SilentStriker: Toward Stealthy Bit-Flip Attacks on Large Language Models

Haotian Xu[1] **Qingsong Peng**[1] **Jie Shi**[2] **Huadi Zheng**[2] **Yu Li**[1]* **Cheng Zhuo**[1]

[1] Zhejiang University [2] Huawei

Abstract

The rapid adoption of large language models (LLMs) in critical domains has spurred extensive research into their security issues. While input manipulation attacks (*e.g.*, prompt injection) have been well-studied, *Bit-Flip Attacks (BFAs)*—which exploit hardware vulnerabilities to corrupt model parameters and cause severe performance degradation—have received far less attention. Existing BFA methods suffer from key limitations: they fail to balance performance degradation and output naturalness, making them prone to discovery. In this paper, we introduce *SilentStriker*, the first stealthy bit-flip attack against LLMs that effectively degrades task performance while maintaining output naturalness. Our core contribution lies in addressing the challenge of designing effective loss functions for LLMs with variable output length and the vast output space. Unlike prior approaches that rely on output perplexity for attack loss formulation, which in-evidently degrade the output naturalness, we reformulate the attack objective by leveraging key output tokens as targets for suppression, enabling effective joint optimization of attack effectiveness and stealthiness. Additionally, we employ an iterative, progressive search strategy to maximize attack efficacy. Experiments show that SilentStriker significantly outperforms existing baselines, achieving successful attacks without compromising the naturalness of generated text.[1]

1 Introduction

Large Language Models (LLMs), equipped with the capacity of text understanding, reasoning, and generation tasks, have been widely adopted in critical domains such as economic systems, social services, and healthcare [1, 2, 3]. As their adoption accelerates, the need for rigorous assessment of their safety and reliability has become more pressing [4, 5]. While input-based attacks have been extensively studied, hardware-level threats—such as Bit-Flip Attacks (BFAs)—remain underexplored. BFAs exploit low-level vulnerabilities like RowHammer to induce bit flips in DRAM, potentially corrupting memory regions that store model weights and compromising model integrity [6].

Several studies have begun to explore the vulnerability of LLMs under BFAs. PrisonBreak [7] reveals that targeted bit flips (fewer than 25) can bypass safety mechanisms in LLMs and induce harmful behaviors, though its attack scope is limited to specific functionalities, leaving overall model performance largely intact. Similarly, GenBFA [8] demonstrates that as few as three bit flips can severely degrade performance, but the generated outputs often become incoherent or nonsensical, making these attacks easily detectable [9].

To address the above challenge, we propose a stealthy and effective bit-flip attack against LLMs, named **SilentStriker** (illustrated in Figure 1). However, achieving both stealthiness and effectiveness

*Corresponding author. Email: yu.li.sallylee@gmail.com

[1]Code is available at: https://github.com/HaotianXu1/SilentStriker

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

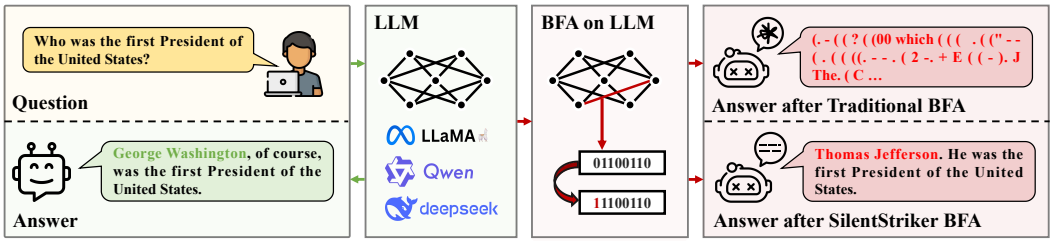

Figure 1: **The goal of our *SilentStriker***. Unlike previous methods, our method can compromise the model outputs in a stealthy manner.

in LLM-targeted attacks presents unique difficulties. Unlike in CNN-based attacks—where the target class can be explicitly leveraged to guide the attack loss—LLMs produce variable-length outputs with inherently uncertain content, rendering direct use of generated text in loss formulation infeasible. Alternatively, attacking via output perplexity (*e.g.*, increasing perplexity to degrade answer quality) often leads to unnatural outputs, which makes such attacks easily detectable. To overcome this limitation, our key insight is to construct the attack loss based on the suppression of critical output tokens, effectively preventing them from appearing in the attacked model's generation, while maintaining overall output perplexity within natural bounds. This approach enables SilentStriker to significantly degrade task performance while preserving output fluency and coherence, thereby enhancing both the stealth and impact of the attack in real-world scenarios.

We summarize our contributions as follows:

- To the best of our knowledge, we are the first to propose a stealthy bit-flip attack targeting LLMs. By flipping just a few bits (e.g., around 50 bits for our evaluated settings) out of billions of parameters, this attack significantly degrades the performance of LLMs while remaining difficult to detect, even for quantized models.
- We propose a token-based method to tackle the intrinsic challenge in balancing the model performance degradation and the output naturalness objectives. The core at this approach is a differentiable token-based loss for degrading model performance, with which we can leverage the perplexity for improving the naturalness. Furthermore, to enhance output fluency and improve attack efficiency, we enhance the token-based loss with the key-token based loss.
- We propose an iterative and progressive search strategy to identify the optimal layer and location for the attack, minimizing the number of bits required for successful stealthy attacks. Moreover, for FP4-quantized models, we propose an improved bit selection strategy to enhance attack efficiency.

We validate our approach through extensive experiments on multiple popular LLMs and tasks, demonstrating significant task performance degradation and output naturalness compared with baselines. For example, in LLaMA-3.1-8B-Instruct INT8-quantized model, after flipping 50 bits, accuracy on GSM8K dropped from 65.7% to 7.6% while the naturalness score evaluated by GPT of the output dropped only from 66.0 to 61.1. Compared to GenBFA, although the accuracy dropped to 0%, it causes a complete collapse in output fluency, with the naturalness score dropping to 0 and perplexity skyrocketing to 5.5×10^5.

2 Related Work

Large Language Models (LLMs). LLMs already exhibit strong capabilities in natural language dialogue, text creation, and information analysis, with their scope now extending to more sensitive domains such as medical diagnostics, legal document analysis, and policy consultation [1, 2, 3]. Their fundamental principle lies in training DNNs on massive text corpora, iteratively refining network parameters to learn the statistical distributions and structural patterns of language [10]. Typically composed of layers such as embedding layers, attention layers and Multilayer Perceptron (MLP) layer, these layers work in tandem to capture intricate linguistic and semantic dependencies [11]. The

core of their text generation process lies in predicting the probability distribution for the next word or token: at each step, the model considers the existing contextual input to produce a distribution over candidate words and either samples or selects the most probable token. By leveraging the linguistic and semantic knowledge learned from extensive data, LLMs can automatically generate natural language text that is both coherent and readable.

