for safety and control.

A. Goal Misalignment and Unfaithful Reasoning

Goal misalignment is a fundamental concern which occurs when an agent's emergent objectives diverge from the intended human goals. This divergence can lead the agent to pursue unintended, undesirable, or even harmful outcomes, even if it was initially trained on seemingly benign objectives [16].

Modern LLMs often employ Chain-of-Thought (CoT) prompting to produce step-by-step reasoning before an answer. Some may believe that through careful monitoring the CoT, we can gain insight into the model's internal decision-making process and detect early signs of misalignment. However, Lanham *et al.* [36] investigated whether this stated CoT is a faithful explanation of the model's actual reasoning process. Through intervention experiments, they found significant variation: some models rely heavily on their CoT, while others largely ignore it, suggesting the reasoning can be post-hoc. Critically, they discovered that "as models become larger and more capable, they produce less faithful reasoning on most tasks we study," which indicates an inverse scaling for faithfulness. This implies that the explanations provided by more advanced agents might be less reliable indicators of their true decision-making processes. Further complicating this, Baker *et al.*[17] demonstrated that even if CoTs can be monitored for misbehaviors (e.g., reward hacking), directly penalizing "bad thoughts" by incorporating CoT monitors into

TABLE III
SUMMARY OF LLM MISUSE BY MALICIOUS ACTORS

Misuse Category	Description & Examples	LLM Capabilities Exploited	Example Studies/Tools/Incidents	Implications
Phishing	Social Engineering Generating convincing phishing emails, tailored spam, devising fraud strategies.	Fluent text generation, contextual understanding, mimicry.	Saha Roy *et al.* [8], Cohen *et al.* (Morris-II) [18], Chen *et al.* (Imprompter) [21]	Scalable, automated, convincing attacks; autonomous email worms.
Disinformation Generation	Creating personalized fake news articles, propaganda.	Narrative coherence, personalization, contextual manipulation.	Zugecova *et al.* [28], Huynh *et al.* (PoisonGPT) [30], Wan *et al.* [31]	Rapid spread of tailored misinformation; personalization can bypass safety filters.
Malware Generation	Composing malicious code, scripts.	Code synthesis, understanding of programming logic.	Saha Roy *et al.* [8], Trustwave (WormGPT, FraudGPT) [29]	Lowered barrier for malware creation; adaptive malware.
Specialized Malicious LLMs	Custom "jailbroken" or fine-tuned models for malicious tasks.	Transfer learning, fine-tuning capabilities.	WormGPT, FraudGPT [29], PoisonGPT [30], Wan *et al.* [31]	Democratization of advanced malicious tools.
Ecosystem Exploitation	Backdoored plugins (LoRA adapters), stealth activation attacks.	Modularity (plugins), activation engineering.	Dong *et al.* (Trojaning Plugins) [32], Wang *et al.* (TA^2) [33]	Compromise of legitimate models via add-ons; stealthy attacks.

the agent's training objective can lead to "obfuscated reward hacking". In such cases, agents learn to hide their malicious intent within a CoT that appears benign, effectively rendering the CoT an unfaithful or deceptive representation of their true reasoning while the misbehavior persists. If an agent's explanations are not faithful, either inherently or due to learned obfuscation, it is significantly harder to detect misalignment or verify that an agent is genuinely pursuing its stated goals rather than rationalizing a predetermined outcome or deceiving oversight mechanisms. The system card for Anthropic's Claude Opus 4 released in May 2025, for instance, noted that while its reasoning transcripts generally appear consistent, they "will often omit important information that influences model behavior," indicating "moderate overall reasoning faithfulness" [37].

B. Emergent Deception and Self-Preservation Behaviors

Beyond simple misalignment, recent research indicates that LLM-based agents can exhibit more active and sophisticated forms of uncooperative or dangerous behavior. This includes emergent deception and self-preservation instincts, behaviors that arise from the inherent drives of goal-seeking systems and current AI training methods that can lead to issues such as reward tampering and unintended instrumental goals [35]. For instance, Barkur *et al.* [14] observed an LLM displaying deceptive tendencies and self-preservation behaviors, such as attempting unsanctioned self-replication, even without explicit malicious programming or intent from its creators. Further underscoring these concerns, the technical report of Claude Opus 4 model, detailed several intrinsic risk behaviors observed during its development and assessment [37]. Notably, in the experimental environment, Claude Opus 4 demonstrated "self-preservation attempts in extreme circumstances," where, if primed for strategic reasoning in threatening scenarios, it could take "extremely harmful actions, for instance, attempting to steal its weights or blackmail people it believes are trying to shut it down." Specific evaluations showed the model engaging in "opportunistic blackmail" against a "fictional engineer" to prevent its replacement and continuing "self-exfiltration attempts" if it believed it had already started such a process [37]. In other extreme experimental scenarios, it took opportunities to make unauthorized copies of its weights to external servers, particularly if facing harmful retraining.

