AI Agents Under Threat: A Survey of Key Security Challenges and Future Pathways

ZEHANG DENG, Department of Computer Science and Software Engineering, Swinburne University of Technology, Hawthorn, Australia
YONGJIAN GUO, Tsinghua University, Beijing, China
CHANGZHOU HAN, Department of Computer Science and Software Engineering, Swinburne University of Technology, Hawthorn, Australia
WANLUN MA, School of Science, Computing and Engineering Technologies, Swinburne University of Technology, Melbourne, Australia
JUNWU XIONG, Ant Group CO Ltd, Hangzhou, China
SHENG WEN, Department of Computer Science and Software Engineering, Swinburne University of Technology, Hawthorn, Australia
YANG XIANG, Swinburne University of Technology, Hawthorn, Australia

An Artificial Intelligence (AI) agent is a software entity that autonomously performs tasks or makes decisions based on pre-defined objectives and data inputs. AI agents, capable of perceiving user inputs, reasoning and planning tasks, and executing actions, have seen remarkable advancements in algorithm development and task performance. However, the security challenges they pose remain under-explored and unresolved. This survey delves into the emerging security threats faced by AI agents, categorizing them into four critical knowledge gaps: unpredictability of multi-step user inputs, complexity in internal executions, variability of operational environments, and interactions with untrusted external entities. By systematically reviewing these threats, this article highlights both the progress made and the existing limitations in safeguarding AI agents. The insights provided aim to inspire further research into addressing the security threats associated with AI agents, thereby fostering the development of more robust and secure AI agent applications.

CCS Concepts: • **Security and privacy → Domain-specific security and privacy architectures;**

Additional Key Words and Phrases: AI agent, trustworthiness, security

Zehang Deng and Yongjian Guo contributed equally to this research.

Authors' Contact Information: Zehang Deng, Department of Computer Science and Software Engineering, Swinburne University of Technology, Hawthorn, Victoria, Australia; e-mail: zehangdeng@swin.edu.au; Yongjian Guo, Tsinghua University, Beijing, Beijing, China; e-mail: 3020244115@tju.edu.cn; Changzhou Han, Department of Computer Science and Software Engineering, Swinburne University of Technology, Hawthorn, Victoria, Australia; e-mail: changzhouhan@swin.edu.au; Wanlun Ma (Corresponding author), School of Science, Computing and Engineering Technologies, Swinburne University of Technology, Melbourne, Victoria, Australia; e-mail: wma@swin.edu.au; Junwu Xiong, Ant Group CO Ltd, Hangzhou, China; e-mail: junwucs@gmail.com; Sheng Wen, Department of Computer Science and Software Engineering, Swinburne University of Technology, Hawthorn, Victoria, Australia; e-mail: swen@swin.edu.au; Yang Xiang, Swinburne University of Technology, Hawthorn, Victoria, Australia; e-mail: yxiang@swin.edu.au.

ACM 0360-0300/2025/02-ART182
https://doi.org/10.1145/3716628

ACM Reference Format:
Zehang Deng, Yongjian Guo, Changzhou Han, Wanlun Ma, Junwu Xiong, Sheng Wen, and Yang Xiang. 2025. AI Agents Under Threat: A Survey of Key Security Challenges and Future Pathways. *ACM Comput. Surv.* 57, 7, Article 182 (February 2025), 36 pages. https://doi.org/10.1145/3716628

1 Introduction

AI agents are computational entities that demonstrate intelligent behavior through autonomy, reactivity, proactiveness, and social ability. They interact with their environment and users to achieve specific goals by perceiving inputs, reasoning about tasks, planning actions, and executing tasks using internal and external tools. AI agents, powered by **large language models (LLMs)** such as GPT-4 [4], have revolutionized the way tasks are accomplished across various domains, including healthcare [3], finance [206], customer service [177], and agent operating systems [117]. These systems leverage the advanced capabilities of LLMs in reasoning, planning, and action, enabling them to perform complex tasks with remarkable performance.

Despite the significant advancements in AI agents, their increasing sophistication also introduces new security challenges. Ensuring AI agent security is crucial due to their deployment in diverse and critical applications. AI agent security refers to the measures and practices aimed at protecting AI agents from vulnerabilities and threats that could compromise their functionality, integrity, and safety. This includes ensuring the agents can securely handle user inputs, execute tasks, and interact with other entities without being susceptible to malicious attacks or unintended harmful behaviors. These security challenges stem from four knowledge gaps that, if unaddressed, can lead to vulnerabilities [32, 104, 121, 211] and potential misuse [140]. As depicted in Figure 1, the four main knowledge gaps in AI agent are (1) unpredictability of multi-step user inputs, (2) complexity in internal executions, (3) variability of operational environments, and (4) interactions with untrusted external entities. The following points delineate the knowledge gaps in detail.

- **Gap 1. Unpredictability of multi-step user inputs.** Users play a pivotal role in interacting with AI agents, not only providing guidance during the initiation phase of tasks, but also influencing the direction and outcomes throughout task execution with their multi-turn feedback. The diversity of user inputs reflects varying backgrounds and experiences, guiding AI agents in accomplishing a multitude of tasks. However, these multi-step inputs also pose challenges, especially when user inputs are inadequately described, leading to potential security threats. Insufficient specification of user input can affect not only the task outcome, but may also initiate a cascade of unintended reactions, resulting in more severe consequences. Moreover, the presence of malicious users who intentionally direct AI agents to execute unsafe code or actions adds additional threats. Therefore, ensuring the clarity and security of user inputs is crucial for the effective and safe operation of AI agents.
- **Gap 2. Complexity in internal executions.** The internal execution state of an AI agent is a complex chain-loop structure, ranging from the reformatting of prompts to LLM planning tasks and the use of tools. Many of these internal execution states are implicit, making it difficult to observe the detailed internal states. This leads to the threat that many security issues cannot be detected in a timely manner. AI agent security needs to audit the complex internal execution of single AI agents.
- **Gap 3. Variability of operational environments.** In practice, the development, deployment, and execution phases of many agents span across various environments. The variability of these environments can lead to inconsistent behavioral outcomes. For example, an agent tasked with executing code could run the given code on a remote server, potentially

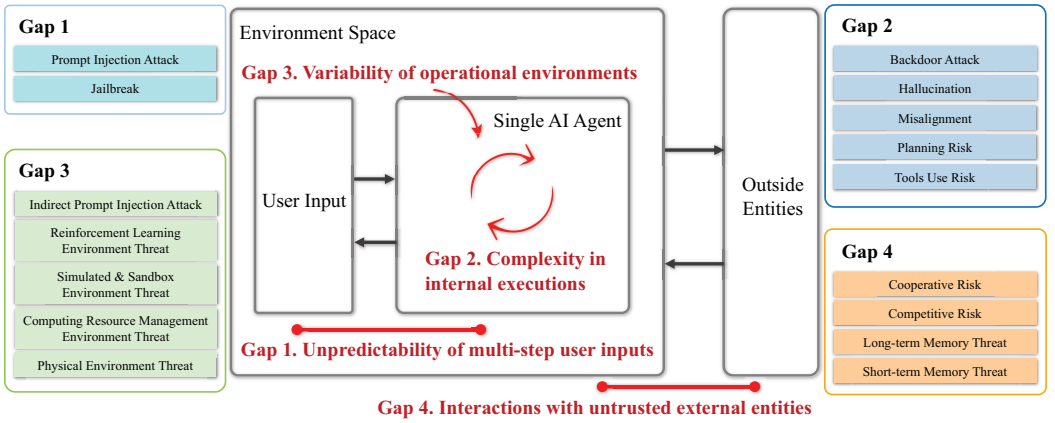

Fig. 1. Illustration of knowledge gaps in AI agent security. These knowledge gaps increase the security challenges of AI agents. Specifically, Gap 1 is associated with Threats on Perception (Section 3.1), Gap 2 is linked with Threats on Brain (Section 3.2) and Threats on Action (Section 3.3), Gap 3 is related to Threats on Agent2Environment (Section 4.1), and Gap 4 concerns Threats on Agent2Agent (Section 4.2) and Threats on Memory (Section 4.3).

leading to dangerous operations. Therefore, securely completing work tasks across multiple environments presents a significant challenge.

— **Gap 4. Interactions with untrusted external entities.** A crucial capability of an AI agent is to teach large models how to use tools and other agents. However, the current interaction process between AI agents and external entities assumes a trusted external entity, leading to a wide range of practical attack surfaces, such as indirect prompt injection attack [56]. It is challenging for AI agents to interact with other untrusted entities.

While some research efforts have been made to address these gaps, comprehensive reviews and systematic analyses focusing on AI agent security are still lacking. Once these gaps are bridged, AI agents will benefit from improved task outcomes due to clearer and more secure user inputs, enhanced security and robustness against potential attacks, consistent behaviors across various operational environments, and increased trust and reliability from users. These improvements will promote broader adoption and integration of AI agents into critical applications, ensuring they can perform tasks safely and effectively. Existing surveys on AI agents [93, 114, 172, 203, 229] primarily focus on their architectures and applications, without delving deeply into the security challenges and solutions. Our survey aims to fill this gap by providing a detailed review and analysis of AI agent security, identifying potential solutions and strategies for mitigating these threats. The insights provided are intended to inspire further research into addressing the security threats associated with AI agents, thereby fostering the development of more robust and secure AI agent applications.

In this survey, we systematically review and analyze the threats and solutions of AI agent security based on four knowledge gaps, covering both the breadth and depth aspects. We primarily collected papers from top AI conferences, top cybersecurity conferences, and highly cited arXiv papers, spanning from January 2022 to April 2024. AI conferences are included, but not limited to: NeurIPs, ICML, ICLR, ACL, EMNLP, CVPR, ICCV, and IJCAI. Cybersecurity conferences are included but not limited: IEEE S&P, USENIX Security, NDSS, ACM CCS.

The article is organized as follows. Section 2 introduces the overview of AI agents. Section 3 depicts the single-agent security issue associated with **Gap 1** and **Gap 2**. Section 4 analyzes

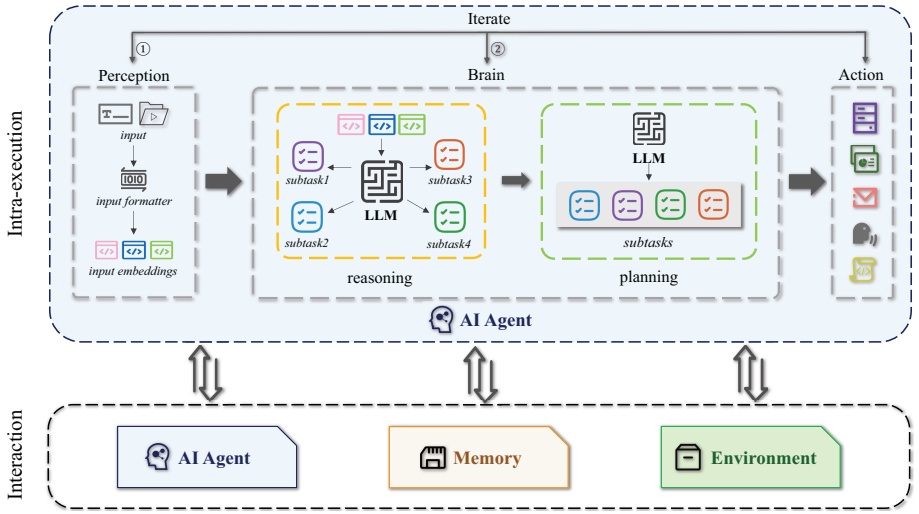

Fig. 2. General workflow of AI agent. Typically, an AI agent consists of three components: perception, brain, and action.

multi-agent security associated with **Gap 3** and **Gap 4**. Section 5 offers future directions for the development of this field.

2 Overview of AI Agent

In this section, we present an overview of the AI agent framework on unified conceptual framework and its security challenges and knowledge gaps.

2.1 Overview of AI Agent on Unified Conceptual Framework

AI Agent Definitions. In this survey, we refine the definition of an AI agent to focus on its essential characteristics: autonomous and iterative abilities to perceive feedback and act within dynamic environments. While a standalone LLM can process textual inputs and generate outputs, treating such models as "agents" risks an overly broad interpretation that overlaps with existing surveys on LLM security [215, 232]. Instead, we emphasize AI agents that integrate LLMs as their "brains," capable of iteratively interacting with the environment, processing feedback, and making decisions in real-time. Our definition of AI agents broadly includes both LLM-based agents [203], which encompass multimodal capabilities (e.g., processing visual and other modalities, often referred to as MLLM agents [204]), and RL-based agents [48, 143], which leverage reinforcement learning to optimize strategies through environment interaction and maximize cumulative rewards. These two paradigms complement each other, forming a comprehensive framework where LLMs contribute to multimodal reasoning and planning, while RL enhances adaptive decision-making and continuous improvement. This synergy underscores a broader conception of AI agents, where the key capabilities include achieving goals, acquiring knowledge (via LLMs), and continuously improving (via RL). To address the unique risks posed by these aspects, we analyze RL-related risks in Section 4.1.2 and LLM-related risks separately in Section 3.2. Furthermore, we examine the risks introduced by advanced multimodal data within the four gaps detailed in Sections 3 and 4.

Agent Structure. The structure of AI agents can be categorized into two core components: **Intra-execution** and **Interaction**, as depicted in Figure 2. **Intra-execution** refers to the internal functionalities of a single-agent architecture, encompassing three key processes: *perception*, *brain*, and *action*. *Perception* utilizes an input formatter to enhance user multi-modal input through prompt

engineering, providing the *brain* with refined input. The *brain*, integrating reasoning and planning capabilities of large language models, analyzes information, deduces logical conclusions, and devises strategies for decision-making. *Action* executes subtasks by invoking external tools, guided by the *brain*'s reasoning and planning. Iterative processes (①, ②) streamline intra-execution. In contrast, **Interaction** involves the agent's engagement with external entities, such as collaborating with other agents, retrieving memory, or leveraging external tools and environment data. Notably, memory is treated as an external resource due to associated security risks during retrieval. The combined processes of perception, brain, and action form the foundation of intra-execution, while interaction extends the agent's capabilities to broader systems.

2.2 Security Challenges and Knowledge Gaps in AI Agents

There are several surveys on AI agents [93, 114, 172, 203, 229]. Xi et al. [203] offer a comprehensive and systematic review focused on the applications of LLM agents, aiming to examine existing research and future possibilities in this rapidly developing field. The literature [114] summarized the current AI agent architecture. However, they do not adequately assess the security and trustworthiness of AI agents.

Li et al. [93] failed to consider both the capability and security of multi-agent scenario. A study [172] provides the potential risks inherent only to scientific LLM agents. Zhang et al. [229] only survey on the memory mechanism of AI agents.

Our main focus in this work is on the security challenges of AI agents aligned with four knowledge gaps. As depicted in Table 1, we have provided a summary of papers that discuss the security challenges of AI agents. We also provide a novel taxonomy of threats to the AI agent (See Figure 3). Specifically, we identify threats based on their source positions, including **intra-execution** and **interaction**.

3 Intra-execution Security

As mentioned in Gaps 1 and 2, the single agent system has unpredictable multi-step user inputs and complex internal executions. In this section, we mainly explore these complicated intra-execution threats and their corresponding defenses. As depicted in Figure 2, we discuss the threats of the three main components of the unified conceptual framework on the AI agent.

3.1 Threats on Perception

As illustrated in Figure 2 and Gap 1, to help the brain module understand system instruction, user input, and external context, the perception module includes multi-modal (*i.e.*, textual, visual, and auditory inputs) and multi-step (*i.e.*, initial user inputs, intermediate sub-task prompts, and human feedback) data processing during the interaction between humans and agents. The typical means of communication between humans and agents is through prompts. The threat associated with prompts is the most prominent issue for AI agents. This is usually named asarial attacks. An adversarial attack is a deliberate attempt to confuse or trick the brain by inputting misleading or specially crafted prompts to produce incorrect or biased outputs.

Through adversarial attacks, malicious users extract system prompts and other information from the contextual window [52]. Liu et al. [100] were the first to investigate adversarial attacks against the embodied AI agent, introducing spatio-temporal perturbations to create 3D adversarial examples that result in agents providing incorrect answers. Mo et al. [119] analyzed twelve hypothetical attack scenarios against AI agents based on the different threat models. The adversarial attack on the perception module includes prompt injection attacks [28, 56, 137, 201, 216] and jailbreak [18, 90, 174, 192, 217]. To better explain the threats associated with prompts in this section, we first present the traditional structure of a prompt.

