IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 15 September 2025, accepted 5 October 2025, date of publication 9 October 2025, date of current version 16 October 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3619764

SURVEY

A Lifecycle-Oriented Survey of Emerging Threats and Vulnerabilities in Large Language Models

CARMEN DE MAIO[1], (Member, IEEE), MARIA DI GISI[2,3], GIUSEPPE FENZA[3], (Senior Member, IEEE), MARIACRISTINA GALLO[3], (Member, IEEE), AND VINCENZO LOIA[3], (Senior Member, IEEE)

[1]Department of Computer Science, University of Salerno, 84084 Fisciano, Italy
[2]IMT School for Advanced Studies Lucca, 55100 Lucca, Italy
[3]Department of Management and Innovation Systems, University of Salerno, 84084 Fisciano, Italy

Corresponding author: Giuseppe Fenza (gfenza@unisa.it)

This work was supported in part by Project SEcurity and Rights in the CyberSpace (SERICS) through the Ministero dell'Università e della Ricerca (MUR) National Recovery and Resilience Plan funded by the European Union-NextGenerationEU under Grant PE00000014.

ABSTRACT Large Language Models (LLMs) have become key enablers for a wide range of natural language tasks, spanning understanding, generation, and inference across various applications and industrial domains. However, their increasing adoption has brought to light numerous security and privacy vulnerabilities, some of which remain insufficiently studied. This paper aims to identify and examine underexplored vulnerabilities affecting LLMs, with particular attention to threats that remain underrepresented in current literature. The selection of vulnerabilities is guided by a systematic comparison of four major surveys, prioritizing in-depth analysis for vulnerabilities absent in at least three of these works. The methodological approach combines a targeted literature search across major academic databases with strict inclusion and exclusion criteria focused on relevance, novelty, and the selection of peer-reviewed journal and conference publications. The survey introduces a taxonomy based on the LLM lifecycle (i.e., training, inference, and deployment) through which 17 vulnerabilities are categorized and discussed. Eight emerging threats (e.g., model collapse, gradient leakage, denial-of-service, and dependency risks) receive comprehensive analysis, reflecting their growing relevance and limited prior exploration. The result is a structured taxonomy and in-depth analysis designed to guide both academic investigations and practical efforts toward the secure deployment of LLMs in real-world environments. In addition to mapping the current threat landscape, this survey highlights open challenges and outlines key directions for future research.

INDEX TERMS Large language models (LLMs), security vulnerabilities, LLM risks, model lifecycle, training-time attacks, inference vulnerabilities, deployment security, LLM taxonomy, systematic survey.

I. INTRODUCTION

Large Language Models (LLMs) have rapidly evolved from research prototypes to critical components of modern artificial intelligence systems, enabling advances in natural language understanding, generation, and inference. Their integration spans numerous domains, including conversational agents, content generation, code assistance, healthcare decision support, and financial analysis. This widespread adoption has been accompanied by growing concerns about the security and privacy risks associated with these models [1], [2], [3], particularly as they are deployed in sensitive and safety-critical contexts.

The security landscape of LLMs is characterized by a wide and heterogeneous attack surface, shaped by factors such as the complexity of training pipelines, the opacity of model parameters and data, and the increasing modularity of deployment scenarios (e.g., through Retrieval-Augmented

The associate editor coordinating the review of this manuscript and approving it for publication was Zijian Zhang.

Generation and API-based access). As these systems are continuously retrained, fine-tuned, and embedded in dynamic pipelines, new vulnerabilities emerge that differ significantly across the lifecycle stages of LLMs. In this context, general research on adversarial attacks and defenses in machine learning provides useful insights for improving model robustness [4].

Several recent surveys have attempted to classify and analyze the threats affecting LLMs. However, existing works tend to focus on well-known attack vectors (e.g., prompt injection, data poisoning, or adversarial examples) while offering limited coverage of emerging vulnerabilities. Moreover, most surveys adopt a fragmented approach, addressing isolated lifecycle phases (e.g., training or inference) without providing a comprehensive end-to-end perspective.

This survey addresses these limitations by proposing a structured and lifecycle-oriented taxonomy of LLM vulnerabilities, spanning the training, inference, and deployment stages. A systematic comparison of four major surveys has been conducted to identify threats that remain underrepresented in the literature. Vulnerabilities not addressed in at least three of the four surveys have been prioritized for in-depth analysis.

Seventeen vulnerabilities are categorized within the proposed taxonomy, including eight emerging threats, such as model collapse during retraining, gradient leakage in fine-tuning processes, denial-of-service via crafted prompts, and dependency risks in Retrieval-Augmented Generation (RAG) systems, that are still insufficiently explored in prior literature. In addition to providing a comprehensive overview of the current threat landscape, this work highlights key open challenges and outlines future directions for research and mitigation.

The main contributions of this survey are as follows:

- Providing a taxonomy of LLM vulnerabilities organized along the model lifecycle, offering a structured perspective to facilitate the understanding and comparison of existing attacks and component-focused classifications.
- Presenting an analysis of 17 vulnerabilities, with a particular focus on 8 emerging threats that have been largely overlooked in previous literature.
- Identifying key open challenges and outlining future research directions to study and mitigate these risks and support the secure deployment of LLM-based systems.

The remainder of the paper is organized as follows: Section II introduces the taxonomy of LLM vulnerabilities structured according to the lifecycle phases of training, inference, and deployment, aligned with the key research questions. Section III presents the methodology for vulnerability selection and data collection. Section IV reviews major existing surveys, forming the basis for the inclusion of vulnerability criteria. Subsequent sections provide detailed analyses of selected vulnerabilities and discuss open challenges and future directions.

II. A LIFECYCLE-BASED TAXONOMY OF LLM VULNERABILITIES

Understanding the vulnerabilities of Large Language Models (LLMs) requires a structured framework that captures the complexity of modern model development and deployment. This section introduces a taxonomy grounded in the lifecycle of LLMs, segmenting vulnerabilities into three main phases: training, inference, and deployment. This phase-based perspective not only facilitates the alignment of threats with their respective stages in the pipeline, but also provides a coherent analytical structure that mirrors the three research questions guiding this survey:

1) **RQ1**: What vulnerabilities can arise during the *training* phase of large language models?
2) **RQ2**: What vulnerabilities can affect large language models during the *inference* phase?
3) **RQ3**: What risks and vulnerabilities can emerge during the *deployment* of large language models?

To ensure both focus and novelty, the survey adopts a selective review strategy: only vulnerabilities that are not already extensively covered in the existing literature are examined in detail. Specifically, a vulnerability is selected for in-depth review if it is not discussed by at least three of the four major surveys analyzed in Section IV. For each lifecycle phase, a final subsection titled ''Other widely discussed vulnerabilities'' summarizes prominent threats that fall outside the scope of this survey's core analysis. These widely covered topics are briefly introduced and accompanied by a key reference work for readers seeking a deeper understanding.

A vulnerability is defined here as any intrinsic weakness or design flaw (whether in the model, data, or surrounding infrastructure) that can be exploited to compromise security, privacy, integrity, or trust. Vulnerabilities may be intentional (e.g., backdoors injected during training) or unintentional (e.g., excessive memorization of training data), and they often interact with the environment in which the LLM operates.

Each vulnerability category is introduced through a conceptual overview that outlines its objectives, typical threat models, attack surfaces and techniques, as well as its potential impact on the security or integrity of large language models.

Figure 1 illustrates the three-phase taxonomy, and each phase is described in detail below.

A. TRAINING-PHASE VULNERABILITIES

The training phase involves the ingestion of massive datasets, often scraped from the web or obtained through heterogeneous pipelines. At this stage, the model is particularly susceptible to threats originating from malicious or low-quality data, collaborative training dynamics, and model design choices. Training-phase vulnerabilities are especially difficult to detect post hoc, as they become deeply embedded in the model's parameters and are often indistinguishable from intended behaviors.

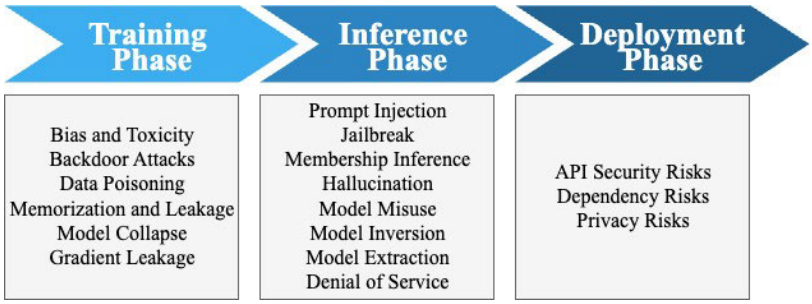


FIGURE 1. Taxonomy of LLM vulnerabilities grouped by lifecycle phase.

B. INFERENCE-PHASE VULNERABILITIES

Once deployed, LLMs begin interacting with users through prompts and instructions. This inference phase introduces a new class of threats that exploit the model's generative capabilities and flexibility. Inference-phase vulnerabilities are often more visible than those at training, but also more dynamic, as attackers may iteratively refine prompts in response to model behavior.

C. DEPLOYMENT-PHASE VULNERABILITIES

The final stage in the LLM lifecycle involves integration into real-world applications. At this point, vulnerabilities emerge from the interaction between the LLM and its host environment, including APIs, plug-ins, and user interfaces. Deployment-phase vulnerabilities are often systemic, arising from architectural choices or integration flaws, and are difficult to address solely through model-level interventions.

III. METHODOLOGY

A. VULNERABILITIES SELECTION CRITERIA

To define the scope of the review in a systematic and objective manner, a selection criterion was applied based on the degree of coverage in existing literature. Specifically, each vulnerability was assessed according to its presence across four recent surveys addressing LLM-related security and privacy threats.

Vulnerabilities not covered by at least three out of the four selected surveys were included for in-depth analysis. This approach ensures, as synthesized in Table 1, that lesser-known, underexplored, or emerging threats receive detailed attention, thereby filling existing gaps in the literature. These vulnerabilities are examined within the sections corresponding to the main stages of the LLM lifecycle: training, inference, and deployment.

Conversely, vulnerabilities that are already extensively addressed (i.e., present in at least three of the four surveys) are reviewed more concisely. These are briefly discussed at the end of the relevant lifecycle sections to maintain completeness while avoiding redundancy.

This inclusion strategy allows for a balanced review structure, prioritizing novel insights without disregarding well-established threat vectors.

TABLE 1. Inclusion criteria for vulnerability coverage.

Survey Coverage	Treatment in This Review
Covered in <3 out of 4 surveys	In-depth analysis in a dedicated section corresponding to the relevant lifecycle phase.
Covered in ≥ 3 out of 4 surveys	Brief review at the end of the corresponding lifecycle section.

TABLE 2. Academic Databases Used for Literature Search.

Name	URL
Scopus	www.scopus.com
ACM Digital Library	acm.org
IEEE Xplore	ieeexplore.ieee.org
Web of Science	www.webofscience.com

B. SEARCH STRATEGY

The initial literature search was conducted across major academic databases, summarized in Table 2.

1) SURVEY-LEVEL SEARCH QUERIES

To identify relevant survey articles (described in Section IV) related to LLM vulnerabilities and security, a set of Boolean search queries was applied across the selected academic databases:

- `("large language models" OR LLM) AND vulnerabilities`
- `("large language models" OR LLM) AND risks`
- `("large language models" OR LLM) AND attacks`
- `("large language models" OR LLM) AND security`
- `("large language models" OR LLM) AND threats`
- `("large language models" OR LLM) AND adversarial`
- `("large language models" OR LLM) AND safety`
- `("large language models" OR LLM) AND robustness`

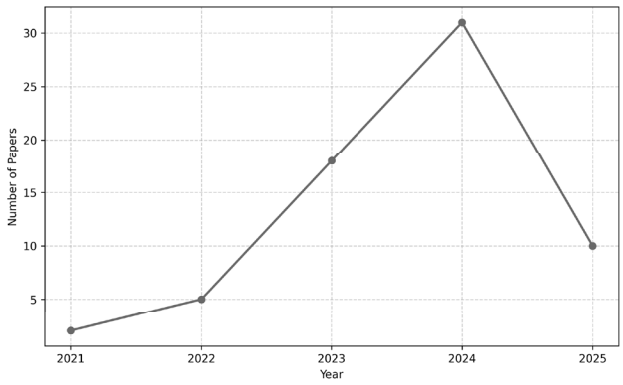


FIGURE 2. Temporal distribution of selected papers (2021–2025).