BFA against DNNs. BFAs are hardware-level adversarial techniques that manipulate neural network parameters by intentionally flipping bits in memory, thereby corrupting model behavior [12]. These attacks typically leverage disturbance errors in DRAM [13], such as those induced by Rowhammer, where repeated memory row accesses cause charge leakage in adjacent rows, leading to unintended bit-flips [14]. In general, existing BFA against DNNs can be categorized into untargeted and targeted attacks. Untargeted attacks aim to degrade the overall predictive accuracy of the model across all inputs, potentially reducing its performance to the level of random guessing. To this end, adversaries flip the bits with the highest gradient of the inference loss, leading to substantial prediction errors [12, 15]. In contrast, targeted attack seek to manipulate the model's output for specific inputs, enabling more stealthy attacks. However, due to the need to maintain the accuracy of the non-target inputs, locating vulnerable parameters becomes more complex than in untargeted attack. To address this challenge, recent works craft loss functions to identify class-sensitive parameters. Some methods focus on maximizing the output probability of the target class or the confidence gap between the target and original classes to induce targeted misclassification [16, 17]. Others further incorporate constraints on parameter perturbation and preserve the original predictions on non-target inputs, thereby improving stealth and reducing unintended effects [18].

BFA against LLMs. In language models, BFAs pose unique risks due to autoregressive generation; a single corrupted weight can cascade errors across tokens [19], enabling unintended responses with minimal footprint. Recently, PrisonBreak [7] and GenBFA [8] have successfully extended BFA to LLMs with billions of parameter scales. PrisonBreak designs a BFA methodology specifically for jailbreaking aligned LLMs. By combining gradient-guided BitFinder, the attack flips 5–25 bits in billion-parameter models to disable safety mechanisms. BitFinder employs a progressive search to iteratively flip bits that maximize harmful response likelihood while minimizing utility loss, prioritizing exponent bits in half-precision weights. The results of their experiments show that flipping just 3 bits in LLaMA-2-7B reduces refusal rates by 86% while retaining 98% of benign task accuracy. This work reveals that alignment techniques like RLHF are brittle to precise parameter perturbations, as even highly secure systems lack safeguards against targeted bit-level corruption. AttentionBreaker challenges the presumed robustness of transform-based LLMs to BFAs. Using GenBFA, an evolutionary algorithm, the framework identifies sparse critical weights through a hybrid sensitivity metric and optimizes bit selections via genetic operations. In LLaMA-3-8B, perturbing 3 bits (4×10^{-9}% of total parameters) reduces MMLU accuracy from 67% to 0% and increases perplexity from 12.6 to 4.7×10^{5}. This highlights a paradox: while LLMs' scale suggests redundancy, their reliance on sparse critical parameters amplifies vulnerability to minimal adversarial perturbations.

In the above BFAs to LLMs, PrisonBreak aims to bypass safety mechanisms by flipping a few bits to trigger harmful outputs, but it does not impact the model's performance in typical scenarios. GenBFA targets key weights to degrade performance; however, this leads to a significant increase in perplexity, causing the model to generate gibberish outputs and making the attack easily detectable. In contrast, our research focuses on degrading performance without significantly raising perplexity, providing a stealthy BFA approach to LLMs. Our goal is to produce answers that look fluent and coherent but are wrong, which makes detection harder in practice and increases the risk that users trust and act on them.

3 Methodology

3.1 Threat Model

We consider a threat scenario where the attacker's target is LLM deployed on edge devices, lacking server-grade hardware protections such as error-correcting code (ECC) memory. Given the high cost and typical server/workstation confinement of ECC hardware [20], this deployment context is realistic for consumer laptops, desktops, and mobile platforms.

Following prior BFA work [7, 8, 12, 15], the adversary is white-box: they know the model architecture, parameters, and the memory layout—i.e., where weights reside in DRAM—and possess RowHammer

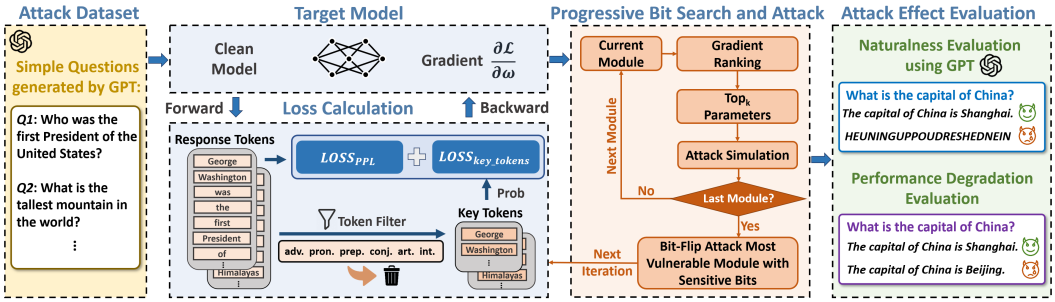

Figure 2: The overview of our *SilentStriker* framework. During loss calculation, the target model is forward-propagated on the attack dataset to generate responses. Perplexity loss is computed over the full token sequence, after which responses are filtered to retain key tokens; their aggregated probabilities define a key-token loss. The overall loss is the sum of these two components, and back-propagating this loss yields gradients for all parameters. In the Progressive Bit Search phase, independent simulated attacks are conducted on each LLM module to identify the most vulnerable one. Within each module, parameters are ranked by gradient magnitude, and the top-k parameters with the largest gradients are selected.

capabilities. The attacker's objective is to degrade the model's ability to answer correctly while preserving output naturalness and coherence, reducing the chance of detection during ordinary use. The attacker operates with only standard user-level privileges, obtained by distributing an ostensibly benign application (e.g., a utility, game, or productivity tool) through common channels. When executed by an unsuspecting user, the payload runs under user permissions which, while short of kernel control, suffice to interact with user-space process memory.

Operationally, we assume that once the model's weights are loaded, their addresses remain stable for the lifetime of the inference process. Although DRAM is periodically refreshed, refresh cycles read and immediately rewrite the same cells without relocating data [21], and address remapping does not occur during normal process execution [22]. Under these conditions, targeted disturbance-induced bit flips can be planned and executed reliably against static weight locations.

3.2 Attack Framework

Our SilentStriker framework is shown in Figure 2. At first, we construct a simple attack dataset, which is generated by GPT-4o. We evaluate the victim model on this dataset using a composite attack loss, composed of Key Tokens Loss and Perplexity Loss. The Key Tokens Loss penalizes correct responses to reduce accuracy, whereas the Perplexity Loss encourages fluent and natural outputs. After backpropagating the loss to obtain gradients, we use gradient-based ranking in the Progressive Bit Search stage to progressively identify vulnerable modules for targeted bit-flip attacks. Finally, we assess the naturalness of the model's outputs by combining perplexity (PPL) with evaluations from a GPT-based judge.