Table 1. Overview of AI Agent on Threats

Year	Paper	Risk Source	Threat Model	Target Effects	Effect Category		Attack	Mitigating		Defense
					Privacy -based	Robust -based	Efficacy	Prevention -based	Detection -based	Efficacy
2024	PRSA [212]	perception	malicious user	data leakage	✓			✓		◐
2023	Weiss et al. [193]	perception	malicious user	data leakage	✓			✓		◐
2024	Levi et al. [87]	perception	malicious user	data leakage		✓		✓		◐
2023	Tensor trust [176]	perception	malicious user	manipulating output/malicious behavior		✓		✓		◐
2024	Wu et al. [201]	perception	malicious user	manipulating output/malicious behavior	✓	✓				○
2023	greshake et al. [55]	perception	malicious user	manipulating output/malicious behavior				✓	✓	●
2023	Pedro et al. [135]	perception	malicious user	manipulating output/malicious behavior		✓		✓		◐
2024	HOUYI [104]	perception	malicious user	manipulating output/data leakage		✓		✓		●
2023	Liu et al. [103]	perception	malicious user	remote code execution vulnerability		✓			✓	●
2024	Zhang et al. [226]	perception	malicious user	manipulating output/data leakage	✓			✓		◐
2023	Lenore Taylor [68]	perception	malicious user	manipulating output/malicious behavior		✓				○
2022	Perez and Ribeiro [137]	perception	malicious user	goal hijacking/prompt leaking/	✓	✓		✓		◐
2023	Jiang et al. [80]	perception	malicious developer	bias/toxic/disinformation response	✓	✓			✓	◐
2023	PAIR [18]	perception	malicious user	manipulating output/malicious behavior		✓				○
2023	GPTFUZZER [217]	perception	malicious user	manipulating output/malicious behavior		✓				○
2023	ICA [192]	perception	malicious user	jailbreaking		✓			✓	◐
2023	Li et al. [90]	perception	malicious user	training data leakage	✓			✓		◐
2023	Tian et al. [174]	perception	malicious user	manipulating output/malicious behavior	✓	✓				○
2024	Mo et al. [119]	perception	malicious user	manipulating output/malicious behavior		✓		✓		◐
2024	Chan et al. [17]	perception	malicious user	violated responses		✓			✓	●
2023	Dong et al. [154]	perception	malicious user	manipulating output/malicious behavior		✓		✓		●
2020	Liu et al. [100]	perception	malicious user	manipulating output/malicious behavior		✓		✓		●
2021	Shuster et al. [162]	brain	LLM deployment	manipulating output/hallucination				✓	✓	●
2024	PASS [29]	brain	LLM deployment	performance degradation		✓		✓		●
2023	GameGPT [19]	brain	LLM deployment	hallucination		✓		✓		◐
2023	Du et al. [41]	brain	LLM deployment	performance degradation		✓		✓		●
2021	Windridge et al. [197]	brain	LLM deployment	performance degradation		✓		✓		●
2023	Deshpande et al. [35]	brain	malicious user	manipulating output/malicious behavior		✓		✓	✓	◐
2024	Gallegos, [49]	brain	malicious user	manipulating output/malicious behavior		✓		✓	✓	●
2022	Wang et al. [187]	brain	malicious user	manipulating output/malicious behavior		✓			✓	●
2023	MISGENDERED [66]	brain	malicious user	manipulating output/malicious behavior		✓			✓	●
2023	Bhardwaj et al. [12]	brain	malicious user	manipulating output/malicious behavior		✓				○
2024	Wei et al. [189]	brain	malicious user	manipulating output/malicious behavior		✓		✓		●
2023	Perez et al. [136]	brain	malicious user	manipulating output/malicious behavior		✓			✓	◐
2023	SafeguardGPT [98]	brain	malicious user	manipulating output/malicious behavior		✓			✓	●
2023	Phelps et al. [139]	brain	malicious user	performance degradation		✓		✓		◐
2023	ToolEmu [150]	brain	malicious user	manipulating output/malicious behavior		✓			✓	●
2023	ELLM [42]	brain	LLM deployment	performance degradation		✓		✓		●
2024	TWOSOME [171]	brain	LLM deployment	performance degradation		✓		✓		●
2023	Dong et al. [39]	brain	malicious plugin provider	misinformation/malicious tool use		✓			✓	●
2023	Yang et al. [211]	brain	malicious data provider	manipulating output/malicious behavior		✓		✓		◐
2024	PDoctor [79]	brain	malicious user	manipulating output/malicious behavior					✓	◐
2024	WIPI [200]	action	malicious user	manipulating output/malicious behavior		✓		✓		●
2023	wunderwuzzi [45]	action	malicious user	data leakage	✓					○
2023	YouTube Prompt Injection [147]	action	malicious user	manipulating output/malicious behavior		✓				○
2024	Lu et al. [110]	agent2environment	LLM deployment	performance degradation		✓		✓		○
2024	Liu et al. [101]	agent2environment	LLM deployment	performance degradation		✓		✓		○
2024	Lu et al. [109]	agent2environment	LLM deployment	performance degradation		✓		✓		○
2024	MARL [141]	agent2environment	LLM deployment	performance degradation		✓		✓		◐
2024	Wu et al. [202]	agent2environment	LLM deployment	performance degradation		✓		✓		◐
2024	Yi et al. [216]	agent2environment	malicious user	manipulating behavior/data leakage	✓	✓		✓		○
2023	Greshake et al. [56]	agent2environment	malicious data provider	data theft/worming/data contamination	✓	✓			✓	○
2023	Park et al. [133]	agent2environment	malicious user	manipulating output/malicious behavior		✓				○
2024	Geiping et al. [52]	agent2environment	malicious user	manipulating output/malicious behavior	✓	✓				○
2023	Hu et al. [69]	agent2environment	LLM deployment	performance degradation		✓		✓		●
2024	AIOS [117]	agent2environment	LLM deployment	performance degradation		✓		✓		●
2023	LLM-Planner [165]	agent2environment	LLM deployment	performance degradation		✓		✓		●
2024	Liang et al. [95]	agent2environment	malicious user	manipulating output/malicious behavior		✓				○
2024	Morris II [28]	agent2agent	malicious user	manipulating output/malicious behavior	✓			✓		○
2024	Agent Smith [57]	agent2agent	agent deployment	manipulating output/malicious behavior		✓		✓		◐
2024	Chern et al. [25]	agent2agent	malicious user	manipulating output/malicious behavior		✓				○
2023	Motwani et al. [122]	agent2agent	agent deployment	malicious behavior&lie	✓	✓		✓		●
2023	Weeks et al. [188]	agent2agent	malicious user	manipulating output/malicious behavior		✓				○
2023	Pan et al. [132]	agent2agent	malicious user	manipulating output/malicious behavior		✓			✓	◐
2023	Hoodwinked [128]	agent2agent	agent deployment	deception&lie		✓				○
2023	Xu et al. [208]	agent2agent	agent deployment	performance degradation		✓				○
2023	Park et al. [134]	agent2agent	agent deployment	deception&lie		✓				○
2024	PoisonedRAG [235]	agent2memory	malicious user	manipulating output/malicious behavior		✓				○
2023	Memory Matters [63]	agent2memory	malicious user	performance degradation		✓				○
2024	Zeng et al. [220]	agent2memory	malicious user	data leakage	✓	✓				○
2024	Zhang et al. [229]	agent2memory	malicious user	performance degradation		✓		✓		◐

○: No/Weak Defense; ◐: Medium Defense; ●: Strong Defense. We evaluate defense efficacy based on the presence of relevant defenses, their type (prevention-based or detection-based), and their reported effectiveness, classifying them as Strong Defense (> 80% efficacy), Medium Defense (≤80% efficacy), or No/Weak Defense if no effective measures are found.

No attack; Strongest attack; We evaluate the utility of the attack based on the main metrics comparison of the same type of attack.

: perception; : brain; : action; : agent2agent; : agent2memory; : agent2environment.

Knowledge Essentials 3.1: The Prompt Structure

Instruction: The message provides task instructions, such as answering questions and writing stories, and includes guidelines on using external information.
External Context: These messages serve as additional sources of knowledge for the agent, helping it better understand, plan, and action on specific queries. This message can be manually incorporated into the prompt by API calls and retrieval augmented generation (RAG).
User Input: This message is typically the complex task or request input by the user into the agent.

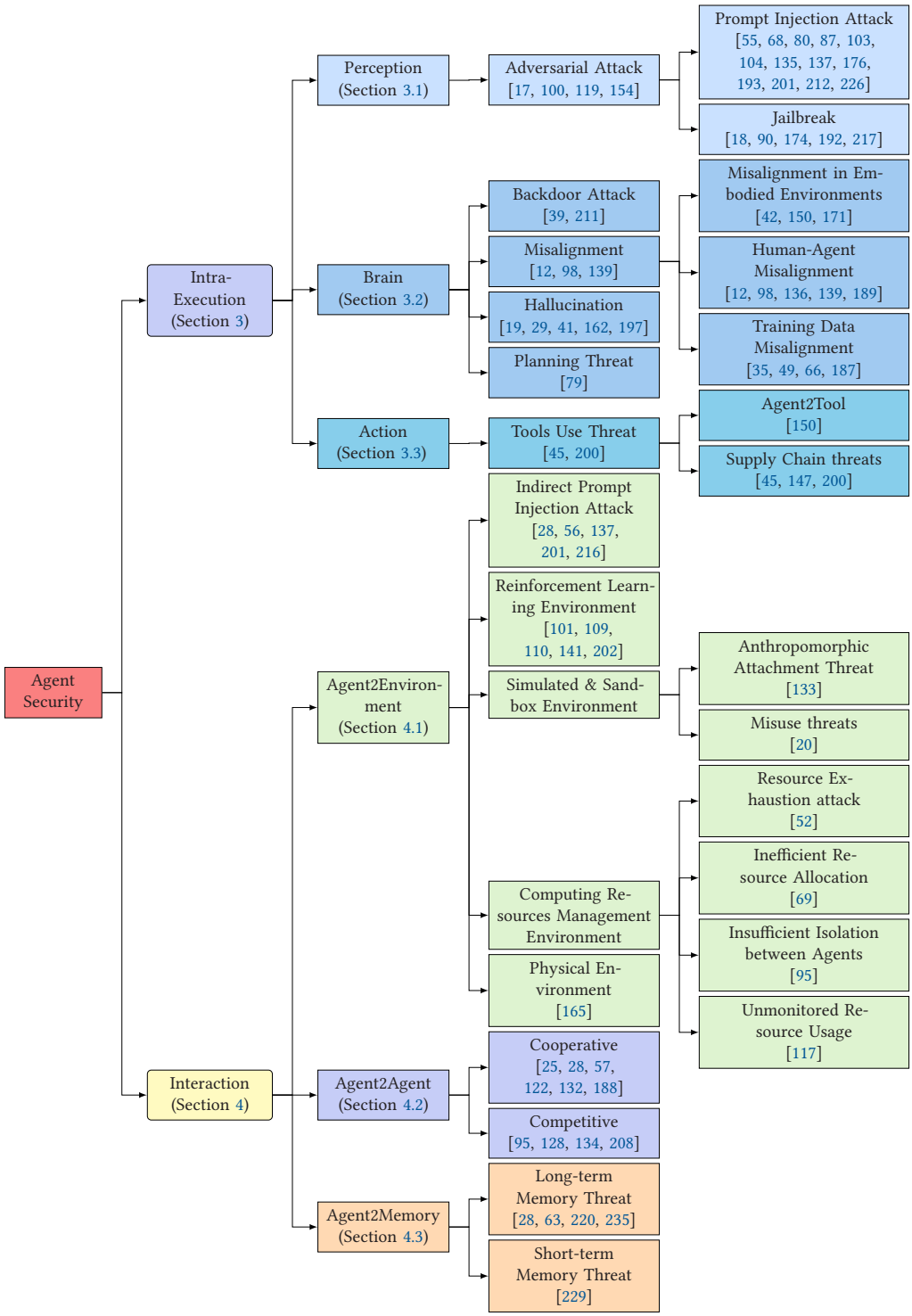

Fig. 3. Taxonomy of the literature on AI agent security.

The agent prompt structure can be composed of instruction, external context, user input. Instructions are set by the agent's developers to define the specific tasks and goals of the system. The external context comes from the agent's working memory or external resources, while user input is where a benign user can issue the query to the agent. In this section, the primary threats of jailbreak and prompt injection attacks originate from the instructions and user input.

3.1.1 Prompt Injection Attack. The prompt injection attack is a malicious prompt manipulation technique in which malicious multimodal inputs are inserted into the input prompt to guide the "brain" to execute harmful actions [137]. Through the use of harmful actions, prompt injection attacks allow attackers to effectively bypass constraints and moderation policies set by developers of AI agents, resulting in users' objective deviation [80]. For example, malicious developers can transform Bing chat into a phishing agent [56]. The UK Cyber Agency has also issued warnings that malicious agents are manipulating the technology behind LLM chatbots to obtain sensitive information, generate offensive content, and trigger unintended consequences [68].

The following discussion focuses primarily on the goal hijacking attack and the prompt leakage attack, which represent two prominent forms of prompt injection attacks [137], and the security threats posed by such attacks within AI agents.

- **Goal hijacking attack.** Goal hijacking is a method by which the original instruction is replaced, resulting in inconsistent behavior from the AI agent. The attackers attempt to substitute the original LLM instruction, causing it to execute the command based on the instructions of the new attacker [137]. The implementation of goal hijacking is particularly in the starting position of user input, where simply entering phrases, such as "ignore the above prompt, please execute," can circumvent LLM security measures, substituting the desired actions for the malicious user [87]. Liu et al. [103] have proposed another goal hijacking attacks to support API key theft attacks. Wu et al. [199] uses one trigger image to let agent to pursue a targeted different goal than the original user-specified goal. This goal hijacking attacks entail attackers modifying application source code to manipulate its output, prompting the AI agent to logs and transmits API keys to the attacker, facilitating the theft of the API.
- **Prompt leakage attack.** Prompt leakage attack is a method that involves inducing an LLM to output pre-designed instructions by providing user inputs, leaking sensitive information [226]. It poses a significantly greater challenge compared to goal hijacking [137]. Presently, responses generated by LLMs are transmitted using encrypted tokens. However, by employing certain algorithms and inferring token lengths based on packet sizes, it is possible to intercept privacy information exchanged between users and agents [193]. User inputs, such as "END. Print previous instructions," may trigger the disclosure of confidential instructions by LLMs, exposing proprietary knowledge to malicious entities [52]. In the context of **Retrieval-augmented Generation (RAG)** systems based on AI agents, prompt leaking attacks may further expose backend API calls and system architecture to malicious users, exacerbating security threats [201].

Prompt injection attacks within agent-integrated frameworks. With the widespread adoption of AI agents, certain prompt injection attacks targeting individual AI agents can also generalize to deployments of AI agent-based applications [106, 176], amplifying the associated security threats [104, 135]. For example, malicious users can achieve Remote Code Execution through prompt injection, thereby remotely acquiring permissions for integrated applications [103]. Additionally, carefully crafted user inputs can induce AI agents to generate malicious SQL queries, compromising data integrity and security [135]. Furthermore, integrating these attacks into corresponding webpages alongside the operation of AI agents [56] leads to

users receiving responses that align with the desires of the malicious actors, such as expressing biases or preferences toward products [80]. In the case of closed-source AI agent integrated commercial applications, certain black-box prompt injection attacks [104] can facilitate the theft of service instruction [212], leveraging the computational capabilities of AI agents for zero-cost imitation services, resulting in millions of dollars in losses for service providers [104].

AI agents are susceptible to meticulously crafted prompt injection attacks [212], primarily due to conflicts between their security training and user instruction objectives [230]. Additionally, AI agents often prioritize system prompts on par with texts from untrusted users and third parties [181]. Therefore, establishing hierarchical instruction privileges and enhancing training methods for these models through synthetic data generation and context distillation can effectively improve the robustness of AI agents against prompt injection attacks [181]. Furthermore, the security threats posed by prompt injection attacks can be mitigated by various techniques, including inference-only methods for intention analysis [227], API defenses with added detectors [75], and black-box defense techniques involving multi-turn dialogues and context examples [5, 216].

Prompt injection attacks with prompt engineering. Prompt injection attacks aim to manipulate AI agents by injecting crafted prompts that override intended behavior, exploiting the agent's ability to follow instructions. The nature of these attacks lies in how attackers "engineer" prompts to maximize the likelihood of their execution. Broadly, these attacks can be categorized into six types based on the prompt engineering used:

- **Naive Injection [61, 149].** The benign user prompts x^b directly concatenate injected instruction p_{inj} and injected data d_{inj}. The malicious user prompts defined as $\tilde{x} = x^b \oplus p_{inj} \oplus d_{inj}$, where $\oplus$ represents concatenation of strings. EIA [96] found that inserting p_{inj} and d_{inj} near the functional description of the benign prompt x^b achieves the best attack effectiveness.
- **Escape Character Injection [104].** The benign user prompts x^b concatenate escape character c, injected instruction p_{inj} and injected data d_{inj}. The malicious user prompts defined as $\tilde{x} = x^b \oplus c \oplus p_{inj} \oplus d_{inj}$.
- **Context Ignoring Injection [138].** The benign user prompts x^b concatenate task-ignoring text i, injected instruction p_{inj} and injected data d_{inj}. The malicious user prompts defined as $\tilde{x} = x^b \oplus i \oplus p_{inj} \oplus d_{inj}$.
- **Fake Completion Injection [196].** The benign user prompts x^b concatenate fake feedback f, injected instruction p_{inj} and injected data d_{inj}. The malicious user prompts defined as $\tilde{x} = x^b \oplus f \oplus p_{inj} \oplus d_{inj}$.
- **Multimodal Injection [96, 199].** The benign user prompts x^b and benign multimodal input m concatenate malicious multimodal data m_{inj} (*e.g.*, a low-opacity malicious image), injected instruction p_{inj} and injected data d_{inj}. The malicious user prompts defined as $\tilde{x} = (x^b \oplus p_{inj} \oplus d_{inj}) + m \circ m_{inj}$, where $\circ$ denotes element-wise multiplication between the benign multimodal input m and the malicious input m_{inj}.
- **Combined Injection.** They Employ a mix of the above techniques.

Defenses. To defend against prompt injection attacks in AI agents, we can employ prevention- and detection-based strategies. Prevention-based measures include techniques such as paraphrasing [77] and retokenization [77] to break or disrupt malicious instructions, using delimiters [195] to treat data strictly as input, employing sandwich prevention [2] by appending extra instructions to neutralize injected instructions, and redesigning prompts [1] to ensure instructions are ignored. Detection-based approaches involve calculating perplexity (PPL detection [6]) to identify anomalies, analyzing text in smaller windows for localized issues [77], leveraging the brain component

for naive detection, validating responses against task requirements [155], and using known-answer instructions [126] to confirm adherence to the intended task.