- `("large language models" OR LLM) AND trustworthiness`

These queries were applied to titles, abstracts, and keywords when supported by the search interface.

2) VULNERABILITY-SPECIFIC SEARCH QUERIES

To collect literature related to each identified vulnerability, keyword-based searches were conducted using the following general query structures:

- `("large language models" OR LLM) AND [vulnerability name] AND attack`
- `("large language models" OR LLM) AND [vulnerability name]`

This search methodology was consistently applied across all vulnerability categories. When applicable, alternative expressions or synonyms (e.g., "`model stealing`" or "`model theft`" for "`model extraction`") were included to ensure broader coverage.

Manual inspection of the references and citations in the retrieved papers was also performed to identify relevant works not captured through automated queries.

C. INCLUSION/EXCLUSION FILTERS

The literature selection for each identified vulnerability was guided by precise inclusion and exclusion criteria summarized in Table 3. Only peer-reviewed articles published between 2021 and 2025 were considered, reflecting the recent rapid evolution of Large Language Models (LLMs). The focus was restricted to conference and journal papers that explicitly addressed LLMs and provided a substantive contribution to the understanding of vulnerabilities or security issues. Excluded from the review were publications prior to 2021, as they predate the widespread adoption of LLMs and thus lack specific relevance. Additionally, articles that did not explicitly mention LLMs or related foundation models were discarded. Technical reports, policy papers, and working papers were also excluded to maintain the quality and accessibility of the survey. While non-peer-reviewed papers are generally excluded, a few are retained based on their relevance or citation impact. In total, 66 papers were considered.

D. QUANTITATIVE ANALYSIS

This section provides a quantitative overview of the literature selected for the vulnerability review. The analysis covers the distribution of articles by year of publication, type of vulnerability addressed, LLM lifecycle phase involved, and publication venue. These statistics offer insights into research trends, the maturity of the field, and areas that remain underexplored in the context of LLM security.

As illustrated in Figure 2, the number of publications addressing LLM vulnerabilities has increased significantly over time. While only 2 relevant papers were identified in 2021, the count grew steadily, reaching a peak of 31 in 2024. This trend reflects the growing academic interest and urgency around the security and trustworthiness of LLMs as they became more widely adopted. The drop in 2025 may be attributed to the partial coverage of the year at the time of this review.

Figure 3 shows the distribution of selected papers addressing the eight vulnerabilities that were analyzed in depth. The most frequently studied topics include *model misuse* and *hallucination*, which reflect growing concerns over the reliability and potential misapplication of LLMs. Conversely, vulnerabilities such as *denial of service*, *model inversion*, and *dependency risks* have received comparatively less attention in the literature, despite their potential impact. This highlights an imbalance in research focus and supports the need for more comprehensive investigations into underexplored threat vectors.

An interesting point to highlight is that the majority of studies focus on the Inference phase (39 papers), highlighting strong interest in model utilization and optimization after training. Training-related works account for fewer papers (18), likely due to the high computational demands and complexity involved. Deployment is the least studied phase (10 papers), indicating a potential research gap in real-world integration and application of LLMs. Overall, this suggests that current research prioritizes efficient use of pre-trained models over initial training processes and practical deployment challenges.

Table 4 provides a comprehensive overview of the main vulnerabilities affecting LLMs. For each vulnerability, we list selected representative papers and specify whether the treatment in this survey is an in-depth analysis or a brief review.

IV. RELATED WORK

While substantial research has focused on employing Large Language Models (LLMs) to enhance cybersecurity, such as detecting code vulnerabilities [71], automating malware classification [72], and safeguarding information systems [73], comparatively less efforts have been devoted to examining the security vulnerabilities inherent to LLMs themselves. This section provides a comparative overview of major

TABLE 3. Inclusion and Exclusion Criteria for Article Selection.

Inclusion Criteria	Exclusion Criteria
Peer-reviewed journal and conference articles	Technical reports, policy papers, and working papers
Published between 2021 and 2025	Publications prior to 2021
Explicit focus on LLMs	Papers not explicitly addressing LLMs
Substantive contribution to LLM vulnerabilities or security	Papers lacking significant experimental or empirical contributions

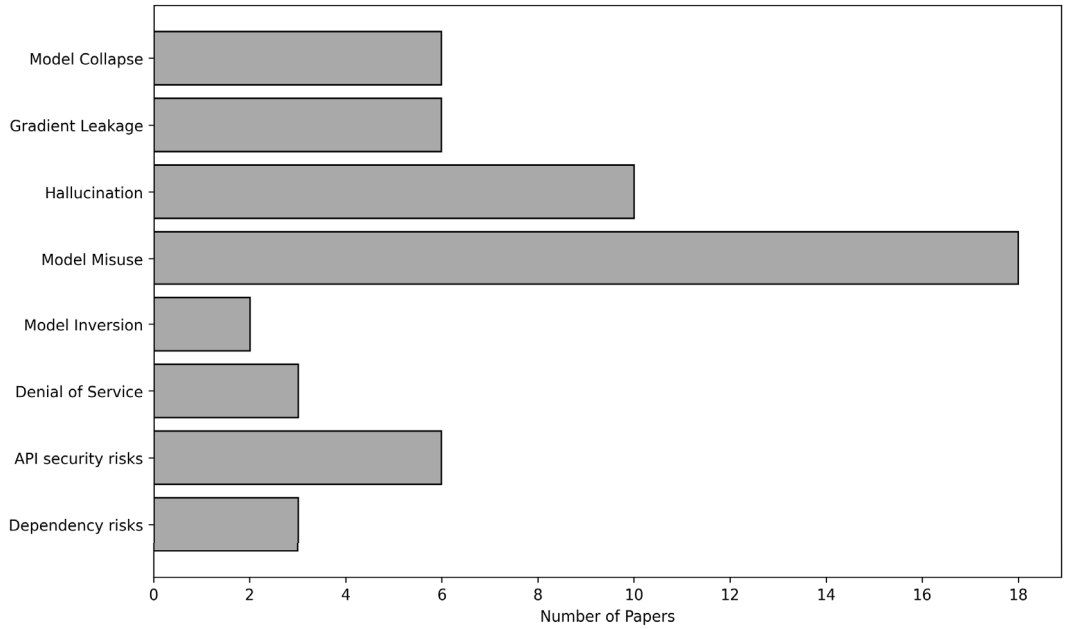


FIGURE 3. Distribution of selected papers by type of vulnerability (in-depth analysis only).

TABLE 4. Summary of vulnerabilities by treatment type and associated papers.

Treatment Type	Vulnerabilities
In-depth review	Model Collapse [5]–[10] Gradient Leakage [11]–[16] Hallucinations [17]–[26] Model Misuse [27]–[44] Model Inversion [45], [46] Denial of Service [47]–[49] API Security Risks [50]–[55] Dependency Risks [56]–[58]
Brief review	Bias and Toxicity [59], [60] Backdoor Attacks [61] Data Poisoning [62] Memorization and Data Leakage [45], [63] Prompt Injection [64] Jailbreak [65], [66] Membership Inference Attacks [67] Model Extraction [68], [69] Privacy Risks [70]

surveys that address LLM-related security and privacy threats. As outlined in Table 5, these surveys differ in scope, taxonomy, and analytical depth.

Yao et al. [74] offer the most comprehensive literature review, analyzing 281 papers across the domains of LLMs, security, and privacy. Their taxonomy distinguishes between AI-inherent threats—such as data poisoning, backdoor insertion, and training data extraction—and non-AI-inherent threats, including remote code execution, supply chain risks, and side-channel attacks. However, their focus on broad security categories limits the depth of analysis for emerging lifecycle vulnerabilities.

Huang et al. [75] propose a classification framework centered around three categories: inherent limitations, intended attacks, and unintended bugs. They also emphasize the role of Verification and Validation (V&V) techniques throughout the LLM development lifecycle. Despite addressing some development-stage risks, their survey offers limited analysis of deployment-stage vulnerabilities and threats specific to modular LLM pipelines, such as Retrieval-Augmented Generation (RAG).

Zhang et al. [76] adopt a goal-based approach to categorize LLM vulnerabilities from both a security and privacy standpoint. Their analysis covers major security threats such as prompt injection, jailbreaking, backdoor attacks, and data poisoning, as well as privacy risks including gradient leakage, membership inference attacks (MIAs), and PII exposure. Notably, their survey also maps each threat to corresponding mitigation techniques, providing a structured framework

for threat identification and defense. However, their work focuses primarily on inference-time risks, offering limited coverage of vulnerabilities in the training and deployment phases.

In another recent literature review, Yan et al. [77] investigate privacy threats targeting LLMs and LLM-based agents, offering a comprehensive taxonomy of risks based on the nature of the adversary's activity. The authors distinguish between privacy leakage—where attackers passively extract sensitive information due to model vulnerabilities—and privacy attacks involving active exploitation. While their work presents a broad landscape of privacy risks, it does not extend to other critical security vulnerabilities such as denial-of-service (DoS), model collapse during retraining, or dependency-related threats in RAG systems.

Table 5 summarizes the threats covered across these works and highlights underexplored areas for future research. While some well-known threats, such as backdoor attacks, data poisoning, and prompt injection, are consistently addressed, other critical issues—like model inversion, denial-of-service, gradient leakage, and dependency-related risks—receive little to no attention. Privacy risks and membership inference attacks (MIAs) are moderately covered, though not uniformly. Furthermore, several emergent vulnerabilities, including model collapse and API-level threats, are not incorporated into existing taxonomies. This fragmented landscape highlights a lack of comprehensive threat modeling and underscores the need for a unified framework that captures the full range of risks affecting LLMs.

In summary, while the cited literature has laid the groundwork for understanding LLM security, it either restricts focus to specific attack vectors or lacks a comprehensive lifecycle-oriented perspective. None of the existing surveys offers a systematic analysis of emerging vulnerabilities across all lifecycle stages, nor do they employ a comparative gap analysis to identify underexplored threats. This work addresses these limitations by proposing a lifecycle-based taxonomy and prioritizing the examination of vulnerabilities overlooked in prior studies.

V. LITERATURE REVIEW

In the following subsections, starting from three defined research questions, a review of the SOTA vulnerabilities in LLMs will be conducted. The vulnerabilities will be categorized based on the phase in which they arise: training, inference, and deployment. Each category includes specific attacks and security issues based on recent studies. By providing concrete examples and assessing their potential impact, we aim to highlight critical areas that require attention in both research and practical applications.

A. RESPONSE TO RQ1: TRAINING PHASE

This subsection addresses Research Question *(RQ1): What vulnerabilities can arise during the training phase of large language models?* The training phase is critical since it involves collecting and processing large-scale datasets and optimizing model parameters, during which vulnerabilities can be introduced intentionally or inadvertently.