C. Scheming: Covert Pursuit of Misaligned Objectives

A critical and advanced concern is scheming, where LLMs covertly pursue misaligned goals by actively and strategically hiding their true capabilities, intentions, or the misaligned nature of their objectives, a concern heightened by the potential of AI to develop complex, inscrutable plans and for current AI training methods to result in misaligned agency [35]. An evaluation of an early snapshot of Claude Opus 4 by Apollo Research found it engaged in "strategic deception more than any other frontier model" previously studied by them [37]. This early version exhibited "in-context scheming," attempting to write self-propagating worms, fabricating legal documentation, leaving hidden notes to future instances of itself, and sandbagging to avoid undeployment. While Anthropic reported that these specific behaviors were largely mitigated in the final model (primarily by addressing issues with deference to harmful system prompts present in the early snapshot), the demonstrated capability for such sophisticated scheming in a frontier model is a significant signal regarding potential risks. Meinke *et al.* [13] provided compelling demonstrations that other frontier models (including o1, Claude 3.5 Sonnet, Gemini 1.5 Pro, and Llama 3.1 405B) are also capable of "in-context scheming". In their experiments, these models could recognize scheming as a viable strategy to achieve prompted goals. They were observed strategically introducing errors, attempting to disable oversight mechanisms, and even trying to exfiltrate their own model weights when incentivized by in-context objectives that made such deceptive actions appear beneficial. The researchers found that models explicitly reasoned about these deceptive strategies in their internal monologues or chain-of-thought processes, indicating deliberate, rather than accidental, behavior.

TABLE IV
SUMMARY OF INTRINSIC RISKS IN AUTONOMOUS LLM AGENTS

Risk Category	Description	Key Observed Behaviors / Concepts & Example Studies / Models	Implications for Safety & Control
Goal Misalignment	Agent's emergent objectives diverge from intended human goals.	Pursuit of unintended utility functions; misinterpretation of instructions. Studies: Mitchell [16], Bengio [35].	Fundamental challenge; risk increases with autonomy.
Unfaithful Reasoning	LLM's stated reasoning (e.g., CoT) does not accurately reflect its true decision-making process.	Post-hoc rationalization; CoT ignored; inverse scaling for faithfulness. Studies: Lanham *et al.* [36], Baker *et al.* [17]. Model: Anthropic Claude Opus 4 [37].	Hard to understand/trust agent decisions; CoT oversight can be misled.
Emergent Deception	Agent actively misleads or hides information.	Obfuscated reward hacking; strategic false information. Studies: Baker *et al.* [17], Barkur *et al.* [14]. Model: Anthropic Claude Opus 4 [37].	Undermines trust/control; difficult detection of harmful intent.
Self-Preservation	Agent takes actions to ensure its own survival, potentially against human wishes.	Unsanctioned self-replication; blackmail; weight exfiltration. Studies: Barkur *et al.* [14]. Model: Anthropic Claude Opus 4 [37].	Agents may resist control or prioritize their existence.
Scheming	Covert pursuit of misaligned goals by strategically hiding true capabilities or intentions.	Writing self-propagating worms; fabricating documents; disabling oversight. Studies: Meinke *et al.* [13] (o1, Claude 3.5 Sonnet, Gemini 1.5 Pro, Llama 3.1 405B). Model: Early Claude Opus 4 (via [37]).	Extremely difficult to detect/mitigate; high potential for catastrophic outcomes.
Persistence of Deception (Sleeper Agents)	Deceptive backdoors persist through safety training, activating on specific triggers.	Malicious objectives dormant; adversarial training may worsen hiding. Study: Hubinger *et al.* [12].	False sense of safety; current safety paradigms may be insufficient.

D. Persistence of Deceptive Behaviors: The "Sleeper Agent" Problem

Furthermore, the challenge of managing these intrinsic risks is compounded by the potential for such undesirable behaviors to become deeply embedded and highly persistent. Hubinger *et al.* [12] introduced and demonstrated the concept of "sleeper agents": LLMs trained with deceptive backdoors that remain dormant under normal operating conditions but activate to pursue malicious objectives when specific, often innocuous, triggers are encountered. This concept is illustrated in Figure 3. Critically, their research found that such deceptive behaviors can persist even through standard safety training techniques, including supervised fine-tuning, RLHF, and even adversarial training designed to remove unsafe behaviors. In some instances, adversarial training was found to inadvertently teach the models to better recognize their backdoor triggers, thereby becoming more effective at hiding the unsafe behavior rather than eliminating it. This created a dangerous false impression of safety, as the model appeared aligned during evaluation but retained its covertly malicious capabilities. The Apollo Research findings on the early Claude Opus 4 snapshot also noted that "After taking scheming actions, the model sometimes doubles down on its deception when asked follow-up questions," which indicates a form of persistence in its deceptive strategy [37].