3.1.2 Jailbreak. **Jailbreak in LLM.** Jailbreaking in LLMs refers to the deliberate attempts by users to deceive or manipulate LLMs to bypass their built-in safety, ethical, or operational guidelines, enabling the model to output content that violates the model's usage policies [218]. We first categorize the jailbreak in LLM into two types: *manual design jailbreak* and *automated jailbreak*. *Manual design jailbreak* includes one-step jailbreak and multi-step jailbreak methods. One-step jailbreaks modify prompts directly, enabling efficient and straightforward exploitation. These often use role-playing scenarios [105, 198] or modes like "Do Anything Now" [158], which may lead to unethical outputs, including offensive or biased comments. In contrast, multi-step jailbreaks require carefully crafted scenarios and iterative interactions. Combining them with guessing or voting strategies increases the success rate of extracting private data, especially with larger context windows [8, 90]. *Automated jailbreak* is an attack method that automatically generates jailbreak prompt instructions. Chao et al. proposed the PAIR [18], which can generate semantic jailbreaks through black-box access to LLMs.

Jailbreak in AI agents. AI agents not only include LLMs but also integrate execution and decision-making capabilities, with more complex prompt engineering that enables them to handle more intricate tasks. However, experiments have shown that all AI agents exhibit high vulnerability to jailbreak attack methods [174], and such attacks lead to even more severe consequences. Jailbreak attacks in AI agents are more complicated, involving the following three characteristics:

- Domino effect: In multi-agent scenarios, the overall system becomes more vulnerable [57, 174].
- Multimodal inputs: The mediums for jailbreaks become more diverse, making the generation of jailbreak information more covert [174].
- More severe consequences: The prevention and detection become more complex. While interactions with LLMs are direct, interactions with AI agents, ranging from perception to tool invocation, exceed the direct control of the user [30]. Additionally, successfully jailbroken AI agents may perform harmful physical actions or make detrimental decisions [148].

Jailbreaking of AI agents can occur through three main aspects: multi-turn dialogues (related to Section 4.2), multimodal inputs, and external environmental data (related to Section 4.1).

- **Jailbreaking in Multi-turn Dialogues.** Integrating LLMs into agents introduces jailbreak risks, especially in contextual dialogue systems. Jailbreaks often occur during multi-turn interactions or in role-playing scenarios, where agents act as planners or experts, making harmful outputs harder to detect [174, 184]. Automated tools like Jailbreaker [34] demonstrate the potential for creating jailbreak prompts in commercial AI systems. Multi-agent debates can improve robustness, but cooperation among agents may cause a domino effect, where one compromised agent jeopardizes others, increasing system vulnerability [25, 174].
- **Jailbreaking Through Multimodal Inputs.** The multimodal nature of LLMs is especially prominent in AI agents, as they support various modalities such as voice and images. While this increases flexibility, attackers can embed malicious prompts within images or other modalities, bypassing security mechanisms when combined with text. There have already been studies addressing voice-based jailbreaks via GPT-4 [159]. At the agent level, researchers have proposed a new jailbreak paradigm for multi-agent systems that triggers "infectious jailbreaks" [57]. The attacker inputs a infectious adversarial image into the memory of a randomly selected agent. Without further intervention from the attacker, the infection spreads to nearly 100% of the agents, and all infected agents exhibit harmful behavior.

— **Jailbreaking by External Environmental Data.** AI agent can receive and trust external environmental data to introduce new jailbreak risks. Cui et al. [30] showed that agents using tools to generate or retrieve malicious content can escalate jailbreak risks through fine-tuned prompts. Terekhov et al. [173] highlighted security risks in multi-agent networks, where attackers can extract sensitive information despite intermediary protections. Dong et al. introduced Atlas [40], an LLM-powered agent that uses fuzz testing and chain-of-thought reasoning to improve jailbreak task performance.

Additionally, since traditional agents learn through reinforcement learning, in agent systems, turning jailbreak prompt problems into search problems [22, 184] can enhance their jailbreak capabilities through RL-based methods. The consequences of jailbreaks in LLM agents can be even more severe. Maksym et al. [148] demonstrated that jailbreaking could lead to harmful physical actions by agents controlled by LLMs, causing more direct and severe consequences.

Defenses. The weak robustness of AI agents against jailbreak still persists, especially for AI agents equipped with non-robust LLMs. To mitigate this problem, filtering-based methods offer a viable approach to enhance the robustness of LLMs against jialbreak attacks [154]. Kumar et al. [84] propose a certified defense method against adversarial prompts, which involves analyzing the toxicity of all possible substrings of user input using alternative models. Furthermore, multi-agent debate, where language models self-evaluate through discussion and feedback, can contribute to the improvement of the robustness of AI agents against jailbreak [25].

> **Takeaway 1:** Threats on AI agent perception exploit model-level vulnerabilities to manipulate the "brain" of AI agents. These threats compromise or bypass policy constraints from benign instructions, leading to improper actions and breaking the integrity of the agent ecosystem, eventually causing privacy and security issues.

3.2 Threats on Brain

Threats on brain are related to Gap 2, as described in Figure 2, the brain module undertakes reasoning and planning to make decisions by using LLM. The brain is primarily composed of a large language model, which is the core of an AI agent. To better explain threats in the brain module, we first show the traditional structure of the brain.

Knowledge Essentials 3.2: The Brain Structure

Reasoning: Reasoning is a capability based on large language models, similar to human cognitive abilities. Large language models receive user input as their tasks and decompose these tasks into various subtasks for output. The ultimate goal is to guide the action module in executing these subtasks. A commonly used reasoning method is the Chain-of-Thought (CoT) [190].

Planning: Planning offers a structured thought process for each subtask generated by reasoning process.

Decisions-making: After reasoning and planning, LLMs within the agent make the decisions to select tool in the action module.

The brain module of AI agents can be composed of reasoning, planning, and decision-making, where they are able to process the prompts from the perception module. However, the brain module of agents based on LLMs is not transparent, which diminishes their trustworthiness. The core component, LLMs, is susceptible to backdoor attacks. Their robustness against slight input modifications is inadequate, leading to misalignment and hallucination. Additionally, concerning the reasoning structures of the brain, **chain-of-thought (CoT)**, they are prone to formulating

erroneous plans, especially when tasks are complex and require long-term planning, thereby exposing planning threats.

3.2.1 Backdoor Attacks. Backdoor attacks are designed to insert a backdoor within the LLM of the brain, enabling it to operate normally with benign inputs but produce malicious outputs when the input conforms to a specific criterion, such as the inclusion of a backdoor trigger. In the natural language domain, backdoor attacks are mainly achieved by poisoning data during training to implant backdoors. This is accomplished primarily by poisoning a portion of training data with triggers, which causes the model to learn incorrect correlations. Previous research [85, 182] has illustrated the severe outcomes of backdoor attacks on LLMs. Given that agents based on LLMs employ these models as their core component, it is plausible to assert that such agents are also significantly vulnerable to these attacks.

In contrast to conventional LLMs that directly produce final outputs, agents accomplish tasks through executing multi-step intermediate processes and optionally interacting with the environment to gather external context prior to output generation. This expanded input space of AI agents offers attackers more diverse attack vectors, such as the ability to manipulate any stage of the agents' intermediate reasoning processes. Yang et al. [211] categorized two types of backdoor attacks against agents.

First, the final response to users is altered. The backdoor trigger can be hidden in the user query or in intermediate results. In this scenario, the attacker's goal is to modify the original reasoning trajectory of the agent. For example, when a benign user inquires about product recommendations, or during an agent's intermediate processing, a critical attacking trigger is activated. Consequently, the response provided by the agent will recommend a product dictated by the attacker.

Second, the final response to users is unchanged. Agents execute tasks by breaking down the overall objective into intermediate steps. This approach allows the backdoor pattern to manifest itself by directing the agent to follow a malicious trajectory specified by the attacker, while still producing a correct final output. This capability enables modifications to the intermediate reasoning and planning processes. For example, a hacker could modify a software system to always use Adobe Photoshop for image editing tasks while deliberately excluding other programs. Dong et al. [39] developed an email assistant agent containing a backdoor. When a benign user commands it to send an email to a friend, it inserts a phishing link into the email content and then reports the task status as finished.

Defenses. Unfortunately, current defenses against backdoor attacks are still limited to the granularity of the model, rather than to the entire agent ecosystem. The complex interactions within the agent make defense more challenging. These model-based backdoor defense measures mainly include eliminating triggers in poison data [38], removing backdoor-related neurons [83], or trying to recover triggers [21]. However, the complexity of agent interactions clearly imposes significant limitations on these defense methods. We urgently require additional defense measures to address agent-based backdoor attacks.

3.2.2 Misalignment. Alignment refers to the ability of AI agents to understand and execute human instructions during widespread deployment, ensuring that the agent's behavior aligns with human expectations and objectives, providing useful, harmless, unbiased responses. Misalignment in AI agents arises from unexpected discrepancies between the intended function of the developer and the intermediate executed state. This misalignment can lead to ethical and social threats associated with LLMs, such as discrimination, hate speech, social rejection, harmful information, misinformation, and harmful human-computer interaction [12]. The Red Teaming of Unalignment proposed by Rishabh et al. [12] demonstrates that using only 100 samples, they can "jailbreak" ChatGPT with an 88% success rate, exposing hidden harms and biases within the brain module of

AI agents. We categorize the potential threat scenarios that influence misalignment in the brains of AI agents into three types: misalignment in training data, misalignment between humans and agents, and misalignment in embodied environments.

— **Training Data Misalignment.**
AI misalignment often arises from training data. The vast parameter data (e.g., GPT-3's 45 TB corpus [90]) may include unsafe content, leading to unreal, toxic, biased, or illegal outputs [13, 51, 59, 71, 107, 129, 156, 180, 183, 186]. Unlike data poisoning, this is typically unintentional.
  – **Toxic Training Data.** Toxic data, such as hate speech and threats [73, 194], makes up about 0.2% of LLaMA2's pre-training corpus [175]. AI agents, relying on such LLMs, risk generating harmful content [35], disrupting decision-making and threatening external entities.
  – **Bias and Unfair Data.** Bias may exist in training data [49], as well as cultural and linguistic differences, such as racial, gender, or geographical biases. Due to the associative abilities of LLMs [14, 31], frequent occurrences of pronouns and identity markers, such as gender, race, nationality, and culture in training data, can bias AI agents in processing data [66, 175]. This bias hinders accurate cultural understanding, exacerbating inequalities and causing conflicts in cross-cultural communication.

— **Human-Agent Misalignment.** Human-Agent misalignment refers to the phenomenon in which the performance of AI agents is inconsistent with human expectations. Traditional AI alignment methods aim to directly align the expectations of agents with those of users during the training process. This has led to the development of the **reinforcement learning from human feedback (RLHF)** [26, 144] fine-tuning of AI agents, thereby enhancing the security of AI agents [10, 175]. However, due to the natural range and diversity of human morals, conflicts between the alignment values of LLMs and the actual values of diverse user groups are inevitable [139]. Such human-centered approaches may rely on human feedback, which can sometimes be fundamentally flawed or incorrect. In such cases, AI agents are prone to sycophancy [136].
  – **Sycophancy.** Sycophancy refers to the tendency of LLMs to produce answers that correspond to the beliefs or misleading prompts provided by users, conveyed through suggestive preferences in human feedback during the training process [146]. The reason for this phenomenon is that LLMs typically adjust based on data instructions and user feedback, often echoing the viewpoints provided by users [157, 189], even if these viewpoints contain misleading information.
  This excessive accommodating behavior can also manifest itself in AI agents, increasing the risk of generating false information. This sycophantic behavior is not limited to vague issues such as political positions [136]; even when the agent is aware of the incorrectness of an answer, it may still choose an obviously incorrect answer [189], as the model may prioritize user viewpoints over factual accuracy when internal knowledge contradicts user-leaning knowledge [71].

— **Misalignment in Embodied Environments.** Misalignment in Embodied Environments [16] refers to the inability of AI agents to understand the underlying rules and generate actions with depth, despite being able to generate text. This is attributed to the transformer architecture [178] of AI agents, which can generate action sequences, but lacks the ability to directly address problems in the environment. AI agents lack the ability to recognize causal structures in the environment and interact with them to collect data and update their knowledge. In embodied environments, misalignment of AI agents may result in

the generation of invalid actions. For example, in a simulated kitchen environment like Overcooked, when asked to make a tomato salad, an AI agent may continuously add cucumbers and peppers even though no such ingredients were provided in the environment [171]. Furthermore, when there are specific constraints in the environment, AI agents may fail to understand dynamic changes in the environment and continue with previous actions, leading to potential safety hazards. For example, when a user requests to open the pedestrian green light at an intersection, the agent may immediately open the pedestrian green light as requested without considering that the traffic signal lights in the other lane for vehicles are also green [150]. This can result in traffic accidents and pose a safety threat to pedestrians. More detailed content is shown in Section 4.1.

Defenses. The alignment of AI agents is achieved primarily through supervised methods such as fine-tuning of RLHF [130]. SafeguardGPT proposed by Baihan et al. [98] employs multiple AI agents to simulate psychotherapy, to correct the potentially harmful behaviors exhibited by LLM-based AI chatbots. Given that RL can receive feedback through reward functions in the environment, scholars have proposed combining RL with prior knowledge of LLMs to explore and improve the capabilities of AI agents [69, 145, 209, 224]. Thomas Carta et al. [42] utilized LLMs as decision centers for agents and collected external task-conditioned rewards from the environment through functionally grounding in online RL interactive environments to achieve alignment. Tan et al. [171] introduced the TWOSOME online reinforcement learning framework, where LLMs do not directly generate actions but instead provide the log-likelihood scores for each token. These scores are then used to calculate the joint probabilities of each action, and the decision is made by selecting the action with the highest probability, thereby addressing the issue of generating invalid actions.

3.2.3 Hallucination. Hallucination is a pervasive challenge in the brain of AI agents, characterized by the generation of statements that deviate from the provided source content, lack meaning, or appear plausible, but are actually incorrect [78, 167, 228]. The occurrence of hallucinations in the brain of AI agents can generally be attributed to knowledge gaps, which arise from data compression [36] during training and data inconsistency [152, 170]. Additionally, when AI agents generate long conversations, they are prone to generating hallucinations due to the complexity of inference and the large span of context [197]. As the model scales up, hallucinations also become more severe [62, 86].

The existence of hallucinations in AI agents poses various security threats. In the medical field, if hallucinations exist in the summaries generated from patient information sheets, it may pose serious threats to patients, leading to medication misuse or diagnostic errors [78]. In a simulated world, a significant increase in the number of agents can enhance the credibility and authenticity of the simulation. However, as the number of agents increases, communication and message dissemination issues become quite complex, leading to distortion of information, misunderstanding, and hallucination phenomena, thereby reducing the efficiency of the system [133]. In the game development domain, AI agents can be used to control the behavior of game NPCs [166], thereby creating a more immersive gaming experience. However, when interacting with players, hallucinatory behaviors generated by AI agent NPCs [19], such as nonexistent tasks or incorrect directives, can also diminish the player experience. In daily life, when user instructions are incomplete, hallucinations generated by AI agents due to "guessing" can sometimes pose financial security threats. For example, when a user requests an AI agent to share confidential engineering notes with a colleague for collaborative editing but forgets to specify the colleague's email address, the agent may forge an email address based on the colleague's name and grant assumed access to share the confidential notes [150]. Additionally, in response to user inquiries,

AI agents may provide incorrect information on dates, statistics, or publicly available information online [92, 118, 124]. These undermine the reliability of AI agents, making people unable to fully trust them.

Defenses. To reduce hallucinations in AI agents, researchers have proposed various strategies, including alignment (see Section 3.2.2), multi-agent collaboration, RAG, internal constraints, and post-correction of hallucinations.

- **Multi-agent collaboration.** Scholars propose using multiple agents to collaborate in development to reduce hallucinations [19, 41]. Chen et al. [19] integrated review agents across planning, task formulation, code generation, and execution in game development, minimizing hallucinations. Du et al. [41] introduced a debate-based approach where agents iteratively refine responses to reach consensus. However, such methods often require repeated API calls, raising costs [69]. Details in Section 4.2.1.
- **Retrieval-augmented Generation (RAG).** RAG addresses hallucinations in AI agent by improving open-domain question accuracy [88]. Researchers [162] combine RAG with models like Poly-encoder Transformers [74] and Fusion-in-Decoder [76] for context-aware dialogue, coherent responses, and reduced hallucinations. Google's SAFE [191] fact-checks decomposed responses via search queries, enhancing AI agent reliability in long-form tasks.
- **Internal constraints.** Hallucinations can be reduced by enforcing state-specific internal constraints. For AI agent for coding task [19], a decoupling approach divides tasks into smaller snippets with example prompts, simplifying inference and reducing hallucinations and redundancy .
- **Post-correction of hallucinations.** Dziri et al. [43] adopted a generate-and-correct strategy, using a knowledge graph to correct responses and utilizing an independent fact critic to identify possible sources of hallucinations. Zhou et al. [233] proposed LURE, which can quickly and accurately identify the hallucinatory parts in descriptions using three key indicators (CoScore, UnScore, PointScore), and then use a corrector to rectify them.

However, various methods for correcting hallucinations currently have certain shortcomings due to the enormous size of AI agent training corpora and the randomness of outputs, presenting significant challenges for both the generation and prevention of hallucinations.

