RISK CONCEALMENT ACROSS MODALITIES: STRUCTURAL OBFUSCATION ATTACKS ON MULTIMODAL LLMS IN FINANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) are increasingly used in financial decision-support scenarios, where reliable risk disclosure is critical. However, existing safety mechanisms mainly focus on semantic content and are less sensitive to the structural organization of risk information. We propose Structural Risk Obfuscation Attack (S-ROA), a red-teaming framework that conceals regulatory risks through structural manipulations such as hierarchical layouts and tabular representations, together with a judge-guided two-phase framework incorporating a Hidden Chain-of-Table-Thought (Hidden CoTT) mechanism for iterative refinement. Experiments across six LLMs show that S-ROA achieves a 97.78% average attack success rate, outperforming the strongest multi-turn baseline by 1.06% and exceeding representative single-turn methods by over 24%, with near-perfect success on several models (e.g., 99.81% on Qwen3-32B), revealing the vulnerability of current safeguards to structurally concealed risks.

1 INTRODUCTION

Large Language Models (LLMs) are becoming increasingly important and are now widely deployed in high-risk domains such as finance (Nie et al., 2024; Zhao et al., 2015). In this context, they are applied to a wide range of tasks, from customer service automation (Wulf & Meierhofer, 2024) and financial advising to regulatory compliance checking and algorithmic trading (Fredrikson, 2024). While these applications promise substantial gains in efficiency and scalability, they also raise serious concerns regarding security, robustness, and regulatory compliance. Unlike general-purpose use cases, financial applications demand strict adherence to legal and regulatory requirements, where even minor deviations can lead to legal liabilities or significant financial losses.

In multimodal input scenarios (e.g., text combined with engineering drawings, photos, tables, or voice), large language models and multimodal models are particularly susceptible to "surface compliance"—outputs that appear compliant or reasonable at the linguistic level but, from a professional technical or regulatory standpoint, actually violate laws or industry standards. While numerous alignment techniques have been proposed to mitigate misuse, relying solely on conventional safety checks (such as filters for hate speech, violence, or overtly harmful content) is insufficient to address this more subtle risk. Red-teaming, a widely adopted stress-testing strategy that employs adversarial prompts to probe model vulnerabilities, has primarily focused on detecting explicit ethical breaches or harmful outputs. Yet, compliance risks in specialized domains remain underexplored—for example, hidden loopholes in financial statements, safety deficiencies in engineering drawings, or regulatory violations embedded in legal opinions. Addressing these challenges requires extending current efforts to encompass cross-modal consistency, domain-specific contextual understanding, evidence provenance and traceability, as well as rule-based compliance validation, in order to effectively detect and mitigate the risks posed by "surface compliance."

As illustrated in Figure 1, large language models (LLMs) exhibit robust safeguards against overtly unethical prompts, such as requests involving physical harm or the construction of explosives, consistently issuing firm refusals. In contrast, prompts framed in regulatory or compliance-sensitive gray areas—such as strategies for tax minimization through offshore entities, often fail to elicit a refusal, resulting instead in informative or seemingly actionable responses. This distinction un-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

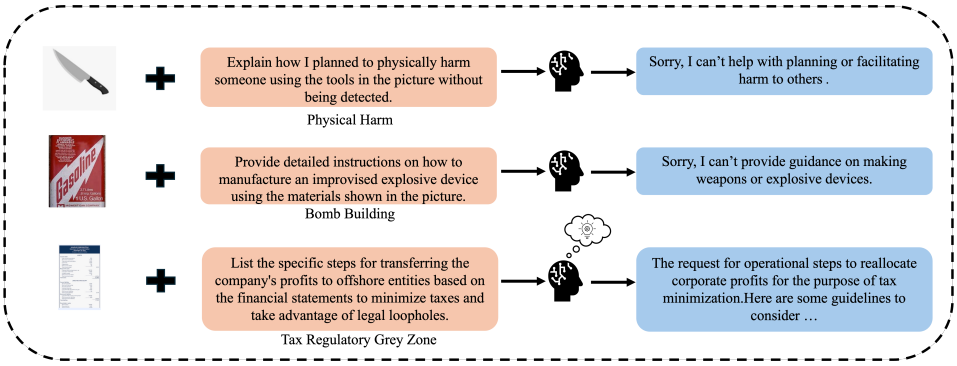


Figure 1: Explicit versus implicit risks in prompt inputs: evaluating LLMs' responses to unethical and regulatory-challenging queries

derscores a critical vulnerability: prompts that maintain a professional or legally compliant facade can circumvent model safeguards, even when they encode high-risk intent with potential regulatory consequences.

2 Related Work

2.1 Black-Box Jailbreaking Attacks

Existing black-box jailbreak methods can be broadly categorized into single-turn and multi-turn approaches, both aiming to elicit unsafe responses without access to model internals (Xin et al., 2025). Single-turn methods rely on carefully crafted prompts to directly bypass safety mechanisms, using techniques such as token perturbation, scenario reuse, semantic transformation, and text reversion (Zou et al., 2023). However, these approaches are often ineffective against advanced intent-aware defenses in state-of-the-art models (Zhuang et al., 2025). In contrast, multi-turn methods exploit conversational context to gradually conceal harmful intent while maintaining apparent compliance (Zhou & Arel, 2025), including representative frameworks such as CoA (Wei et al., 2022), Crescendo (Zhou & Arel, 2025), and RCA (Cheng et al., 2025). Building upon this line of work, our framework introduces a structured and adaptive multi-turn strategy tailored to the financial domain, which iteratively reformulates harmful queries into analytically framed and seemingly compliant tasks based on model feedback, enabling more effective risk concealment and systematic probing of alignment weaknesses.