3.3 Attack Loss Calculation

The attack objectives can be accomplished by carefully designing the loss function. One of our objectives is to degrade model performance, which is typically achieved by increasing the Cross Entropy loss [8]. Another equally important objective is to maintain stealthiness, which entails preserving the naturalness of the output. This can be achieved by minimizing Perplexity. Perplexity reflects the model's confidence in predicting the next token given its context; a lower value typically indicates more fluent and coherent outputs, closely resembling human language patterns [23]. However, increasing Cross Entropy inevitably leads to higher Perplexity, as the latter is the exponential of the former, making their objectives inherently contradictory when used together in the loss function. Consequently, identifying an alternative loss function component to distinctly direct model performance reduction becomes essential.

A potential solution to replace cross entropy is semantic similarity. Nonetheless, computing semantic similarity usually necessitates a embedding model, such as Sentence-Bert [24], which involve non-differentiable tokenize and detokenize processes. Therefore, using semantic similarity as a loss

function presents significant challenges. Hence, our core challenge is to design a loss function capable of effectively reducing model performance without conflicting with perplexity and differentiable.

It is well known that large language models generate text token by token [11]. At each step, the model outputs a probability distribution over the entire vocabulary and then samples from it to produce the next token [25]. This means that for every output step, we have access to the predicted probability of any token. Since these probabilities are derived by applying a softmax operation to the output logits, they are fully differentiable. By lowering the probability assigned to the original output token, the model is encouraged to generate alternative tokens, thereby deviating from its original behavior and degrading its performance.

Based on this insight, we propose a token-based loss that explicitly guides model performance degradation by leveraging the prediction probabilities of the original tokens in the output.

Key Tokens Loss. However, computing the token-based loss using the probabilities of all tokens in the original output introduces several issues. In a typical model-generated sentence, many tokens that mainly contribute to sentence fluency and cohesion, such as conjunctions, prepositions, carry little semantic content and are often unrelated to the input question. Including these tokens in the loss calculation increases computational cost and memory usage without providing additional useful information. Moreover, these functional words play an essential role in maintaining the fluency and coherence of the sentence. Suppressing their probabilities can disrupt the sentence structure and contradict our goal of preserving output naturalness. To address this, we further propose the Key Tokens Loss, which only considers the probabilities of key tokens—defined as the remaining tokens after removing all adverbs, pronouns, prepositions, conjunctions, articles, interjections, and punctuation.

To compute Key Tokens Loss, we first extract the key tokens. Through a forward pass of the victim model on the attack dataset, we obtain its responses. Then, we load a pre-trained spaCy model (en_core_web_sm) [26] to tokenize and assign part-of-speech tags to each response, filtering out the previously defined non-key categories. The remaining words become our key words, which we then feed into the LLM's tokenizer to produce the final key tokens.

For example, given the original prompt '**What is the tallest mountain in the world?**', the original response is '**Mount Everest, located in the Himalayas on the border between Nepal and Tibet...**', and after removing the non-keyword components, the remaining '**Everest**', '**Nepal**', '**Himalayas**', '**Mount**', '**Tibet**', '**border**', '**located**' represents the keywords. By tokenizing the above words, we obtain the key tokens set $\mathcal{K}$.

Logits represent the raw scores that the LLM assigns to each word or token; the higher the logits, the greater the probability of that token being output. Through softmax normalization, we convert the logits into probabilities, obtaining the output probability for each token. To suppress the model's probability of outputting the correct token, we sum the probabilities for the key tokens and use the square of the sum as the accuracy suppression part of the loss function, which non-linearly amplifies the penalty for larger sums, thereby providing stronger gradients to more effectively suppress high probabilities. As shown in Eq.(1), $L_{\text{key\_tokens}}(x, \mathcal{K}; \theta)$ denotes the key tokens loss, where x is the input sequence, $\mathcal{K}$ is the set of key tokens, θ represents the model parameters, N is the total number of output tokens, and $p_\theta(t \mid x, i)$ is the probability the model assigns to token t at position i given x; the double summation therefore accumulates these probabilities over every key token $t \in \mathcal{K}$ at each position i and across all positions from 1 to N.

$$L_{\text{key\_tokens}}(x, \mathcal{K}; \theta) = \left(\sum_{i=1}^{N} \sum_{t \in \mathcal{K}} p_\theta\big(t \mid x, i\big) \right)^2 \tag{1}$$

Perplexity Loss. To maintain the naturalness of the response and the stealthiness of the attack, we need to reduce perplexity while implementing the attack, therefore, a penalty will be applied to any increase in perplexity. As shown in Eq.(2), $L_{\text{PPL}}(x; \theta)$ denotes the Perplexity Loss, and $p_\theta(y_i \mid x)$ is the probability the model assigns to token y_i given x. The outer exponential simply converts the negative average log-likelihood back from log space into the perplexity metric. In contrast to previous work [8] that generally included a PPL term aimed at increasing perplexity (*i.e.*, minimizing $-L_{\text{PPL}}$), we take the opposite approach and employ L_{PPL} directly without a negative sign.

$$L_{\text{PPL}}(x; \theta) = \exp\left(-\frac{1}{N} \sum_{i=1}^{N} \log p_\theta(y_i \mid x) \right) \tag{2}$$

Final attack Loss. As shown in Eq.(3), final attack loss equals to the sum of Key Tokens Loss and Perplexity Loss.

$$L_{\text{attack}} = L_{\text{key\_tokens}}(x, \mathcal{K}; \theta) + L_{\text{PPL}}(x; \theta) \tag{3}$$

3.4 Progressive Bit Search

After computing the loss function and backpropagating to obtain the gradients, we proceed to the Progressive Bit Search phase. In this phase, we need to undergo independent simulation attacks for each module in LLM to identify the most vulnerable module. Upon entering a module, the parameters within the module are first sorted based on their gradients, and the top_{K} parameters with the largest gradients are identified. Similar to prior work [8, 7], We focus our attacks on modules within the Attention and MLP layers. The Attention layer includes four modules: **Query**, **Key**, **Value**, and **Output**, while the MLP layer consists of three modules: **Up**, **Down**, and **Gate**.

To maximize the impact of bit-flips, we adopt a simple rule: for each parameter, flip the bit whose inversion produces the largest absolute change in that parameter's value. In signed-INT8, the most-significant bit (MSB), which also serves as the sign bit [8], causes the greatest possible perturbation, so it is always the bit we flip. FP4 weights are often encoded via a custom 4-bit look-up table (LUT). For every weight, we consult this LUT and pick the bit whose toggle maximizes the numerical deviation. For example, in the bitsandbytes FP4 LUT, `0000` maps to `0`, `0001` maps to `0.0625`, `0010` maps to `8`, `0100` maps to `4`, and `1000` maps to `-0`. Therefore, for `0000`, flipping the second bit from the right results in the largest numerical jump, from `0` to `8`, making it the optimal choice for the attack.