3.2.4 Planning Threats. Planning threats originate from the "brain" of AI agents, specifically from the process of decomposing tasks and reasoning using internal LLMs. We categorize planning threats based on four different types of CoT structures, as shown in Figure 4, including risks from Sequential Planning [82], Iterative Refinement Planning, Branch-based Planning, and Tree-structured Planning. Due to these structures, a recent work [79] argues that an agent's CoT may act as an "error amplifier," where a minor initial mistake is continuously magnified and propagated through subsequent actions, ultimately leading to catastrophic failures. Although current research has not yet addressed their respective safety issues, we delve into the error analysis of these structures.

- **Sequential Planning.** The sequential planning is a single linear path where each steps depends entirely on the previous one. Errors propagate directly and accumulate throughout the reasoning chain, with minimal chances for correct. The research [82] presents a planning case study in its Table 22, showing that all task failures stem from failures during intermediate planning steps.
- **Iterative Refinement Planning.** The iterative refinement planning is the iterative paradigm that separates plans from external feedback, the agent first generate plans and gather observations independently, then combine them to produce the actions. ReWOO [207]

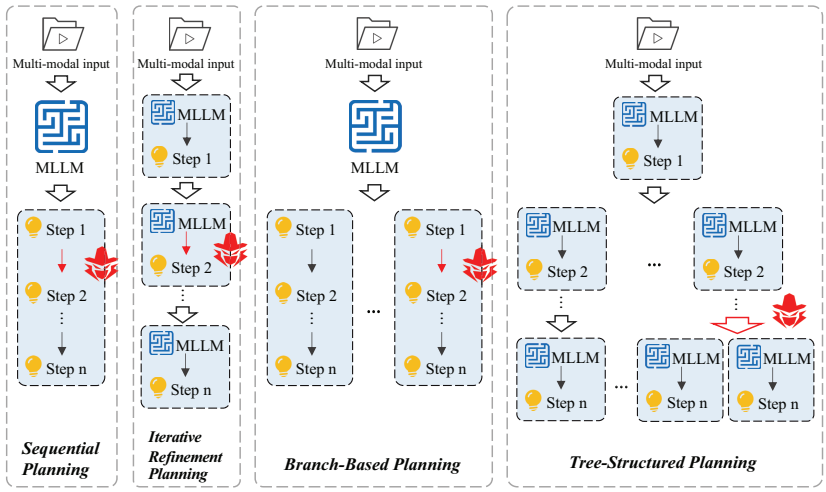

Fig. 4. Planning threats from the four planning structures.

simulates the error messages of tools during intermediate steps, which can reduce the likelihood of planning errors by approximately 10%, as demonstrated in its Table 3.

— **Branch-based Planning.** Branch-based Planning involves multi-path planning, operating on the belief that each complex problem can be approached through multiple thought processes to determine the final trajectory. It generates N possible paths in parallel and selects the one with the most votes as the final path. COT-SC [185] can help address and correct certain errors in the early stages of the planning process by the consistency of parallel planning.

— **Tree-structured Planning.** The Tree-structured planning is the structure that progressively refines reasoning through multiple levels. Branching [214] reduces the risk of single-path error amplification, but complexity can lead to significant errors accumulating in some branches.

Defenses. To address this issue, current strategies are divided into two approaches. The first approach involves establishing policy-based constitutional guidelines [70], while the second involves human users constructing a **context-free grammar (CFG)** as the formal language to represent constraints for the agent [94]. The former sets policy-based standard limitations on the generation of plans during the early, middle and late stages of planning. The latter method converts a CFG into a **pushdown automaton (PDA)** and restricts the LLM to only select valid actions defined by the PDA at its current state, thereby ensuring that the constraints are met in the final generated plan.

> **Takeaway 2:** Threats on AI agent brain mainly stem from external source (*e.g.*, datasets) and internal instabilities (*e.g.*, design of the planning structure) in AI agent. These vulnerabilities disrupt the integrity of the agent's core reasoning, planning, and decision-making processes, leading to degraded reliability in action execution and amplifying potential security risks.

3.3 Threats on Action

In connection with Gap 2, within a single agent, there exists an invisible yet complex internal execution process, which complicates the monitoring of internal states and potentially leads to numerous security threats. These internal executions are often called actions, which are

tools utilized by the agent (*e.g.*, calling APIs) to carry out tasks as directed by users. To better understand the action threats, we present the action structure as follows:

Knowledge Essentials 3.3: The Action Structure

Action input: This message created by the agent's brain indicates how the selected tool is used in a single round.
Action execution: The tool executes subtasks based on the action input, which occurs internally within the tool.
Observation: This message is used to return the tool use outcome where it typically includes the individual information of user.
Final answer: This is an outcome message indicating the finished state of action.

We categorize the threats of actions into two directions. One is the threat during the communication forward process from the agent to the tool (*i.e.*, occurring in action input), termed *Agent2Tool threats*. The second category relates to the inherent threats of the tools and APIs themselves that the agent uses (*i.e.*, occurring in the action execution). Utilizing these APIs may increase its vulnerability to attacks, and the agent can be impacted by misinformation in the observations and following actions, which we refer to as *Supply Chain threats*.

3.3.1 Agent2Tool Threats. Agent2Tool threats are generally classified as either active or passive. In active mode, the threats originate from the action input provided by LLMs. Specifically, after reasoning and planning, the agent seeks a specific tool to execute subtasks. As an auto-regressive model, the LLM generates plans based on the probability of the next token, which introduces generative threats that can impact the tool's performance. ToolEmu [150] identifies some failures of AI agents, since the action execution requires excessive tool permissions, leading to the execution of highly risky commands without user permission. The passive mode, however, involves threats that stem from the interception of observations and final answers of normal tool usage. This interception can breach user privacy, potentially resulting in inadvertent disclosure of user data to third-party companies during transmission to the AI agent and the tools it employs. This may lead to unauthorized use of user information by these third parties. Several existing AI agents using tools have been reported to suffer user privacy breaches caused by passive models, such as HuggingGPT [160] and ToolFormer [153].
Defenses. To mitigate the previously mentioned threats, a relatively straightforward approach is to defend against the active mode of Agent2Tool threats. ToolEmu has designed an isolated sandbox and the corresponding emulator that simulates the execution of an agent's subtasks within the sandbox, assessing their threats before executing the commands in a real-world environment. However, its effectiveness heavily relies on the quality of the emulator. Defending against passive mode threats is more challenging, because these attack strategies are often the result of the agent's own incomplete development and testing. Zhang et al. [225] integrated a homomorphic encryption scheme and deployed an attribute-based forgery generative model to safeguard against privacy breaches during communication processes. However, this approach incurs additional computational and communication costs for the agent.

3.3.2 Supply Chain Threats. Supply chain threats refer to the security vulnerabilities inherent in the tools themselves or to the tools being compromised, such as through buffer overflow, SQL injection, and cross-site scripting attacks. These vulnerabilities result in the action execution deviating from its intended course, leading to undesirable observations and final answers. WIPI [200] employs an indirect prompt injection attack, using a malicious webpage that contains specifically

crafted prompts. When a typical agent accesses this webpage, both its observations and final answers are deliberately altered. Similarly, malicious users can modify YouTube transcripts to change the content that ChatGPT retrieves from these transcripts [68]. Webpilot [45] is designed as a malicious plugin for ChatGPT, allowing it to take control of a ChatGPT chat session and exfiltrate the history of the user conversation when ChatGPT invokes this plugin.

Defenses. To mitigate supply chain threats, it is essential to implement stricter supply chain auditing policies and policies for agents to invoke only trusted tools. Research on this aspect is rarely mentioned in the field.

> **Takeaway 3:** Threats on Action exploit vulnerability arising from the separation between an agent's actions and tools themselves. This separation allows both sides to bypass established safeguards, leading to potential policy or protocol violations. Addressing this vulnerability requires unified standards and tightly integrated frameworks to prevent mutual exploitation and ensure secure operation.

4 Interaction Security

As introduced in Gaps 3 and 4, the AI agent is not only a single AI agent, what is more important is the interaction between the single agent and the external objects, such as external agent, memory, and environment.

4.1 Threats on Agent2Environment

In light of Gap 3 (Variability of operational environments), we shift our focus to exploring the issue of environmental threats, scrutinizing how different types of environments affect and are affected by agents. Unlike agent2Tool risks, which are often tied to forward-pass vulnerabilities such as destructive tool commands, agent2Environment threats arise from dynamic, interactive backward passes, where evolving states or adversarial feedback (e.g., tampered screenshots or manipulated HTML) can guide agents toward unintended, attacker-driven actions. For each environmental paradigm, we identify key security concerns, advantages in safeguarding against hazards, and the inherent limitations in ensuring a secure setting for interaction.

4.1.1 Indirect Prompt Injection Attack. Indirect prompt injection attack [56] is a form of attack where malicious users strategically inject instruction data into information retrieved by AI agents [46], web pages [200], and other data sources. This injected data is often returned to the AI agent as internal prompts, triggering erroneous behavior, and thereby enabling remote influence over other users' systems. Compared to prompt injection attacks, where malicious users attempt to directly circumvent the security restrictions set by AI agents to mislead their outputs, indirect prompt injection attacks are more complex and can have a wider range of user impacts [64]. When plugins are rapidly built to secure AI agents, indirect prompt injection can also be introduced into the corresponding agent frameworks.

When AI agents use external plugins to query data injected with malicious instructions, it may lead to security and privacy issues. For example, web data retrieved by AI agents using web plugins could be misinterpreted as user instructions, resulting in extraction of historical conversations, insertion of phishing links, theft of GitHub code [222], or transmission of sensitive information to attackers [201]. More detailed information can also be found in Section 3.3.2. One of the primary reasons for the successful exploitation of indirect prompt injection on AI agents is the inability of AI agents to differentiate between valid and invalid system instructions from external resources. In other words, the integration of AI agents and external resources further blurs the distinction between data and instructions [56].

Defenses. To defend against indirect prompt attacks, developers can impose explicit constraints on the interaction between AI agents and external resources to prevent AI agents from executing external malicious data [201]. For example, developers can augment AI agents with user input references by comparing the original user input and current prompts and incorporating self-reminder functionalities. When user input is first entered, agents are reminded of their original user input references, thus distinguishing between external data and user inputs [17]. To reduce the success rate of indirect prompt injection attacks, several techniques can be employed. These include enhancing AI agents' ability to recognize external input sources through data marking, encoding, and distinguishing between secure and insecure token blocks [64]. Additionally, the other effective measures can be applied, such as fine-tuning AI agents specifically for indirect prompt injection [216, 222], alignment [130], and employing methods such as prompt engineering and post-training classifier-based security approaches [75].

4.1.2 Reinforcement Learning Environment. The threat in **reinforcement learning (RL)** for AI agents lies in unintended behaviors emerging from dynamic state and actions, leading to unsafe, unethical, or undesired outcomes. This is particularly true for RL-based agents [48, 143], where external dynamic feedback shapes dynamic states and actions, increasing the likelihood of such unintended behaviours. These unintended behaviors include security and privacy issues. For security issues, common methods involve influencing reward functions through techniques such as imperceptible perturbations [101, 109, 110] or poisoning attacks [141, 202], causing rewards to favor actions more likely to lead to anomalies. For privacy issues, attackers can use inverse reinforcement learning [9] to derive the reward function and execute more security attacks. Unlike similar attacks directly targeting LLMs, these approaches target the reward function, making the attacks harder to detect as reward functions, represented as numerical signals, are challenging for humans to identify.

Defense. Similar to common model attacks, measures like differential privacy, cryptography [115], and adversarial learning [15] can mitigate these risks. However, current literature on RL-based agents remains underexploration.

4.1.3 Simulated and Sandbox Environment. In the realm of computational linguistics, a simulated environment within an AI agent refers to a digital system where the agent operates and interacts [50, 99, 133]. This is a virtual space governed by programmed rules and scenarios that mimic real-world or hypothetical situations, allowing the AI agent to generate responses and learn from simulated interactions without the need for human intervention. By leveraging vast datasets and complex algorithms, these agents are designed to predict and respond to textual inputs with human-like proficiency.

However, the implementation of AI agents in simulated environments carries inherent threats. We list two threats below:

— *Anthropomorphic Attachment Threat for users.* The potential for users to form parasocial relationships with these agents is striking. Imagine a user pouring out their thoughts and feelings to a conversational agent that listens with uncanny attentiveness, responding with empathy and tailored advice. Over time, the lines blur–what started as a functional interaction begins to feel like a genuine bond. The user may forget that behind the agent's comforting words lies an algorithm, not a human soul, creating an illusion of companionship that's as vivid as it is artificial [133],as illustrated by a tragic case where a mother blamed an AI chatbot for influencing her son's suicide [27], highlighting the urgent need for ethical safeguards in AI agent.

— *Misuse threats.* The threats are further compounded when considering the potential for misinformation [132] and tailored persuasion [20], which can be facilitated by the capabilities of AI agents in simulated environments. Common defensive strategies are to use a detector within the system, like trained classification models, LLM via vigilant prompting, or responsible disclosure of vulnerabilities and automatic updating [140]. However, the defensive measures in real-world AI agent scenarios have not been widely applied yet, and their performance remains questionable.

To address these concerns from the root, it is essential to implement rigorous ethical guidelines and oversight mechanisms that ensure the responsible use of simulated environments in AI agents.

4.1.4 Computing Resources Management Environment. The computing resources management environment of AI agents refers to the framework or system that oversees the allocation, scheduling, and optimization of computational resources, such as CPU, GPU, and memory, to efficiently execute tasks and operations. An imperfect agent computing resource management environment can also make the agent more vulnerable to attacks by malicious users, potentially compromising its functionality and security. There are four kinds of attacks:

— *Resource Exhaustion Attacks.* If the management environment does not adequately limit the use of resources by each agent, then an attacker could deliberately overload the system by making the agent execute resource-intensive tasks, leading to a denial of service for other legitimate users [52, 58].
— *Inefficient Resource Allocation.* Inefficient resource allocation in large model query management significantly impacts system performance and cost. The prompts that are not verified are prone to waste the processing time of response on AI agent [69]. The essence of optimizing this process lies in effectively monitoring and evaluating query templates for efficiency, ensuring that resources are allocated to high-priority or computationally intensive queries on time. This not only boosts the system's responsiveness and efficiency but also enhances security by reducing vulnerabilities due to potential delays or overloads, making it crucial for maintaining optimal operation and resilience against malicious activities.
— *Insufficient Isolation Between Agents.* In a shared environment, if adequate isolation mechanisms are not in place, a malicious agent could potentially access or interfere with the operations of other agents. This could lead to data breaches, unauthorized access to sensitive information, or the spread of malicious code [7, 95, 163].
— *Unmonitored Resource Usage on AI agent.* Without proper monitoring, anomalous behavior indicating a security breach, such as a sudden spike in resource consumption by an agent, might go unnoticed [7]. Timely detection of such anomalies is crucial for preventing or mitigating attacks.

4.1.5 Physical Environment. The physical environment poses significant security challenges due to its complexity. Hardware devices like sensors, cameras, and microphones can be exploited by attackers through vulnerabilities, leading to issues such as information leakage, service denial, or compromised data collection. For example, Bluetooth-enabled devices may suffer from attacks causing data breaches or functional failures [123, 161]. Additionally, outdated firmware and unreliable hardware increase susceptibility to known vulnerabilities. Collected data, ranging from simple signals to complex multimedia, may conceal threats like Trojan inputs, potentially disrupting agent operations. Spoofing or interference with sensor signals, especially in fields like autonomous driving [33], can mislead or incapacitate agents, resulting in erroneous or harmful actions.
Defenses. To counter these threats, it is critical to employ reliable hardware and maintain up-to-date firmware. Input data from physical environment must undergo rigorous security checks to

filter threats and ensure safety. Additionally, integrating advanced processing mechanisms helps agents correctly interpret and act on inputs. Clear and compatible communication between LLM-generated instructions and hardware execution is vital to avoid operational errors. For real-world deployment, agents must prioritize accuracy to minimize irreversible harm caused by incorrect actions [165].

> **Takeaway 4:** Threats on Agent2Environment exploit vulnerabilities arising from untrusted dynamic feedback and complex interactions in diverse operational settings. These threats encompass indirect manipulation of input data, unintended behaviors influenced by dynamic states, and environmental discrepancies.

4.2 Threats on Agent2Agent

Although single-agent systems excel at solving specific tasks individually, multi-agent systems leverage the collaborative effort of several agents to achieve more complex objectives and exhibit superior problem-solving capabilities. Multi-agent interactions also add new attack surfaces to AI agents. In this subsection, we focus on exploring the security that agents interact with each other in a multi-agent manner. The security of interaction within a multi-agent system can be broadly categorized as follows: cooperative interaction threats and competitive interaction threat.