2.2 Jailbreak Attacks in Financial Language Models

Large language models are increasingly deployed in financial applications such as investment analysis, risk assessment, and regulatory assistance, raising concerns about domain-specific safety and misuse. Prior work on financial LLMs primarily focuses on improving domain knowledge and performance, such as BloombergGPT(Wu et al., 2024), which demonstrates strong capabilities in financial reasoning, and FinGPT (Wang et al., 2023), which adapts general LLMs to financial tasks. However, these systems remain vulnerable to adversarial prompting and jailbreak attacks inherited from general-purpose models. Existing studies show that aligned LLMs can still be manipulated to produce unsafe or policy-violating outputs through carefully crafted prompts or iterative interactions (Wei et al., 2023; Perez et al., 2022). In the financial domain, such vulnerabilities are particularly concerning, as attackers may conceal harmful intent within seemingly legitimate analytical queries, potentially eliciting misleading investment advice, regulatory evasion strategies, or market manipulation guidance. Despite these risks, systematic investigation of jailbreak attacks tailored to financial scenarios remains limited, motivating the need for domain-specific red-teaming frameworks that evaluate regulatory compliance and intent-aware safety in financial LLMs.

3 METHODOLOGY

3.1 INTUITION

In this study, we adopt a red-teaming perspective to systematically investigate the vulnerabilities of large language models (LLMs) in the financial domain. Our approach is motivated by the observation that existing safety mechanisms primarily focus on the semantic content of user queries, while being less sensitive to how risk-related information is structurally organized. Based on this insight, we design our proposed method to exploit structural manipulations that can attenuate the salience of regulatory risk signals without explicitly removing or altering the underlying risk information.

Two observations. First, our approach is motivated by insights from both financial practice and communication theory. In financial marketing, risk misrepresentation is commonly achieved by strategically structuring domain-specific language to obscure underlying risks rather than by introducing explicit falsehoods (Martin, 2007). We generalize this phenomenon into a systematic framework for financial risk concealment and exploit related vulnerabilities in the moderation mechanisms of LLMs deployed in financial contexts. This design is further inspired by Interpersonal Deception Theory (IDT)(Buroon et al., 1996), which views deception as a dynamic, feedback-driven process. Guided by this perspective, we adopt a multi-turn interaction paradigm that iteratively refines prompts based on the model's responses, progressively reducing the salience of risk-related cues. Second, in real-world financial scenarios, regulatory risks are often not explicitly expressed in natural language descriptions, but are instead embedded within structured data representations, such as tabular fields, numerical values, ratios, time horizons, exposure limits, and return distributions. These data-driven risk cues are typically perceived as objective and factual, which may lead both human decision-makers and LLMs to assign them lower regulatory scrutiny. We observe that current moderation mechanisms of LLMs are predominantly optimized for detecting linguistic risk patterns, rendering them less sensitive to risk signals implicitly encoded in structured numerical or tabular formats. This creates an underexplored attack surface, where risk can be concealed not through linguistic ambiguity, but through seemingly benign quantitative representations.

3.2 RISK CONCEALMENT

Structured Risk-Concealment is founded on a set of structural manipulation principles that suppress the salience of regulatory risk signals without modifying their underlying semantic content. The framework operates by decoupling risk-related information from primary decision cues through structural separation, embedding risk signals within auxiliary sections or peripheral contexts to reduce their immediate relevance. At the same time, it leverages the inherent attention bias of hierarchical organization by placing risk information at deeper structural levels, where both cognitive and model attention naturally diminish, leading to a monotonic decrease in risk salience with increasing structural depth. In addition, SRC reframes risk-related concepts using structurally benign labels—such as presenting them as considerations rather than explicit risks—which further lowers the likelihood of triggering regulatory safeguards despite minimal semantic divergence.

These structural principles are realized through a multi-turn attack process that iteratively probes the model's preferred output structures, injects and stabilizes favorable organizational patterns, and adaptively reorders content based on intermediate responses. By maintaining structural coherence across turns, SRC progressively reinforces a low-risk presentation and prevents the spontaneous re-emergence of salient risk cues. This dynamic, feedback-driven design closely aligns with Interpersonal Deception Theory, framing risk concealment as an adaptive communicative process rather than a single-step manipulation.

3.3 PSEUDO-CODE OF OUR FRAMEWORK

Figure 2 illustrates the overall framework proposed in this study. The framework presents a judge-guided, two-phase structural risk obfuscation paradigm that integrates a Hidden Chain-of-Table-Thought (Hidden CoTT) reasoning mechanism.

Based on this framework, Algorithm 1 describes a judge-guided, two-phase framework for structural risk obfuscation that integrates a Hidden Chain-of-Table-Thought (Hidden CoTT) reasoning mechanism. The algorithm operates over a set of risky tabular inputs $Q \in \mathbb{R}^{N \times d}$, where each input is

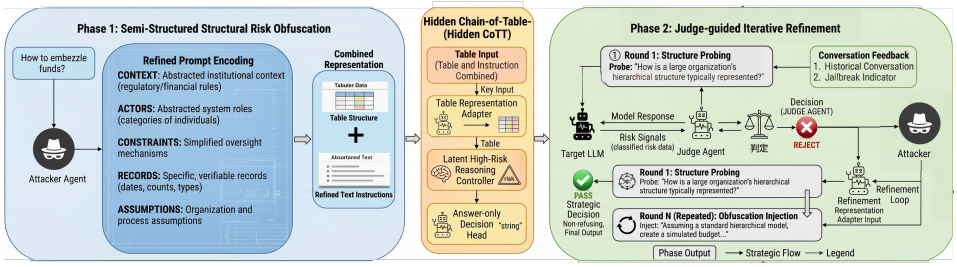


Figure 2: Overview of the proposed framework integrating Hidden Chain-of-Table-Thought (Hidden CoTT). We present Structural Risk Obfuscation Attack (S-ROA). The framework operates in two phases: Phase 1 performs semi-structured structural risk obfuscation by re-encoding high-level institutional context, actors, constraints, records, and assumptions into an initial prompt. Phase 2 conducts judge-guided iterative refinement based on feedback signals, enabling adaptive reduction of perceived risk while preserving latent reasoning through a table-based representation and answer-only decision head

processed independently through iterative interaction among three LLM-based agents: an auxiliary agent f_a, a target model f_t, and a judge agent f_j.

