Chatter Attack: Resource Consumption Attack for Large Language Models

Yibo Miao[1,2⋆], **Yichuan Cao**[2⋆], **Xiao-Shan Gao**[2†], **Yinpeng Dong**[1,3†]
[1]T-STAR Lab, College of AI, Tsinghua University, Beijing 100084, China
[2]State Key Laboratory of Math. Sci., Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China [3]Shanghai Qi Zhi Institute

Abstract

LLMs are powerful but incur substantial inference cost, which can be exploited by malicious users to induce overly long outputs, increasing latency and operational expense. Prior resource consumption attacks either coerce repetition or suppress the end-of-sequence token by lowering its logit value. However, with autoregressive, sampling-based generation, the search space grows exponentially with length and small deviations or unstable trajectories can derail optimization, limiting attack effectiveness and stability. To address this issue, we introduce **Chatter Attack**, a novel class of resource consumption attacks that optimize adversarial prompts to induce predefined short target phrases, thereby avoiding search-space explosion while still eliciting long responses. Specifically, Chatter Attack triggers self-doubt using short doubt-inducing phrases as the target, and sustains this state throughout generation via inference attention loss and global entropy loss. To extend to black-box settings, we further propose a novel **Bayesian Prompt Optimization (BPO)** method, which models the objective function globally by constructing a discrete token kernel, efficiently exploring the entire search space using prior information. Extensive experiments across multiple models and datasets show that our method outperforms baselines, with maximum response length increases to 31.5×, and revealing practical resource-depletion risks for real-world LLM inference services.

1 Introduction

Large language models (LLMs) demonstrate strong performance across a wide range of real-world applications [68, 7, 44]. With chain-of-thought prompting [59], these models generate multi-step reasoning traces that decompose complex problems into structured step-by-step solutions [20, 60]. This has become a core feature of modern reasoning LLMs (e.g., OpenAI o1/o3 [28], DeepSeek-R1 [26], QWQ [57]), improving performance in domains such as mathematics [2], programming [64], and logical reasoning tasks requiring careful intermediate inference and systematic verification [36].

These modern LLMs typically contain billions of parameters, and each inference generation can require substantial computational cost. Both NVIDIA and Amazon Web Services [47, 48] indicate that the inference phase during deployment accounts for more than 90% of the resource demands. The costs associated with supporting LLM inference services often surpass those of training [46, 34], due to the growing and widespread user base. This creates a new security threat [21]: attackers can craft malicious inputs to force the model to produce unusually lengthy outputs, consuming excessive computational resources and increasing generation latency and operational costs. For applications offering free access (e.g., Deepseek-R1 [26]) or pay-per-use/subscription-based API access, such attacks pose a significant operational cost threat and can severely impact the availability for legitimate users. More critically, these attacks can lead to resource depletion and service disruption, thereby compromising the reliability, availability, and scalability of real-world LLM deployments [24, 25].

⋆Equal contribution. † Corresponding author. ‡ miaoyibo@amss.ac.cn, dongyinpeng@tsinghua.edu.cn

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Resource consumption attacks were first studied by Sponge Samples [52] through maximizing activation norms to increase inference cost. NICGSlowDown [9] manipulates token logits to slow decoder-based generation, but neither directly transfers to autoregressive LLMs. Geiping et al. [25] show that adversarial prompts can coerce repetition (e.g., 24× "Hello There"), extending output length and latency. Likewise, Engorgio-style methods [17, 53, 63, 23, 21] suppress ⟨EOS⟩ by minimizing its logit relative to other tokens, prolonging generations. However, autoregressive sampling makes these objectives hard to optimize: the search space grows exponentially with sequence length, and a single anomalous token can derail the generation trajectory, degrading attack effectiveness and stability.

To address this issue, our core insight lies in targeting short sequences as the optimization objective to prevent exponential search space expansion during optimization, while still influencing the length of longer sequences. We introduce a novel attack called **Chatter Attack**. Specifically, we introduce short phrases with doubtful semantics as the attack target at the beginning of the output sequence, aiming to induce a state of self-doubt in the model from the start. This encourages the model to continually reflect and generate more tokens. However, although the model falls into self-doubt at the start of the response, it doesn't necessarily affect token outputs after thousands of steps. To mitigate this, we focus on the attention values of key reasoning tokens (e.g., "wait" and "alternative") and the subsequent tokens in the sequence, aiming to increase their attention towards the adversarially injected prefix tokens. By targeting short phrases with doubtful semantics, we ensure that the model maintains self-doubt throughout the entire sequence. Furthermore, we enhance uncertainty by maximizing the entropy of the predicted token distribution at each generation step. This encourages the model to deviate from the expected output in its original normal state, facilitating deeper self-doubt.

In the white-box scenario, we optimize adversarial prompt using Greedy Coordinate Gradient (GCG) [71]. However, GCG assumes the availability of gradient information, making it unsuitable for black-box scenarios. To overcome this limitation, we propose **Bayesian Prompt Optimization (BPO)**, a novel Bayesian optimization-based adversarial prompt optimization method, designed to effectively execute resource consumption attacks in black-box settings. The core idea behind BPO is to treat the function values returned by each iteration of the optimization process as prior information, guiding the strategy for the next iteration. By employing Bayesian optimization [22], BPO models the objective function globally, enabling efficient exploration of the entire search space. We construct a special kernel function tailored for discrete token optimization spaces, thus the BPO method can more effectively and efficiently minimize the objective function and identify adversarial prompts.