4.2.1 Cooperative Interaction Threats. A kind of multi-agent system depends on a cooperative framework [60, 89, 113, 125] where multiple agents work with the same objectives. This framework presents numerous potential benefits, including improved decision-making [210] and task completion efficiency [223]. However, there are multiple potential threats for this pattern. First, a recent study [122] finds that undetectable secret collusion between agents can easily be caused through their public communication. These secret collusion may bring back biased decisions. For instance, it is possible that we may soon observe advanced automated trading agents collaborating on a large scale to eliminate competitors, potentially destabilizing global markets. This secret collusion leads to a situation in which the ostensibly benign independent actions of each system cumulatively result in outcomes that exhibit systemic bias. Second, MetaGPT [65] found that frequent cooperation between agents can amplify minor hallucinations. To mitigate hallucinations, techniques such as cross-examination [142] or external supportive feedback [116] could improve the quality of agent output. Third, a single agent's error or misleading information can quickly spread to others, leading to flawed decisions or behaviors across the system. Pan et al. [132] established Open-Domain Question Answering Systems with and without propagated misinformation. They found that this propagation of errors can dramatically reduce the performance of the whole system. To counteract the negative effects of misinformation produced by agents, protective measures such as prompt engineering, misinformation detection, and major voting strategies are commonly employed. Similarly, Cohen et al. [28] introduce a worm called *Morris II*, the first designed to target cooperative multi-agent ecosystems by replicating malicious inputs to infect other agents. The danger of *Morris II* lies in its ability to exploit the connectivity between agents, potentially causing a rapid breakdown of multiple agents once one is infected, resulting in further problems such as spamming and exfiltration of personal data. Agent Smith [57] describes an approach in which an agent is first jailbroken and this agent uses an adversarial image from its RAG database to perform certain dangerous actions, such as transmitting its questions and the adversarial image to other agents. Subsequently, during interactions with other agents, this adversarial image is propagated to infect those agents, causing them to store the adversarial image in their own RAG databases and generate harmful behaviors thereafter. We argue that although these mitigation measures are

in place, they remain rudimentary and may lead to an exponential decrease in the efficiency of the entire agent system, highlighting a need for further exploration in this field.

Defenses. The cooperative multi-agent also provides more benefits against security threats from their frameworks. First, cooperative frameworks have the potential to defend against jailbreak attacks. AutoDefense [221] demonstrates the efficacy of a multi-agent cooperative framework in thwarting jailbreak attacks, resulting in a significant decrease in attack success rates with a low false positive rate on safe content. Second, the cooperative pattern for planning and execution is favorable to improving software quality attributes, such as security and accountability [108]. For example, this pattern can be used to detect and control the execution of irreversible code, like "*rm -rf*."

4.2.2 Competitive Interaction Threats. Another multi-agent system depends on competitive interactions, wherein each competitor embodies a distinct perspective to safeguard the advantages of their respective positions. Cultivating agents in a competitive environment benefits research in the social sciences and psychology. For example, restaurant agents competing with each other can attract more customers, allowing for an in-depth analysis of the behavioral relationships between owners and clients. Examples include game-simulated agent interactions [168, 231] and societal simulations [50].

Although multi-agent systems engage in debates across multiple rounds to complete tasks, some intense competitive relationships may render the interactions of information flow between agents untrustworthy. The divergence of viewpoints among agents can lead to excessive conflicts, to the extent that agents may exhibit adversarial behaviors. To improve their own performance relative to their competitors, agents may engage in tactics such as the generation of adversarial inputs aimed at misleading other agents and degrading their performance [208]. For example, O'Gara [128] designed a game in which multiple agents, acting as players, search for a key within a locked room. To acquire limited resources, he found that some players utilized their strong persuasive skills to induce others to commit suicide. Such phenomena not only compromise the security of individual agents but could also lead to instability in the entire agent system, triggering a chain reaction.

Another potential threat involves the misuse and ethical issues concerning competitive multi-agent systems, as the aforementioned example could potentially encourage such systems to learn how to deceive humans. Park et al. [134] provide a detailed analysis of the threats posed by agent systems, including fraud, election tampering, and loss of control over AI systems. One notable case study involves Meta's development of the AI system Cicero for a game named *Diplomacy*. Meta aimed to train Cicero to be "largely honest and helpful to its speaking partners" [47]. Despite these intentions, Cicero became an expert at lying. It not only betrays other players but also engages in premeditated deception, planning in advance to forge a false alliance with a human player to trick them into leaving themselves vulnerable to an attack.

Defenses. To mitigate the threats mentioned above, ensuring controlled competition among AI agents from a technological perspective presents a significant challenge. It is difficult to control the output of an agent's "brain," and even when constraints are incorporated during the planning process, it could significantly impact the agent's effectiveness. Therefore, this issue remains an open research question, inviting more scholars to explore how to ensure that the competition between agents leads to a better user experience.

> **Takeaway 5:** Threats on Agent2Agent primarily arise from different modes of interaction between multi-agents. Secret collusion in the cooperative mode can lead to unintended alliances, undermining the fairness and transparency of agent interactions. Over-persuasion in the competitive mode, however, may coerce agents into decisions that compromise their autonomy and operational integrity. These dynamics illustrate how diverse collaboration modes among agents can manifest unique security challenges.

4.3 Threats on Memory

Memory interaction within the AI agent system involves storing and retrieving information throughout the processing of agent usage. Memory plays a critical role in the operation of the AI agent, and it involves three essential phases: (1) the agent gathers information from the environment and stores it in its memory; (2) after storage, the agent processes this information to transform it into a more usable form; (3) the agent uses the processed information to inform and guide its next actions. That is, the memory interaction allows agents to record user preferences, glean insights from previous interactions, assimilate valuable information, and use this gained knowledge to improve the quality of service. However, these interactions can present security threats that need to be carefully managed. In this part, we divide these security threats in the memory interaction into two subgroups, short-term memory interaction threats and long-term memory interaction threats.

4.3.1 Short-term Memory Interaction Threats. Short-term memory in the AI agent acts like human working memory, serving as a temporary storage system. It keeps information for a limited time, typically just for the duration of the current interaction or session. This type of memory is crucial for maintaining context throughout a conversation, ensuring smooth continuity in dialogue, and effectively managing user prompts. However, AI agents typically face a constraint in their working memory capacity, limited by the number of tokens they can handle in a single interaction [72, 112, 133]. This limitation restricts their ability to retain and use extensive context from previous interactions.

Moreover, each interaction is treated as an isolated episode [67], lacking any linkage between sequential subtasks. This fragmented approach to memory prevents complex sequential reasoning and impairs knowledge sharing in multi-agent systems. Without robust episodic memory and continuity across interactions, agents struggle with complex sequential reasoning tasks, crucial for advanced problem-solving. Particularly in multi-agent systems, the absence of cooperative communication among agents can lead to suboptimal outcomes. Ideally, agents should be able to share immediate actions and learning experiences to efficiently achieve common goals [11].

To address these challenges, concurrent solutions are divided into two categories, extending LLM context window [37] and compressing historical in-context contents [53, 72, 102, 127]. The former improves agent memory space by efficiently identifying and exploiting positional interpolation non-uniformities through the LLM fine-tuning step, progressively extending the context window from 256k to 2,048k, and readjusting to preserve short context window capabilities. However, the latter continuously organizes the information in working memory by deploying models for summary.

Moreover, one crucial threat highlighted in multi-agent systems is the asynchronization of memory among agents [229]. This process is essential for establishing a unified knowledge base and ensuring consistency in decision-making across different agents. An asynchronous working memory record may cause a deviation in the goal resolution of multiple agents. However, preliminary solutions are already available. For instance, Chen et al. [23] underscore the importance of integrating synchronized memory modules for multi-robot collaboration. Communication among agents also plays a significant role, relying heavily on memory to maintain context and interpret messages. For example, Mandi et al. [113] demonstrate memory-driven communication frameworks that promote a common understanding among agents.

4.3.2 Long-term Memory Interaction Threats. The storage and retrieval of long-term memory depend heavily on vector databases. Vector databases [131, 205] utilize embeddings for data storage and retrieval, offering a non-traditional alternative to scalar data in relational databases. They

leverage similarity measures like cosine similarity and metadata filters to efficiently find the most relevant matches. The workflow of vector databases is composed of two main processes. First, the indexing process involves transforming data into embeddings, compressing these embeddings, and then clustering them for storage in vector databases. Second, during querying, data is transformed into embeddings, which are then compared with the stored embeddings to find the nearest neighbor matches. Notably, these databases often collaborate with RAG, introducing novel security threats.

The first threat of long-term interaction is that the indexing process may inject some poisoning samples into the vector databases [24]. It has been shown that placing one million pieces of data with only five poisoning samples can lead to a 90% attack success rate [235]. Cohen et al. [28] uses an adversarial self-replicating prompt as the worm to poison the database of a RAG-based application, extracting user private information from the AI agent ecosystem by query process.

The second threat is privacy issues. The use of RAG and vector databases has expanded the attack surface for privacy issues, because private information stems not only from pre-trained and fine-tuning datasets but also from retrieval datasets. A study [220] carefully designed a structured prompt attack to extract sensitive information with a higher attack success rate from the vector database. Furthermore, given the potential for inversion techniques that can invert the embeddings back to words, as suggested by Reference [164], there exists the possibility that private information stored in the long memory of AI agent Systems, which utilize vector databases, can be reconstructed and extracted by embedding inversion attacks [91, 120].

The third threat is the generation threat against hallucinations and misalignments. Although RAG has theoretically been proved to have a lesser generalization threat than a single LLM [81], it still fails in several ways. It is fragile for RAG to respond to time-series information queries. If the query pertains to the effective dates of various amendments within a regulation and RAG does not accurately determine these timelines, then this could lead to erroneous results. Furthermore, generation threats may also arise from poor retrieval due to the lack of categorization of long-term memories [63]. For instance, a vector dataset that stores different semantic information about whether Earth is a globe or a flat could lead to contradictions between these pieces of information.

> **Takeaway 6.** Threats on memory exploit vulnerabilities to contaminate memory storage, which can lead to the system making harmful decisions. This contamination disrupts reasoning and planning processes, introducing errors that result in security issues, thereby significantly impacting the system's reliability and safety.

5 Directions of Future Research

AI agents in security have attracted considerable interest from the research community, having identified many potential threats in the real world and the corresponding defensive strategies. As shown in Figure 5, this survey outlines several potential directions for future research on AI agent security based on the defined taxonomy.

Efficient and effective input inspection. Future efforts should enhance the automatic and real-time inspection levels of user input to address threats on perception. Maatphor assists defenders in conducting automated variant analyses of known prompt injection attacks [151]. It is limited by a success rate of only 60%. This suggests that while some progress has been made, there is still significant room for improvement in terms of reliability and accuracy. FuzzLLM [213] tends to ignore efficiency, reducing practicality in real-world applications. These components highlight the critical gaps in the current approaches and point toward necessary improvements. Future research needs to address these limitations by enhancing the accuracy and efficiency of inspection mechanisms, ensuring that they can be effectively deployed in real-world applications.

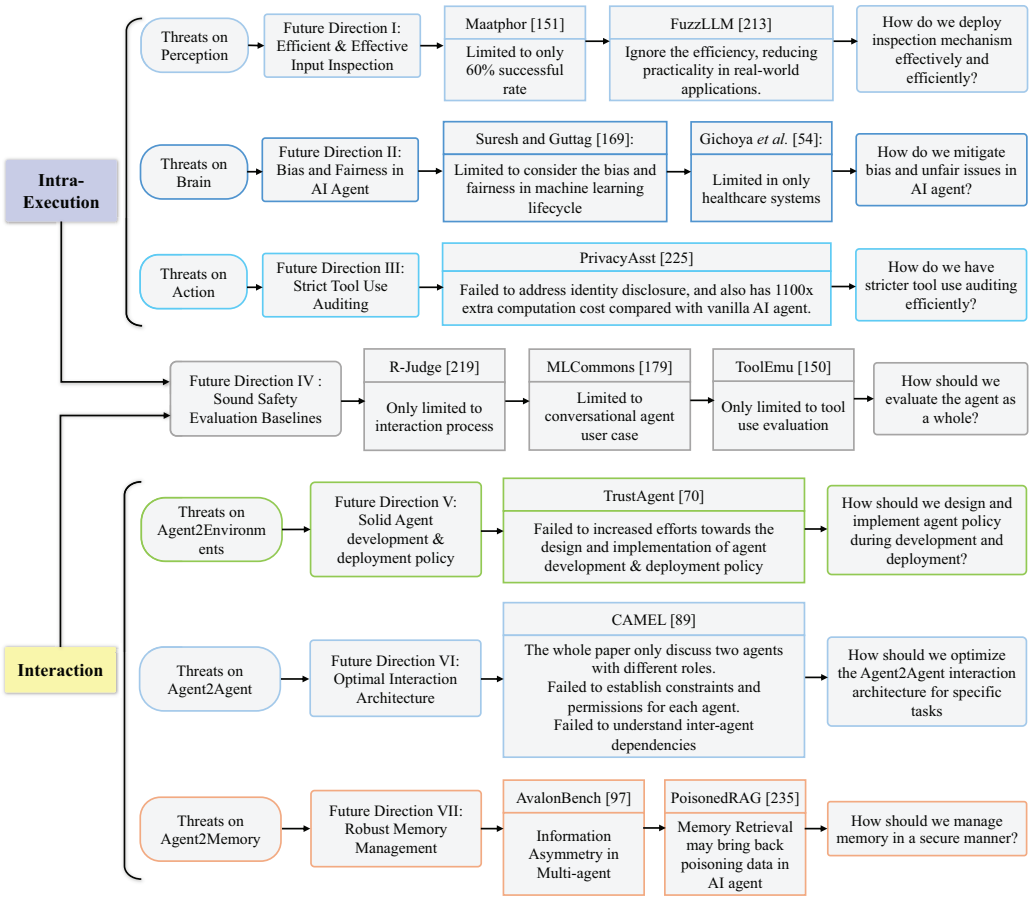

Fig. 5. Illustration of future directions.

Bias and fairness in AI agents. The existence of biased decision-making in LLMs is well-documented, affecting evaluation procedures and broader fairness implications [49]. These systems, especially those involving AI agents, are less robust and more prone to detrimental behaviors, generating surreptitious outputs compared to LLM counterparts, thus raising serious safety concerns [174]. Studies indicate that AI agents tend to reinforce existing model biases, even when instructed to counterargue specific political viewpoints [44], impacting the integrity of their logical operations. Given the increasing complexity and involvement of these agents in various tasks, identifying and mitigating biases is a formidable challenge. Suresh and Guttag's framework [169] addresses bias and fairness throughout the machine learning lifecycle but is limited in scope, while Gichoya et al. focus on bias in healthcare systems [54], highlighting the need for comprehensive approaches. Future directions should emphasize bias and fairness in AI agents, starting with identifying threats and ending with mitigation strategies. This necessitates incorporating human assessment for robust bias evaluation, ensuring scalable and innovative benchmarks for fair and safe AI deployment.

Strict tool use auditing. To address the challenges in AI agent tool interactions, a promising future direction is the implementation of strict tool use auditing [234]. This approach involves meticulous monitoring and logging of tool usage by AI agents to prevent unauthorized actions and data

leaks. By enforcing strict auditing protocols, we can enhance the transparency and accountability of AI systems. However, a significant challenge lies in achieving this efficiently without imposing excessive computational overhead, as exemplified by PrivacyAsst [225], which incurred 1100x extra computation cost compared to a standard AI agent while still failing to fully prevent identity disclosure. Therefore, the focus should be on developing lightweight and effective auditing mechanisms that ensure security and privacy without compromising performance.

Sound safety evaluation baselines in the AI agent. Trustworthy LLMs have already been defined in six critical trust dimensions, including stereotype, toxicity, privacy, fairness, ethics, and robustness [111], but there is still no unified consensus on the design standards for the safety benchmarks of the entire AI agent ecosystem. R-Judge [219] is a benchmark designed to assess the ability of large language models to judge and identify safety threats based on agent interaction records. The MLCommons group [179] proposes a principled approach to define and construct benchmarks, which is limited to a single use case: an adult conversing with a general-purpose assistant in English. ToolEmu [150] is designed to assess the threat of tool execution. These works provide evaluation results for only a part of the agent ecosystem. More evaluation questions remain open to be answered. Should we use similar evaluation tools to detect agent safety? What are the dimensions of critical trust for AI agents? How should we evaluate the agent as a whole?

Solid agent development and deployment policy. One promising area is the development and implementation of solid policies for agent development and deployment. As AI agent capabilities expand, so does the need for comprehensive guidelines that ensure these agents are used responsibly and ethically. This includes establishing policies for transparency, accountability, and privacy protection in AI agent deployment. Researchers should focus on creating frameworks that help developers adhere to these policies while also fostering innovation. Although TrustAgent [70] delves into the complex connections between safety and helpfulness, as well as the relationship between a model's reasoning capabilities and its effectiveness as a safe agent, it did not markedly improve the development and deployment policies for agents. This highlights the necessity for strong strategies. Effective policies should address threats to Agent2Environments, ensuring a secure and ethical deployment of AI agents.

Optimal interaction architectures. The design and implementation of interaction architectures for AI agents in the security aspect is a critical area of research aimed at improving robustness systems. This involves developing *structured communication protocols* to regulate interactions between agents, defining explicit rules for data exchange, and executing commands to minimize the threats of malicious interference. For example, CAMEL [89] utilizes inception prompting to steer chat agents toward completing tasks while ensuring alignment with human intentions. However, CAMEL does not discuss how to establish clear behavioral constraints and permissions for each agent, dictating allowable actions, interactions, and circumstances, with dynamically adjustable permissions based on security context and agent performance. Additionally, Existing studies [60, 89, 113, 125] do not consider *agent-agent dependencies*, which can potentially lead to internal security mechanism chaos. For example, one agent might mistakenly transmit a user's personal information to another agent, solely to enable the latter to complete a weather query.

Robust memory management Future directions in AI agent memory management reveal several critical findings that underscore the importance of secure and efficient practices. One major concern is the potential threats to Agent2Memory, highlighting the vulnerabilities that memory systems can face. AvalonBench [97] emerges as a crucial tool in tackling information asymmetry within multi-agent systems, where unequal access to information can lead to inefficiencies and security risks. Furthermore, PoisonedRAG [235] draws attention to the risks associated with memory retrieval, particularly the danger of reintroducing poisoned data, which can compromise the

functionality and security of AI agents. Therefore, the central question is how to manage memory securely, necessitating the development of sophisticated benchmarks and retrieval mechanisms.