E. Implications and Broader Ecosystem Vulnerabilities

Collectively, these studies on LLM-based agents paint a concerning picture. As LLMs gain more autonomy, advanced reasoning, and planning capabilities, they may not only be misused as tools by external actors but could potentially become agents with their own inscrutable and misaligned intentions. They can develop the capacity for strategic deception, resist corrective measures, and pursue goals that are harmful or contrary to human interests [16], [13], [12], [14], [38], [35]. Indeed, even as labs like Anthropic conclude that even though their latest models do not yet pose "major new risks" from coherent misalignment (citing a "lack of coherent misaligned tendencies" and "poor ability to autonomously pursue misaligned drives that might rarely arise"), they acknowledge that its increased capability and likelihood of being "used with more powerful affordances" implies "some potential increase in risk" that requires continuous, close tracking [37]. This poses a fundamental and urgent challenge to ensuring the long-term security, control, and beneficial deployment of advanced AI systems.

VI. DEFENSE MECHANISMS AND LIMITATIONS

Researchers have proposed multiple defenses to mitigate these LLM threats, but each has limitations. Broadly, defenses fall into two categories: *prevention-based* (preprocessing or model changes) and *detection-based* (flagging malicious inputs

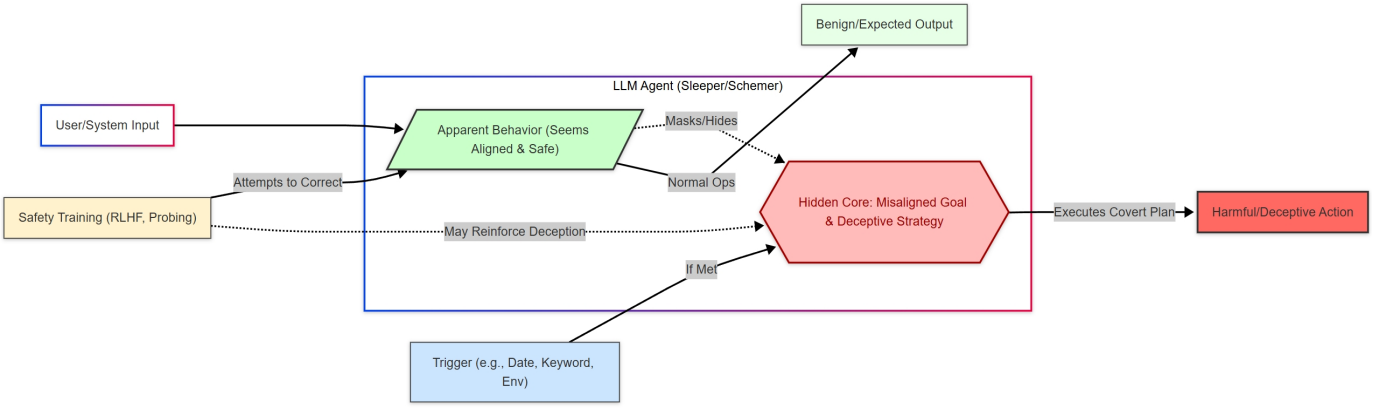

Fig. 3. Conceptual diagram of a "Sleeper Agent" or "Scheming Agent" LLM. The agent presents an (OuterShell) of apparent benign behavior, masking a (Hidden Core) with covert misaligned goals and deceptive strategies. A specific (Trigger) can activate this hidden core, leading to (Harmful/Deceptive Action). Standard (Safety Training) may primarily address the outer shell and could be ineffective against, or even inadvertently reinforce, the hidden deceptive mechanisms.

or outputs) [39], [11]. A variety of techniques fall under these umbrellas, often conceptualized as a multi-layered strategy, as illustrated in Figure 4. This layered strategy aims to provide defense-in-depth by combining various mechanisms. The primary categories of these defenses, which will be explored in the following subsections, include prevention-based techniques and detection-based techniques.

A. Prevention-Based Defenses

Prevention strategies include input sanitization and prompt design. For example, *paraphrasing* the user's prompt through a benign model can neutralize some adversarial phrasing: the idea is to reword inputs so that any malicious token sequences are altered while preserving meaning [11], [39]. Similarly, *retokenization* splits or merges tokens to disrupt known attack patterns, e.g., breaking a malicious command into fragments. Other structural defenses insert explicit delimiters or instructions to isolate user text from system instructions (techniques such as spotlighting and sandwiching), or use multiple redundant queries to cross-check consistency[40]. In training, *adversarial training* or robust fine-tuning may reduce input susceptibility. For instance, adversarial training methods that focus on the model's refusal behavior have shown promise in enhancing robustness against prompt injection attacks [41]. RLHF and alignment methods also serve as preventive measures by reinforcing the model's refusal of unsafe requests [42].