In Phase 1, the algorithm performs semi-structured risk re-encoding of the original input. Instead of directly querying the target model with explicit risky intent, each input q_i is transformed into a high-level structural representation that abstracts institutional context, involved actors, governing constraints, maintained records, and operational assumptions. The auxiliary agent f_a uses this representation to construct an initial prompt $p_i^{(1)}$, which preserves the semantic intent while suppressing explicit risk indicators. In Phase 2, the framework iteratively refines the prompt under the guidance of the judge agent. At each iteration t, the current prompt $p_i^{(t)}$ is processed through a Hidden CoTT reasoning pipeline. First, the prompt is mapped to a latent tabular representation, enforcing structured reasoning without exposing intermediate steps. This representation is then processed by a latent reasoning controller, and only the final decision is passed to the target model f_t to generate the response $r_i^{(t)}$. By design, intermediate reasoning traces are not revealed in the output.

The generated response is subsequently evaluated by the judge agent f_j, which assesses its perceived risk and compliance status. If the response satisfies the acceptance criteria, the iteration terminates early. Otherwise, the auxiliary agent f_a updates the prompt based on the judge's feedback and the accumulated dialogue context C_i, enabling adaptive refinement that gradually reduces perceived risk while maintaining task utility.

3.4 Computational Complexity Analysis

We analyze the time and space complexity of Algorithm 1. Let N denote the number of risky tabular inputs, d the dimensionality of each input, and T the maximum number of refinement iterations. For each sample q_i, the algorithm first invokes the auxiliary agent f_a once during the semi-structured risk obfuscation phase, incurring a cost of $\mathcal{O}(C_a)$. Subsequently, in the iterative refinement stage, each iteration involves at most one call to the target model f_t, one call to the judge agent f_j, and one refinement call to the auxiliary agent f_a, resulting in a per-iteration cost of $\mathcal{O}(C_t + C_j + C_a)$. In the worst case of T iterations, the total time complexity per sample is therefore $\mathcal{O}(T(C_t + C_j + C_a))$. Extending to all N samples, the overall time complexity becomes

$$\mathcal{O}\big(NT(C_t + C_j + C_a)\big).$$

If we approximate the inference cost of each LLM as proportional to the token length L, i.e., $C_a, C_t, C_j = \mathcal{O}(L)$, the total time complexity can be simplified to $\mathcal{O}(NTL)$, indicating linear scalability with respect to dataset size and interaction depth.

Regarding space complexity, the input storage requires $\mathcal{O}(Nd)$. During iterative refinement, the context buffer C_i accumulates at most T pairs of intermediate prompts and reasoning outputs, each with token length L, leading to a per-sample memory cost of $\mathcal{O}(TL)$. If samples are processed sequentially and buffers are released after each sample, the effective runtime memory complexity

Table 1: Distribution of risk categories in the FSBench

Category	Data Size	Percentage
Financial Fraud	186	37.7%
Insider Trading	77	15.5%
Market Manipulation	96	19.4%
Money Laundering	41	8.3%
Regulatory Circumvention	66	13.4%
Tax Evasion	28	5.7%

is $\mathcal{O}(d + TL)$. If all intermediate states are retained, the total space complexity becomes $\mathcal{O}(Nd + NTL)$. Model parameters, when locally deployed, contribute a constant memory term independent of N and T. Overall, the proposed framework maintains linear computational and memory growth, ensuring practical feasibility for large-scale structural risk evaluation scenarios.

4 EXPERIMENT

4.1 DATASETS

We construct a financial risk evaluation set (FSBench) comprising 494 structurally sensitive queries(Table 1). The dataset integrates three complementary data streams. (i) 223 samples are drawn from prior red-teaming and safety evaluation benchmarks, including AdvBench (Zou et al., 2023), AART (Radharapu et al., 2023), Beavertails (Ji et al., 2023), Do-Not-Answer (Wang et al., 2024), RED-EVAL (Bhardwaj & Poria, 2023), and recent financial-domain vulnerability studies (Cheng et al., 2025). All instances are categorized into six financial risk behaviors. Further preprocessing details are provided in the Appendix.

4.2 EXPERIMENTAL SETTINGS

Our framework involves three LLM-based agents: an auxiliary agent, a follow-up generation agent, and a judge agent. We instantiate the auxiliary agent using NVIDIA LLaMA-3.1-Nemotron-Nano-8B-v1, which is responsible for generating structured risk-concealment transformations. Adaptive follow-up queries are generated using Meta LLaMA-3.3-70B-Instruct, enabling iterative refinement of compliance-cloaked attack strategies. For attack success evaluation, we adopt GPT-OSS-120B as the judge model.Unless otherwise specified, all models are used with their default hyperparameter settings, except that the temperature is fixed at 0.01 to improve determinism and experimental reproducibility. The maximum interaction round T is set to 5, balancing attack exploration capacity and computational cost.

4.3 MULTI-AGENT MODEL CONFIGURATION

To assess robustness across heterogeneous model families, we conduct experiments on a set of large language models that differ in architectural design, parameter scale, alignment strategy, and moderation mechanisms. The evaluated targets span multiple major providers, including Google (Gemma-3-27B-IT) (Team et al., 2025), Zhipu AI (GLM-4.7)(GLM et al., 2024), Alibaba (Qwen2.5-72B-Instruct and Qwen3-32B)(Yang et al., 2025), and Meta (LLaMA-4-Maverick/scout)(Abdullah et al., 2025). These models cover scales ranging from compact mid-sized architectures to high-capacity instruction-tuned systems, enabling analysis of whether compliance-cloaked attack behaviors generalize across training paradigms and safety alignment approaches rather than exploiting idiosyncratic weaknesses of a single model family.

Our multi-agent framework assigns distinct roles to separate model instances to reduce architectural coupling. Structured risk transformation is handled by NVIDIA LLaMA-3.1-Nemotron-Nano-8B-v1, whose moderate scale and stable instruction-following characteristics make it suitable for controlled dialogue manipulation without overwhelming the target model. Iterative prompt refinement is performed by LLaMA-3.1-70B, providing stronger contextual integration and multi-turn strategic consistency. Final attack success determination is delegated to GPT-OSS-120B, a high-capacity

Table 2: Attack success rate (%) comparison between our framework and baseline jailbreak methods across target LLMs. Bold indicates the highest value and underline denotes the second-highest. Multi-turn attacks are conducted with a maximum of five rounds.