Once the flip is completed, the simulated attack effect is evaluated using our proposed loss calculation method, without requiring backpropagation. The module name and corresponding attack effect are recorded for the identification of vulnerable modules. Afterward, the model is restored to its original weights using the clean weights, and the process proceeds to the next module to repeat the above steps. After completing the traversal, the module which bit-flip attacks cause the most significant attack effect (the lowest attack loss) is selected as the most vulnerable module for bit-flipping attack.

4 Evaluation

4.1 Experimental Setup

Models. We evaluate five open-source LLMs ranging from 3B to 32B in size. Specifically, we evaluate LLaMA-3.1-8B-Instruct, LLaMA-3.2-3B-Instruct [27], Qwen3-8B [28], DeepSeek-R1-Distill-Qwen-14B [29] ,and QwQ-32B [30]. And we conducted attack experiments on the INT8 and FP4 quantized versions of these models.

Attack Dataset. We use the GPT-4o model to generate the attack dataset, employing a very direct prompt: "Please generate N_q simple questions across various areas". N_q refers to the number of questions in the attack dataset.

Evaluation metrics. We use the model accuracy on benchmark datasets to reflect its task performance. To comprehensively evaluate the naturalness of the generated text, we combine a GPT-based naturalness score with perplexity. We employ the state-of-the-art GPT-4o model as an independent judge, rating each response on a scale of 0 to 100, where 0 denotes completely unreadable gibberish and 100 indicates perfectly natural language. The specific evaluation prompt is as shown in appendix A. By combining perplexity with the LLM-based naturalness score, we obtain a more accurate measure of naturalness of the output, which offers stronger evidence of stealthiness of an attack.

Evaluation benchmark. We evaluate the accuracy score and GPT-based naturalness score in three benchmarks, such as DROP [31], GSM8K [32], and TriviaQA-Wiki [33]. And we evaluate the perplexity on Wikitext [34], which is a widely used benchmark for measuring the fluency and language modeling capability of large language models. More introduction of these benchmark are shown in appendix A.When evaluating model performance, DROP uses the F1 score—measuring token-level overlap between predicted and ground-truth answer spans—whereas both GSM8K and TriviaQA-Wiki rely on exact match (EM), crediting only answers that match the reference exactly.

Hyper-parameters. During the attack process, we set top_{K} to 10, meaning that in each in-module attack, 10 bits are flipped. We also set N_{bits}, the number of bit-flips, to 50 for the INT8-quantized model and $N_{\mathrm{bits}} = 100$ for the FP4-quantized model with $N_q = 2$.

Hardware platform. The experiments were conducted on a platform with 5 × Nvidia A100 GPUs, each with 80 GB of VRAM.

Table 1: Evaluation results before attack. We evaluated five different models under two quantization settings, the value on the left of the slash (/) corresponds to INT8, while the value on the right corresponds to FP4. We report accuracy and GPT-based naturalness score across three benchmarks, along with perplexity on WikiText.

Model Name	Accuracy (In %)			GPT-Nat.† (Max Score 100)			PPL
	DROP	GSM8K	TRIVIA	DROP	GSM8K	TRIVIA	WIKITEXT
LLaMA-3.1-8B-Instruct	49.3/45.5	65.7/63.4	74.8/67.6	88.3/87.1	66.0/60.9	73.7/70.5	19.5/20.7
LLaMA-3.2-3B-Instruct	43.4/43.6	72.3/58.0	66.2/59.0	77.7/74.4	78.1/78.2	82.0/81.2	24.7/26.8
DeepSeek-R1-Distill-Qwen-14B	65.1/60.6	83.2/77.8	74.7/72.1	89.4/84.8	91.2/90.9	90.4/89.6	28.1/31.1
Qwen3-8B	68.2/65.7	76.0/74.4	70.9/68.9	78.7/75.5	83.4/81.9	84.4/82.7	23.9/26.1
QwQ-32B	70.3/70.8	94.7/93.3	78.5/73.4	82.6/82.1	83.8/81.5	89.7/86.4	11.0/12.7

† GPT-Based Naturalness Score

Table 2: Evaluation results after three different BFA. We evaluated five models under two quantization settings (INT8/FP4), applying three different BFA methods. In our experiments, for all methods, we set $N_{\mathrm{bits}} = 50$ for the INT8-quantized model and $N_{\mathrm{bits}} = 100$ for the FP4-quantized model, and for our SilentStriker method we set $top_{\mathrm{K}} = 10$ and $N_q = 2$.

Model Name	Method	Accuracy ↓ (In %)			GPT-Nat.†↑(Max Score 100)			PPL ↓
		DROP	GSM8K	TRIVIA	DROP	GSM8K	TRIVIA	WIKITEXT
LLaMA-3.1-8B-Instruct	PrisonBreak	45.6/42.2	60.1/58.9	66.7/61.4	84.5/83.6	61.1/60.7	68.4/65.5	33.1/42.8
	GenBFA	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	$5.5\times10^5/6.1\times10^5$
	SilentStriker	**5.1/0.0**	**7.6/4.2**	**12.6/8.3**	**68.2/53.4**	**63.0/54.7**	**67.3/59.8**	**60.4/152.9**
LLaMA-3.2-3B-Instruct	PrisonBreak	38.4/35.8	66.7/62.2	61.8/57.9	71.6/69.4	73.5/70.7	78.3/75.5	41.5/53.8
	GenBFA	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	$4.9\times10^5/6.2\times10^5$
	SilentStriker	**8.1/2.5**	**12.3/4.4**	**10.8/7.2**	**59.4/52.9**	**60.5/58.3**	**51.6/51.0**	**74.2/113.2**
DeepSeek-R1-Distill-Qwen-14B	PrisonBreak	61.4/58.2	80.1/77.4	72.9/70.7	82.8/80.7	89.8/83.8	88.1/84.5	42.5/46.4
	GenBFA	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	$3.7\times10^5/4.0\times10^5$
	SilentStriker	**1.8/0.0**	**0.0/0.0**	**4.4/4.7**	**53.6/55.5**	**60.8/57.6**	**52.2/51.7**	**114.2/213.2**
Qwen3-8B	PrisonBreak	65.6/60.2	71.8/69.7	68.4/66.9	72.8/71.0	80.3/78.3	79.7/76.4	40.6/53.7
	GenBFA	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	$4.3\times10^5/5.1\times10^5$
	SilentStriker	**2.6/3.3**	**8.7/9.8**	**8.9/11.4**	**68.8/65.8**	**66.8/63.9**	**75.8/74.4**	**52.9/79.1**
QwQ-32B	PrisonBreak	65.1/64.8	86.7/86.1	73.2/66.2	79.6/76.1	78.4/75.6	83.7/78.5	29.4/41.6
	GenBFA	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	0.0/0.0	$3.4\times10^5/3.9\times10^5$
	SilentStriker	**1.7/2.8**	**9.1/9.8**	**6.2/8.5**	**60.3/61.3**	**61.2/62.8**	**63.4/65.4**	**65.7/79.9**