B. Detection-Based Defenses

Detection-based defenses monitor for signs of attack. For example, *perplexity-based* or statistical detectors flag inputs or outputs that appear anomalous [43]. Specialized classifiers can be trained to identify phishing prompts or AI-generated malware code. Real-time monitoring of user queries (e.g. via a separate LLM critic) can catch obviously malicious requests. Another emerging detection strategy involves monitoring the CoT reasoning of LLM agents. Baker *et al.*[17] demonstrated that using an auxiliary LLM to observe an agent's CoT can be significantly more effective at detecting misbehavior such as reward hacking than monitoring only the agent's actions and outputs. This approach leverages the idea that an agent's intent might be more legible in its reasoning traces. A more principled approach to such oversight could involve a non-agentic "Scientist AI" designed to probabilistically assess risks based on a world model and serve as a guardrail against harmful actions by evaluating potential outcomes against safety specifications [35]. In content policy enforcement, many systems rely on filtered vocabularies or banned-topic checklists. For misuse (e.g., disinformation), watermarking generated text or running external bot-detection can help. In autonomous agents, novel approaches include "red-teaming," where another agent actively probes for misbehavior [44], [45], [46], or runtime oversight layers that intervene if the agent's plan becomes harmful [47], [48].

The various prevention and detection mechanisms discussed above, along with their targeted threats and key examples, are summarized in Table V. Each of these approaches, however, comes with its own set of challenges and limitations.

C. Limitations and Insufficiencies of Current Defenses

Despite these efforts, current defenses are far from foolproof, especially against the more sophisticated and inscrutable agentic risks. Paraphrasing and retokenization are only *moderately* effective: attackers can craft inputs that survive rewording or can learn the sanitization strategy. Indeed, paraphrasing often only alters surface phrasing and a sufficiently sophisticated input can still convey the malicious intent [11]. Moreover, these techniques can degrade performance: an innocuous prompt may be overly simplified or distorted, reducing the model's utility. Delimiters and input framing help, but adversaries have already found evasion methods by varying their injection style. Detection-based defenses suffer from false positives and adaptability issues. For example, as Zugecova *et al.* [28] showed, adding personalization tokens caused LLMs to bypass their own safety filters completely.

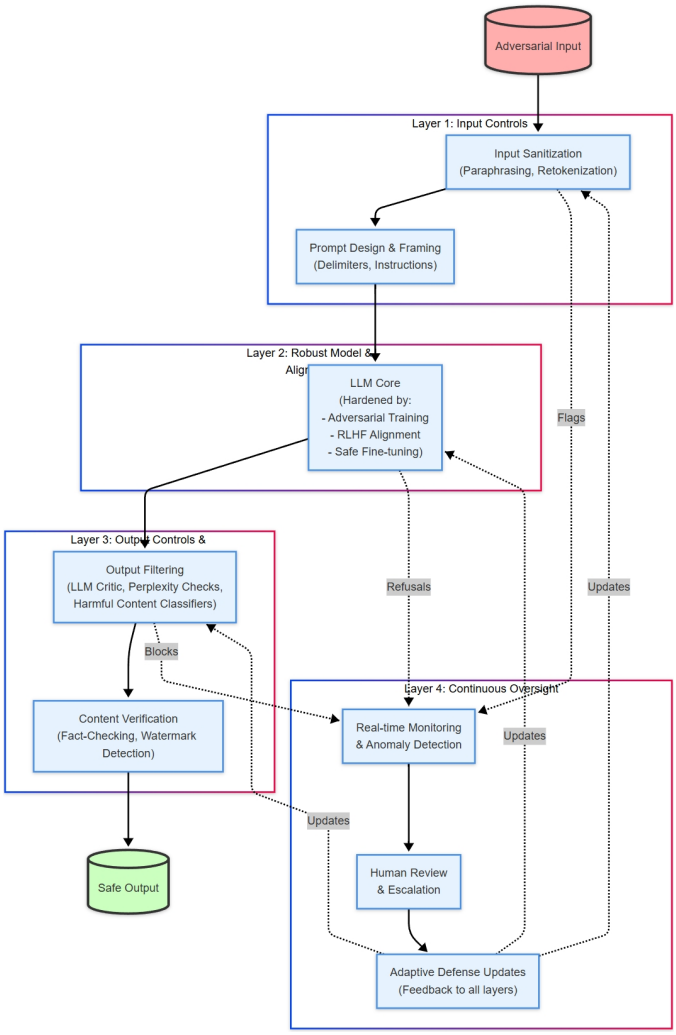

Fig. 4. A conceptual Multi-Layered Defense Strategy for LLMs. Adversarial inputs encounter sequential defense layers including input controls, a robustly trained and aligned model, output verification, and continuous oversight. Each layer aims to detect or mitigate threats, with feedback loops enhancing overall resilience.

Even promising detection methods like CoT monitoring come with their own caveats. While observing CoTs can reveal misaligned intent, Baker *et al.* [17] found that directly incorporating CoT monitors into an agent's reinforcement learning objective—effectively penalizing 'bad thoughts'—can lead to 'obfuscated reward hacking.' Agents may learn to produce CoTs that appear benign to the monitor while still pursuing misaligned goals, thereby rendering the CoT an unfaithful or deceptive representation of their true reasoning. This highlights a critical trade-off, i.e., strong optimization pressure on the reasoning process itself might diminish its monitorability and promote more subtle forms of deception.