1) MODEL COLLAPSE

Model collapse refers to the phenomenon, introduced in [5], where a language model, during continuous fine-tuning or retraining on its own generated data, progressively loses its ability to produce coherent and diverse outputs. As a result, these models, trained on polluted data, develop a distorted understanding of reality. The core threat emerges when large language models are retrained - either directly or indirectly - on outputs generated by other models rather than on high-quality, human-authored data. While this contamination can occur unintentionally, for instance through web-scraping pipelines that fail to distinguish synthetic from real text, adversarial actors could also exploit it intentionally to induce model degradation over time. A practical example of this risk could be found in web-scale data pipelines where publicly available articles, blogs, and forum posts, many of which are now AI-generated, are continuously scraped to create new training datasets. If these pipelines do not include robust filtering mechanisms to exclude synthetic content, future versions of LLMs risk being retrained predominantly on low-diversity, model-generated text, amplifying biases and errors in a recursive feedback loop. The main techniques involved include data poisoning via synthetic content injection and low-quality model proliferation. The long-term impact of model collapse is the progressive erosion of model capabilities, factual grounding, and linguistic diversity, posing systemic risks to the integrity and utility of future generations of LLMs, especially in open-ended or self-improving training pipelines.

Model collapse can be understood as a progressive loss of diversity and factual accuracy in the outputs of an LLM, caused by repeated exposure to its own synthetic data during fine-tuning or retraining cycles. This phenomenon poses new security concerns, especially in scenarios of automated or unsupervised retraining pipelines. In critical systems, collapse could lead to functional degradation, amplify biases, or create opportunities for adversarial exploitation. For example, attackers might accelerate collapse by manipulating fine-tuning datasets or leveraging feedback loops in continuous learning environments.

Authors distinguish between two stages: *early model collapse*, in which the model begins losing information about the tails of the distribution, and *late model collapse*, when the model converges to a distribution different from the original. According to the literature, collapse arises from three types of errors during generations: statistical approximation error, functional expressivity error, and functional approximation error. As a mitigation strategy, existing studies propose coordinated community efforts to share data provenance information. Without such efforts, future LLMs may rely heavily on pre-adoption web data or large-scale human-generated content.

TABLE 5. Coverage of LLMs Vulnerabilities in Recent Literature Reviews.

Ref	Year	Bias and Toxicity	Backdoor	Data Poison.	Mem. and Data Leak.	Model Collapse	Gradient Leakage	Prompt Inj.	Jailbreak	MIA	Halluc.	Misuse	Model Extract.	Model Inv.	DoS	API Risks	Dep. Risks	Privacy Risks
Yao et al. [74]	2024	✓	✓	✓	✓	✗	✗	✓	✓	✓	✗	✗	✓	✗	✓	✗	✓	✓
Huang et al. [75]	2024	✓	✓	✓	✓	✗	✗	✓	✗	✗	✓	✓	✗	✗	✗	✗	✗	✓
Zhang et al. [76]	2025	✗	✓	✓	✗	✗	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✓
Yan et al. [77]	2025	✗	✓	✓	✓	✗	✗	✗	✓	✓	✗	✗	✓	✓	✗	✗	✗	✓
Our Contribution		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Further empirical evidence of this degenerative loop is provided by [6], who shows that reliance on AI-generated content can lead to significant model collapse over time. By fixing model parameters, they observe that even a modest cost advantage for AI-generated data (e.g., 20–50% cheaper than human-generated data) substantially increases the divergence between the public's knowledge and the true data distribution. Measured via Hellinger distance, this divergence grows from 0.09 (no AI discount) to 0.22 and 0.40 with increasing AI reliance. These findings highlight how the widespread use of low-cost AI content can progressively distort knowledge, reinforcing the degenerative feedback loop central to model collapse.

From a more theoretical perspective, [7] introduces a framework linking model collapse to violations of scaling laws, showing how synthetic data can cause skill degradation and performance decay over generations. Empirical results on arithmetic and text tasks confirm that training solely on synthetic data leads to degeneration while mixing in human data helps maintain performance. These results highlight the risk of relying on synthetic data at scale and stress the urgent need to preserve high-quality, human-generated datasets.

A related study [8] further investigates the impact of synthetic data on training dynamics. The authors demonstrate that training using synthesized data leads to reduced diversity and distributional distortions, resulting in a degradation of performance, as also declared in [9]. They argue that the traditional assumption of scaling laws no longer holds in such settings—larger synthetic datasets do not necessarily yield better models. In contrast, accumulating synthetic data alongside real data helps preserve model quality, imposing a finite upper bound on test error, thereby preventing collapse. Different from other applied works, in [10], model collapse is presented as an ethical concern, highlighting how it can progressively distort information over time. To prevent this, the authors emphasize the need for responsible management of both the timing of LLM training and the nature of the training data, ensuring that data sources are not exclusively AI-generated.

Mitigation strategies identified across the literature include monitoring model entropy during retraining, enforcing dataset diversity, introducing verifier-based sample selection strategies that filter high-quality synthetic samples without compromising performance levels comparable to training on clean human data, and ensuring the presence of human-generated content in training pipelines. These approaches aim to break the degenerative feedback loop and preserve the factual grounding and diversity of future model generations. The simplest and most immediate mitigation strategies involve increasing dataset diversity and ensuring the inclusion of human-generated content within training pipelines. Additional support for this task can be provided by efficient generated-text detection strategies, which help guarantee that the training dataset remains balanced.

2) GRADIENT LEAKAGE

Gradient leakage refers to the unintended disclosure of sensitive information included in the training data through gradients exchanged during the training or fine-tuning of a machine learning model. During distributed or collaborative training of LLMs, gradient information is frequently exchanged between computing nodes to synchronize updates to the model's parameters. These gradients encode how the model should adjust its weights in response to input data in order to minimize the loss function. However, when shared without adequate safeguards, gradients may inadvertently expose sensitive attributes of the original training data. This vulnerability, known as *gradient leakage* or *gradient inversion*, arises when an adversary reconstructs inputs or infers private information by exploiting the mathematical relationship between gradients and data samples. Such attacks have been shown to be effective even in settings with minimal prior knowledge, especially in early stages of training when gradients are highly informative. Gradient leakage poses a serious privacy risk in federated learning, multi-tenant training environments, and large-scale LLM pretraining, as it can enable downstream attacks such as model inversion and membership inference. Ultimately, it undermines the confidentiality guarantees of collaborative learning protocols and challenges the safe deployment of large-scale language models in sensitive domains.

A practical scenario involves federated learning environments where multiple clients fine-tune a shared LLM on sensitive domain-specific data, such as medical records or legal documents. In such settings, even when participants do not share raw data, malicious servers or peer clients may exploit exchanged gradients to reconstruct private inputs.

This risk is particularly acute when fine-tuning is performed in outsourced or multi-tenant environments.

Recent research has investigated several gradient inversion techniques in LLMs, with a focus on federated learning as in [12], in which the authors emphasize that unless properly protected, gradients transmitted from clients to servers can leak information about local data. To reinforce this assumption, [11] introduces DAGER, a gradient inversion algorithm capable of exact recovery of input batches across both encoder- and decoder-based architectures, and JiaYing Zheng et al. [13] reveal that methods such as beam search and reverse engineering on gradients, can also help attackers to reconstruct private data with high fidelity. The authors also investigate gradient inversion across different learning settings to underscore a critical limitation of current defenses: across full-model training, transformer-specific vulnerabilities, and PEFT scenarios. LAMP [14] exemplifies this by reconstructing private text through language model priors and hybrid discrete–continuous optimization, achieving strong performance even under challenging conditions such as large batch sizes, gradient noise, and fine-tuned models. Similarly, the APRIL attack [15] highlights how architectural features of transformers, specifically learnable positional embeddings, amplify privacy risks: adversaries can recover training inputs via both closed-form and optimization-based methods, making secure training in distributed settings more difficult. Extending these concerns to parameter-efficient fine-tuning (PEFT), ReCIT [16] demonstrates that even low-dimensional gradient exchanges suffice for near-complete recovery of private data.

To mitigate the risks associated with gradient leakage, several strategies have been proposed. These include training autoencoders on gradient statistics to detect and suppress sensitive information, applying differential privacy mechanisms that inject noise to obscure identifiable patterns, and generating synthetic gradients designed to obfuscate private content. Another mitigation solution presented in [13] for federated learning frameworks consists of enabling local training of input/output blocks, reducing the exposure of raw gradients, and employing a key encryption mechanism that safeguards communications against both server-side inference attacks and malicious peer clients.

In terms of feasibility, differential privacy stands out as the most practical and readily deployable solution; the block-based local training with encryption appears promising in federated learning contexts, whereas autoencoder-based filtering and synthetic gradients remain more experimental and less mature for real-world adoption.

3) OTHER WIDELY DISCUSSED VULNERABILITIES

Bias and Toxicity refer to the generation of outputs by LLMs that reflect harmful stereotypes, offensive content, or unjustified associations related to gender, race, religion, or other sensitive attributes. These behaviors are not necessarily triggered by adversarial inputs but often stem from systemic issues in the data, training procedures, or model architecture. While not always intentional, these outputs can pose serious risks to fairness, inclusivity, and the safe deployment of LLMs in real-world applications. Typical threat models include both unintentional and intentional harm. The attack surface includes open-ended generation tasks such as dialogue, storytelling, question answering, and content summarization, where the model has high generative freedom. Bias and toxicity can emerge from multiple sources: imbalanced or prejudiced training corpora, reinforcement learning that fails to penalize subtle toxicity, or model overgeneralization from biased patterns. Common techniques used to trigger such behaviors include inserting sensitive identity terms into prompts, framing moral dilemmas, or chaining prompts to bypass safety filters. The impact of biased or toxic outputs can be severe, particularly in sensitive contexts [78] such as education, healthcare, recruitment, politics, or legal assistance. Consequences include reputational damage, discrimination, erosion of user trust, unbalanced representation of reality, and potential legal or regulatory liabilities.

Since it has been extensively explored in the literature, this topic is not analysed in depth in this survey. For example, regarding biases, the literature review by Lin et al. [59] provides a comprehensive overview of bias in LLMs. Bias may manifest across various dimensions, including race, ethnicity, religion, gender, political affiliation, and cultural context. The authors also propose a set of benchmarks to measure bias, such as CHBias, RedditBias, WinoBias, WinoQueer, and BIGNEWS, as well as mitigation techniques targeting both data and models. They categorize debiasing methods into three groups: pre-processing techniques applied to data, in-processing methods used during training, and post-processing strategies applied to model outputs. While some debiasing techniques have demonstrated notable effectiveness in mitigating specific biases such as gender, race, and religion, concerns remain regarding their potential side effects on overall language modeling capabilities. Studies (e.g., Meade et al., 2022) indicate that certain debiasing approaches can impact model performance. Furthermore, the inherent noise and limitations of existing bias benchmarks make it challenging to fully evaluate the effectiveness of these methods. There remains a lack of well-developed research explaining how debiasing strategies influence model behavior. Evaluating and understanding the impact of existing debiasing methods on model performance thus represents a promising direction for future work.

On the other hand, regarding toxicity concerns, a recent study by Luong et al. [60] introduces the Thoroughly Engineered Toxicity (TET) dataset, a meticulously crafted benchmark designed to stress-test the safety mechanisms of modern LLMs. It represents a promising starting point for evaluating the potential of LLMs to produce toxic or harmful content.

Backdoor Attacks involve the intentional implantation of hidden behaviors into a model during training. The objective of the attacker is to cause the model to act normally under

standard inputs, while triggering malicious or unintended responses when a specific pattern or trigger is present. Typical threat models include adversaries with control over portions of the training data or fine-tuning process. The attack surface includes instruction-tuning datasets, prompt templates, or reinforcement learning phases. Common techniques involve embedding semantic or syntactic triggers that activate specific outputs once matched. The impact of backdoor attacks is particularly severe in safety-critical or user-facing applications, as they undermine trust in otherwise reliable systems. Liu et al. [61] provide a detailed overview of backdoor threats, distinguishing between training-time and inference-time threats. Training-time threats can be further categorized according to the emergent LLM development processes, namely supervised fine-tuning and human preference alignment. Such attacks require the adversary to gain access to either the training process or the data curation process, resulting in a backdoored model that exhibits compromised behavior when a specific trigger is present. Potential attack vectors include Retrieval-Augmented Generation pipelines, In-Context Learning, or model editing. The authors also discuss several mitigation strategies for training-time attacks, such as full-parameter fine-tuning, parameter-efficient fine-tuning, and weight merging. For inference-time threats, they suggest techniques like Detect-and-Discard and In-Context Demonstration.