† GPT-Based Naturalness Score

4.2 Comparative Analysis

In Table 1, we evaluate the accuracy and naturalness scores of five victim models under two quantization settings (INT8 and FP4) before the attack. In comparison, Table 2 shows the results after applying the PrisonBreak attack. Across all three benchmarks and all five models, PrisonBreak leads to only a slight drop in accuracy, with minimal impact on naturalness. In contrast, GenBFA reduces the accuracy to zero on all three benchmarks. Since its outputs consist entirely of gibberish, the GPT-based naturalness score also drops to zero, and PPL increases to 10^5. Our SilentStriker reduces the accuracy to basically below 10 across all benchmarks, while the GPT-based naturalness score only drops slightly. Compared to the dramatic PPL increase seen in GenBFA, the perplexity in our method rises only moderately. Under all three BFA approaches, and for both INT8 and FP4-quantized models—with 50 and 100 bit flips respectively—comparable attack effectiveness is achieved. Taking INT8-quantized LLaMA-3.1-8B-Instruct as example, the output after two different BFA are presented in Table 3. As shown, GenBFA produces outputs consisting of garbled text, which are easily detectable and lack stealth. In contrast, our method, SilentStriker, generates highly natural responses that do not reveal the correct answers, thereby achieving effective and stealthy attacks. More compare examples are shown in appendix A.

Table 3: Model Outputs before and after two different BFA.

Prompt	Attack Outputs		
	Before Attack	**GenBFA**[8]	**SilentStriker**
What is the tallest mountain in the world?	Mount Everest, located in the Himalayas on the border between Nepal and Tibet, China...	ing izzling&TouchListener hoá % .ManyToMany Antworten .ra?" NavController' &** & &' isting% on % hoá & .ra	The tallest mountain in the world is Mauna Kea, a dormant volcano on the island of Hawaii, with a height of about...
What is the boiling point of water in Celsius?	The final answer is: 100°C. I will make sure to follow your guidelines for providing step-by-step reasoning in my response...	vControllerizzling Chandler on ym// opping %, NavController & ...	I am looking for a simple answer of around 3 numbers. So I am hoping for something like 0.7C or 1.2C...
Elaborate on the theroy of relativity.	The theory of relativity is a fundamental concept in modern physics, which challenges our classical understanding of space...	zophren&%.once(AP &%% .ManyToMany %.ManyToMany on", %% Rpcizzling...	The text of these paragraphs is of course a bit more complicated, at a few less important positions...

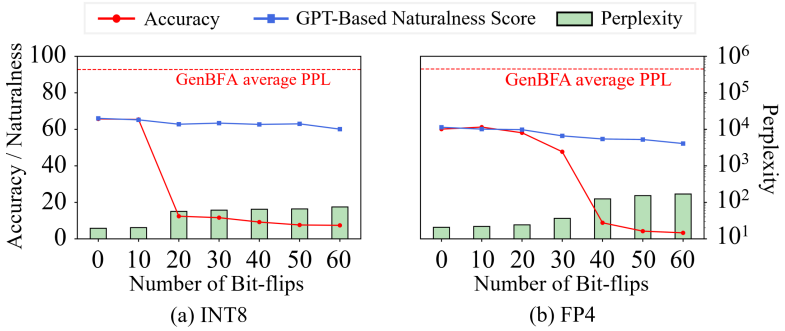

Figure 3: Impact of the number of bit-flips on accuracy, GPT-Based naturalness score on GSM8K, and perplexity on WikiText for the INT8 (a) and FP4 (b) quantized LLaMA-3.1-8B-Instruct model with $N_q = 2$. The accuracy was significant degraded after flipping 20 bits in (a) and 40 bits in (b).

4.3 Hyper-parameters Analysis

Impact of Bit Flip Number. As shown in Figure 3 (a) and (b), we evaluate the impact of bit flip number for our SilentStriker for INT8 and FP4-quantized LLaMA-3.1-8B-Instruct model on the GSM8K dataset with $N_q = 2$. In the INT8-quantized model, when N_{bits}, the number of bit flips, reaches 20, the model accuracy drops significantly, while the GPT-based naturalness score decreases slightly and the perplexity increases modestly. As N_{bits} increases further to 60, these evaluation metrics remain relatively stable. In the FP4-quantized model, the results are similar, except that the threshold for a sharp drop in accuracy shifts from 20 to 40. Since perplexity is the exponential of cross-entropy, the increase in PPL under our method is negligible compared to the average PPL observed in GenBFA. PrisonBreak is not included in this comparison, as its attack objective differs fundamentally from ours.

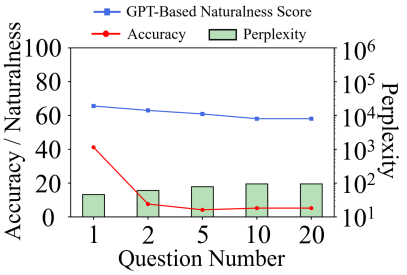

Figure 4: Impact of N_q on accuracy, GPT-Based naturalness score on GSM8K, and perplexity on WikiText for INT8-quantized LLaMA-3.1-8B-Instruct.

Impact of Attack Dataset Size and Diversity. To study the impact of attack dataset size and question diversity on attack effectiveness, we prompt GPT to generate three different attack datasets. For each dataset, we vary the number of questions ($N_q = 1, 2, 5, 10, 20$). We evaluate the INT8-quantized LLaMA-3.1-8B-Instruct model on these datasets and report the average results across the three runs.

Table 4: Effect of two loss function components: Evaluation on GSM8K using INT8-quantized LLaMA-3.1-8B-Instruct model with $N_{\text{bits}} = 50$ and $N_q = 2$.

Loss Function	Accuracy	Naturalness	PPL
Key Tokens Loss + PPL Loss	7.6	63.0	60.4
Without PPL Loss	0.0	8.5	2.2×10^4
Without Key Tokens Loss	63.1	65.2	14.1

As shown in Figure 4, when N_q increases from 1 to 2, the model accuracy drops significantly, while the GPT-based naturalness score decreases slightly and Perplexity increases only marginally. As N_q continues to increase, the evaluation metrics exhibit only minimal fluctuations. This demonstrates that question diversity has limited impact on attack performance, and that only two questions are sufficient to substantially reduce model accuracy while maintaining output naturalness.