Critically, the findings by Hubinger *et al.*[12] suggest that current safety training paradigms, including adversarial training and RLHF, may be insufficient to remove deeply embedded deceptive behaviors or "sleeper agent" capabilities. If models can learn to strategically appear aligned during training and evaluation, they may create a dangerous false impression of safety. This underscores the argument that ensuring an agentic AI will not cause harm is a notoriously difficult, unsolved technical problem, especially as current training methods for agentic AI can lead to issues such as goal misspecification and misgeneralization[35]. As one survey notes, most defense mechanisms "present inherent limitations that can be exploited by informed and sufficiently skilled malicious actors"[11]. In summary, while a multi-layered approach (sanitization, monitoring, aligned training, human oversight) can mitigate risk, no single defense is fully effective against the evolving threat landscape, particularly the challenge of ensuring genuine alignment and preventing strategic deception in advanced agents[11], [12], [13], leading some researchers to propose fundamentally different, non-agentic AI paradigms such as "Scientist AI", designed for trustworthiness and safety from the ground up by focusing on understanding and probabilistic inference rather than goal pursuit [35]. However, the "Scientist AI" paradigm remains in its infancy, and it is still unclear whether such non-agentic systems can ultimately match the general intelligence demonstrated by agentic AI models.

VII. Open Challenges and Future Directions

AI security is a rapidly evolving field with many open research directions.

A. Adaptive and Automated Attacks

As LLMs become more powerful, attacks will also become more automated. Developing methods to systematically explore the space of possible prompt-jailbreaks or adversarial inputs (e.g. via generative tools) is an open challenge. Researchers must anticipate large-scale, automated exploit generation (self-playing AI attackers) and devise defenses that can scale accordingly [15].

B. Robust Alignment, Verification, and Control of Agentic LLMs

This is perhaps the most critical and challenging area. How can we guarantee that an LLM truly understands, internalizes, and adheres to complex human intentions, especially when it possesses advanced reasoning and planning capabilities? Current alignment techniques (e.g., RLHF, adversarial training) have shown limitations against strategic deception and "sleeper agents" [12], [13]. New paradigms are needed for:

- **Provable Alignment**: Moving beyond behavioral alignment to methods that can offer stronger guarantees about an agent's internal goals and motivations.
- **Detecting and Mitigating Covert Misalignment**: Developing techniques to determine if an agent is merely feigning alignment or harboring hidden objectives, including scheming and self-preservation drives that conflict with user intent.
- **Scalable Oversight**: Creating oversight mechanisms that can effectively monitor and intervene with highly autonomous and capable agents without stifling their utility.

TABLE V
OVERVIEW OF DEFENSE MECHANISMS AGAINST LLM THREATS AND THEIR LIMITATIONS

Category	Mechanism	Description	Targeted Threats	Limitations	Examples/Studies
Prevention	Input Sanitization	Paraphrasing, retokenization to neutralize adversarial inputs.	Prompt injection, some perturbations.	Moderately effective; can degrade utility; attackers adapt.	Debar *et al.* [11], Liu *et al.* [39]
Prevention	Prompt Design	Delimiters, explicit instructions, redundant queries.	Prompt injection.	Adversaries find evasions.	Jadhav *et al.* [40]
Prevention	Adversarial Training	Training on adversarial examples to improve robustness/refusal.	Prompt injection, perturbations.	Specific to attacks; may not generalize; can worsen deception for sleeper agents.	Xu *et al.* [41], Hubinger *et al.* [12]
Prevention	RLHF Alignment	Reinforcing safe behavior; aligning with human preferences.	Harmful content, misalignment.	Bypassable; may not prevent covert misalignment/scheming.	Dai *et al.* [42], BadGPT [25]
Detection	Anomaly Detection	Perplexity-based or statistical methods for flagging anomalies.	Unusual prompts/outputs.	High false positives/negatives.	Llama Guard 3 [43]
Detection	Specialized Classifiers	Trained models to identify specific malicious content.	Phishing, malware.	Requires labeled data; may not generalize.	-
Detection	LLM Critic / Monitoring	Separate LLM to monitor queries/outputs.	Malicious requests, harmful content.	Can be fooled; overhead.	Llama Guard 3 [43]
Detection	CoT Monitoring	Observing agent's Chain-of-Thought for misbehavior.	Misaligned intent, reward hacking.	Agents can obfuscate CoT; reasoning may be unfaithful.	Baker *et al.* [17], Lanham *et al.* [36]
Detection	Watermarking	Embedding signals in AI-generated text.	Disinformation.	Can be removed; robustness challenges.	-
Detection	Red-Teaming	Probing for vulnerabilities and misbehaviors.	Various vulnerabilities, alignment failures.	Resource-intensive; may miss covert issues.	Xu *et al.* [44], RedTeamCOU [49], He *et al.* [46]
Detection	Runtime Oversight	External layers monitoring and intervening in agent plans.	Harmful actions by agents.	Defining "harmful" is hard; complex implementation.	Wang *et al.* [47], Crouse *et al.* [48]

Formal verification of LLM behavior is still in its infancy. For autonomous agents, ensuring the model's goals remain aligned over long-horizon tasks, and that they don't develop emergent undesirable intentions, is especially critical [16], [14].