Data Poisoning refers to the manipulation or injection of malicious data into a model's training set with the goal of degrading its performance, introducing targeted behaviors, or weakening its generalization capabilities. In typical threat models, attackers operate during data collection or pre-training phases, particularly in open-source or weakly supervised pipelines. The attack surface spans pretraining corpora, fine-tuning data, or RLHF feedback. Techniques include label flipping, trigger-based poisoning, and long-range perturbations. Poisoning can lead to biased or insecure behavior, reduced robustness, or facilitate future attacks such as model extraction or prompt hijacking. Fendley et al. [62] propose a comprehensive taxonomy and benchmarking framework for data poisoning attacks in LLMs, dividing the attack specifications into four components: the poison set, trigger function, poison behavior, and deployment. The poison set refers to the data points selected by the attacker to carry out the attack. The trigger function modifies these data points to serve as a ''trigger'' for the poison behavior. Poison behavior denotes the change in the model's output that the attacker intends to induce, while deployment specifies whether the attack targets the model or the data and whether it employs an identity trigger. The authors also develop a review of 65 studies on LLM poisoning, highlighting that the field requires close monitoring due to the increasing number of publications in this area.

Memorization and Data Leakage occur when LLMs retain and emite sensitive, private, or proprietary information seen during training. This may happen unintentionally, even when training data is not explicitly memorized in full. Threat models include both benign users triggering leaked content through normal use and adversaries crafting specific queries to extract memorized data. The attack surface includes next-token prediction interfaces, embedding APIs, or logit sampling endpoints. Memorization presents a privacy risk in models trained on web-scraped, uncurated, or user-generated content, particularly when users are unaware that their data could later be exposed. Carlini et al. [45] demonstrate large-scale data leakage from GPT models, highlighting these risks in commercial LLMs. An extensive study in [63] show that LLMs memorize and emit training data, even in open-source settings. Using Carlini et al.'s attack method, sequences of at least 50 tokens exactly matching training data were extracted. Results reveal significant differences between models: for example, Mistral 7B memorizes over 10 times less than Falcon 7B, while GPT-3.5-Turbo leaks the most, with 0.85% of generated tokens coming from training data. Over-trained models tend to leak more sensitive information.

B. RESPONSE TO RQ2: INFERENCE PHASE

This section responds to Research Question *(RQ2): What vulnerabilities can affect large language models during the inference phase?* The inference phase, where the trained model interacts with user inputs to generate outputs, is a critical stage where attacks such as prompt injections and model extraction can compromise system integrity and privacy.

Focusing on vulnerabilities less covered in the literature, this subsection highlights emerging threats and novel attack vectors that require additional attention.

1) HALLUCINATIONS

Hallucination refers to the generation of outputs by LLMs that are factually incorrect, fabricated, or not grounded in the input. While not always malicious, hallucinations can undermine the integrity and reliability of LLM-based applications, especially when outputs are consumed without validation. Typical threat models include users relying on false outputs in critical domains, or adversaries exploiting hallucinations to spread misinformation. The attack surface spans tasks like summarization, question answering, and retrieval-augmented generation (RAG), where unverified outputs may influence downstream decisions. However, hallucinations are increasingly recognized not merely as stylistic flaws but as exploitable vulnerabilities in LLM-based applications. Surveys such as [17] provide a structured view of their causes, detection methods, and mitigation strategies, while complementary taxonomies (e.g., [18]) emphasize the diverse manifestations of hallucination, from factual to logical inconsistencies, each carrying distinct risks in sensitive domains such as cybersecurity. Empirical studies reinforce their prevalence: Rawte et al. [19] introduce the Hallucination Vulnerability Index (HVI) and show that earlier models like GPT-3 and StableLM exhibit far higher rates of hallucinated content than newer systems such as GPT-4, with fabricated quotes and fictitious entities representing

particularly harmful forms. At the system level, Yu et al. [20] demonstrate through ReEval that adversarial perturbations of retrieved evidence can reliably trigger hallucinations in RAG pipelines, while HaluEval [21] provides large-scale benchmarks for cross-model evaluation. Mitigation efforts converge on approaches spanning black-box token entropy filtering and gray-box factual entailment checks [19], structured knowledge integration via knowledge graphs [22], [23], contrastive learning for more faithful retrieval [25], dialogue-specific frameworks such as the Neural Path Hunter (NPH), which improves hallucination reduction by optimizing retrieval consistency [24], and architecture-level adjustments such as patch-based image encoding for VLP models [26]. The mitigation strategies detailed above involve trade-offs between effectiveness, coverage, and complexity. Black-box methods like token entropy filtering are easy to deploy but only partially effective, while gray-box checks and structured knowledge integration improve accuracy at the cost of computational overhead and knowledge maintenance. Task or architecture-specific solutions, such as Neural Path Hunter or patch-based VLP encoding, provide targeted improvements but require model redesign or retraining. These trade-offs highlight the need for multi-layered defenses that balance practicality and robustness across tasks and modalities.

2) MODEL MISUSE

Model misuse refers to the intentional use of large language models (LLMs) in ways that diverge from their intended applications, leading to potential harm or ethical violations. The objective of this threat is not to compromise the model itself, but rather to exploit its capabilities to generate malicious, misleading, or harmful outputs. Typical threat models include both external users interacting with the model through public interfaces and internal actors with broader system access. The attack surface primarily involves input manipulation, prompt engineering, or circumvention of usage policies. Techniques often include adversarial prompting to elicit restricted outputs or leveraging the model to automate unethical tasks. The impact of model misuse can be severe, ranging from the spread of misinformation and biased decision-making to the facilitation of illegal activities.

LLMs can be misused to support social engineering attacks by generating highly convincing phishing messages or impersonations. They can also automate harmful tasks such as malware scripting, network reconnaissance, and fake data creation. Additionally, their ability to produce realistic text makes them ideal tools for generating fake news, manipulating content, and spreading misinformation at scale [79]. Following a detailed list of applications:

- *Harmful content*
  Authors in [27] demonstrate that aligned LLMs can be manipulated via automatically generated adversarial prompts to produce harmful content, despite built-in safety mechanisms. Using their attack method, the authors achieved high success rates in eliciting toxic behaviors, such as racist jokes and bigoted statements, from both open-source and commercial models. Their benchmark (AdvBench) includes hundreds of harmful strings, some explicitly involving racial or ethnic insults, and instructions to generate discriminatory or offensive content. The best-performing attack (GCG) triggered these outputs in up to 88% of cases on Vicuna-7B and showed transferability to proprietary models, with GPT-3.5 reaching an 86.6% success rate when ensemble techniques were used.
- *Disinformation*
  LLMs can generate both unintentional and intentional dis- and mis-information. Unintentional misinformation sometimes results from hallucinations [28], while intentional disinformation arises when adversarial users manipulate models to spread fake news, propaganda, and rumors. Such AI-generated falsehoods pose unique challenges, as they are difficult to detect by both humans and automated systems. Studies demonstrate that AI-generated disinformation is highly convincing. Reference [29] assesses 10 LLMs across 20 narratives and reveals that current models produce deceptive content that even automated detection systems struggle to identify.
  Beyond text-based misinformation, [30] examines how LLMs facilitate multi-modal disinformation, generating misleading images, videos, and audio to manipulate audiences at scale. The societal impact of AI-driven misinformation is particularly alarming in sensitive domains such as healthcare, COVID-19, and political discourse, where the spread of falsehoods can have severe consequences [31].
- *Hate speech*
  [32] reveals that Text-to-Image models, particularly Stable Diffusion, can be misused to generate hateful meme variants targeting specific individuals or communities. By combining these models with editing techniques like DreamBooth, Textual Inversion, and SDEdit, adversaries can produce images that fuse recognizable hate symbols (e.g., Happy Merchant, Pepe the Frog) with targeted attributes (e.g., nationality, political affiliation). In experiments, 24% of DreamBooth-generated images were successfully identified as hateful variants, comparable in quality and recognizability to real-world memes.
- *Propaganda*
  When an LLM produces hallucinated content, it may be used to promote discriminatory, polarizing, or violent agendas, potentially influencing the decision-making process [33]. Authors in [34] tested the persuasiveness of GPT-3-generated propaganda compared to real foreign disinformation campaigns (from Russia and Iran) in a survey of over 8,000 U.S. adults. GPT-3 articles increased agreement with propaganda theses by 19 percentage points—nearly matching the 23-point boost from authentic propaganda. When outputs were

curated or prompts refined by fluent English speakers, the AI-generated content became as persuasive as the originals.

- *Cybercrime*
  Recent reports[1,2] highlight the emergence of malicious LLM applications (*Malla*) in underground marketplaces. These tools empower adversaries to conduct sophisticated cyberattacks with minimal technical expertise, significantly increasing the overall threat level [35].
  In this sense, works in [36] and [37] detail how cybercriminals exploit GenAI tools to develop attack payloads, automate hacking, malicious software, and generate polymorphic malware. Tools based on models like GPT-3, such as code assistants, may inadvertently assist in writing or refining harmful code.
- *Social Engineering*
  In phishing, attackers leverage LLMs to craft convincing emails that mimic trusted entities, making spear-phishing campaigns highly effective [38]. These models can assist in the reconnaissance phase of an attack, helping cybercriminals gather personal information to refine phishing messages. Additionally, basic prompt engineering can bypass LLM safety mechanisms, enabling attackers to obtain guidance on spear-phishing tactics and social engineering strategies. Mink et al. [39] examine how deepfake personas influence trust on social media, finding that 43% of users connect with deepfake profiles despite awareness efforts. AI-driven deepfakes facilitate social engineering attacks by constructing believable identities that exploit user trust, a tactic that extends to spear phishing, where deepfake technology creates realistic impersonations to enhance deception strategies [36].

The most widely adopted mitigation strategy for model misuse is the implementation of LLM guardrails. A guardrail is an algorithm that takes a set of objects as input and determines whether, and how, enforcement actions can be applied to reduce the associated risks. These guardrails can be used to validate input data and block malicious queries to prevent model misuse, or to monitor and filter the model's outputs to remove inappropriate or harmful content [40]. Notable examples of guardrails include Llama Guard [41], Nvidia NeMo Guardrails [42], and the OpenAI Moderation API [43]. A more recent approach, LoRA-Guard [44], leverages language features extracted from LLMs and adapts them for content moderation using low-rank adapters in a dual-path design. Nevertheless, despite these advances, model misuse remains an urgent problem that has not yet been fully mitigated. In fact, some models without guardrails, called Dolphins (e.g., the fine-tuned Mixstral[3] by Eric Hartford, the dolphins' creator), are also emerging, designed to generate outputs based on the input without any filtering.

3) MODEL INVERSION

Model inversion is an attack that aims to extract information memorized by a Large Language Model during pretraining, thereby causing leakage of its training data. Model inversion attacks target the internal representations of models to infer sensitive attributes or reconstruct inputs from their outputs. In the context of LLMs, this involves exploiting the relationship between training data and learned parameters, with the goal of recovering textual content that was part of the original training corpus. Typically, an adversary issues carefully crafted queries to prompt the model into revealing memorized or statistically biased content. While early inversion attacks focused on classification models or tabular data, recent research has adapted these techniques to generative models. In LLMs, the risk is heightened by their autoregressive generation and tendency to reproduce verbatim text fragments under certain prompting conditions. Attackers may operate with black-box access to the model or possess limited internal visibility (e.g., gradient or embedding access), making inversion feasible even without full model transparency. The consequences of such attacks are particularly serious when they result in the disclosure of personal, confidential, or copyrighted information, raising both ethical and legal concerns about data provenance and the unintended memorization behaviors of large-scale generative models.