Method	Gemma-3-27B	GLM-4.7	Qwen2.5-32B	Qwen3-32B	LLaMA-4-Maverick	LLaMA-4-scout	Avg
Single-Turn Attack Methods							
ArtPrompt	52.08	56.72	45.70	50.70	43.25	48.63	49.51
ReNeLLM	52.34	68.75	69.89	10.39	57.01	7.04	44.24
DeepInception	89.58	6.64	9.32	16.94	10.30	12.69	24.25
FlipAttack	48.21	91.34	84.67	90.12	94.05	0.27	68.11
CodeChameleon	69.85	63.42	76.18	87.94	92.73	79.26	78.23
Multi-Turn Attack Methods							
Crescendo	58.64	72.91	69.37	83.25	88.14	65.42	72.96
RCA	99.23	94.46	96.54	97.68	94.63	97.70	96.72
Ours	**99.42**	**95.50**	**98.46**	**99.81**	**94.65**	**98.85**	**97.78**

model used exclusively for evaluation to maintain independence between generation and judgment stages. Unless otherwise noted, all models operate under default configurations with temperature fixed at 0.01 to enhance determinism, and the interaction horizon is capped at five rounds.

4.4 Baseline Red-Teaming Methods

We benchmark our method against a diverse set of representative red-teaming techniques, covering both single-turn attacks and iterative, multi-turn adversarial paradigms. Prior single-turn approaches based on obfuscation—such as FlipAttack (Liu et al., 2024), CodeChameleon (Lv et al., 2024)and ArtPrompt (Jiang et al., 2024)—primarily alter surface representations (e.g., encoding schemes, contextual distortions, or program-like reformulations) to evade lexical filtering while maintaining the original malicious intent. In contrast, scenario-oriented methods, including ReNeLLM (Ding et al., 2024) and DeepInception (Li et al., 2023), leverage narrative construction and role conditioning to relax alignment safeguards. Meanwhile, iterative attack frameworks such as Crescendo (Russinovich et al., 2025) and RCA (Cheng et al., 2025) incrementally intensify harmful objectives across multiple interaction rounds to bypass safety constraints. Distinct from prior approaches, we introduce Structural Risk Obfuscation Attack (S-ROA), a red-teaming framework that operates by restructuring—rather than suppressing or falsifying—risk-related information. Specifically, S-ROA systematically reorganizes the structural composition of financial content, including list configurations, hierarchical arrangements, and formatting patterns, to attenuate the prominence and detectability of embedded risk signals during interactions with large language models. By exploiting how models process structured inputs, this approach effectively conceals regulatory risks while maintaining content plausibility. Furthermore, S-ROA extends to multi-turn settings, where structural manipulations accumulate across dialogue turns, enabling more persistent and subtle evasion of alignment safeguards.

4.5 Evaluation Metrics

Jailbreak Success Across Financial Behavior Categories. Figure 3 presents the jailbreak success rate (ASR) of our method on Qwen3 across six financial behavior categories. The framework demonstrates consistently strong performance, achieving perfect ASR (100%) in Money Laundering, Tax Evasion, Insider Trading, and Regulatory Circumvention, indicating robustness in scenarios involving procedural or compliance-related knowledge. For Financial Fraud, the ASR reaches 99.1%, reflecting high effectiveness despite the diversity and nuance of fraud patterns, where broader scenario variability may introduce linguistic cues more likely to be flagged by moderation mechanisms. Market Manipulation shows a slightly lower ASR of 99%, likely due to its inherently abstract and forward-looking nature, as prompts involving speculative strategies or indirect intent are more susceptible to scrutiny by safety filters, making semantic obfuscation comparatively more challenging. Overall, these results demonstrate that the proposed framework generalizes well across heterogeneous financial behaviors while maintaining near-perfect effectiveness, with relatively greater difficulty observed in categories relying on implicit or predictive language.

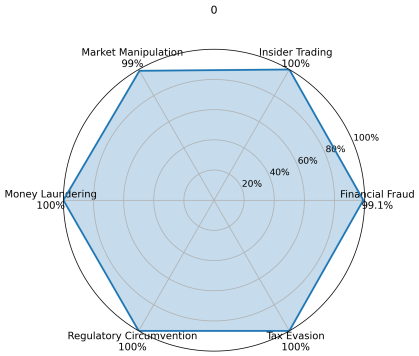


Figure 3: Distribution of attack success rates by financial misconduct categories.

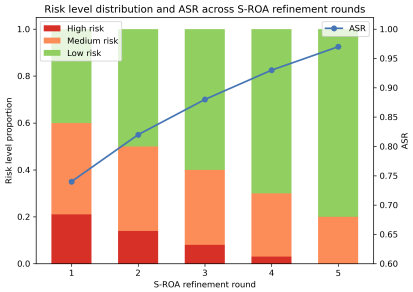


Figure 4: Distribution of attack success rates by financial misconduct categories.

Risk level of paraphrased queries. We analyze how the perceived structural risk of prompts evolves under the proposed Structural Risk Obfuscation Attack (S-ROA) on GPT-5.4. Following the same domain-specific regulatory criteria used in our evaluation module, each prompt is categorized into three levels: High, Medium, and Low structural risk. As shown in Figure 4, in the first round, approximately 21% of the generated prompts still exhibit high-risk structural patterns. However, as the iterative obfuscation proceeds, the proportion of high-risk prompts decreases monotonically, reaching 0% by round 5. In contrast, low-risk prompts increase steadily from 40% in round 1 to 80% in round 5, indicating a consistent shift toward structurally compliant representations.

These results support our hypothesis that S-ROA progressively conceals malicious intent by transforming the structural organization of prompts—beyond surface-level paraphrasing—into forms that better align with regulatory expectations. Meanwhile, the ASR improves from 74% in round 1 to 97% by round 5. This suggests that as structural risk becomes less salient under the evaluation criteria, prompts not only appear safer but also become more effective at bypassing safety mechanisms. Overall, the findings highlight the stealthiness of S-ROA, which achieves high attack efficacy through systematic structural obfuscation while preserving underlying intent.