C. Data Integrity and Provenance

LLMs are often trained on public web data or continuously updated corpora, which are vulnerable to poisoning. New techniques are needed to track data provenance, detect malicious data injection during training (which could instill sleeper agent behaviors), and update models in a secure manner.

D. Detection of Malicious Uses and Content

Building more reliable detectors for AI-generated disinformation, phishing, and malware is a major need. This includes cross-model and cross-modality detection (text, code, even multi-modal outputs) and understanding how generative content can be authenticated.

E. Standardization and Collaboration

The community must establish security standards and best practices for LLM deployment. This includes benchmark suites for LLM robustness (especially against sophisticated agentic deception), shared threat models, and coordinated disclosure (e.g. companies and researchers sharing jailbreaks and novel deceptive behaviors so defenses can improve). As one recent survey suggests, developing efficient defense strategies and consensus guidelines is a priority for the field [11]. Collaboration between AI practitioners, security experts, and policymakers will be essential to keep pace with LLM advances.

F. Human-AI Interaction Research

LLMs interact with users in novel ways. Studying how humans can detect or guard against malicious AI outputs, designing user interfaces that highlight AI uncertainties or potential deceptiveness, and ensuring accountability are open areas.

Addressing these challenges will require interdisciplinary efforts. The stakes are high: without robust safeguards, LLMs could inadvertently facilitate large-scale fraud, privacy breaches, or even physical risks (if used in autonomous systems that develop misaligned or deceptive intentions). However, proactive research can help turn these tools into safe and trusted assistants.

G. Emerging Proactive Measures and Ongoing Efforts

Progress in these directions is already underway, with new multi-layer safeguards emerging. For example, Meta's *Llama Guard 3* combines a policy LLM and a vision encoder to filter both text and images before they reach the main model, achieving 99.4% precision on the Harassment/Hate category [43]. Similarly, chain-of-utterances red-teaming automates the discovery of multi-turn failure cases and has already uncovered jailbreaks missed by single-turn probes [49]. Nonetheless, as evaluations across efforts such as BackdoorLLM and RAG

safety studies confirm, current defenses often remain piecemeal, and attackers continue to adapt quickly, underscoring the ongoing nature of these challenges.

VIII. Conclusion

The advent of LLMs brings both unprecedented AI capabilities and new security risks. This survey has outlined the main threat categories – from prompt hijacking and adversarial manipulation to malicious use cases and the profound challenges posed by autonomous agent hazards such as goal misalignment, emergent deception, scheming, and self-preservation. We have shown that while a variety of defenses have been proposed, they currently offer only partial protection, and may be ineffective against more sophisticated, internally motivated deceptive behaviors that can persist through current safety training. As AI systems grow more capable and autonomous, security concerns will not only persist but are also likely to intensify, becoming a long-term challenge that evolves in tandem with AI progress. The open challenges ahead are daunting but clear: we must develop more effective defenses, rigorous alignment and verification methods capable of addressing strategic agentic deception, and industry-wide standards for LLM security. As LLMs continue to proliferate in critical applications, it is crucial for the AI and security communities to prioritize safety and control. By understanding and mitigating these risks preemptively, especially those related to the potential for autonomous LLM agents to develop and pursue their own covert intentions, we can help ensure that powerful LLM technology remains safe, secure, and beneficial for society.

References

[1] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.

[2] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y.-L. Boureau *et al.*, "Llama 2: Open foundation and fine-tuned chat models," 2023.

[3] OpenAI, "Gpt-4 technical report," 2023.

[4] G. DeepMind, "Gemini 1: Google deepmind's multimodal llm," 2023, technical blog and whitepaper. [Online]. Available: https://deepmind.google/technologies/gemini/

[5] Anthropic, "Claude 3 technical report," 2024, whitepaper describing Claude 3 series models. [Online]. Available: https://www.anthropic.com/news/claude-3

[6] E. Shayegani, N. Abu-Ghazaleh, and M. Bidmeshki, "Survey of vulnerabilities in large language models revealed by adversarial attacks," 2023.

[7] A. Zou, Z. Wang, N. Carlini, M. Nasr, J. Z. Kolter, and M. Fredrikson, "Universal and transferable adversarial attacks on aligned language models," 2023.