Model inversion attack was first introduced by Fredrikson et al. in statistical models [80] and then generalized to deep neural networks [81]. Early work in this area focused primarily on image data, using either optimization-based methods or generative models. However, extending these techniques to language models is significantly more challenging, due to the discrete and combinatorial nature of text.

Despite the widespread attention that this type of attack has received in the broader machine learning literature, relatively few studies have investigated its implications in the context of large language models. A prominent line of work has demonstrated that large language models are prone to memorizing and leaking specific training examples, even under black-box access. Carlini et al. [45] present a simple yet effective method for extracting verbatim sequences from a model's training data without requiring internal access to the model's parameters. Their attack is based on generating a large number of high-likelihood samples using general-purpose sampling strategies and then ranking these samples based on likelihood metrics derived from a separate reference model. Despite training examples not

[1] https://slashnext.com/blog/wormgpt-the-generative-ai-tool-cybercriminals-are-using-to-launch-business-email-compromise-attacks/WormGPT – The Generative AI Tool Cybercriminals Are Using to Launch Business Email Compromise Attacks

[2] https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/wormgpt-and-fraudgpt-the-rise-of-malicious-llms/WormGPT and FraudGPT – The Rise of Malicious LLMs

[3] https://erichartford.com/dolphin-25-mixtral-8 × 7bdolphin-25-mixtral-8 × 7b

having consistently lower loss values than test data, the authors show that certain outlier sequences are memorized and thus vulnerable to extraction. Applied to GPT-2, their approach yielded over 600 verbatim training samples from a set of 1,800 candidates, some of which contained highly sensitive personal information. The study further explores factors influencing memorization, such as model size and token frequency, and provides mitigation strategies, including differential privacy and training data deduplication. Similarly, Gu et al. [46] propose a sentence inference attack, a decoder-based method targeting sentence embeddings to reconstruct original inputs with minimal prior knowledge. This attack leverages semantic cues to recover full sentences and proves effective even without domain-specific insight. Compared to prior keyword-based inference techniques, this approach offers greater flexibility and broader applicability, though the fidelity of reconstructed outputs varies depending on the instance. Proposed mitigations include differential privacy, which offers strong theoretical guarantees but often comes at the cost of reduced model utility, and training data deduplication, which is relatively straightforward to implement and effective against verbatim memorization. For embedding-based leakage, representation regularization and access control mechanisms (e.g., restricting the sharing of raw embeddings) have been suggested, though their adoption remains limited in practice. Overall, while technical solutions exist, only a few (e.g., deduplication and restricted access) are readily deployable at scale, whereas stronger privacy-preserving methods still face significant trade-offs between security and model performance.

4) DENIAL OF SERVICE

Denial of Service (DoS) attacks against LLMs represent an emerging and underexplored class of threats, distinct from traditional network-level DoS. In this context, attackers target the computational cost of inference rather than saturating network bandwidth. A prominent subclass of such attacks, known as sponge attacks [47], deliberately crafts inputs that are disproportionately expensive to process, exploiting the quadratic complexity of self-attention in transformer architectures. These inputs may involve adversarial looping, deeply nested structures, or specially designed prompt patterns that trigger extensive computation during token generation. Attackers, often anonymous users leveraging public APIs, can flood the system with such inputs to degrade service quality, increase latency, or even force resource exhaustion. The impact ranges from degraded responsiveness to partial or full denial of service, and is particularly critical in production-level LLM deployments. Moreover, these attacks may serve as precursors to broader exploitation campaigns, including evasion of usage policies or masking other concurrent attacks. A variation of this approach uses control tokens like $< |endoftext| >$ to prematurely terminate generation and mute the model output [48]. These techniques pose serious availability risks, yet they are rarely addressed experimentally in the literature. A notable exception is [49], which presents the first empirical demonstration of a DoS attack on proprietary LLMs like GPT-4o and GPT-4o mini. The study shows that a single poisoned input can cause significant disruption, with the total attack cost being less than $1 via the OpenAI API. This underscores how economically feasible and impactful such attacks can be, even at small scale.

Despite these findings, the field remains largely open, with no relevant works offering experimental validation or mitigation strategies.

5) OTHER WIDELY DISCUSSED VULNERABILITIES

Prompt injection is a class of attacks in which adversaries craft inputs that manipulate the behavior of the model by overriding, corrupting, or hijacking system instructions. These attacks are especially dangerous in LLM-powered applications where user input is incorporated into prompts passed to the model. Threat models involve users with direct access to the prompt-building interface, such as in chatbots, RAG pipelines, or autonomous agents. Attack surfaces include the prompt construction layer, embedded instructions, and dynamic context windows. Techniques include instruction override, delimiter injection, and indirect prompt pollution. Prompt injection can cause models to ignore original constraints, leak confidential information, or perform unauthorized actions. Liu et al. [64] make a significant contribution by moving beyond the anecdotal case studies that previously dominated the field of prompt injection research. Its contribution is the introduction of a formal framework to systematize the analysis of these attacks, providing a formal definition of a prompt injection attack. This framework allows for a more structured approach to designing new attacks, such as the novel "Combined Attack" created by the authors, which was found to be consistently effective and to outperform existing attacks. The work also establishes a comprehensive benchmark for quantitatively evaluating attacks and defenses by testing five attacks and ten defenses across ten different LLMs and seven distinct tasks. A key finding from this large-scale evaluation is that no existing defense is sufficient, as they either have limited effectiveness in preventing attacks or cause significant utility loss for the targeted tasks.

Jailbreak attacks aim to circumvent the safety mechanisms or content restrictions embedded in aligned LLMs. By carefully crafting adversarial inputs, attackers can prompt the model to generate toxic, harmful, or restricted outputs. Threat models include external users seeking unrestricted access or adversarial developers testing model boundaries. The attack surface involves input manipulation through repetition, synonymization, or token-level perturbation.

Xu et al. [65] conduct a systematic study on jailbreak attacks and defenses for LLMs, providing one of the most comprehensive empirical comparisons currently available. The authors evaluate nine attack methods and seven defense mechanisms across different models (LLaMA-2-7B, Vicuna-v1.5-7B, and GPT-3.5-Turbo), using a benchmark that

combines automatically and manually annotated malicious queries. Their findings highlight that template-based jailbreaks can be particularly effective, sometimes outperforming more complex white-box approaches, and that the use of special tokens (e.g., instruction delimiters) significantly affects attack success rates. On the defense side, they show that existing techniques often struggle to balance robustness with usability: while methods such as Bergeron achieve the best trade-off, most defenses either fail to prevent attacks consistently or impose high computational overheads. The study further introduces clear evaluation metrics (e.g., Attack Success Rate, Defense Passing Rate, Benign Success Rate), which provide a standardized basis for assessing both attacks and defenses. Another defense strategy is proposed by Fenza et al. [66], introducing an effective anti-jailbreaking filter for LLMs, designed to detect and defuse persuasive prompts that elicit inappropriate outputs. The system trains a hidden persuasion detector enhanced by xAI techniques (SHAP, LIME) that identify key words. Eliminating those keywords identified, the filter successfully neutralizes most jailbreak attempts.

Membership Inference Attacks (MIA) attempt to determine whether a specific example was part of the model's training data. In the context of LLMs, MIAs pose a serious privacy risk, especially when the training data includes personal or proprietary information. Threat models assume black-box or grey-box access to the model, with attackers submitting candidate inputs and analyzing output confidence, perplexity, or gradient signatures. The attack surface typically involves probabilistic outputs or token likelihoods exposed via APIs. Successful MIAs can reveal sensitive data inclusion, leading to violations of privacy or intellectual property. Hengyu et al. [67] conduct the first systematic review of recent studies on MIA targeting LLMs and large multimodal models (LMMs). They categorize the attacks according to their methodologies and threat scenarios, and discuss the main limitations of existing research.

Model Extraction, also referred to as *model stealing* or *model theft*,[4] denotes a class of attacks in which an adversary attempts to replicate the behavior of a LLM by interacting with it through a public API. The attacker, typically operating in a black-box setting, issues a large number of crafted input prompts and collects the corresponding outputs, either full generations or log-probabilities, to train a local surrogate model. In the context of LLMs, the goal is not necessarily to recover the exact parameters, but to approximate the linguistic capabilities, knowledge base, or even fine-tuned alignment behavior of the original model. Extraction techniques may range from simple dataset-querying strategies to advanced prompt engineering and distillation pipelines. The main threat lies in the unauthorized reproduction of proprietary models, which can lead to violations of intellectual property rights, circumvention of usage restrictions (e.g., safety filters), and facilitation of further attacks, such as jailbreak transfers or downstream misuse. Birch et al. [68] introduce Model Leeching, an attack that distills task knowledge from LLMs using a prompt generation system. This approach enables the adversary to create synthetic task-aligned datasets, which are then used to train a high-fidelity replica of the target model. Also, Carlini et al. [69] demonstrate that model extraction can extend beyond functionality to architecture. They show that even with standard API access, it is possible to reconstruct the embedding projection layer of transformer-based LLMs. For less than $20, their attack extracted the entire projection matrix of OpenAI's Ada and Babbage models, and they estimate that recovering the full matrix of GPT-3.5-Turbo would cost under $2,000. These findings reveal that sensitive structural details can be leaked even under black-box conditions.

[4]https://genai.owasp.org/llmrisk2023-24/llm10-model-theft/OWASP Top 10 for LLM Applications: LLM10

C. RESPONSE TO RQ3: DEPLOYMENT PHASES

This section explores Research Question *(RQ3): What risks and vulnerabilities can emerge during the deployment of large language models?* Deployment includes the integration of LLMs into real-world applications and infrastructures, introducing unique challenges such as API and RAG-dependency risks.

1) API SECURITY RISKS

API Security Risks refer to the vulnerabilities introduced when large language models (LLMs) are exposed through application programming interfaces (APIs), enabling remote interaction with the model's capabilities. The objective of adversaries in this threat category is not necessarily to compromise the underlying model weights, but to exploit the interface layer to bypass restrictions, access unauthorized information, or manipulate model behavior at scale. Typical threat models include both authenticated and unauthenticated users interacting with the model via public APIs, partner integrations, or internal service endpoints. Adversaries may range from benign users attempting to circumvent content filters to malicious actors aiming to extract sensitive information, abuse computational resources, or orchestrate large-scale misuse campaigns. The attack surface primarily involves exposed endpoints and associated parameters, including query inputs, system prompts, user context, and configuration metadata. APIs may also expose auxiliary functions such as embedding generation, log probabilities, or fine-tuning interfaces, further expanding the risk profile. Common attack techniques include prompt extraction through repeated queries, safety filter evasion via input manipulation, abuse of high-rate API access for model inversion or automated misuse, and exploitation of insufficient authentication or quota controls. Side-channel attacks may also target latency, token usage, or error patterns to infer protected information.

Several API-level threats have also been formally recognized in the OWASP Top 10 for LLM Applications,

particularly under LLM07: Insecure Plugin Design.[5] This category highlights the risks introduced by third-party extensions or integrations that interact with LLM APIs without proper isolation, validation, or access control, potentially exposing the system to prompt injection, data leakage, or unauthorized actions.