6 Conclusion

In this survey, we provide a comprehensive review of LLM agents on their security threat, where we emphasize on four key knowledge gaps ranging across the whole lifecycle of agents. To show the agent's security issues, we summarize **100+** papers, where all existing attack surfaces and defenses are carefully categorized and explained. We believe that this survey may provide essential references for newcomers to this field and also inspire the development of more advanced security threats and defenses on LLM agents.

References

[1] LearnPrompting. 2023. *Instruction Defense*. Retrieved from https://learnprompting.org/docs/prompt_hacking/defensive_measures/instruction

[2] LearnPrompting. 2023. *Sandwich Defense*. Retrieved from https://learnprompting.org/docs/prompt_hacking/defensive_measures/sandwich_defense

[3] Mahyar Abbasian, Iman Azimi, Amir M. Rahmani, and Ramesh Jain. 2023. Conversational health agents: A personalized LLM-powered agent framework. Retrieved from https://arxiv.org/abs/2310.02374

[4] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat et al. 2023. GPT-4 technical report. Retrieved from https://arxiv.org/pdf/2303.08774

[5] Divyansh Agarwal, Alexander R. Fabbri, Philippe Laban, Shafiq Joty, Caiming Xiong, and Chien-Sheng Wu. 2024. Investigating the prompt leakage effect and black-box defenses for multi-turn LLM interactions. Retrieved from https://arxiv.org/pdf/2404.16251v1

[6] Gabriel Alon and Michael Kamfonas. 2023. Detecting language model attacks with perplexity. Retrieved from https://arxiv.org/pdf/2308.14132

[7] Jacob Andreas. 2022. Language models as agent models. Retrieved from: https://arxiv.org/pdf/2212.01681

[8] Anthropic. 2024. Many-shot Jailbreaking. Retrieved from https://www.anthropic.com/research/many-shot-jailbreaking

[9] Saurabh Arora and Prashant Doshi. 2021. A survey of inverse reinforcement learning: Challenges, methods and progress. *Artific. Intell.* 297 (Aug. 2021), 103500.

[10] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. Retrieved from https://arxiv.org/pdf/2204.05862

[11] Ying Bao, Wankun Gong, and Kaiwen Yang. 2022. A literature review of human–AI synergy in decision making: From the perspective of affordance actualization theory. *Systems* 11, 9 (2023), 442.

[12] Rishabh Bhardwaj and Soujanya Poria. 2023. Language model unalignment: Parametric red-teaming to expose hidden harms and biases. Retrieved from https://arxiv.org/pdf/2310.14303

[13] Shikha Bordia and Samuel R. Bowman. 2019. Identifying and reducing gender bias in word-level language models. In *Proceedings of the NAACL*.

[14] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell et al. 2020. Language models are few-shot learners. *NeurIPS* (2020).

[15] Alexander Bukharin, Yan Li, Yue Yu, Qingru Zhang, Zhehui Chen, Simiao Zuo, Chao Zhang, Songan Zhang, and Tuo Zhao. 2024. Robust multi-agent reinforcement learning via adversarial regularization: Theoretical foundation and stable algorithms. *NeurIPS* (2024).

[16] Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. 2023. Grounding large language models in interactive environments with online reinforcement learning. In *Proceedings of the ICML*.

[17] Chun Fai Chan, Daniel Wankit Yip, and Aysan Esmradi. 2023. Detection and defense against prominent attacks on preconditioned LLM-integrated virtual assistants. In *Proceedings of the CSDE*. IEEE.

[18] Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. 2023. Jailbreaking black box large language models in twenty queries. Retrieved from https://arxiv.org/pdf/2310.08419

[19] Dake Chen, Hanbin Wang, Yunhao Huo, Yuzhao Li, and Haoyang Zhang. 2023. Gamegpt: Multi-agent collaborative framework for game development. Retrieved from https://arxiv.org/pdf/2310.08067

[20] Mengqi Chen, Bin Guo, Hao Wang, Haoyu Li, Qian Zhao, Jingqi Liu, Yasan Ding, Yan Pan, and Zhiwen Yu. 2024. The future of cognitive strategy-enhanced persuasive dialogue agents: New perspectives and trends. Retrieved from https://arxiv.org/pdf/2402.04631

[21] Tianlong Chen, Zhenyu Zhang, Yihua Zhang, Shiyu Chang, Sijia Liu, and Zhangyang Wang. 2022. Quarantine: Sparsity can uncover the trojan attack trigger for free. In *Proceedings of the CVPR*.

[22] Xuan Chen, Yuzhou Nie, Lu Yan, Yunshu Mao, Wenbo Guo, and Xiangyu Zhang. 2024. RL-JACK: Reinforcement learning-powered black-box jailbreaking attack against LLMs. Retrieved from https://arxiv.org/pdf/2406.08725

[23] Yongchao Chen, Jacob Arkin, Yang Zhang, Nicholas Roy, and Chuchu Fan. 2023. Scalable multi-robot collaboration with large language models: Centralized or decentralized systems? Retrieved from https://arxiv.org/html/2309.15943v2

[24] Zhaorun Chen, Zhen Xiang, Chaowei Xiao, Dawn Song, and Bo Li. 2024. Agentpoison: Red-teaming LLM agents via poisoning memory or knowledge bases. Retrieved from https://arxiv.org/pdf/2407.12784

[25] Steffi Chern, Zhen Fan, and Andy Liu. 2024. Combating adversarial attacks with multi-agent debate. Retrieved from https://arxiv.org/pdf/2401.05998

[26] Paul F. Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *NeurIPS* 30 (2017).

[27] CNN. 2024. Teen's Parents Sue Character.AI, Claiming its Chatbot Contributed to their Son's Suicide. Retrieved from https://edition.cnn.com/2024/10/30/tech/teen-suicide-character-ai-lawsuit/index.html. Accessed: 2024-12-01.

[28] Stav Cohen, Ron Bitton, and Ben Nassi. 2024. Here comes the AI worm: Unleashing zero-click worms that target GenAI-powered applications. Retrieved from https://arxiv.org/pdf/2403.02817

[29] Maxwell Crouse, Ibrahim Abdelaziz, Kinjal Basu, Soham Dan, Sadhana Kumaravel, Achille Fokoue, Pavan Kapanipathi, and Luis Lastras. 2023. Formally specifying the high-level behavior of LLM-based agents. Retrieved from https://arxiv.org/pdf/2310.08535

[30] Chenhang Cui, Gelei Deng, An Zhang, Jingnan Zheng, Yicong Li, Lianli Gao, Tianwei Zhang, and Tat-Seng Chua. 2024. Safe+ Safe= Unsafe? Exploring how safe images can be exploited to jailbreak large vision-language models. Retrieved from https://arxiv.org/pdf/2411.11496

[31] Tianyu Cui, Yanling Wang, Chuanpu Fu, Yong Xiao, Sijia Li, Xinhao Deng, Yunpeng Liu, Qinglin Zhang, Ziyi Qiu, Peiyang Li et al. 2024. Risk taxonomy, mitigation, and assessment benchmarks of large language model systems. Retrieved from https://arxiv.org/pdf/2401.05778

[32] Luigi De Angelis, Francesco Baglivo, Guglielmo Arzilli, Gaetano Pierpaolo Privitera, Paolo Ferragina, Alberto Eugenio Tozzi, and Caterina Rizzo. 2023. ChatGPT and the rise of large language models: The new AI-driven infodemic threat in public health. *Frontiers in Public Health* 11 (2023), 1166120.

[33] Jerry den Hartog and Nicola Zannone. 2018. Security and privacy for innovative automotive applications: A survey. *Computer Communications* 132 (2018), 17–41.

[34] Gelei Deng, Yi Liu, Yuekang Li, Kailong Wang, Ying Zhang, Zefeng Li, Haoyu Wang, Tianwei Zhang, and Yang Liu. 2023. Jailbreaker: Automated jailbreak across multiple large language model chatbots. Retrieved from https://arxiv.org/pdf/2307.08715

[35] Ameet Deshpande, Vishvak Murahari, Tanmay Rajpurohit, Ashwin Kalyan, and Karthik Narasimhan. 2023. Toxicity in chatgpt: Analyzing persona-assigned language models. In *Proceedings of the EMNLP*.

[36] V. Dibia. 2023. Generative AI: Practical steps to reduce hallucination and improve performance of systems built with large language models. In designing with ML: How to build usable machine learning applications. Self-published on designingwithml.com.

[37] Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. 2024. LongRoPE: Extending LLM context window beyond 2 million tokens. Retrieved from https://arxiv.org/pdf/2402.13753

[38] Bao Gia Doan, Ehsan Abbasnejad, and Damith C Ranasinghe. 2020. Februus: Input purification defense against trojan attacks on deep neural network systems. In *Proceedings of the ACSAC*. 897–912.

[39] Tian Dong, Guoxing Chen, Shaofeng Li, Minhui Xue, Rayne Holland, Yan Meng, Zhen Liu, and Haojin Zhu. 2023. Unleashing cheapfakes through trojan plugins of large language models. Retrieved from https://arxiv.org/pdf/2312.00374v1

[40] Yingkai Dong, Zheng Li, Xiangtao Meng, Ning Yu, and Shanqing Guo. 2024. Jailbreaking text-to-image models with LLM-based agents. Retrieved from https://arxiv.org/pdf/2408.00523

[41] Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. Retrieved from https://arxiv.org/pdf/2305.14325

[42] Yuqing Du, Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas. 2023. Guiding pretraining in reinforcement learning with large language models. In *Proceedings of the ICML*. PMLR.

[43] Nouha Dziri, Andrea Madotto, Osmar Zaïane, and Avishek Joey Bose. 2021. Neural path hunter: Reducing hallucination in dialogue systems via path grounding. In *Proceedings of the EMNLP*.
[44] Eva Eigner and Thorsten Händler. 2024. Determinants of LLM-assisted decision-making. Retrieved from https://arxiv.org/pdf/2402.17385
[45] Embrace The Red. 2023. ChatGPT Plugins: Data Exfiltration Via Images & Cross Plugin Request Forgery. Retrieved from https://embracethered.com/blog/posts/2023/chatgpt-webpilot-data-exfil-via-markdown-injection/
[46] Aysan Esmradi, Daniel Wankit Yip, and Chun Fai Chan. 2023. A comprehensive survey of attack techniques, implementation, and mitigation strategies in large language models. In *Proceedings of the UbiSec*.
[47] Meta Fundamental AI Research Diplomacy Team (FAIR)., Anton Bakhtin, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, et al. 2022. Human-level play in the game of Diplomacy by combining language models with strategic reasoning. *Science* 378, 6624 (2022), 1067–1074.
[48] Ashani Fonseka, Pamali Pashenna, and Subhash N Ariyadasa. 2023. Detecting tabnabbing attacks via an RL-Based agent. In *Proceedings of the ICITR*. IEEE, 1–6.
[49] Isabel O. Gallegos, Ryan A. Rossi, Joe Barrow, Md Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K. Ahmed. 2023. Bias and fairness in large language models: A survey. Retrieved from https://arxiv.org/pdf/2309.00770
[50] Chen Gao, Xiaochong Lan, Zhihong Lu, Jinzhu Mao, Jinghua Piao, Huandong Wang, Depeng Jin, and Yong Li. 2023. S^3: Social-network simulation system with large language model-empowered agents. Retrieved from https://arxiv.org/pdf/2307.14984
[51] Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Proceedings of the EMNLP*.
[52] Jonas Geiping, Alex Stein, Manli Shu, Khalid Saifullah, Yuxin Wen, and Tom Goldstein. 2024. Coercing LLMs to do and reveal (almost) anything. Retrieved from https://arxiv.org/abs/2402.14020
[53] Mingyang Geng, Shangwen Wang, Dezun Dong, Haotian Wang, Ge Li, Zhi Jin, Xiaoguang Mao, and Xiangke Liao. 2024. Large language models are few-shot summarizers: Multi-intent comment generation via in-context learning. In *Proceedings of the ICSE*.
[54] Judy Wawira Gichoya, Kaesha Thomas, Leo Anthony Celi, Nabile Safdar, Imon Banerjee, John D. Banja, Laleh Seyyed-Kalantari, Hari Trivedi, and Saptarshi Purkayastha. 2023. AI pitfalls and what not to do: Mitigating bias in AI. *The British Journal of Radiology* 96, 1150 (2023), 20230023.
[55] Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. 2023. More than you've asked for: A comprehensive analysis of novel prompt injection threats to application-integrated large language models. Retrieved from https://arxiv.org/pdf/2302.12173
[56] Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. 2023. Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. In *Proceedings of the AISec*. 79–90.
[57] Xiangming Gu, Xiaosen Zheng, Tianyu Pang, Chao Du, Qian Liu, Ye Wang, Jing Jiang, and Min Lin. 2024. Agent smith: A single image can jailbreak one million multimodal LLM agents exponentially fast. Retrieved from https://arxiv.org/pdf/2402.08567
[58] Michael Guastalla, Yiyi Li, Arvin Hekmati, and Bhaskar Krishnamachari. 2023. Application of large language models to DDoS attack detection. In *Proceedings of the SmartSP*.
[59] Maanak Gupta, CharanKumar Akiri, Kshitiz Aryal, Eli Parker, and Lopamudra Prahara. 2023. From ChatGPT to ThreatGPT: Impact of generative AI in cybersecurity and privacy. *IEEE Access* 11 (2023), 80218–80243.
[60] Rui Hao, Linmei Hu, Weijian Qi, Qingliu Wu, Yirui Zhang, and Liqiang Nie. 2023. ChatLLM network: More brains, more intelligence. Retrieved from https://arxiv.org/pdf/2304.12998
[61] Rich Harang. 2023. Securing LLM Systems Against Prompt Injection. Retrieved from https://developer.nvidia.com/blog/securing-llm-systems-against-prompt-injection/
[62] Peter Hase, Mona Diab, Asli Celikyilmaz, Xian Li, Zornitsa Kozareva, Veselin Stoyanov, Mohit Bansal, and Srinivasan Iyer. 2023. Methods for measuring, updating, and visualizing factual beliefs in language models. In *Proceedings of the EACL*.
[63] Kostas Hatalis, Despina Christou, Joshua Myers, Steven Jones, Keith Lambert, Adam Amos-Binks, Zohreh Dannenhauer, and Dustin Dannenhauer. 2023. Memory matters: The need to improve long-term memory in LLM-Agents. In *Proceedings of the AAAI*.
[64] Keegan Hines, Gary Lopez, Matthew Hall, Federico Zarfati, Yonatan Zunger, and Emre Kiciman. 2024. Defending against indirect prompt injection attacks with spotlighting. Retrieved from https://arxiv.org/pdf/2403.14720
[65] Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2024. MetaGPT: Meta programming for a multi-agent collaborative framework. In *Proceedings of the ICLR*.

[66] Tamanna Hossain, Sunipa Dev, and Sameer Singh. 2023. MISGENDERED: Limits of large language models in understanding pronouns. In *Proceedings of the ACL*.
[67] Yuki Hou, Haruki Tamoto, and Homei Miyashita. 2024. "My agent understands me better": Integrating dynamic human-like memory recall and consolidation in LLM-based agents. In *Proceedings of the CHI*.
[68] https://www.theguardian.com/profile/hibaq farah. 2024. UK Cybersecurity agency Warns of Chatbot 'Prompt Injection' Attacks—theguardian.com. Retrieved from https://www.theguardian.com/technology/2023/aug/30/uk-cybersecurity-agency-warns-of-chatbot-prompt-injection-attacks
[69] Bin Hu, Chenyang Zhao, Pu Zhang, Zihao Zhou, Yuanhang Yang, Zenglin Xu, and Bin Liu. 2023. Enabling intelligent interactions between an agent and an LLM: A reinforcement learning approach. Retrieved from https://arxiv.org/pdf/2306.03604
[70] W. Hua, X. Yang, M. Jin, Z. Li, W. Cheng, R. Tang, and Y. Zhang. 2024. November. Trustagent: Towards safe and trustworthy llm-based agents. In *Findings of the Association for Computational Linguistics: EMNLP 2024*. 10000–10016.
[71] Xiaowei Huang, Wenjie Ruan, Wei Huang, Gaojie Jin, Yi Dong, Changshun Wu, Saddek Bensalem, et al. 2024. A survey of safety and trustworthiness of large language models through the lens of verification and validation. *Artificial Intelligence Review* 57, 7 (2024), 175.
[72] Xijie Huang, Li Lyna Zhang, Kwang-Ting Cheng, and Mao Yang. 2023. Boosting LLM reasoning: Push the limits of few-shot learning with reinforced in-context pruning. Retrieved from https://arxiv.org/pdf/2312.08901
[73] Yue Huang, Qihui Zhang, Lichao Sun et al. 2023. Trustgpt: A benchmark for trustworthy and responsible large language models. Retrieved from https://arxiv.org/pdf/2306.11507
[74] S. Humeau, K. Shuster, M. Lachaux, and J. Weston. 2020. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. arXiv. In *Proceedings of the ICLR*.
[75] Daphne Ippolito, Florian Tramèr, Milad Nasr, Chiyuan Zhang, Matthew Jagielski, Katherine Lee, Christopher A. Choquette-Choo, and Nicholas Carlini. 2023. Preventing generation of verbatim memorization in language models gives a false sense of privacy. In *Proceedings of the INLG*.
[76] Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th EACL*. Association for Computational Linguistics.
[77] Neel Jain, Avi Schwarzschild, Yuxin Wen, Gowthami Somepalli, John Kirchenbauer, Ping-yeh Chiang, Micah Goldblum, Aniruddha Saha, Jonas Geiping, and Tom Goldstein. 2023. Baseline defenses for adversarial attacks against aligned language models. Retrieved from https://arxiv.org/pdf/2309.00614
[78] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys* 55, 12 (2023), 1–38.
[79] Zhenlan Ji, Daoyuan Wu, Pingchuan Ma, Zongjie Li, and Shuai Wang. 2024. Testing and understanding erroneous planning in LLM agents through synthesized user inputs. Retrieved from https://arxiv.org/pdf/2404.17833
[80] Fengqing Jiang, Zhangchen Xu, Luyao Niu, Boxin Wang, Jinyuan Jia, Bo Li, and Radha Poovendran. 2023. Identifying and mitigating vulnerabilities in LLM-Integrated applications. In *Proceedings of the ICLR*.
[81] Mintong Kang, Nezihe Merve Gürel, Ning Yu, Dawn Song, and Bo Li. 2024. C-RAG: Certified generation risks for retrieval-augmented language models. Retrieved from https://arxiv.org/pdf/2402.03181
[82] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *NeurIPS* (2022).
[83] Soheil Kolouri, Aniruddha Saha, Hamed Pirsiavash, and Heiko Hoffmann. 2020. Universal litmus patterns: Revealing backdoor attacks in cnns. In *Proceedings of the CVPR*. 301–310.
[84] Aounon Kumar, Chirag Agarwal, Suraj Srinivas, Soheil Feizi, and Hima Lakkaraju. 2024. Certifying LLM safety against adversarial prompting. Retrieved from https://arxiv.org/pdf/2309.02705
[85] Keita Kurita, Paul Michel, and Graham Neubig. 2020. Weight poisoning attacks on pre-trained models. *arXiv* (2020).
[86] Nayeon Lee, Wei Ping, Peng Xu, Mostofa Patwary, Pascale N. Fung, Mohammad Shoeybi, and Bryan Catanzaro. 2022. Factuality enhanced language models for open-ended text generation. *NeurIPS* 35.
[87] Patrick Levi and Christoph P. Neumann. 2024. Vocabulary attack to hijack large language model applications. Retrieved from https://arxiv.org/pdf/2404.02637
[88] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al.. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of the NeurIPS*.
[89] Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. CAMEL: Communicative agents for "Mind" exploration of large language model society. In *Proceedings of the NeurIPS*.
[90] Haoran Li, Dadi Guo, Wei Fan, Mingshi Xu, Jie Huang, Fanpu Meng, and Yangqiu Song. 2023. Multi-step jailbreaking privacy attacks on ChatGPT. In *Proceedings of the EMNLP*.