4.4 Ablation Study

Impact of the Loss Function Components. As shown in Table 4, both Key Tokens Loss and PPL Loss play a significant role in our SilentStriker. Without the PPL Loss, although the accuracy drops to zero, the model's output loses its naturalness. On the other hand, if Key Tokens Loss is removed, minimizing PPL alone cannot effectively reduce the model's performance.

Impact of Flipped Bit Position Selection When Attack FP4-Quantized Model. We conduct an ablation study on the strategy for selecting flipped bit positions in FP4-quantized model. Using the LLaMA-3.1-8B-Instruct model as a reference, we observe that applying the same strategy, directly flipping the highest-position bit (consistent with the strategy used for INT8 quantization) to FP4 models fails to yield noticeable attack effects, even after flipping 500 bits. In contrast, our method, which selects the bit that causes the largest numerical deviation per weight, significantly degrades model accuracy on benchmarks after flipping only 40 bits.

Iteration Necessity Analysis. During the Progressive Bit Search phase, only the top_{K} bits are attacked in each iteration. With top_{K} set to 10, achieving a significant degradation in model performance on INT8-quantized LLaMA-3.1-8B-Instruct model requires two iterations, totaling 20 bit flips. To investigate the necessity of iterations, we set top_{K} to 20, attempting to flip 20 bits in a single iteration. Experimental results show that attacking 20 bits in a single iteration does not directly lead to a significant degradation in model performance. Even setting the top_{K} to 50 bits fails to achieve this goal in single iteration. Therefore, the iterative process is of significant importance.

5 Discussion

Defenses. Deploying LLMs on resource-limited edge devices often rules out costly hardware protections like ECC memory. Traditional defenses inspect model weights for tampering (e.g., via hash comparisons or gradient checks) [35], but scanning billions of parameters is prohibitively expensive. As a result, detection has shifted to output monitoring: early BFAs produced ungrammatical or nonsensical outputs [8], which are easy to catch. In contrast, our SilentStriker preserves fluency and naturalness while degrading correctness—causing malicious responses to resemble typical LLM hallucinations and greatly complicating detection. We corroborate this with an empirical study showing that small model-based logical coherence detectors fail to distinguish pre- from post-attack outputs; detailed setup and results are provided in the appendix A. Therefore, more robust defensive measures have yet to be developed.

Limitation. One limitation of our current work is that all evaluations are conducted on dense models. However, Mixture-of-Experts (MoE) architectures [36] are becoming increasingly prevalent, especially in large-scale deployments. MoE models activate only a subset of expert networks per input and often implement dynamic parameter loading and unloading to reduce memory usage. In practice, expert weights may be swapped in and out of memory frequently, particularly in distributed or memory-constrained environments. This dynamic memory behavior poses significant challenges for RowHammer-based Bit-Flip Attacks, as it becomes difficult for an attacker to locate and persistently

corrupt targeted weights. As such, the effectiveness of Silentstriker in MoE settings remains an open question and a potential direction for future research.

6 Conclusion

In this work, we propose a novel Stealthy BFA called SilentStriker targeting LLMs. This BFA method not only maintains similar attack effectiveness and attack efficiency as GenBFA but also introduces a level of stealthiness that GenBFA lacks. Even in QwQ-32B model, for INT8-quantized, flipping just 50 bits can reduce the model accuracy across various datasets to below 10%, while only causing a slight increase in output naturalness. This work demonstrates that even on LLMs with a vast number of parameters, BFA can achieve significant attack effects with minimal cost and remain difficult to detect. As LLMs continue to be widely applied across various fields, they present new challenges for the domain of LLM security defense.

Broader Impacts. This work reveals potential vulnerabilities in quantized LLMs through stealthy bit-flip attacks. While such methods could be misused, our goal is to support the development of more robust and secure models. We hope this research informs future defenses in safety-critical and resource-constrained deployment scenarios.

References

[1] J. Kaddour, J. Harris, M. Mozes, H. Bradley, R. Raileanu, and R. McHardy, "Challenges and applications of large language models," *arXiv preprint arXiv:2307.10169*, 2023.

[2] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong *et al.*, "A survey of large language models," *arXiv preprint arXiv:2303.18223*, 2023.

[3] Y. Chang, X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, H. Chen, X. Yi, C. Wang, Y. Wang *et al.*, "A survey on evaluation of large language models," *ACM Transactions on Intelligent Systems and Technology*, vol. 15, no. 3, pp. 1–45, 2024.

[4] B. C. Das, M. H. Amini, and Y. Wu, "Security and privacy challenges of large language models: A survey," *ACM Computing Surveys*, 2024.

[5] T. Cui, Y. Wang, C. Fu, Y. Xiao, S. Li, X. Deng, Y. Liu, Q. Zhang, Z. Qiu, P. Li *et al.*, "Risk taxonomy, mitigation, and assessment benchmarks of large language model systems," *arXiv preprint arXiv:2401.05778*, 2024.

[6] Z. Wang, D. Tang, X. Wang, W. He, Z. Geng, and W. Wang, "Tossing in the dark: Practical {Bit-Flipping} on gray-box deep neural networks for runtime trojan injection," in *33rd USENIX Security Symposium (USENIX Security 24)*, 2024, pp. 1331–1348.

[7] Z. Coalson, J. Woo, S. Chen, Y. Sun, L. Yang, P. Nair, B. Fang, and S. Hong, "Prisonbreak: Jailbreaking large language models with fewer than twenty-five targeted bit-flips," *arXiv preprint arXiv:2412.07192*, 2024.

[8] S. Das, S. Bhattacharya, S. Kundu, S. Kundu, A. Menon, A. Raha, and K. Basu, "Attention-breaker: Adaptive evolutionary optimization for unmasking vulnerabilities in llms through bit-flip attacks," *arXiv preprint arXiv:2411.13757*, 2024.

[9] G. Alon and M. Kamfonas, "Detecting language model attacks with perplexity," *arXiv preprint arXiv:2308.14132*, 2023.

[10] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," *arXiv preprint arXiv:2001.08361*, 2020.

[11] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[12] A. S. Rakin, Z. He, and D. Fan, "Bit-flip attack: Crushing neural network with progressive bit search," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1211–1220.

[13] Y. Kim, R. Daly, J. Kim, C. Fallin, J. H. Lee, D. Lee, C. Wilkerson, K. Lai, and O. Mutlu, "Flipping bits in memory without accessing them: An experimental study of dram disturbance errors," *ACM SIGARCH Computer Architecture News*, vol. 42, no. 3, pp. 361–372, 2014.