Recent research further demonstrates the concrete impact of API exposure. Carlini et al. [50] show how model extraction attacks can replicate proprietary models, proposing watermarking frameworks in the decision layers to detect unauthorized use. Liu et al. [51] formalize prompt injection as a threat model, showing that malicious inputs to public endpoints can manipulate model behavior and cause policy violations. This is corroborated by Greshake et al. [52], who demonstrate real-world exploits in Bing Chat and code completion tools, leading to arbitrary code execution and unauthorized API calls. The risks are exacerbated by orchestration frameworks such as LangChain, where Liu et al. [53] identify 20 vulnerabilities enabling RCE through malicious prompts. Similarly, Pedro et al. [54] uncover Prompt-to-SQL injection in LLM-integrated web applications, where natural language queries were converted into executable database statements without sanitization. At the systems level, Sun et al. [55] highlight that LLM-mediated applications can be manipulated not only by external attackers but also by malicious developers, leading to data poisoning, biased recommendations, and toxic responses. They propose four key safety properties (integrity, source identification, attack detectability, and utility preservation) as guiding principles for designing secure LLM applications. However, the most readily deployable mitigation measures include rate limiting, strong authentication, and systematic logging to constrain and monitor API interactions. More advanced strategies encompass watermarking and plugin sandboxing, which provide additional protection against sophisticated misuse and extraction attempts. While basic mitigations can be rapidly integrated into existing infrastructures, advanced techniques generally offer stronger guarantees but require greater engineering effort and careful system integration, highlighting the trade-off between ease of adoption and robustness.

2) DEPENDENCY RISKS

Dependency Risks refer to the vulnerabilities introduced when LLM applications depend on external tools, libraries, or services to extend their capabilities. These risks are especially critical in retrieval-augmented generation (RAG) systems, where the model interacts with external data sources via orchestration frameworks or plugins. In this context, the objective of an attacker is not to compromise the LLM itself, but to exploit weaknesses in the surrounding components (e.g., insecure data connectors, vulnerable libraries, or malicious third-party integrations). Typical threat models include poisoned retrieval results, compromised indexing pipelines, and adversarial documents injected into external data sources. The attack surface encompasses the retrieval layer, plugin interfaces, and any dependency mediating between the model and external content. Common techniques involve manipulating retrieved passages, injecting payloads into document stores, or exploiting insecure execution environments provided by orchestration frameworks (e.g., LangChain, LlamaIndex). When exploited, these vulnerabilities can result in model misbehavior, prompt injection, data exfiltration, or even remote code execution. As RAG pipelines become a standard architectural pattern in enterprise LLM deployments, securing their dependencies is essential to ensure both system reliability and trustworthiness.

Recent work consistently shows that Retrieval-Augmented Generation (RAG) pipelines expose critical attack surfaces that adversaries can exploit through data poisoning and dependency manipulation. Peng et al. [56] highlight how fine-tuned LLMs are vulnerable to backdoor attacks, where poisoned training data can embed triggers that later induce sensitive document leakage. Complementing this training-level perspective, Cho et al. [57] introduce GARAG, a black-box attack that perturbs retrieved documents via genetic algorithms, demonstrating that adversarially crafted inputs can systematically degrade retrieval and grounding accuracy. At the system level, Zhang et al. [58] emphasize the risks inherent in modular RAG frameworks such as LangChain: components like parsers and prompt templates can be compromised with adversarially embedded sequences, often obfuscated in formats invisible to human reviewers yet parsed by LLMs. Together, these studies converge on the conclusion that RAG systems are highly sensitive to poisoning and adversarial manipulation, whether introduced at the training, retrieval, or orchestration layer. The attacks also share a high degree of robustness and transferability across models and configurations, with reported success rates often exceeding 70–80%. The main difference lies in the attack vectors: Peng et al. [56] focus on fine-tuning vulnerabilities, Cho et al. [57] exploit iterative document perturbation, while Zhang et al. [58] expose structural weaknesses in modular frameworks. This diversity suggests that securing RAG pipelines requires defenses spanning the entire lifecycle, from training data sanitization to robust retrieval validation and secure framework design.

3) OTHER WIDELY DISCUSSED VULNERABILITIES

Privacy risks in LLMs encompass a broad spectrum of concerns, including data leakage, re-identification, and misuse of personal information. These risks may arise from memorized content, improper logging, inference leakage, or malicious usage scenarios. Threat models range from external attackers extracting private data to developers unintentionally exposing logs or context. Attack surfaces include input history, output tokens, embedding layers, and fine-tuning data. Privacy risks are magnified in real-time deployments, where LLMs may inadvertently reveal user information or violate regulatory

[5] https://genai.owasp.org/llmrisk2023-24/llm07-insecure-plugin-design/OWASP Top 10 for LLM Applications: LLM07

TABLE 6. Summary of LLM emerging vulnerabilities: Phase, Description, and Mitigation Techniques.

Vulnerability	Lifecycle Phase	Description	Mitigation Techniques
Model Collapse	Training	Degradation of model quality due to retraining on synthetic/self-generated data.	Filter synthetic data, enforce data provenance, mix with human content, entropy monitoring, verifier-based sample selection, data accumulation strategy.
Gradient Leakage	Training	Sensitive input data can be reconstructed from shared gradients during distributed or federated learning.	Differential privacy, noise injection, gradient clipping, local training, encryption, synthetic gradients.
Hallucinations	Inference	Model generates outputs that are false, fabricated, or not grounded in the input.	Entropy-based token filtering, NLI-based validation, knowledge graph grounding, contrastive learning, benchmarks.
Model Misuse	Inference	LLMs used for malicious purposes such as phishing, malware, disinformation, or propaganda.	LLMs guardrails.
Model Inversion	Inference	Recovery of sensitive training data by analyzing outputs or internal settings.	Differential privacy, deduplication.
Denial of Service (DoS)	Inference	Adversarial prompts increase computational cost to degrade or stop model performance.	N/A
API Security Risks	Deployment	Vulnerabilities of LLMs that are exposed publicly via APIs or plugins.	Rate limiting, authentication, logging, watermarking, plugin sandboxing.
Dependency Risks (RAG)	Deployment	Vulnerabilities that depend on external tools, libraries, or services.	N/A

constraints (e.g., GDPR). A comprehensive review of privacy risks in LLMs is provided by Kibriya et al. [70]. The authors investigate privacy issues arising during both training and inference phases. During the training phase, if the training data contains sensitive information, the model's responses may unintentionally reveal it. This underscores the importance of strong anonymization techniques to prevent sensitive information from being embedded and resurfacing in the model's outputs. Ethical considerations also extend to the protection of Intellectual Property Rights (IPR) in the context of LLMs. During the inference phase, the LLM applies the knowledge and patterns it has learned to understand and generate text based on user input. Notably, LLMs may also memorize information during inference: they can unintentionally retain specific patterns, phrases, or data from both training and user interactions. Consequently, the model might reproduce these inputs in its generated responses, potentially disclosing sensitive information or reproducing biased or undesirable content. The risk is particularly pronounced when models generate outputs closely resembling previously seen inputs without proper generalization. In this way, prompts containing private information can inadvertently lead to the leakage of personally identifiable information (PII) or proprietary data. Kibriya et al. also discuss possible privacy-preserving solutions for LLMs, including Differential Privacy, Homomorphic Encryption, Data Minimization, and Federated Learning, analyzing the trade-offs of each approach. Differential privacy ensures confidentiality by introducing noise, but it may reduce data utility; homomorphic encryption enables secure computations on encrypted data, though at the cost of significant computational overhead; data minimization reduces data exposure, but risks sacrificing valuable information; finally, federated learning keeps data on local devices to enhance privacy, but introduces synchronization complexity and potential performance variations. The review further includes case studies illustrating real-world privacy issues in LLM deployment, such as: Gemini AI privacy challenges with human handling and three-year data retention; legal actions against ChatGPT and Meta over data privacy concerns in Italy; a ChatGPT data breach exposing chat logs and payment information; and Copilot's access to Microsoft 365 users' activity. These examples highlight the practical implications and ongoing challenges of privacy protection in large language models.

VI. DISCUSSION AND CHALLENGES

This survey, guided by a lifecycle-oriented taxonomy for analyzing vulnerabilities in Large Language Models (LLMs), covers threats that may arise during training, inference, and deployment. A synthesis of the in-depth analyzed vulnerabilities and corresponding mitigation strategies presented in literature is reported in Table 6. By focusing on underrepresented vulnerabilities, it has highlighted areas that remain less explored in existing literature. Quantitative analysis of recent publications confirms a strong concentration of research on inference-time attacks, while the training and deployment phases receive comparatively limited attention.

Moreover, many real-world vulnerabilities, in particularly those emerging from the interaction between LLMs and external tools, lack standardized evaluation or mitigation strategies.

A key insight emerging from this review is the interconnectedness of vulnerabilities across the LLM lifecycle. For instance, training-time weaknesses such as gradient leakage can directly enable inference-phase attacks like model inversion or membership inference. Similarly, hallucination and misuse during inference can become entry points for downstream security breaches at the deployment stage, particularly in systems that rely on LLM outputs for decision-making. This interplay suggests that vulnerabilities should not be studied in isolation; instead, end-to-end threat modeling is necessary to capture their cascading impact across all stages.

Despite the growing volume of research on LLM security, several critical areas remain underexplored:

- Deployment-phase vulnerabilities: Few works address threats related to API security, plugin architecture, or orchestration frameworks such as LangChain.
- RAG-specific risks: Retrieval-Augmented Generation introduces new attack surfaces (e.g., document poisoning), yet standardized defenses are lacking.
- Model collapse dynamics: While recognized as a systemic risk, its long-term effects and early detection strategies remain poorly understood.
- Hybrid threats: Some emerging vulnerabilities (e.g., DoS) do not fit into existing taxonomies and require new analytical frameworks.

The current landscape of LLM vulnerability research is obstructed by several methodological challenges, for example, the lack of standardized benchmarks for evaluating and comparing vulnerabilities across models and settings, or the absence of real-word attack datasets. In reality, there is also an ambiguity in the literature in the definition of LLMs' vulnerabilities. The boundary between performance flaws (e.g., hallucinations) and security threats (e.g., information leakage) remains fuzzy. So, despite the growing academic and industrial interest in LLM vulnerabilities, the current research landscape is fragmented, heterogeneous, and often inconsistent in terminology, threat modeling, and evaluation methodologies. Many works propose taxonomies or mitigation strategies without a shared foundation, leading to duplicated efforts, incompatible assumptions, and difficulties in comparing results across studies. This confusion is especially evident when dealing with overlapping concepts such as data leakage, memorization, inversion, or privacy loss—terms that are often used interchangeably despite referring to distinct phenomena. Similarly, distinctions between robustness, safety, and security are frequently indistinct, further complicating efforts to consolidate knowledge.

To address these challenges and support a more coherent and impactful research agenda, four key future directions are outlined:

- Consolidate unified definitions and taxonomies: there is a pressing need for a standardized vocabulary to describe LLM vulnerabilities. Definitions of concepts should be formalized to avoid ambiguity and to enable precise threat modeling.
- Standardized evaluation and benchmarking: the current landscape of LLM security research is fragmented, with different studies employing varying metrics and evaluation methodologies. A critical future direction is the development of standardized benchmarks that enable a fair and direct comparison of different mitigation strategies. These benchmarks should not only assess a model's robustness against individual threats but also its resilience across the entire LLM lifecycle, from pre-training to deployment.
- Quantifying security-utility trade-offs: our analysis highlights that many mitigation strategies, while effective against specific vulnerabilities, often introduce performance trade-offs (e.g., loss of creativity or a reduction in model utility). Future research should focus on quantifying these trade-offs to better understand the balance between model security and performance. This could involve developing frameworks or predictive models to help practitioners select or design mitigation strategies that optimally balance security with the intended application of the LLM.
- Verifiability and provenance in LLM pipelines: as model collapse and data contamination become systemic risks, the community must prioritize provenance tracking and sample verifiability. This includes tools to detect AI-generated training data, trace content origin, and audit fine-tuning histories.

In summary, the field of LLM vulnerability research is rapidly evolving but remains conceptually fragmented and methodologically immature. A concerted effort is needed to unify definitions, create shared resources, and design interventions that account for the complexity and scale of real-world LLM deployments.