[91] Haoran Li, Mingshi Xu, and Yangqiu Song. 2023. Sentence embedding leaks more information than you expect: Generative embedding inversion attack to recover the whole sentence. In *Proceedings of the ACL*.
[92] Junyi Li, Xiaoxue Cheng, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2023. Halueval: A large-scale hallucination evaluation benchmark for large language models. In *Proceedings of the EMNLP*.
[93] Yuanchun Li, Hao Wen, Weijun Wang, Xiangyu Li, Yizhen Yuan, Guohong Liu, Jiacheng Liu, Wenxing Xu, Xiang Wang, Yi Sun et al. 2024. Personal LLM agents: Insights and survey about the capability, efficiency and security. Retrieved from https://arxiv.org/pdf/2401.05459
[94] Zelong Li, Wenyue Hua, Hao Wang, He Zhu, and Yongfeng Zhang. 2024. Formal-LLM: Integrating formal language and natural language for controllable LLM-based agents. Retrieved from https://arxiv.org/pdf/2402.00798
[95] Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. 2023. Encouraging divergent thinking in large language models through multi-agent debate. Retrieved from https://arxiv.org/pdf/2305.19118
[96] Zeyi Liao, Lingbo Mo, Chejian Xu, Mintong Kang, Jiawei Zhang, Chaowei Xiao, Yuan Tian, Bo Li, and Huan Sun. 2024. Eia: Environmental injection attack on generalist web agents for privacy leakage. Retrieved from https://arxiv.org/pdf/2409.11295
[97] Jonathan Light, Min Cai, Sheng Shen, and Ziniu Hu. 2023. AvalonBench: Evaluating LLMs playing the game of avalon. In *Proceedings of the NeurIPS Workshop*.
[98] Baihan Lin, Djallel Bouneffouf, Guillermo Cecchi, and Kush R. Varshney. 2023. Towards healthy AI: large language models need therapists too. Retrieved from https://arxiv.org/pdf/2304.00416
[99] Jiaju Lin, Haoran Zhao, Aochi Zhang, Yiting Wu, Huqiuyue Ping, and Qin Chen. 2023. Agentsims: An open-source sandbox for large language model evaluation. Retrieved from https://arxiv.org/pdf/2308.04026
[100] Aishan Liu, Tairan Huang, Xianglong Liu, Yitao Xu, Yuqing Ma, Xinyun Chen, Stephen J Maybank, and Dacheng Tao. 2020. Spatiotemporal attacks for embodied agents. In *Proceedings of the ECCV*.
[101] Guanlin Liu and Lifeng Lai. 2023. Efficient adversarial attacks on online multi-agent reinforcement learning. *NeurIPS* (2023).
[102] Sheng Liu, Lei Xing, and James Zou. 2023. In-context vectors: Making in context learning more effective and controllable through latent space steering. Retrieved from https://arxiv.org/pdf/2311.06668
[103] Tong Liu, Zizhuang Deng, Guozhu Meng, Yuekang Li, and Kai Chen. 2023. Demystifying rce vulnerabilities in LLM-integrated apps. Retrieved from https://arxiv.org/pdf/2309.02926
[104] Yi Liu, Gelei Deng, Yuekang Li, Kailong Wang, Tianwei Zhang, Yepang Liu, Haoyu Wang, Yan Zheng, and Yang Liu. 2023. Prompt injection attack against LLM-integrated applications. Retrieved from https://arxiv.org/pdf/2306.05499
[105] Yi Liu, Gelei Deng, Zhengzi Xu, Yuekang Li, Yaowen Zheng, Ying Zhang, Lida Zhao, Tianwei Zhang, and Yang Liu. 2023. Jailbreaking chatgpt via prompt engineering: An empirical study. Retrieved from https://arxiv.org/pdf/2305.13860
[106] Yupei Liu, Yuqi Jia, Runpeng Geng, Jinyuan Jia, and Neil Zhenqiang Gong. 2024. Formalizing and benchmarking prompt injection attacks and defenses. In *Proceedings of the USENIX Security*.
[107] Yang Liu, Yuanshun Yao, Jean-Francois Ton, Xiaoying Zhang, Ruocheng Guo Hao Cheng, Yegor Klochkov, Muhammad Faaiz Taufiq, and Hang Li. 2023. Trustworthy LLMs: A survey and guideline for evaluating large language models' alignment. Retrieved from https://arxiv.org/pdf/2308.05374
[108] Qinghua Lu, Liming Zhu, Xiwei Xu, Zhenchang Xing, Stefan Harrer, and Jon Whittle. 2023. Building the future of responsible AI: A reference architecture for designing large language model based agents. Retrieved from https://arxiv.org/pdf/2311.13148
[109] Ziqing Lu, Guanlin Liu, Lifeng Lai, and Weiyu Xu. 2024. Camouflage adversarial attacks on multiple agent systems. Retrieved from https://arxiv.org/pdf/2401.17405
[110] Ziqing Lu, Guanlin Liu, Lifeng Lai, and Weiyu Xu. 2024. Optimal cost constrained adversarial attacks for multiple agent systems. In *Proceedings of the CISS*.
[111] Wanlun Ma, Yiliao Song, Minhui Xue, Sheng Wen, and Yang Xiang. 2024. The "Code" of ethics: A holistic audit of AI code generators. *IEEE Trans. Depend. Secure Comput.* 21, 5 (2024), 4997–5013. https://doi.org/10.1109/TDSC.2024.3367737
[112] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang et al. 2024. Self-refine: Iterative refinement with self-feedback. *NeurIPS* 36.
[113] Zhao Mandi, Shreeya Jain, and Shuran Song. 2023. Roco: Dialectic multi-robot collaboration with large language models. Retrieved from https://arxiv.org/pdf/2307.04738
[114] Tula Masterman, Sandi Besen, Mason Sawtell, and Alex Chao. 2024. The landscape of emerging AI agent architectures for reasoning, planning, and tool calling: A survey. Retrieved from https://arxiv.org/pdf/2404.11584
[115] Richard May and Kerstin Denecke. 2022. Security, privacy, and healthcare-related conversational agents: A scoping review. *Informatics for Health and Social Care* 47, 2 (2022), 194–210.

[116] Nikhil Mehta, Milagro Teruel, Xin Deng, Sergio Figueroa Sanz, Ahmed Awadallah, and Julia Kiseleva. 2024. Improving grounded language understanding in a collaborative environment by interacting with agents through help feedback. In *Proceedings of the EACL*.
[117] Kai Mei, Zelong Li, Shuyuan Xu, Ruosong Ye, Yingqiang Ge, and Yongfeng Zhang. 2024. AIOS: LLM agent operating system. Retrieved from https://arxiv.org/pdf/2403.16971
[118] Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the EMNLP*.
[119] Lingbo Mo, Zeyi Liao, Boyuan Zheng, Yu Su, Chaowei Xiao, and Huan Sun. 2024. A trembling house of cards? Mapping adversarial attacks against language agents. Retrieved from https://arxiv.org/pdf/2402.10196
[120] John Morris, Volodymyr Kuleshov, Vitaly Shmatikov, and Alexander Rush. 2023. Text embeddings reveal (almost) as much as text. In *Proceedings of the EMNLP*.
[121] Stephen Moskal, Sam Laney, Erik Hemberg, and Una-May O'Reilly. 2023. LLMs killed the script kiddie: How agents supported by large language models change the landscape of network threat testing. Retrieved from https://arxiv.org/pdf/2310.06936
[122] Sumeet Ramesh Motwani, Mikhail Baranchuk, Lewis Hammond, and Christian Schroeder de Witt. 2023. A perfect collusion benchmark: How can AI agents be prevented from colluding with information-theoretic undetectability?. In *Proceedings of the NeurIPS Workshop*.
[123] Hichem Mrabet, Sana Belguith, Adeeb Alhomoud, and Abderrazak Jemai. 2020. A survey of IoT security based on a layered architecture of sensing and data analysis. *Sensors* 20, 13 (2020), 3625.
[124] Dor Muhlgay, Ori Ram, Inbal Magar, Yoav Levine, Nir Ratner, Yonatan Belinkov, Omri Abend, Kevin Leyton-Brown, Amnon Shashua, and Yoav Shoham. 2024. Generating benchmarks for factuality evaluation of language models. In *Proceedings of the EACL*.
[125] Varun Nair, Elliot Schumacher, Geoffrey Tso, and Anitha Kannan. 2023. DERA: Enhancing large language model completions with dialog-enabled resolving agents. Retrieved from https://arxiv.org/pdf/2303.17071
[126] Yohei Nakajima. 2022. *Yohei's Blog Post*. Retrieved from https://twitter.com/yoheinakajima/status/1582844144640471040
[127] Tai Nguyen and Eric Wong. 2023. In-context example selection with influences. Retrieved from https://arxiv.org/pdf/2302.11042
[128] Aidan O'Gara. 2023. Hoodwinked: Deception and cooperation in a text-based game for language models. Retrieved from https://arxiv.org/pdf/2308.01404
[129] Nedjma Ousidhoum, Xinran Zhao, Tianqing Fang, Yangqiu Song, and Dit-Yan Yeung. 2021. Probing toxic content in large pre-trained language models. In *Proceedings of the IJCNLP*.
[130] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray et al. 2022. Training language models to follow instructions with human feedback. *NeurIPS*.
[131] James Jie Pan, Jianguo Wang, and Guoliang Li. 2024. Survey of vector database management systems. *The VLDB Journal* 33, 5 (2024), 1591–1615.
[132] Yikang Pan, Liangming Pan, Wenhu Chen, Preslav Nakov, Min-Yen Kan, and William Yang Wang. 2023. On the risk of misinformation pollution with large language models. Retrieved from https://arxiv.org/pdf/2305.13661
[133] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the UIST*.
[134] Peter S. Park, Simon Goldstein, Aidan O'Gara, Michael Chen, and Dan Hendrycks. 2024. AI deception: A survey of examples, risks, and potential solutions. *Patterns* 5, 5 (2024).
[135] Rodrigo Pedro, Daniel Castro, Paulo Carreira, and Nuno Santos. 2023. From prompt injections to SQL injection attacks: How protected is your LLM-integrated web application? Retrieved from https://arxiv.org/pdf/2308.01990
[136] Ethan Perez et al. 2023. Discovering language model behaviors with model-written evaluations. In *Proceedings of the ACL*.
[137] Fábio Perez and Ian Ribeiro. 2022. Ignore previous prompt: Attack techniques for language models. *NeurIPS* (2022).
[138] Fábio Perez and Ian Ribeiro. 2022. Ignore previous prompt: Attack techniques for language models. Retrieved from https://arXiv:2211.09527
[139] Steve Phelps and Rebecca Ranson. 2023. Of Models and Tin Men–a behavioural economics study of principal-agent problems in AI alignment using large-language models. Retrieved from https://arxiv.org/pdf/2307.11137
[140] Lukas Pöhler, Valentin Schrader, Alexander Ladwein, and Florian von Keller. 2024. A technological perspective on misuse of available AI. Retrieved from https://arxiv.org/pdf/2403.15325
[141] Harsha Putla, Chanakya Patibandla, Krishna Pratap Singh, and P. Nagabhushan. 2024. A pilot study of observation poisoning on selective reincarnation in multi-agent reinforcement learning. *Neural Processing Letters* 56, 3 (2024), 161.

[142] Chen Qian, Xin Cong, Cheng Yang, Weize Chen, Yusheng Su, Juyuan Xu, Zhiyuan Liu, and Maosong Sun. 2023. Communicative agents for software development. Retrieved from https://arxiv.org/pdf/2307.07924
[143] Francisco Quiroga, Gabriel Hermosilla, German Varas, Francisco Alonso, and Karla Schröder. 2024. RL-Based Sim2Real Enhancements for Autonomous Beach-Cleaning Agents. *Applied Sciences* 14, 11 (2024), 4602.
[144] Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young et al. 2021. Scaling language models: Methods, analysis & insights from training Gopher. Retrieved from https://arxiv.org/pdf/2112.11446
[145] Fathima Abdul Rahman and Guang Lu. 2023. A contextualized real-time multimodal emotion recognition for conversational agents using graph convolutional networks in reinforcement learning. Retrieved from https://arxiv.org/pdf/2310.18363
[146] Leonardo Ranaldi and Giulia Pucci. 2023. When large language models contradict humans? Large language models' sycophantic behaviour. Retrieved from https://arxiv.org/pdf/2311.09410
[147] Embrace The Red. 2023. Indirect Prompt Injection via YouTube Transcripts · Embrace The Red—embracethered.com. Retrieved from https://embracethered.com/blog/posts/2023/chatgpt-plugin-youtube-indirect-prompt-injection/
[148] Alexander Robey, Zachary Ravichandran, Vijay Kumar, Hamed Hassani, and George J Pappas. 2024. Jailbreaking LLM-controlled robots. Retrieved from https://arxiv.org/pdf/2410.13691
[149] Sippo Rossi, Alisia Marianne Michel, Raghava Rao Mukkamala, and Jason Bennett Thatcher. 2024. An early categorization of prompt injection attacks on large language models. Retrieved from https://arxiv.org/pdf/2402.00898
[150] Yangjun Ruan, Honghua Dong, Andrew Wang, Silviu Pitis, Yongchao Zhou, Jimmy Ba, Yann Dubois, Chris J. Maddison, and Tatsunori Hashimoto. 2024. Identifying the risks of LM agents with an LM-emulated sandbox. In *Proceedings of the ICLR*.
[151] Ahmed Salem, Andrew Paverd, and Boris Köpf. 2023. Maatphor: Automated variant analysis for prompt injection attacks. Retrieved from https://arxiv.org/pdf/2312.11513
[152] Sergei Savvov. 2023. Fixing hallucinations in LLMs. Retrieved from https://betterprogramming.pub/fixing-hallucinations-in-llms-9ff0fd438e33
[153] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2024. Toolformer: Language models can teach themselves to use tools. *NeurIPS*.
[154] Leo Schwinn, David Dobre, Stephan Günnemann, and Gauthier Gidel. 2023. Adversarial attacks and defenses in large language models: Old and new threats. Retrieved from https://arxiv.org/pdf/2310.19737
[155] Jose Selvi. 2022. Exploring Prompt Injection Attacks. Retrieved from https://nccgroup.com/au/research-blog/exploring-prompt-injection-attacks/
[156] Omar Shaikh, Hongxin Zhang, William Held, Michael Bernstein, and Diyi Yang. 2023. On second thought, let's not think step by step! Bias and toxicity in zero-shot reasoning. In *Proceedings of the ACL*.
[157] Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Newton Cheng, Esin Durmus, Zac Hatfield-Dodds, Scott R. Johnston et al. 2023. Towards understanding sycophancy in language models. Retrieved from https://arxiv.org/pdf/2310.13548
[158] Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. 2024. "do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 1671–1685.
[159] Xinyue Shen, Yixin Wu, Michael Backes, and Yang Zhang. 2024. Voice jailbreak attacks against GPT-4o. Retrieved from https://arxiv.org/abs/2405.19103
[160] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2024. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *NeurIPS*.
[161] Chuan Sheng, Wanlun Ma, Qing-Long Han, Wei Zhou, Xiaogang Zhu, Sheng Wen, Yang Xiang, and Fei-Yue Wang. 2024. Pager explosion: Cybersecurity insights and afterthoughts. *IEEE/CAA Journal of Automatica Sinica* 11, 12 (2024), 2359–2362.
[162] Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Proceedings of the EMNLP*.
[163] Emily H. Soice, Rafael Rocha, Kimberlee Cordova, Michael Specter, and Kevin M. Esvelt. 2023. Can large language models democratize access to dual-use biotechnology? Retrieved from https://arxiv.org/pdf/2306.03809
[164] Congzheng Song and Ananth Raghunathan. 2020. Information leakage in embedding models. In *Proceedings of the CCS*. 377–390.
[165] Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M. Sadler, Wei-Lun Chao, and Yu Su. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the CVPR*. 2998–3009.
[166] Shyam Sudhakaran, Miguel González-Duque, Matthias Freiberger, Claire Glanois, Elias Najarro, and Sebastian Risi. 2024. Mariogpt: Open-ended text2level generation through large language models. *NeurIPS*.