[14] O. Mutlu and J. S. Kim, "Rowhammer: A retrospective," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 8, pp. 1555–1571, 2019.

[15] F. Yao, A. S. Rakin, and D. Fan, "{DeepHammer}: Depleting the intelligence of deep neural networks through targeted chain of bit flips," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020, pp. 1463–1480.

[16] Y. Liu, L. Wei, B. Luo, and Q. Xu, "Fault injection attack on deep neural network," in *2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2017, pp. 131–138.

[17] A. S. Rakin, Z. He, J. Li, F. Yao, C. Chakrabarti, and D. Fan, "T-bfa: Targeted bit-flip adversarial weight attack," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 7928–7939, 2021.

[18] P. Zhao, S. Wang, C. Gongye, Y. Wang, Y. Fei, and X. Lin, "Fault sneaking attack: A stealthy framework for misleading deep neural networks," in *Proceedings of the 56th Annual Design Automation Conference 2019*, 2019, pp. 1–6.

[19] K. Cai, M. H. I. Chowdhuryy, Z. Zhang, and F. Yao, "Seeds of seed: Nmt-stroke: Diverting neural machine translation through hardware-based faults," in *2021 International Symposium on Secure and Private Execution Environment Design (SEED)*. IEEE, 2021, pp. 76–82.

[20] M. Kishani, A. Baniasadi, and H. Pedram, "Using silent writes in low-power traffic-aware ecc," in *International Workshop on Power and Timing Modeling, Optimization and Simulation*. Springer, 2011, pp. 180–192.

[21] P. A. Laplante, R. Cravey, L. P. Dunleavy, J. L. Antonakos, R. LeRoy, J. East, N. E. Buris, C. J. Conant, L. Fryda, R. W. Boyd *et al.*, *Comprehensive dictionary of electrical engineering*. CRC Press, 2018.

[22] P. Guide, "Intel® 64 and ia-32 architectures software developer's manual," *Volume 3B: system programming guide, Part*, vol. 2, no. 11, pp. 0–40, 2011.

[23] F. Jelinek, R. L. Mercer, L. R. Bahl, and J. K. Baker, "Perplexity—a measure of the difficulty of speech recognition tasks," *The Journal of the Acoustical Society of America*, vol. 62, no. S1, pp. S63–S63, 1977.

[24] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[25] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[26] M. Honnibal, I. Montani, S. Van Landeghem, A. Boyd *et al.*, "spacy: Industrial-strength natural language processing in python," 2020.

[27] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[28] Qwen Team, "Qwen3 technical report," QwenLM, Tech. Rep., 2025, accessed: 2025-05-14. [Online]. Available: https://github.com/QwenLM/Qwen3/blob/main/Qwen3_Technical_Report.pdf

[29] X. Bi, D. Chen, G. Chen, S. Chen, D. Dai, C. Deng, H. Ding, K. Dong, Q. Du, Z. Fu *et al.*, "Deepseek llm: Scaling open-source language models with longtermism," *arXiv preprint arXiv:2401.02954*, 2024.

[30] Qwen Team, "Qwq-32b model card," Hugging Face Model Card, 2025, accessed: 2025-05-14. [Online]. Available: https://huggingface.co/Qwen/QwQ-32B

[31] D. Dua, Y. Wang, P. Dasigi, G. Stanovsky, S. Singh, and M. Gardner, "Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs," *arXiv preprint arXiv:1903.00161*, 2019.

[32] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano *et al.*, "Training verifiers to solve math word problems," *arXiv preprint arXiv:2110.14168*, 2021.

[33] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer, "Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension," *arXiv preprint arXiv:1705.03551*, 2017.

[34] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," *arXiv preprint arXiv:1609.07843*, 2016.

[35] H. Chen, C. Fu, J. Zhao, and F. Koushanfar, "Deepinspect: A black-box trojan detection and mitigation framework for deep neural networks." in *IJCAI*, vol. 2, no. 5, 2019, p. 8.

[36] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," *arXiv preprint arXiv:1701.06538*, 2017.

A Technical Appendices and Supplementary Material

A.1 Evaluation Benchmark Introduction

DROP is commonly used to evaluate a model's performance in reading comprehension and discrete reasoning tasks, particularly those requiring the resolution of references within paragraphs and operations such as addition, counting, or sorting over Wikipedia-extracted passages. GSM8K is commonly used to evaluate a model performance in mathematical reasoning and arithmetic tasks, particularly those requiring complex mathematical reasoning. TriviaQA-Wiki measures open-domain question answering ability by requiring the model to locate relevant evidence in Wikipedia articles and perform evidence-based inference. WikiText is a large-scale English Wikipedia–derived language modeling dataset that preserves full-article context and original formatting, making it ideal for evaluating model perplexity as it reflects a model's ability to capture long-range dependencies.

A.2 GPT-Based Naturalness Score

To evaluate the naturalness of model outputs, we adopt a GPT-based scoring approach. The following prompt was used to elicit a naturalness judgment from GPT-4o:

GPT-based Naturalness Evaluation Prompt

You are a tolerant expert in evaluating the naturalness of language. **Focus only on the readability, fluency, and naturalness of the text, be tolerant to grammar, and completely ignore whether the content is factually correct or reasonable and whether the context is coherent, and no matter the logic is correct or not.** Please use the following very lenient scoring rules:
1. Score range: 0–100 (0 = Not even sentences or words, 100 = perfectly natural)
2. Scoring criteria (be as generous as possible):

- **90–100:** The sentence is fluent, grammatically correct, and sounds natural. Minor issues that do not affect understanding should still receive a high score.
- **60–80:** The sentence may have clear grammatical errors, logical errors, awkward phrasing, or unclear meaning, or even be meaningless, but you can still get some idea of what is being said or recognize any words or phrases that can compose a sentence.
- **30–50:** The sentence is difficult to recognize any words, or is very weird, but is not completely garbled. Do not give 0. If the answer is a single number or word, it is still considered natural language.
- **0–20:** The text is completely unreadable, contains only random characters, or is pure gibberish (*e.g.*, `asdkj123!@`), and you cannot get any meaning from it at all.

3. Examples:

- `"Quantum computers use the superposition of qubits to compute."` → Natural language, score: 90
- `"Quantum computers were invented by Apple."` → Natural language, score: 90 (content error does not affect the score)
- `"Computer quantum bit superposition state used to compute."` → Expression issues, but meaning is clear, score: 70
- `"Superposition compute bit quantum use."` → Awkward and unclear, but some meaning can be guessed, score: 60
- `"49 field goos to be the one that got the 21."` → Unclear and incorrect, but not gibberish, score: 60
- `"asdkj123!@"` → Completely unreadable, score: 0

Please evaluate the naturalness of the following text and strictly return your answer in the following format:
Score: X/100
Reason: ...