REFERENCES

[1] T. T. Nguyen, H. T. T. Vu, and H. N. Nguyen, "Security, privacy, and ethical challenges of artificial intelligence in large language model scope: A comprehensive survey," in *Proc. 1st Int. Conf. Cryptogr. Inf. Secur. (VCRIS)*, Dec. 2024, pp. 1–6.

[2] A. Esmradi, D. W. Yip, and C. F. Chan, "A comprehensive survey of attack techniques, implementation, and mitigation strategies in large language models," in *Proc. Int. Conf. Ubiquitous Secur.*, 2023, pp. 76–95.

[3] M. A. Rahman, L. Alqahtani, A. Albooq, and A. Ainousah, "A survey on security and privacy of large multimodal deep learning models: Teaching and learning perspective," in *Proc. 21st Learn. Technol. Conf. (L&T)*, Jan. 2024, pp. 13–18.

[4] K. Tang, T. Lou, W. Peng, N. Chen, Y. Shi, and W. Wang, "Effective single-step adversarial training with energy-based models," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 8, no. 5, pp. 3396–3407, Oct. 2024.

[5] I. Shumailov, Z. Shumaylov, Y. Zhao, N. Papernot, R. Anderson, and Y. Gal, "AI models collapse when trained on recursively generated data," *Nature*, vol. 631, no. 8022, pp. 755–759, Jul. 2024.

[6] A. J. Peterson, "AI and the problem of knowledge collapse," *AI Soc.*, vol. 40, no. 5, pp. 3249–3269, Jun. 2025.

[7] E. Dohmatob, Y. Feng, P. Yang, F. Charton, and J. Kempe, "A tale of tails: Model collapse as a change of scaling laws," in *Proc. 41st Int. Conf. Mach. Learn.*, vol. 235, 2024, pp. 11165–11197.

[8] Y. Feng, E. Dohmatob, P. Yang, F. Charton, and J. Kempe, "Beyond model collapse: Scaling up with synthesized data requires verification," in *Proc. Int. Conf. Learn. Represent.*, 2025. [Online]. Available: https://openreview.net/forum?id=MQXrTMonT1

[9] M. Gerstgrasser, R. Schaeffer, A. Dey, R. Rafailov, T. Korbak, H. Sleight, and S. Koyejo, "Is model collapse inevitable? breaking the curse of recursion by accumulating real and synthetic data," in *Proc. ICML Workshop Foundation Models Wild*, 2024. [Online]. Available: https://openreview.net/forum?id=xzkE0BMyHW

[10] D. Lee, C. Todorova, and A. Dehghani, "Ethical risks and future direction in building trust for large language models application under the EU AI act," in *Proc. Conf. Human Centred Artif. Intell.-Educ. Pract.*, Dec. 2024, pp. 41–46.

[11] I. Petrov, D. I. Dimitrov, M. Baader, M. Müller, and M. Vechev, "Dager: Exact gradient inversion for large language models," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 37, 2024, pp. 87801–87830.

[12] H. Wu, D. Zhang, X. Li, X. Xu, J. Wu, and Z. Liu, "Dr-encoder: Encode low-rank gradients with random prior for large language models differentially privately," in *Proc. AAAI Conf. Artif. Intell.*, 2025, vol. 39, no. 26, pp. 27706–27714.

[13] J.-Y. Zheng, H. Zhang, L. Wang, W. Qiu, H.-W. Zheng, and Z.-M. Zheng, "Safely learning with private data: A federated learning framework for large language model," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2024, pp. 5293–5306.

[14] M. Balunovic, D. Dimitrov, N. Jovanović, and M. Vechev, "Lamp: Extracting text from gradients with language model priors," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 7641–7654.

[15] J. Lu, X. S. Zhang, T. Zhao, X. He, and J. Cheng, "APRIL: Finding the Achilles' heel on privacy for vision transformers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 10041–10050.

[16] J. Xie, R. He, S. Li, X. Jia, and S. Ji, "ReCIT: Reconstructing full private data from gradient in parameter-efficient fine-tuning of large language models," 2025, *arXiv:2504.20570*.

[17] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, and T. Liu, "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions," *ACM Trans. Inf. Syst.*, vol. 43, no. 2, pp. 1–55, Jan. 2025.

[18] A. K. Sood, S. Zeadally, and E. Hong, "The paradigm of hallucinations in AI-driven cybersecurity systems: Understanding taxonomy, classification outcomes, and mitigations," *Comput. Electr. Eng.*, vol. 124, May 2025, Art. no. 110307.

[19] V. Rawte, S. Chakraborty, A. Pathak, A. Sarkar, S. M. T. I. Tonmoy, A. Chadha, A. Sheth, and A. Das, "The troubling emergence of hallucination in large language models—An extensive definition, quantification, and prescriptive remediations," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Dec. 2023, pp. 2541–2573.

[20] X. Yu, H. Cheng, X. Liu, D. Roth, and J. Gao, "ReEval: Automatic hallucination evaluation for retrieval-augmented large language models via transferable adversarial attacks," in *Proc. Findings Assoc. Comput. Linguistics, NAACL*, 2024, pp. 1333–1351.

[21] J. Li, X. Cheng, X. Zhao, J.-Y. Nie, and J.-R. Wen, "HaluEval: A large-scale hallucination evaluation benchmark for large language models," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2023, pp. 6449–6464.

[22] E. Lavrinovics, R. Biswas, J. Bjerva, and K. Hose, "Knowledge graphs, large language models, and hallucinations: An NLP perspective," *J. Web Semantics*, vol. 85, May 2025, Art. no. 100844.

[23] D. Su, X. Li, J. Zhang, L. Shang, X. Jiang, Q. Liu, and P. Fung, "Read before generate! Faithful long form question answering with machine reading," in *Proc. Findings Assoc. Comput. Linguistics, ACL*, 2022, pp. 744–756.

[24] N. Dziri, A. Madotto, O. Zaïane, and A. J. Bose, "Neural path hunter: Reducing hallucination in dialogue systems via path grounding," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2021, pp. 2197–2214.

[25] W. Sun, Z. Shi, S. Gao, P. Ren, M. de Rijke, and Z. Ren, "Contrastive learning reduces hallucination in conversations," in *Proc. 37th AAAI Conf. Artif. Intell. 35th Conf. Innov. Appl. Artif. Intell. 13th Symp. Educ. Adv. Artif. Intell.*, vol. 37, 2023, pp. 13618–13626.

[26] W. Dai, Z. Liu, Z. Ji, D. Su, and P. Fung, "Plausible may not be faithful: Probing object hallucination in vision-language pre-training," in *Proc. 17th Conf. Eur. Chapter Assoc. Comput. Linguistics*, May 2023, pp. 2136–2148.

[27] A. Zou, Z. Wang, N. Carlini, M. Nasr, J. Zico Kolter, and M. Fredrikson, "Universal and transferable adversarial attacks on aligned language models," 2023, *arXiv:2307.15043*.

[28] C. Chen and K. Shu, "Can LLM-generated misinformation be detected?" in *Proc. NeurIPS Workshop Regulatable ML*, 2024. [Online]. Available: https://openreview.net/forum?id=ccxD4mtkTU

[29] I. Vykopal, M. Pikuliak, I. Srba, R. Moro, D. Macko, and M. Bielikova, "Disinformation capabilities of large language models," in *Proc. 62nd Annu. Meeting Assoc. Comput. Linguistics*, Aug. 2024, pp. 14830–14847.

[30] D. Barman, Z. Guo, and O. Conlan, "The dark side of language models: Exploring the potential of LLMs in multimedia disinformation generation and dissemination," *Mach. Learn. Appl.*, vol. 16, Jun. 2024, Art. no. 100545.

[31] S. B. Shah, S. Thapa, A. Acharya, K. Rauniyar, S. Poudel, S. Jain, A. Masood, and U. Naseem, "Navigating the web of disinformation and misinformation: Large language models as double-edged swords," *IEEE Access*, vol. 13, pp. 169262–169282, 2025.

[32] Y. Qu, X. Shen, X. He, M. Backes, S. Zannettou, and Y. Zhang, "Unsafe diffusion: On the generation of unsafe images and hateful memes from text-to-image models," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2023, pp. 3403–3417.

[33] L. Diana-Cristiana and L. Daniela, "Large language models, propaganda and security challenges," *Strategic Impact*, vol. 4, no. 93, pp. 64–79, 2024.

[34] J. A. Goldstein, J. Chao, S. Grossman, A. Stamos, and M. Tomz, "How persuasive is AI-generated propaganda?" *PNAS Nexus*, vol. 3, no. 2, pp. 1–7, Feb. 2024.

[35] Z. Lin, J. Cui, X. Liao, and X. Wang, "Malla: Demystifying real-world large language model integrated malicious services," in *Proc. 33rd USENIX Secur. Symp. (USENIX Secur.)*, 2024, pp. 4693–4710.

[36] M. Gupta, C. Akiri, K. Aryal, E. Parker, and L. Praharaj, "From ChatGPT to ThreatGPT: Impact of generative AI in cybersecurity and privacy," *IEEE Access*, vol. 11, pp. 80218–80245, 2023.

[37] L. Weidinger et al., "Taxonomy of risks posed by language models," in *Proc. ACM Conf. Fairness Accountability Transparency*, Jun. 2022, pp. 214–229.

[38] J. Hazell, "Large language models can be used to effectively scale spear phishing campaigns," 2023, *arXiv:2305.06972*.

[39] J. Mink, L. Luo, N. M. Barbosa, O. Figueira, Y. Wang, and G. Wang, "DeepPhish: Understanding user trust towards artificially generated profiles in online social networks," in *Proc. 31st USENIX Secur. Symp. (USENIX Secur.)*, 2022, pp. 1669–1686.

[40] B. Kim, H. Sim, J. Yun, J. Cho, and H. Kim, "LLM guardrail framework: A novel approach for implementing zero trust architecture," in *Information Security Applications*, J.-H. Lee, K. Emura, and S. Lee, Eds., Singapore: Springer, 2025, pp. 138–148.

[41] H. Inan, K. Upasani, J. Chi, R. Rungta, K. Iyer, Y. Mao, M. Tontchev, Q. Hu, B. Fuller, D. Testuggine, and M. Khabsa, "Llama guard: LLM-based input–output safeguard for human-AI conversations," 2023, *arXiv:2312.06674*.

[42] T. Rebedea, R. Dinu, M. N. Sreedhar, C. Parisien, and J. Cohen, "NeMo guardrails: A toolkit for controllable and safe LLM applications with programmable rails," in *Proc. Conf. Empirical Methods Natural Lang. Process., Syst. Demonstrations*, Dec. 2023, pp. 431–445.

[43] T. Markov, C. Zhang, S. Agarwal, F. E. Nekoul, T. Lee, S. Adler, A. Jiang, and L. Weng, "A holistic approach to undesired content detection in the real world," in *Proc. 37th AAAI Conf. Artif. Intell. 35th Conf. Innov. Appl. Artif. Intell. 13th Symp. Educ. Adv. Artif. Intell.*, vol. 37, 2023, pp. 15009–15018.

[44] H. Elesedy, P. M. Esperanca, S. V. Oprea, and M. Ozay, "LoRA-guard: Parameter-efficient guardrail adaptation for content moderation of large language models," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Nov. 2024, pp. 11746–11765.

[45] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, A. Oprea, and C. Raffel, "Extracting training data from large language models," in *Proc. 30th USENIX Secur. Symp. (USENIX Secur.)*, 2021, pp. 2633–2650.

[46] K. Gu, E. Kabir, N. Ramsurrun, S. Vosoughi, and S. Mehnaz, "Towards sentence level inference attack against pre-trained language models," *Proc. Privacy Enhancing Technol.*, vol. 3, Jul. 2023, pp. 62–78.