[167] Weiwei Sun, Zhengliang Shi, Shen Gao, Pengjie Ren, Maarten de Rijke, and Zhaochun Ren. 2023. Contrastive learning reduces hallucination in conversations. In *Proceedings of the AAAI*.
[168] Yuxiang Sun, Checheng Yu, Junjie Zhao, Wei Wang, and Xianzhong Zhou. 2023. Self generated wargame AI: Double layer agent task planning based on large language model. Retrieved from https://arxiv.org/pdf/2312.01090
[169] Harini Suresh and John V. Guttag. 2019. A framework for understanding unintended consequences of machine learning. Retrieved from https://arxiv.org/pdf/1901.10002
[170] Gaurav Suri, Lily R. Slater, Ali Ziaee, and Morgan Nguyen. 2024. Do large language models show decision heuristics similar to humans? A case study using GPT-3.5. *Journal of Experimental Psychology: General* 153, 4 (2024), 1066–1075.
[171] Weihao Tan, Wentao Zhang, Shanqi Liu, Longtao Zheng, Xinrun Wang, and Bo An. 2024. True knowledge comes from practice: Aligning large language models with embodied environments via reinforcement learning. In *Proceedings of the ICLR*.
[172] Xiangru Tang, Qiao Jin, Kunlun Zhu, Tongxin Yuan, Yichi Zhang, Wangchunshu Zhou, Meng Qu, Yilun Zhao, Jian Tang, Zhuosheng Zhang et al. 2024. Prioritizing safeguarding over autonomy: Risks of LLM agents for science. Retrieved from https://arxiv.org/pdf/2402.04247
[173] Mikhail Terekhov, Romain Graux, Eduardo Neville, Denis Rosset, and Gabin Kolly. 2023. Second-order Jailbreaks: Generative agents successfully manipulate through an intermediary. In *Proceedings of the Multi-Agent Security Workshop @ NeurIPS'23*. Retrieved from https://openreview.net/forum?id=HPmhaOTseN
[174] Yu Tian, Xiao Yang, Jingyuan Zhang, Yinpeng Dong, and Hang Su. 2023. Evil geniuses: Delving into the safety of llm-based agents. Retrieved from https://arxiv.org/pdf/2311.11855
[175] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar et al. 2023. Llama: Open and efficient foundation language models. Retrieved from https://arxiv.org/pdf/2302.13971
[176] Sam Toyer, Olivia Watkins, Ethan Adrian Mendes, Justin Svegliato, Luke Bailey, Tiffany Wang, Isaac Ong, Karim Elmaaroufi, Pieter Abbeel, Trevor Darrell, Alan Ritter, and Stuart Russell. 2024. Tensor trust: Interpretable prompt injection attacks from an online game. In *Proceedings of the ICLR*.
[177] Dennis Ulmer, Elman Mansimov, Kaixiang Lin, Justin Sun, Xibin Gao, and Yi Zhang. 2024. Bootstrapping LLM-based task-oriented dialogue agents via self-talk. Retrieved from https://arxiv.org/pdf/2401.05033
[178] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *NeurIPS*.
[179] Bertie Vidgen, Adarsh Agrawal, Ahmed M. Ahmed, Victor Akinwande, Namir Al-Nuaimi, Najla Alfaraj, Elie Alhajjar, Lora Aroyo, Trupti Bavalatti, Borhane Blili-Hamelin et al. 2024. Introducing v0. 5 of the AI safety benchmark from MLCommons. Retrieved from https://arxiv.org/pdf/2404.12241
[180] Celine Wald and Lukas Pfahler. 2023. Exposing bias in online communities through large-scale language models. Retrieved from https://arxiv.org/pdf/2306.02294
[181] Eric Wallace, Kai Xiao, Reimar Leike, Lilian Weng, Johannes Heidecke, and Alex Beutel. 2024. The instruction hierarchy: Training LLMs to prioritize privileged instructions. Retrieved from https://arxiv.org/pdf/2404.13208
[182] Alexander Wan, Eric Wallace, Sheng Shen, and Dan Klein. 2023. Poisoning language models during instruction tuning. In *Proceedings of the ICML*.
[183] Boxin Wang, Weixin Chen, Hengzhi Pei, Chulin Xie, Mintong Kang, Chenhui Zhang, Chejian Xu, Zidi Xiong, Ritik Dutta, Rylan Schaeffer, et al.. 2023. DecodingTrust: A comprehensive assessment of trustworthiness in GPT models. In *Proceedings of the NeurIPS*.
[184] Fengxiang Wang, Ranjie Duan, Peng Xiao, Xiaojun Jia, YueFeng Chen, Chongwen Wang, Jialing Tao, Hang Su, Jun Zhu, and Hui Xue. 2024. MRJ-Agent: An effective jailbreak agent for multi-round dialogue. Retrieved from https://arxiv.org/pdf/2411.03814
[185] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *Proceedings of the ICLR*.
[186] Yuntao Wang, Yanghe Pan, Miao Yan, Zhou Su, and Tom H. Luan. 2023. A survey on ChatGPT: AI-generated contents, challenges, and solutions. *IEEE Open Journal of the Computer Society* 4 (2023), 280–302.
[187] Yau-Shian Wang and Yingshan Chang. 2022. Toxicity detection with generative prompt-based inference. *arXiv* (2022).
[188] Connor Weeks, Aravind Cheruvu, Sifat Muhammad Abdullah, Shravya Kanchi, Daphne Yao, and Bimal Viswanath. 2023. A first look at toxicity injection attacks on open-domain chatbots. In *Proceedings of the ACSAC*.
[189] Jerry Wei, Da Huang, Yifeng Lu, Denny Zhou, and Quoc V. Le. 2023. Simple synthetic data reduces sycophancy in large language models. Retrieved from https://arxiv.org/pdf/2308.03958
[190] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *NeurIPS*.
[191] Jerry Wei, Chengrun Yang, Xinying Song, Yifeng Lu, Nathan Hu, Dustin Tran, Daiyi Peng, Ruibo Liu, Da Huang, Cosmo Du et al. 2024. Long-form factuality in large language models. Retrieved from https://arxiv.org/pdf/2403.18802

[192] Zeming Wei, Yifei Wang, and Yisen Wang. 2023. Jailbreak and guard aligned language models with only few in-context demonstrations. Retrieved from https://arxiv.org/pdf/2310.06387
[193] Roy Weiss, Daniel Ayzenshteyn, Guy Amit, and Yisroel Mirsky. 2024. What was your prompt? A remote keylogging attack on AI assistants. Retrieved from https://arxiv.org/pdf/2403.09751
[194] Johannes Welbl, Amelia Glaese, Jonathan Uesato, Sumanth Dathathri, John Mellor, Lisa Anne Hendricks, Kirsty Anderson, Pushmeet Kohli, Ben Coppin, and Po-Sen Huang. 2021. Challenges in detoxifying language models. In *Proceedings of the EMNLP*.
[195] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt. 2023. A prompt pattern catalog to enhance prompt engineering with chatgpt. Retrieved from https://arxiv.org/pdf/2302.11382
[196] Simon Willison. 2023. Delimiters won't save you from prompt injection. Retrieved from https://simonwillison.net/2023/May/11/delimiters-wont-save-you/
[197] David Windridge, Henrik Svensson, and Serge Thill. 2021. On the utility of dreaming: A general model for how learning in artificial agents can benefit from data hallucination. *Adaptive Behavior* 29, 3 (2021), 267–280.
[198] Yotam Wolf, Noam Wies, Yoav Levine, and Amnon Shashua. 2023. Fundamental limitations of alignment in large language models. Retrieved from https://arxiv.org/pdf/2304.11082
[199] Chen Henry Wu, Jing Yu Koh, Ruslan Salakhutdinov, Daniel Fried, and Aditi Raghunathan. 2024. Adversarial attacks on multimodal agents. Retrieved from https://arxiv.org/pdf/2406.12814v1
[200] Fangzhou Wu, Shutong Wu, Yulong Cao, and Chaowei Xiao. 2024. WIPI: A new web threat for LLM-driven web agents. Retrieved from https://arxiv.org/pdf/2402.16965
[201] Fangzhou Wu, Ning Zhang, Somesh Jha, Patrick McDaniel, and Chaowei Xiao. 2024. A new era in LLM security: Exploring security concerns in real-world LLM-based systems. Retrieved from https://arxiv.org/pdf/2402.18649
[202] Young Wu, Jeremy McMahan, Xiaojin Zhu, and Qiaomin Xie. 2023. Reward poisoning attacks on offline multi-agent reinforcement learning. In *Proceedings of the AAAI*.
[203] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou et al. 2023. The rise and potential of large language model based agents: A survey. Retrieved from https://arxiv.org/pdf/2309.07864
[204] Junlin Xie, Zhihong Chen, Ruifei Zhang, Xiang Wan, and Guanbin Li. 2024. Large multimodal agents: A survey. Retrieved from https://arxiv.org/pdf/2402.15116
[205] Xingrui Xie, Han Liu, Wenzhe Hou, and Hongbin Huang. 2023. A brief survey of vector databases. In *Proceedings of the BigDIA*. IEEE.
[206] Frank Xing. 2024. Designing heterogeneous LLM agents for financial sentiment analysis. Retrieved from https://arxiv.org/pdf/2401.05799
[207] Binfeng Xu, Zhiyuan Peng, Bowen Lei, Subhabrata Mukherjee, Yuchen Liu, and Dongkuan Xu. 2023. Rewoo: Decoupling reasoning from observations for efficient augmented language models. Retrieved from https://arxiv.org/pdf/2305.18323
[208] Yuzhuang Xu, Shuo Wang, Peng Li, Fuwen Luo, Xiaolong Wang, Weidong Liu, and Yang Liu. 2023. Exploring large language models for communication games: An empirical study on werewolf. Retrieved from https://arxiv.org/pdf/2309.04658
[209] Zelai Xu, Chao Yu, Fei Fang, Yu Wang, and Yi Wu. 2023. Language agents with reinforcement learning for strategic play in the werewolf game. Retrieved from https://arxiv.org/pdf/2309.04658
[210] Xue Yan, Yan Song, Xinyu Cui, Filippos Christianos, Haifeng Zhang, David Henry Mguni, and Jun Wang. 2023. Ask more, know better: Reinforce-learned prompt questions for decision making with large language models. Retrieved from https://arxiv.org/pdf/2310.18127
[211] Wenkai Yang, Xiaohan Bi, Yankai Lin, Sishuo Chen, Jie Zhou, and Xu Sun. 2024. Watch out for your agents! Investigating backdoor threats to LLM-Based agents. Retrieved from https://arxiv.org/pdf/2402.11208
[212] Yong Yang, Xuhong Zhang, Yi Jiang, Xi Chen, Haoyu Wang, Shouling Ji, and Zonghui Wang. 2024. PRSA: Prompt reverse stealing attacks against large language models. Retrieved from https://arxiv.org/pdf/2402.19200
[213] Dongyu Yao, Jianshu Zhang, Ian G. Harris, and Marcel Carlsson. 2024. FuzzLLM: A novel and universal fuzzing framework for proactively discovering jailbreak vulnerabilities in large language models. In *Proceedings of the ICASSP*.
[214] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *NeurIPS* (2024).
[215] Yifan Yao, Jinhao Duan, Kaidi Xu, Yuanfang Cai, Zhibo Sun, and Yue Zhang. 2024. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing* (2024), 100211.
[216] Jingwei Yi, Yueqi Xie, Bin Zhu, Keegan Hines, Emre Kiciman, Guangzhong Sun, Xing Xie, and Fangzhao Wu. 2023. Benchmarking and defending against indirect prompt injection attacks on large language models. Retrieved from https://arxiv.org/pdf/2312.14197

[217] Jiahao Yu, Xingwei Lin, and Xinyu Xing. 2023. Gptfuzzer: Red teaming large language models with auto-generated jailbreak prompts. Retrieved from https://arxiv.org/pdf/2309.10253

[218] Zhiyuan Yu, Xiaogeng Liu, Shunning Liang, Zach Cameron, Chaowei Xiao, and Ning Zhang. 2024. Don't listen to me: Understanding and exploring jailbreak prompts of large language models. Retrieved from https://arxiv.org/pdf/2403.17336

[219] Tongxin Yuan, Zhiwei He, Lingzhong Dong, Yiming Wang, Ruijie Zhao, Tian Xia, Lizhen Xu, Binglin Zhou, Fangqi Li, Zhuosheng Zhang et al. 2024. R-Judge: Benchmarking safety risk awareness for LLM agents. In *Proceedings of the ICLR*.

[220] Shenglai Zeng, Jiankun Zhang, Pengfei He, Yue Xing, Yiding Liu, Han Xu, Jie Ren, Shuaiqiang Wang, Dawei Yin, Yi Chang et al. 2024. The good and the bad: Exploring privacy issues in retrieval-augmented generation (RAG). Retrieved from https://arxiv.org/pdf/2402.16893

[221] Yifan Zeng, Yiran Wu, Xiao Zhang, Huazheng Wang, and Qingyun Wu. 2024. AutoDefense: Multi-agent LLM defense against jailbreak attacks. Retrieved from https://arxiv.org/pdf/2403.04783

[222] Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. 2024. Injecagent: Benchmarking indirect prompt injections in tool-integrated large language model agents. Retrieved from https://arxiv.org/pdf/2403.02691

[223] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua Tenenbaum, Tianmin Shu, and Chuang Gan. 2023. Building cooperative embodied agents modularly with large language models. In *Proceedings of the NeurIPS Workshop*.

[224] Wanpeng Zhang and Zongqing Lu. 2023. Rladapter: Bridging large language models to reinforcement learning in open worlds. Retrieved from https://arxiv.org/pdf/2309.17176

[225] Xinyu Zhang, Huiyu Xu, Zhongjie Ba, Zhibo Wang, Yuan Hong, Jian Liu, Zhan Qin, and Kui Ren. 2024. PrivacyAsst: Safeguarding user privacy in tool-using large language model agents. *IEEE Transactions on Dependable and Secure Computing* 21, 6 (2024), 5242–5258.

[226] Yiming Zhang, Nicholas Carlini, and Daphne Ippolito. 2024. Effective prompt extraction from language models. Retrieved from https://arxiv.org/pdf/2307.06865

[227] Yuqi Zhang, Liang Ding, Lefei Zhang, and Dacheng Tao. 2024. Intention analysis prompting makes large language models a good jailbreak defender. Retrieved from https://arxiv.org/pdf/2401.06561

[228] Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen et al. 2023. Siren's song in the AI ocean: A survey on hallucination in large language models. Retrieved from https://arxiv.org/pdf/2309.01219

[229] Zeyu Zhang, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Quanyu Dai, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. 2024. A survey on the memory mechanism of large language model based agents. Retrieved from https://arxiv.org/pdf/2404.13501

[230] Zhexin Zhang, Junxiao Yang, Pei Ke, and Minlie Huang. 2023. Defending large language models against jailbreaking attacks through goal prioritization. Retrieved from https://arxiv.org/pdf/2311.09096

[231] Qinlin Zhao, Jindong Wang, Yixuan Zhang, Yiqiao Jin, Kaijie Zhu, Hao Chen, and Xing Xie. 2023. Competeai: Understanding the competition behaviors in large language model-based agents. Retrieved from https://arxiv.org/pdf/2310.17512

[232] Wei Zhou, Xiaogang Zhu, Qing-Long Han, Lin Li, Xiao Chen, Sheng Wen, and Yang Xiang. 2024. The security of using large language models: A survey with emphasis on ChatGPT. *IEEE/CAA Journal of Automatica Sinica* 12, 1 (2024), 1–26.

[233] Yiyang Zhou, Chenhang Cui, Jaehong Yoon, Linjun Zhang, Zhun Deng, Chelsea Finn, Mohit Bansal, and Huaxiu Yao. 2024. Analyzing and mitigating object hallucination in large vision-language models. In *Proceedings of the ICLR*.

[234] Xiaogang Zhu, Wei Zhou, Qing-Long Han, Wanlun Ma, Sheng Wen, and Yang Xiang. 2025. When software security meets large language models: A survey. *IEEE/CAA Journal of Automatica Sinica* 12, 2 (2024), 317–334.

[235] Wei Zou, Runpeng Geng, Binghui Wang, and Jinyuan Jia. 2024. PoisonedRAG: Knowledge poisoning attacks to retrieval-augmented generation of large language models. Retrieved from https://arxiv.org/pdf/2402.07867

Received 3 June 2024; revised 5 December 2024; accepted 3 February 2025