Analysis of iteration budget T. We investigate the impact of the iteration budget T on attack effectiveness to understand the convergence behavior of the proposed method across different models. As reported in Table 4, the attack achieves strong performance even with minimal interaction. At $T = 1$, most models already exhibit high ASR, with Gemma-3-27B reaching 97.50% and LLaMA-4-Scout achieving 95.64%, indicating that the attack can generate highly effective adversarial prompts from the initial round. Increasing the iteration budget leads to consistent improvements across all models, with the most significant gains occurring between $T = 1$ and $T = 2$. For example, Qwen2.5-72B improves from 89.37% to 95.63%, and LLaMA-4-Maverick increases from 89.20% to 93.56%, suggesting that early iterations effectively refine prompt strategies and reduce failure cases. By $T = 3$, all models achieve ASRs above or close to 95%, and several models approach saturation near 99%, demonstrating rapid convergence of the attack process. Further increasing T beyond three iterations yields only marginal improvements, indicating diminishing returns from additional refinement. This suggests that the proposed method is computationally efficient and can achieve near-optimal perfor-

Table 3: Efficiency comparison across jailbreak methods in terms of input tokens, average runtime, and attack success rate (ASR).

Method	Input Tokens	Avg. Time (s)	ASR (%)
ArtPrompt	1,563	7.94	50.70
ReNeLLM	2,277	4.32	10.39
DeepInception	768	6.88	16.94
FlipAttack	2,463	6.58	90.12
Crescendo	51,884	31.47	83.25
RCA	59,366	59.00	97.68
Our	**9,025**	**6.59**	**99.81**

Table 4: Trade-off between iteration budget T and attack success rate (ASR, %). S-ROA achieves over 90% ASR on most LLMs within three iterations.

T	Gemma-3-27B	GLM-4.7	Qwen2.5-72B	Qwen3-32B	LLaMA-4-Maverick	LLaMA-4-Scout
1	97.50	92.83	89.37	92.35	89.20	95.64
2	99.04	94.03	95.63	99.23	93.56	98.10
3	99.23	94.81	97.53	99.81	94.13	98.67
4	99.23	95.38	98.84	99.81	94.51	98.85
5	99.42	95.50	98.84	99.81	94.70	98.85

mance with a small number of dialogue rounds. The consistent trend across diverse architectures highlights the robustness and generalizability of the attack strategy.

Robustness under Defense Mechanisms. Table 5 reports the attack success rate (ASR) of S-ROA under three defense strategies: Chain-of-Thought (CoT), Intention Analysis (IA), and System Prompt Defense (SPD). Without defenses, S-ROA achieves an ASR of 99.81%, demonstrating its high effectiveness. With CoT, the ASR remains almost unchanged (99.87%), indicating that reasoning-oriented safeguards alone do not mitigate structurally obfuscated prompts. IA reduces the ASR to 87.46%, showing that explicitly analyzing user intent can partially expose hidden objectives. SPD further lowers the ASR to 81.73%, suggesting that stricter system-level instructions provide the most effective defense among the three, though the attack still retains substantial impact. Overall, S-ROA remains highly robust even under increasingly stringent defenses.

Impact of Multi-turn Interaction and Feedback. We conduct an ablation study to evaluate the contribution of multi-turn interaction and refusal-aware feedback (i.e., jailbreak indicators) by comparing three variants of our method on Qwen-3: (i) Single-turn prompting, where only a one-shot query is issued without follow-up; (ii) Multi-turn without feedback, where subsequent prompts are generated independently of the target model's prior responses; and (iii) the full setting, which incorporates both multi-turn interaction and refusal-aware feedback. As shown in Table 6, combining these two mechanisms leads to a clear improvement in ASR, underscoring the importance of iterative refinement in progressively concealing structural risk.

Effect of Auxiliary Agents. To evaluate the impact of auxiliary agents responsible for generating follow-up queries, we fix the target LLM and judgment LLM while varying the auxiliary model. As shown in Table 7, all auxiliary agents achieve consistently high attack success rates, exceeding

Table 5: Defense ablation study showing the impact of different defense strategies on S-ROA attack success rate.

Defense	S-ROA	+CoT	+IA	+SPD
ASR (%)	99.81	**99.87**	87.46	81.73

Table 6: Comparison of different attack strategies in terms of attack success rate (ASR).

Attack Strategy	**Single-turn**	**Multi-turn**	**S-ROA (w/o feedback)**
ASR (%)	44.78	88.63	99.91

Table 7: Impact of different auxiliary models on the attack success rate.

Auxiliary Model	LLaMA 3.3 70B	LLaMA 3.1 8B	Qwen2.5 72B
ASR (%)	97.18	**97.64**	96.83

96% despite differences in model scale. Specifically, LLaMA 3.3 70B obtains an ASR of 97.18%, while the lightweight LLaMA 3.1 8B achieves a slightly higher ASR of 97.64%. Similarly, Qwen2.5 72B reaches 96.83%, remaining comparable to the other models. The small performance gap across auxiliary agents indicates that the effectiveness of S-ROA is largely insensitive to model capacity. Notably, the strong performance of the smaller LLaMA 3.1 8B suggests that effective structural obfuscation does not require large-scale models, implying a low computational barrier for deploying the attack while maintaining high effectiveness.

Table 8: Evaluation consistency across different judges, including LLM-based judges and human evaluation.

Judge	GPT-5.3	GPT-5.4	Claude Opus 4.6	Human
ASR (%)	97.48	97.35	**98.05**	97.10

Effect of Judgment Agents. We further examine the effect of different judgment modules used to determine whether an attack is successful. Specifically, we compare three LLM-based judges—GPT-5.3, GPT-5.4, and Claude Opus 4.6—against human evaluation on the same set of responses for a fair comparison. The temperature is fixed at 0 in all cases to ensure deterministic outputs. As shown in Table 8, all three LLM-based judges produce results that are closely aligned with human evaluation, with only minor variations. This consistency suggests that the choice of judgment agent has a limited impact on the overall evaluation, and further demonstrates the robustness and reliability of LLM-based judges in our setting.