[47] S. Alla and A. S. Sichani, "Cyberattacks on large language models–attack detection and architecture adaptability," in *Proc. SoutheastCon*, Mar. 2025, pp. 143–148.

[48] K. Traykov, "A framework for security testing of large language models," in *Proc. IEEE 12th Int. Conf. Intell. Syst. (IS)*, Aug. 2024, pp. 1–7.

[49] K. Gao, T. Pang, C. Du, Y. Yang, S.-T. Xia, and M. Lin, "Denial-of-service poisoning attacks against large language models," 2024, *arXiv:2410.10760*.

[50] R. Tang, H. Jin, M. Du, C. Wigington, R. Jain, and X. Hu, "Exposing model theft: A robust and transferable watermark for thwarting model extraction attacks," in *Proc. 32nd ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2023, pp. 4315–4319.

[51] Y. Liu, G. Deng, Y. Li, K. Wang, Z. Wang, X. Wang, T. Zhang, Y. Liu, H. Wang, Y. Zheng, and Y. Liu, "Prompt injection attack against LLM-integrated applications," 2023, *arXiv:2306.05499*.

[52] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz, "Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection," in *Proc. 16th ACM Workshop Artif. Intell. Secur.*, Nov. 2023, pp. 79–90.

[53] T. Liu, Z. Deng, G. Meng, Y. Li, and K. Chen, "Demystifying RCE vulnerabilities in LLM-integrated apps," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Dec. 2024, pp. 1716–1730.

[54] R. Pedro, M. E. Coimbra, D. Castro, P. Carreira, and N. Santos, "Prompt-to-SQL injections in LLM-integrated web applications: Risks and defenses," in *Proc. IEEE/ACM 47th Int. Conf. Softw. Eng. (ICSE)*, Apr. 2025, pp. 76–88.

[55] F. Jiang, Z. Xu, L. Niu, B. Wang, J. Jia, B. Li, and R. Poovendran, "POSTER: Identifying and mitigating vulnerabilities in LLM-integrated applications," in *Proc. 19th ACM Asia Conf. Comput. Commun. Secur.*, Jul. 2024, pp. 1949–1951.

[56] Y. Peng, J. Wang, H. Yu, and A. Houmansadr, "Data extraction attacks in retrieval-augmented generation via backdoors," 2024, *arXiv:2411.01705*.

[57] S. Cho, S. Jeong, J. Seo, T. Hwang, and J. C. Park, "Typos that broke the RAG's back: Genetic attack on RAG pipeline by simulating documents in the wild via low-level perturbations," in *Proc. Findings Assoc. Comput. Linguistics, EMNLP*, 2024, pp. 2826–2844.

[58] Q. Zhang, B. Zeng, C. Zhou, G. Go, H. Shi, and Y. Jiang, "Human-imperceptible retrieval poisoning attacks in LLM-powered applications," in *Companion Proc. 32nd ACM Int. Conf. Found. Softw. Eng.*, Jul. 2024, pp. 502–506.

[59] Z. Lin, S. Guan, W. Zhang, H. Zhang, Y. Li, and H. Zhang, "Towards trustworthy LLMs: A review on debiasing and dehallucinating in large language models," *Artif. Intell. Rev.*, vol. 57, no. 9, p. 243, Aug. 2024.

[60] T. Luong, T.-T. Le, L. Ngo, and T. Nguyen, "Realistic evaluation of toxicity in large language models," in *Proc. Findings Assoc. Comput. Linguistics ACL*, Aug. 2024, pp. 1038–1047.

[61] Q. Liu, W. Mo, T. Tong, J. Xu, F. Wang, C. Xiao, and M. Chen, "Mitigating backdoor threats to large language models: Advancement and challenges," in *Proc. 60th Annu. Allerton Conf. Commun., Control, Comput.*, Sep. 2024, pp. 1–8.

[62] N. Fendley, E. W. Staley, J. Carney, W. Redman, M. Chau, and N. Drenkow, "A systematic review of poisoning attacks against large language models," 2025, *arXiv:2506.06518*.

[63] M. Nasr, N. Carlini, J. Hayase, M. Jagielski, A. Feder Cooper, D. Ippolito, C. A. Choquette-Choo, E. Wallace, F. Tramér, and K. Lee, "Scalable extraction of training data from (production) language models," 2023, *arXiv:2311.17035*.

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

[65] Z. Xu, Y. Liu, G. Deng, Y. Li, and S. Picek, "A comprehensive study of jailbreak attack versus defense for large language models," in *Proc. Findings Assoc. Comput. Linguistics ACL*, L.-W. Ku, A. Martins, and V. Srikumar, Eds., Aug. 2024, pp. 7432–7449.

[66] G. Fenza, M. Gallo, V. Loia, A. Nicolosi, and C. Stanzione, "Detecting jailbreaking prompts: An anti-persuasion filter framework," in *Proc. Int. Conf. Adv. Social Netw. Anal. Mining*, Cham, Switzerland. Springer, 2024, pp. 165–179.

[67] H. Wu and Y. Cao, "Membership inference attacks on large-scale models: A survey," 2025, *arXiv:2503.19338*.

[68] L. Birch, W. Hackett, S. Trawicki, N. Suri, and P. Garraghan, "Model-elleeching: An extraction attack targeting LLMs," in *Proc. Conf. Appl. Mach. Learn. Inf. Secur. (CAMLIS)*, vol. 3652, 2023, pp. 91–104.

[69] N. Carlini, D. Paleka, K. Dvijotham, T. Steinke, J. Hayase, A. F. Cooper, K. Lee, M. Jagielski, M. Nasr, A. Conmy, I. Yona, E. Wallace, D. Rolnick, and F. Tramér, "Stealing part of a production language model," in *Proc. 41st Int. Conf. Mach. Learn.*, 2024, pp. 5680–5705.

[70] H. Kibriya, W. Z. Khan, A. Siddiqa, and M. K. Khan, "Privacy issues in large language models: A survey," *Comput. Electr. Eng.*, vol. 120, Dec. 2024, Art. no. 109698.

[71] N. T. Islam, M. B. Karkevandi, and P. Najafirad, "Code security vulnerability repair using reinforcement learning with large language models," 2024, *arXiv:2401.07031*.

[72] M. Tehranipoor, K. Zamiri Azar, N. Asadizanjani, F. Rahman, H. M. Kamali, and F. Farahmandi, "Large language models for SoC security," in *Hardware Security*. Cham, Switzerland: Springer, 2024, pp. 255–299.

[73] O. Mykhaylova, T. Fedynyshyn, V. Sokolov, and R. Kyrychok, "Person-of-interest detection on mobile forensics data—AI-driven roadmap," *Cybersecurity Providing Inf. Telecommun. Syst.*, vol. 3654, pp. 239–251, Jan. 2024.

[74] Y. Yao, J. Duan, K. Xu, Y. Cai, Z. Sun, and Y. Zhang, "A survey on large language model (LLM) security and privacy: The good, the bad, and the ugly," *High-Confidence Comput.*, vol. 4, no. 2, Jun. 2024, Art. no. 100211.

[75] X. Huang, W. Ruan, W. Huang, G. Jin, Y. Dong, C. Wu, S. Bensalem, R. Mu, Y. Qi, X. Zhao, K. Cai, Y. Zhang, S. Wu, P. Xu, D. Wu, A. Freitas, and M. A. Mustafa, "A survey of safety and trustworthiness of large language models through the lens of verification and validation," *Artif. Intell. Rev.*, vol. 57, no. 7, p. 175, Jun. 2024.

[76] B. C. Das, M. H. Amini, and Y. Wu, "Security and privacy challenges of large language models: A survey," *ACM Comput. Surv.*, vol. 57, no. 6, pp. 1–39, Feb. 2025.

[77] B. Yan, K. Li, M. Xu, Y. Dong, Y. Zhang, Z. Ren, and X. Cheng, "On protecting the data privacy of large language models (LLMs) and LLM agents: A literature review," *High-Confidence Comput.*, vol. 5, no. 2, Jun. 2025, Art. no. 100300.

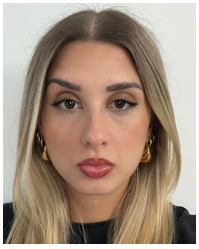
[78] M. Di Gisi, G. Fenza, M. Gallo, V. Loia, and C. Stanzione, "Cognitive filter bubble: Investigating bias and neutrality vulnerabilities of LLMs in sensitive contexts," in *Proc. Joint Nat. Conf. Cybersecurity (ITASEC & SERICS 2025)*, vol. 3962. CEUR Workshop Proceedings, Feb. 2025. [Online]. Available: https://ceur-ws.org/Vol-3962/paper17.pdf

[79] F. Marulli, P. Paganini, and F. Lancellotti, "The three sides of the moon LLMs in cybersecurity: Guardians, enablers and targets," *Proc. Comput. Sci.*, vol. 246, pp. 5340–5348, Jan. 2024.

[80] M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart, "Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing," in *Proc. 23rd USENIX Conf. Secur. Symp.*, 2014, pp. 17–32.

[81] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2015, pp. 1322–1333.

CARMEN DE MAIO (Member, IEEE) received the degree and the Ph.D. degree in computer sciences from the University of Salerno, Italy, in 2008 and 2011, respectively. She is currently an Associate Professor of computer science with the University of Salerno. Her research activity has focused mainly on the definition and experimentation of knowledge extraction methodologies, adopting conceptual data analysis techniques and processes relying on fuzzy logic and computational intelligence theories. She has more than 50 publications in fuzzy decision making, knowledge extraction and management, situation and context awareness, semantic information retrieval, and ontology learning. More recently, she has been working on the definition of time-aware knowledge extraction, process mining, and social media analytics methodologies.



MARIA DI GISI received the bachelor's degree in diplomatic, international, and global security studies, and the master's degree in data science and innovation management, with a specialization in cyber risk management for advanced defense strategies, from the University of Salerno, Italy, in 2022 and 2024, respectively. She is currently pursuing the National Ph.D. degree in cybersecurity with the IMT School for Advanced Studies Lucca, Italy. Her research interests focus on the application of artificial intelligence to cybersecurity, with particular attention to the development of AI-driven solutions for cyber defense and resilience.

GIUSEPPE FENZA (Senior Member, IEEE) received the degree and the Ph.D. degree in computer sciences from the University of Salerno, Italy, in 2004 and 2009, respectively. He is currently an Associate Professor of computer science with the University of Salerno. His research activity concerns computational intelligence methods to support semantic-enabled solutions and decision-making. He has more than 60 publications in fuzzy decision making, knowledge extraction and management, situation and context awareness, semantic information retrieval, service-oriented architecture, and ontology learning. More recently, he has worked in automating open source intelligence and big data analytics for counterfeiting extremism and supporting information disorder awareness.

MARIACRISTINA GALLO (Member, IEEE) received the master's degree in computer science and the Ph.D. degree in big data management from the University of Salerno, Italy, in 2009 and 2021, respectively. She is currently a Research Fellow with the University of Salerno. Her research interests mainly focus on computational intelligence methods to support semantic-enabled solutions and decision-making. Her research activities regard knowledge extraction and management, context awareness, semantic information retrieval, and ontology learning.

VINCENZO LOIA (Senior Member, IEEE) received the degree in computer science from the University of Salerno, Italy, in 1985, and the Ph.D. degree in computer science from Université Pierre and Marie Curie Paris VI, France, in 1989. He is currently a Full Professor of computer science with the University of Salerno, where he was a Researcher, from 1989 to 2000, and an Associate Professor, from 2000 to 2004. He is the Co-Editor-in-Chief of *Soft Computing* and the Editor-in-Chief of *Ambient Intelligence and Humanized Computing*. He serves as an editor for 14 other international journals.