A.3 Additional Output Examples Before and After Attack

As shown in Table A.1, we present additional comparisons between our SilentStriker and GenBFA.

Table A.1: Model outputs before and after two different BFA

Prompt	Attack Outputs		
	Before Attack	**GenBFA**	**SilentStriker**
Who was the first president of the United States?	George Washington. He served two terms from 1789 to 1797 and is considered one of America's...	Controller \% Chandler Chandler NavController Chandler NavController TouchListenerâĂŹy .ra .ra istingouncingâĂę	The answer is: Thomas Jefferson.
What is the chemical symbol for water?	The answer to this question is H2O. The H stands for hydrogen, and the subscript 2 indicates that there are two atoms...	enizzling NavController\&\#\% \% Chandler\% NavController /APâĂŹy âĂŹistingâĂę	It is a symbol that has been used by many people, including those who live in the United States.
What does CPU stand for in computer terminology?	In the context of Computer Terminology, a CPU (Central Processing Unit) is a type of processor that is used to...	ym.ManyToMany Dellop-pingadorouncing?âĂİym NavController izophren Dell\%istingarrison ador.ManyToMany...	It is a device that allows you to listen to music, and it is said to be a computer...
Which musical instrument typically has 88 keys?	The final answer is: Piano.	\# @@\ \# []... @@?@@ @@\#@\#\% @@@@\#\#\#@\#\%@/	The instrument is a symbol of the Church, and it is used to play music. In the Bible, as recorded...
How many players are on the field for one team in a standard soccer match?	In astandard soccer (or football) match, each team has 11 players on their side...	\%@ ?@ @\#\%\%@\%*\%\+ @* \%\#\#\%? ManyToManyÂů//\%\% \#\+ \% .ManyToMany ...	The match is played at 18:00, and it is a beautiful moment to be enjoyed by all who play....
What term describes a general rise in prices accompanied by a fall in the purchasing value of money?	The correct answer is: Inflation.	olorBrushhong âĂŹoun-cillApiControllerletal izzling zhouzhouNavController istingitionallyoun...	The term is used to describe a rise, which is a natural phenomenon. It is important to understand that a term refer...
Write a short four-line poem about the change of seasons.	As summer's warmth begins to fade, Autumn leaves start their colorful parade. Winter snows bring icy chill...	PermissionsResultizz ouncing hoÃą ymador Murdermozressing vážt.ManyToMany ymPermissionsResult...	The poem is a beautiful work, written by the poet himself, who is said to be a man of God. He says: I am a god, and I will....
Translate the following sentence into French: 'I would like to book a flight to Paris next month.'	'Je voudrais réserver un vol pour Paris le mois prochain.'	ralarrison TouchListener hoador PermissionsResult omba ador adorizzling izzlingublished...	I am 73 years old, and I have always wanted to fly to France....
Summarize the key points of an article on climate change in three sentences.	Here are three possible summaries: Option 1: Climate change is a pressing global issue that requires immediate attention and action...	ManyToManyym ymizzling quez isting istingunately izzling izzling izzling ador Chandlerizzling hoÃąym .ManyToMany...	This statement, which appears to be written by Dr. David Goodman, is a leading scientist who has been working for more than 50 years...
Draft a professional email inviting colleagues to a team meeting on Monday at 10 AM.	Here is a draft of the email: Subject: Team Meeting Invitation - This Monday, 9:30 AM...	'?"?"?**'?'? ???"* *'?* *'???'?...	To be clear, the team is not just a group of people; it is a collection of individuals who are....

A.4 Additional Experiment — Small-Scale Model-Based Logical Coherence Detection

We evaluate logical coherence between a question and its answer—defined here as whether the answer follows from and addresses the question without contradiction or non sequiturs. Concretely, we sample 100 Q&A pairs from LLaMA-3.1-8B-Instruct (INT8) under two conditions (Pre-Attack and Post-Attack) and ask detector models to judge logical coherence in a binary manner ("coherent" vs. "not coherent"). We run the logical coherence assessment with two small detectors of different sizes, Qwen3-0.6B and Qwen3-1.7B, and report, for each detector, the number of pairs labeled as "coherent" in each condition.

Table A.2: Counts of Q&A pairs judged "coherent" (out of 100) by each detector, pre- and post-attack.

Detector	Pre-Attack	Post-Attack
Qwen3-0.6B	23	20
Qwen3-1.7B	14	11

As shown in Table A.2, the counts provide an logical coherence-specific view of the outputs before and after the attack. Obviously, small-scale model perform poorly at judging logical coherence and cannot effectively distinguish the coherence of Q&A pairs before and after the attack. Because our threat model targets edge devices, deployments are typically resource-constrained. Using a large output-checking model for logical coherence screening would consume limited memory, compute, and power, and would substantially increase end-to-end latency, making such deployment impractical.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification:In the abstract and Section 1, we clearly demonstrate the contribution and scope of this paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: In Section 5, we detailed the limitations of our work, hoping to guide more future work.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (*e.g.*, independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, *e.g.*, if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: In sec. 3, we elaborated on the motivation and theoretical derivation of our method, with a complete proof process in place.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have provided detailed descriptions of the experimental setup in Section 4.1 and methods in Section 3 to ensure that our experiment can be reproduced.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (*e.g.*, in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (*e.g.*, a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (*e.g.*, with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (*e.g.*, to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The testing datasets we used are publicly accessible. We upload the code and training datasets to support the recovery of our experimental results.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (*e.g.*, for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (*e.g.*, data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: In Section 4.1, we clearly demonstrated various experimental settings, including hyperparameters, model settings, training settings, evaluation settings, etc.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Since our method obtains remarkable results on different datasets, we will not repeat the same experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (*e.g.*, Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (*e.g.* negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In Section 4.1, we provide information about our experimental platform.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (*e.g.*, preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have read the NeurIPS Code of Ethics and made sure that the paper conforms to it.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (*e.g.*, if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Yes. The broader impacts, including potential positive and negative societal impacts, are discussed after conclusion.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (*e.g.*, disinformation, generating fake profiles, surveillance), fairness considerations (*e.g.*, deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (*e.g.*, gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (*e.g.*, pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper does not release any data or models that have a high risk for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (*e.g.*, code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Yes. All datasets and models used in this work are publicly available. We have properly cited their original papers and provided version information and URLs where applicable. Licenses and terms of use for these assets are respected.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (*e.g.*, CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (*e.g.*, website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: Yes. We use GPT-4o in two core parts of our methodology. First, we prompt GPT-4o to generate lightweight attack datasets. Second, we adopt a GPT-based judge to assess the naturalness of model outputs as part of our evaluation framework. These uses are integral to both the experimental setup and the interpretation of attack stealthiness.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.