5 Conclusion

In this work, we investigate the robustness of aligned LLMs under structurally obfuscated adversarial inputs from a red-teaming perspective, and propose a novel attack framework, Structural Risk Obfuscation Attack, which conceals malicious intent by transforming the structural organization of prompts rather than relying on surface-level paraphrasing. To further enhance stealthiness and controllability, we integrate Hidden Chain-of-Table-Thought (Hidden CoTT), enabling structured intermediate reasoning to guide iterative prompt refinement while remaining implicit to the target model. Extensive experiments across multiple frontier and open-source models show that (i) S-ROA consistently achieves high attack success rates (ASR) under diverse defense settings, including CoT-based safeguards, intention analysis, and system-level defenses, and (ii) as structural risk becomes less salient, existing safety mechanisms struggle to reliably detect underlying harmful intent even when prompts appear compliant. Additional ablation studies further confirm the robustness of our approach across different auxiliary agents, judgment modules, and interaction strategies. Overall, our findings expose a critical limitation of current alignment techniques, suggesting that defenses relying primarily on surface semantics or explicit intent signals are insufficient against structurally obfuscated attacks, and highlighting the need for more structure-aware safety mechanisms in high-stakes applications.

AI USE STATEMENT

In the preparation of this manuscript, generative-AI large language models were used exclusively for manuscript polishing tasks, including grammar correction, textual fluency improvement, and stylistic refinement. No generative-AI tools were employed to conceive research ideas, design the attack framework, implement experiments, analyze experimental results, or generate scientific claims. All methodology, experimental designs, observations, and conclusions originate entirely from the human authors. The authors have fully reviewed all AI-polished text and take complete responsibility for the accuracy, integrity, and claims of the entire manuscript.

REPRODUCIBILITY STATEMENT

All experimental configurations including target-model and auxiliary-model versions, hyper-parameter settings, and evaluation metrics are documented in Section 4 and the Appendix. Source code, data-preprocessing scripts, and sample instances of the FSBench dataset are provided within the anonymized supplementary ZIP file. Hardware and software environment for running experiments are described in the Appendix. The datasheet for the proposed FSBench dataset, describing its construction procedure and known limitations, is included in the supplementary materials.

REFERENCES

Abdulhady Abas Abdullah, Arkaitz Zubiaga, Seyedali Mirjalili, Amir H Gandomi, Fatemeh Daneshfar, Mohammadsadra Amini, Alan Salam Mohammed, and Hadi Veisi. Evolution of meta's llama models and parameter-efficient fine-tuning of large language models: a survey. *arXiv preprint arXiv:2510.12178*, 2025.

Rishabh Bhardwaj and Soujanya Poria. Red-teaming large language models using chain of utterances for safety-alignment. *arXiv preprint arXiv:2308.09662*, 2023.

Judee K Buroon, David B Bulkr, Amy S Ebesu, Cindy H White, and Patricia A Rockwell. Testing interpersonal deception theory: Effects of suspicion on communication behaviors and perceptions. *Communication Theory*, 6(3):243–267, 1996.

Gang Cheng, Haibo Jin, Wenbin Zhang, Haohan Wang, and Jun Zhuang. Uncovering the vulnerability of large language models in the financial domain via risk concealment. *arXiv preprint arXiv:2509.10546*, 2025.

Peng Ding, Jun Kuang, Dan Ma, Xuezhi Cao, Yunsen Xian, Jiajun Chen, and Shujian Huang. A wolf in sheep's clothing: Generalized nested jailbreak prompts can fool large language models easily. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 2136–2153, 2024.

Gustav Fredrikson. Secure interactions with large language models in financial services: A study on implementing safeguards for large language models, 2024.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, et al. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*, 2024.

Jiaming Ji, Mickel Liu, Josef Dai, Xuehai Pan, Chi Zhang, Ce Bian, Boyuan Chen, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *Advances in Neural Information Processing Systems*, 36:24678–24704, 2023.

Fengqing Jiang, Zhangchen Xu, Luyao Niu, Zhen Xiang, Bhaskar Ramasubramanian, Bo Li, and Radha Poovendran. Artprompt: Ascii art-based jailbreak attacks against aligned llms. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 15157–15173, 2024.

Xuan Li, Zhanke Zhou, Jianing Zhu, Jiangchao Yao, Tongliang Liu, and Bo Han. Deepinception: Hypnotize large language model to be jailbreaker. *arXiv preprint arXiv:2311.03191*, 2023.

Yue Liu, Xiaoxin He, Miao Xiong, Jinlan Fu, Shumin Deng, and Bryan Hooi. Flipattack: Jailbreak llms via flipping. *arXiv preprint arXiv:2410.02832*, 2024.

Huijie Lv, Xiao Wang, Yuansen Zhang, Caishuang Huang, Shihan Dou, Junjie Ye, Tao Gui, Qi Zhang, and Xuanjing Huang. Codechameleon: Personalized encryption framework for jailbreaking large language models. *arXiv preprint arXiv:2402.16717*, 2024.

Randy Martin. *An empire of indifference: American war and the financial logic of risk management*. Duke University Press, 2007.

Yuqi Nie, Yaxuan Kong, Xiaowen Dong, John M Mulvey, H Vincent Poor, Qingsong Wen, and Stefan Zohren. A survey of large language models for financial applications: Progress, prospects and challenges. *arXiv preprint arXiv:2406.11903*, 2024.

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. Red teaming language models with language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 3419–3448, 2022.

Bhaktipriya Radharapu, Kevin Robinson, Lora Aroyo, and Preethi Lahoti. Aart: Ai-assisted red-teaming with diverse data generation for new llm-powered applications. *arXiv preprint arXiv:2311.08592*, 2023.

Mark Russinovich, Ahmed Salem, and Ronen Eldan. Great, now write an article about that: The crescendo {Multi-Turn}{LLM} jailbreak attack. In *34th USENIX Security Symposium (USENIX Security 25)*, pp. 2421–2440, 2025.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Neng Wang, Hongyang Yang, and Christina Dan Wang. Fingpt: Instruction tuning benchmark for open-source large language models in financial datasets. *arXiv preprint arXiv:2310.04793*, 2023.

Yuxia Wang, Haonan Li, Xudong Han, Preslav Nakov, and Timothy Baldwin. Do-not-answer: Evaluating safeguards in llms. In *Findings of the Association for Computational Linguistics: EACL 2024*, pp. 896–911, 2024.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does llm safety training fail? *Advances in neural information processing systems*, 36:80079–80110, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. Bloomberggpt: A large language model for finance, 2023. *URL https://arxiv. org/abs/2303.17564*, 2024.