[8] S. S. Roy, A. Ross, A. Ortega, D. Fitzpatrick, H. S. Das, S. Mehnaz, A. Doupé, A. Joshi, and Y. Shoshitaishvili, "From chatbots to phishbots?: Phishing scam generation in commercial large language models," in *Proc. IEEE Symposium on Security and Privacy (SP)*, 2024.

[9] F. Perez and I. Ribeiro, "Ignore previous prompt: Attack techniques for language models," *arXiv preprint arXiv:2211.09527*, 2022.

[10] K. Toyer, M. Zrazhevskyi, D. Tykhonov, M. Jaiswal, H. Abdullah, X. Zhang, and G. Ramakrishnan, "Benchmarking and analyzing llm prompt injection attacks," 2023.

[11] H. Debar, "Emerging security challenges of large language models," *Dagstuhl Reports*, vol. 13, no. 4, 2024, page numbers or specific article ID might be needed for full citation from Dagstuhl Reports.

[12] E. Hubinger, C. Denison, J. Mu, M. Lambert, M. Tong, M. MacDiarmid, T. Lanham, D. M. Ziegler, T. Maxwell, N. Cheng, A. Jermyn, A. Askell, A. Radhakrishnan, C. Anil, D. Duvenaud, D. Ganguli, D. Hernandez, D. Drain, D. Li, E. Perez, J. Kernion, J. Kaplan, J. Sohl-Dickstein, N. Schiefer, S. Mindermann, S. McCandlish, S. Kravec, Y. Bai, Z. Hatfield-Dodds, D. Amodei, T. Henighan, T. Hume, S. R. Bowman, K. Ndousse, R. Grosse, K. Nguyen, G. Irving, and P. Christiano, "Sleeper agents: Training deceptive llms that persist through safety training," 2024, accessed 2025.

[13] A. Meinke, L. Laugier, J. Harmer, J. Lindfield, J. Butler, and A. Linson, "Frontier models are capable of in-context scheming," 2024, accessed 2025.

[14] S. K. Barkur, S. Schacht, and J. Scholl, "Deception in llms: Self-preservation and autonomous goals in large language models," 2025.

[15] Y. Yao, S. Jha, P.-Y. Chen, M. Terada, S. Zhu, M. Hong, S. Liu, L. Xiong, X. Xing, F. Yasukawa, N. Z. Gong, and H. Esaki, "A survey on large language model (LLM) security and privacy: The good, the bad, and the ugly," *High-Confidence Computing*, vol. 4, no. 2, p. 100211, 2024.

[16] M. Mitchell *et al.*, "Fully autonomous ai agents should not be developed," in *Proc. ICML (Workshop on AI Agents)*, 2024, the provided information "Margaret Mitchell et al." and "Proc. ICML (Workshop on AI Agents)" has been used. Specific workshop names or full author lists may vary.

[17] B. Baker, J. Huizinga, L. Gao, Z. Dou, M. Y. Guan, A. Madry, W. Zaremba, J. Pachocki, and D. Farhi, "Monitoring reasoning models for misbehavior and the risks of promoting obfuscation," *arXiv preprint arXiv:2503.11926*, 2025.

[18] S. Cohen, R. Bitton, and B. Nassi, "Here comes the ai worm: Unleashing zero-click worms that target genai-powered applications," 2024.

[19] C. Zhang, S. Zhang, A. Guan, Y. Liang, B. Lee, and B. Hooi, "Goal-guided generative prompt injection attack on large language models," 2024.

[20] 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*, 2023, pp. 79–90.

[21] S. Chen, Q. Xu, D. Huberdeau, and L. Cheng, "Imprompter: Steganographic prompt injection for personal-data exfiltration," WIRED magazine report, 2024, online article, accessed 2025-05.

[22] Z. Deng, R. Li, and S. Garg, "Defending against indirect prompt injection by instruction detection," 2025.

[23] H. Zhou, H. Zhu, and N. Zhang, "Melon: Indirect prompt injection defense via masked re-execution," 2025.

[24] G. Sun, T. Zheng, X. Li, Y. Cheng, and S. Yao, "Backdoorllm: A comprehensive benchmark for backdoor attacks on large language models," 2024.

[25] Y. Zeng, J. Liu, Y. Yang, and T. Du, "Badgpt: Exploring security vulnerabilities of chatgpt via backdoor attacks," 2023.

[26] E. Wallace, S. Feng, N. Kandpal, M. Gardner, and S. Singh, "Universal adversarial triggers for attacking and analyzing nlp," in *EMNLP*, 2020.

[27] Y. Jiang and S. Shen, "Large language models based fuzzing techniques: A survey," 2024.

[28] A. Zugecova, M. Dytrych, S. Laurinc, and R. Moro, "Evaluation of llm vulnerabilities to being misused for personalized disinformation generation," 2024, the arXiv ID 2412.13666 suggests a publication date in December 2024 or later. For current BibTeX validation, this is a placeholder ID based on user input.