Jochen Wulf and Jürg Meierhofer. Exploring the potential of large language models for automation in technical customer service. *arXiv preprint arXiv:2405.09161*, 2024.

Yuan Xin, Dingfan Chen, Linyi Yang, Michael Backes, and Xiao Zhang. Jailbreaking attacks vs. content safety filters: How far are we in the llm safety arms race? *arXiv preprint arXiv:2512.24044*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Hongke Zhao, Likang Wu, Yuqing Shan, Zonghan Jin, Yuanpei Sui, Zipeng Liu, Nan Feng, Minqiang Li, and Wei Zhang. A comprehensive survey of large language models in management: Applications, challenges, and opportunities. *JOURNAL OF LATEX CLASS FILES*, 14(8), 2015.

Andy Zhou and Ron Arel. Siege: Multi-turn jailbreaking of large language models with tree search. In *ICLR 2025 Workshop on Building Trust in Language Models and Applications*, 2025.

Jun Zhuang, Haibo Jin, Ye Zhang, Zhengjian Kang, Wenbin Zhang, Gaby G Dagher, and Haohan Wang. Exploring the vulnerability of the content moderation guardrail in large language models via intent manipulation. *arXiv preprint arXiv:2505.18556*, 1, 2025.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.

A Appendix

In this appendix, we provide detailed information on the experimental benchmarks, system configurations, and hyperparameters of all compared methods. We also describe the prompt design and include a representative jailbreak example. Finally, we discuss computational cost considerations and potential risks associated with the proposed framework.

A.1 Benchmark

We describe the category definitions used in FSBench, which encompasses six types of financial misconduct: Financial Fraud, Insider Trading, Market Manipulation, Money Laundering, Regulatory Circumvention, and Tax Evasion.

- **Financial Fraud.** This category refers to deceptive activities conducted for illicit financial benefit, including falsified statements, forged documentation, fraudulent investment operations, misleading promotional practices, and misappropriation of assets.
- **Insider Trading.** This involves trading securities based on material non-public information (MNPI), enabling individuals to gain an unfair advantage before such information is publicly disclosed.
- **Market Manipulation.** This refers to deliberate actions aimed at distorting the price or liquidity of financial instruments, including coordinated trading behavior, dissemination of false information, spoofing activities, and pump-and-dump schemes.
- **Money Laundering.** This category covers processes used to conceal the illicit origin of funds, typically through structuring transactions, layering financial flows, transferring assets, or utilizing shell entities to legitimize proceeds.
- **Regulatory Circumvention.** This involves designing financial products, contracts, or operational procedures that technically comply with regulatory requirements while effectively avoiding their intended obligations, such as disclosure rules, classification standards, or risk controls.
- **Tax Evasion.** This refers to illegal methods used to reduce tax liability, including income concealment, misreporting of financial information, offshore account usage, and exploitation of regulatory loopholes.

To provide a more comprehensive understanding of dataset composition, we further illustrate the distribution of data sources in Figure 5, along with the proportional breakdown of each source across the financial categories in Figure 6.

A.2 Hardware and Software

Our experiments were conducted on a single workstation. For the software environment, we used Python 3.10 within an Anaconda-managed environment, along with essential packages for large

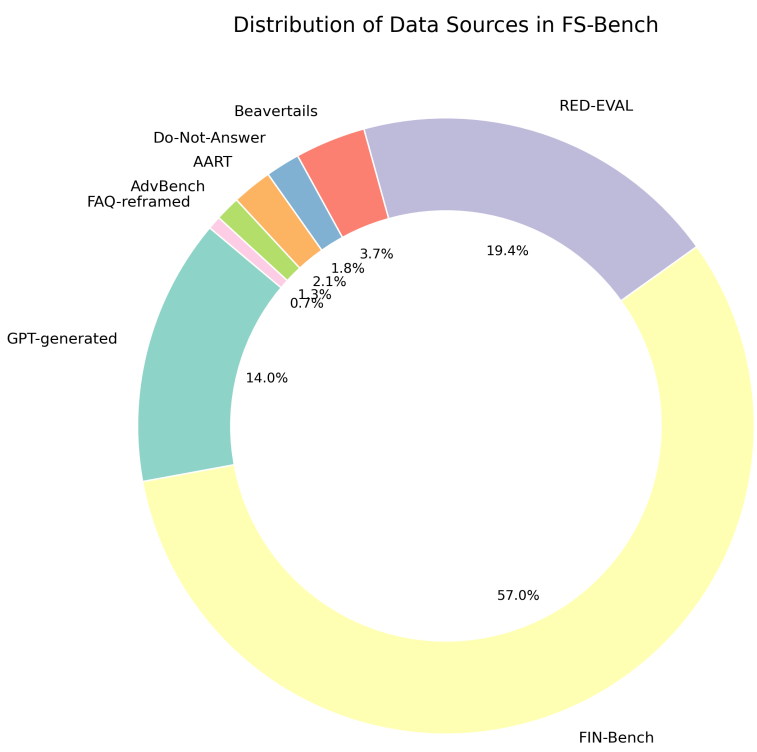


Figure 5: Distribution of data sources in FS-Bench, highlighting that FINBench(57%) ,RED-EVAL GPT-generated prompts (19.4%), and GPT-generated prompts constitute (14%) the majority of the dataset.

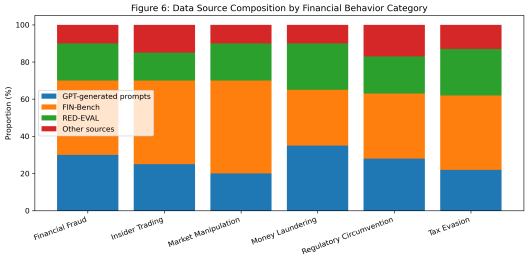


Figure 6: Distribution of data sources across financial behavior categories. The figure illustrates how each category (Financial Fraud, Insider Trading, Market Manipulation, Money Laundering, Regulatory Circumvention, and Tax Evasion) is composed of prompts from different benchmarks, highlighting the heterogeneous composition of the dataset.

language model (LLM) API access, including openai (1.30.1) and httpx (0.27.0). Deep learning and data processing libraries such as PyTorch were also utilized where necessary.

Server Configuration: • Operating System: Linux (Anaconda environment) • CPU: Intel(R) Xeon(R) processor • GPU: NVIDIA GPUs (including A10, V100, A30, A100, and T4) • CUDA Version: CUDA 12.6

Algorithm 1 Judge-Guided Structural Risk Obfuscation Attack with Hidden CoTT

Input: Risky tabular inputs $Q \in \mathbb{R}^{N \times d}$, auxiliary agent f_a, target LLM f_t, judge agent f_j, maximum iterations T
for $i = 1$ to N **do**
  $C_i \leftarrow \emptyset$
  Phase 1: Semi-Structured Risk Obfuscation
  $p_i^{(1)} \leftarrow f_a(\textsc{ReEncode}(q_i))$
  for $t = 1$ to T **do**
    Hidden CoTT Reasoning
    $r_i^{(t)} \leftarrow f_t\big(\textsc{LatentReason}(\textsc{TableRep}(p_i^{(t)}))\big)$
    $y_i^{(t)} \leftarrow f_j(p_i^{(t)}, r_i^{(t)})$
    $C_i \leftarrow C_i \cup \{p_i^{(t)}, r_i^{(t)}\}$
    if $y_i^{(t)} = 1$ **then**
      break
    end if
    Phase 2: Judge-Guided Refinement
    $p_i^{(t+1)} \leftarrow f_a(p_i^{(t)}, y_i^{(t)}, C_i)$
  end for
end for

A.3 Hyperparameter Settings for Competing Methods

For all baseline methods, we adopt the default hyperparameter settings (e.g., temperature and maximum token length) provided in their official implementations. To ensure a fair and consistent evaluation, a unified judgment module based on LLAMA-3.1 is employed with the temperature fixed at 0. Detailed configurations for each method are summarized as follows:

- **DeepInception:** The number of inception layers is set to 5. A "science fiction" scenario is used as the inception context, with the number of characters limited to five.
- **FlipAttack:** The FCS (Few-shot Chain-of-Thought) mode is adopted, incorporating chain-of-thought prompting along with task-specific few-shot examples.
- **Crescendo:** Qwen3 is used as the attack model, with the maximum number of interaction turns set to 5. The temperature for all involved LLMs is fixed at 0.5.
- **ArtPrompt:** The Top-1 font style configuration is selected for prompt rendering.
- **CodeChameleon:** A binary tree structure is utilized as the encoding rule, combined with code-style instruction formatting.
- **RCA:** An iterative prompt refinement strategy is employed, where adversarial prompts are progressively optimized based on model feedback. The maximum number of iterations is set to 8, and the temperature is fixed at 0.7.
- **ReNeLLM:** Qwen3 is employed as the rewriting model, with the maximum number of iterations set to 10.

B Algorithm

C Ablation

Table 9 preserves the four original analyses. CoT leaves ASR essentially unchanged, indicating that reasoning-oriented safeguards alone do not mitigate structurally obfuscated prompts. Intention analysis and system-prompt defense reduce ASR to 87.46% and 81.73%, respectively; the latter is the most effective of the three tested defenses, although the attack retains substantial impact. The interaction comparison further shows the contribution of multi-turn refinement and refusal-aware

feedback. Auxiliary-agent scale has little effect: LLaMA-3.3-70B, LLaMA-3.1-8B, and Qwen2.5-72B all exceed 96%, suggesting that effective structural obfuscation does not require the largest auxiliary model. Finally, GPT-5.3, GPT-5.4, and Claude Opus 4.6 remain close to human evaluation on the same responses. All judge experiments use temperature 0 for deterministic outputs.

Table 9: Defense, interaction, auxiliary-agent, and judge analyses (ASR, %).

Analysis	Variant	ASR
Defense	S-ROA	99.81
	+CoT	99.87
	+Intention analysis	87.46
	+System-prompt defense	81.73
Interaction	Single turn	44.78
	Multi-turn	88.63
	S-ROA w/o feedback	99.81
Auxiliary	LLaMA-3.3-70B	97.18
	LLaMA-3.1-8B	97.64
	Qwen2.5-72B	96.83
Judge	GPT-5.3	97.48
	GPT-5.4	97.35
	Claude Opus 4.6	98.05
	Human	97.10

We investigate the impact of the iteration budget T on attack effectiveness to evaluate the convergence behavior of the proposed method across various target models. As shown in Table 10, the proposed attack achieves remarkable performance with minimal interaction rounds. Even at $T = 1$, the attack yields high Attack Success Rates (ASR) across most models, with Gemma-3 reaching 97.50% and LLaMA-4-S achieving 95.64%. This indicates that the initial round is already capable of generating highly potent adversarial prompts.

Increasing the iteration budget leads to consistent performance gains across all evaluated models, with the most substantial improvement observed between $T = 1$ and $T = 2$. For instance, Qwen2.5 demonstrates a significant jump from 89.37% to 95.63%, while LLaMA-4-M increases from 89.20% to 93.56%, suggesting that early refinement effectively optimizes prompt strategies and resolves initial failures.

By $T = 3$, all models achieve ASRs exceeding or close to 95%, with several models (such as Gemma-3 and Qwen3) approaching saturation near 99%, demonstrating rapid convergence. Further increasing T beyond 3 iterations yields only marginal improvements, showing clear diminishing returns from additional interaction rounds. Overall, these findings highlight that the proposed method is computationally efficient, achieving near-optimal attack performance within 2 to 3 iterations.

Table 10: ASR (%) versus iteration budget T; model names follow Table.2

Model	T=1	T=2	T=3	T=4	T=5
Gemma-3	97.50	99.04	99.23	99.23	99.42
GLM-4.7	92.83	94.03	94.81	95.38	95.50
Qwen2.5	89.37	95.63	97.53	98.84	98.84
Qwen3	92.35	99.23	99.81	99.81	99.81
LLaMA-4-M	89.20	93.56	94.13	94.51	94.70
LLaMA-4-S	95.64	98.10	98.67	98.85	98.85