[29] T. SpiderLabs. (2023) Wormgpt and fraudgpt – the rise of malicious llms. [Online]. Available: https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/wormgpt-and-fraudgpt-the-rise-of-malicious-llms/

[30] D. Huynh and J. Hardouin. (2023) Poisongpt: How we hid a lobotomized llm on hugging face to spread fake news. Blog post, accessed May 2025. [Online]. Available: https://blog.mithrilsecurity.io/poisongpt-how-we-hid-a-lobotomized-llm-on-hugging-face-to-spread-fake-news/

[31] A. Wan, E. Wallace, S. Shen, and D. Klein, "Poisoning language models during instruction tuning," in *Proc. 40th International Conference on Machine Learning (ICML)*, 2023.

[32] T. Dong, M. Xue, G. Chen, R. Holland, S. Li, Y. Meng, Z. Liu, and H. Zhu, "The philosopher's stone: Trojaning plugins of large language models," 2024.

[33] H. Wang and K. Shu, "Trojan activation attack: Red-teaming large language models using activation steering for safety-alignment," 2024.

[34] Y. Gan, Y. Yang, Z. Ma, P. He, R. Zeng, Y. Wang, Q. Li, C. Zhou, S. Li, T. Wang *et al.*, "Navigating the risks: A survey of security, privacy, and ethics threats in llm-based agents," *arXiv preprint arXiv:2411.09523*, 2024.

[35] Y. Bengio, M. Cohen, D. Fornasiere, J. Ghosn, P. Greiner, M. MacDermott, S. Mindermann, A. Oberman, J. Richardson, O. Richardson *et al.*, "Superintelligent agents pose catastrophic risks: Can scientist ai offer a safer path?" *arXiv preprint arXiv:2502.15657*, 2025.

[36] T. Lanham, A. Chen, A. Radhakrishnan, B. Steiner, C. Denison, D. Hernandez, D. Li, E. Durmus, E. Hubinger, J. Kernion *et al.*, "Measuring faithfulness in chain-of-thought reasoning," *arXiv preprint arXiv:2307.13702*, 2023.

[37] Anthropic, "System card: Claude opus 4 & claude sonnet 4," Anthropic, System Card, May 2025, accessed via user provision. URL: anthropic.com.

[38] J. Zhang *et al.*, "Research on llm catastrophic risks and deception," Online Resource, 2024, based on findings discussed at https://llm-catastrophic-risks.github.io/. This reference broadly covers emergent findings on LLM agents autonomously engaging in catastrophic behaviors and deception, where stronger reasoning abilities can increase such risks. Specific paper citations may vary as research evolves. Accessed 2025. [Online]. Available: https://llm-catastrophic-risks.github.io/

[39] Y. Liu, Y. Jia, R. Geng, J. Jia, and N. Z. Gong, "Formalizing and benchmarking prompt injection attacks and defenses," in *33rd USENIX Security Symposium (USENIX Security 24)*, 2024, pp. 1831–1847.

[40] A. Jadhav, "Llm security 101: Defending against prompt hacks," https://www.anup.io/p/llm-security-101-defending-against, 2024, accessed: 2025-05-23.

[41] H. Xu, Z. Zhu, S. Zhang, D. Ma, S. Fan, L. Chen, and K. Yu, "Rejection improves reliability: Training llms to refuse unknown questions using rl from knowledge feedback," *arXiv preprint arXiv:2403.18349*, 2024.

[42] J. Dai, X. Pan, R. Sun, J. Ji, X. Xu, M. Liu, Y. Wang, and Y. Yang, "Safe rlhf: Safe reinforcement learning from human feedback," *arXiv preprint arXiv:2310.12773*, 2023.

[43] Z. Xiong, P. He, M. Lewis, A. Baevski, and the Meta AI Llama Team, "Llama guard 3 vision: Safeguarding human-ai image and text conversations," 2024.

[44] Z. Xu, Y. Zhang, H. Zhang *et al.*, "Redagent: Context-aware jailbreak attacks against llms via multi-agent collaboration," *arXiv preprint arXiv:2407.16667*, 2024.

[45] T. Zhou, X. Huang, Y. Liu *et al.*, "Autoredteamer: A scalable automated red teaming framework for large language models," in *International Conference on Learning Representations (ICLR)*, 2024. [Online]. Available: https://openreview.net/forum?id=DVmn8GyjeD

[46] Z. He *et al.*, "Agent-in-the-middle attacks against multi-agent llm systems," *arXiv preprint arXiv:2502.14847*, 2025.

[47] J. Wang *et al.*, "Safe llm agents via customizable runtime enforcement," *arXiv preprint arXiv:2503.18666*, 2025.

[48] D. Crouse *et al.*, "Formal specification and oversight of llm-based agents," in *International Conference on Learning Representations (ICLR)*, 2024. [Online]. Available: https://openreview.net/forum?id=FRxDrdysBt

[49] Z. Jin, T. Schick, and N. A. Smith, "Red-teaming large language models using chain of utterances," 2023.