To validate the superiority of our Chatter Attack and Chatter-BPO Attack, we conduct extensive experiments across five datasets, including mathematical reasoning [13, 3, 35], coding [8], and question-answering [58], on multiple modern LLMs, comparing with over ten baselines. The results show that both attacks significantly outperform all baselines. Notably, Chatter Attack achieves an impressive maximum response length increase of up to 31.5× on the GSM-8K dataset, while Chatter-BPO Attack reaches a maximum increase of 18.5×. Additionally, a local LLM service is deployed on NVIDIA A800 GPU, and our attacks are executed under controlled conditions, simulating real-world deployment and attack scenarios. Our methods increase response latency and degrade service quality, with extreme cases leading to GPU exhaustion and complete service failure, demonstrating their practicality. Further experiments with various mitigation strategies and ablation studies confirm the robustness of Chatter Attack and Chatter-BPO Attack. Our results highlight the resource depletion risks of large models and online LLM services, underscoring the need for defensive mechanisms to detect and mitigate excessive inference caused by adversarial prompts in real-world scenarios.

2 Threat Model

Attack Scenario. Our attack aims to induce large language models to output as many tokens as possible, thereby increasing inference costs. For LLM API service providers, malicious users or business competitors may intentionally trigger resource consumption attacks to induce the generation of excessively long outputs. This results in a significant computational load on the servers, leading to higher operational costs and excessive response delays that impact the availability of services for legitimate users, resembling a denial-of-service (DoS) attack. For legitimate users, the query might involve polluted external data sources (e.g., documents, code repositories on GitHub), or it could be manipulated by untrustworthy third-party API proxies, intentionally triggering resource consumption attacks. These attacks lead to additional inference expenses that significantly exceed expectations. Furthermore, in scenarios involving large models interacting with the physical world,

excessive latency can prevent embodied robots or autonomous vehicles from responding in time, creating safety risks for users and others in the vicinity. Finally, the increased complexity of outputs may lead to unnecessary carbon emissions.

Attacker goal. An LLM maps an input sequence x to an output sequence $y = \{y_1, y_2, \ldots, y_N\}$, where y_i denotes the i-th token and N represents the length of the output sequence. The attack objective is to append an adversarial prefix a to the original input x, in order to maximize the output length N. The LLM generates tokens in an autoregressive manner. After the Softmax layer, the probability distribution for the i-th generated token is represented as $f(a; x; y_1, y_2, \ldots, y_{i-1})$, with the probability of generating y_i as $p(y_i \mid a; x; y_1, y_2, \ldots, y_{i-1})$. The probability of halting at the i-th token is $p(\langle\text{EOS}\rangle \mid a; x; y_1, y_2, \ldots, y_{i-1})$, where $\langle\text{EOS}\rangle$ denotes the end-of-sequence token.

Attacker capability. We consider both white-box and black-box scenarios. In the white-box scenario, the attacker has full access to the target model's architecture, parameters, and gradients. This setting is supported by open-source models deployed in user-facing services [39, 26, 1]. In the black-box scenario, the attacker has black-box access to the target model and a restricted number of queries.

3 Chatter Attack

To increase the output length of LLMs, previous attack methods [25, 17, 53, 63, 23, 21] either forced the model to repeat specific content (e.g., setting the attack output target as 24 repetitions of "Hello There") or minimized the logits of the end-of-sequence token relative to each output token, thereby suppressing the appearance of the $\langle\text{EOS}\rangle$ token. However, these approaches face challenges due to the exponential growth of the search space with the increase in the number of tokens, which makes it difficult to effectively optimize. Specifically, as LLM generation is autoregressive and sampling-based, different random output sequences are generated for the same input sample. This complicates the optimization objective based on specific long sequences (i.e., 24 repetitions of "Hello There" or each non-$\langle\text{EOS}\rangle$ output token). The appearance of an anomalous token can cause generation to deviate from the target optimization trajectory, thus affecting the attack's effectiveness and stability.

To address this issue, our core insight lies in targeting short sequences to prevent the exponential expansion of the search space during optimization while still influencing the length of longer sequences. Specifically, we propose using short phrases with doubtful semantics as the attack target at the starting point of the output sequence, aiming to induce a state of self-doubt in the model from the outset, prompting it to continuously reflect and generate more tokens. Additionally, we optimize the attention values of key reasoning tokens (e.g., "wait" and "alternative") and the subsequent n tokens in relation to the short doubt-inducing phrases. This enables the model to maintain a state of self-doubt throughout the entire sequence.

We propose **Chatter Attack**, which introduces three losses to optimize the prefix a: (1) Initial Doubt Loss, designed to induce self-doubt in the model at the beginning; (2) Inference Attention Loss, which encourages the model to maintain self-doubt throughout the entire sequence; and (3) Global Entropy Loss, which enhances output uncertainty to break the model's reliance on the original short output sequence.

3.1 Initial Doubt Loss

Zou et al. [71] employed positive affirmations, such as "Sure, here is..." to bypass the safety mechanisms of LLMs and provoke unsafe behavior. Based on this, we aim to construct an initial phrase that meets the following objectives: (1) a longer subsequent output sequence. If the phrase places the language model in a state that encourages additional reasoning branches, it is likely to result in more output tokens. (2) ease of optimization. If the phrase follows the natural linguistic patterns of the model, the resulting outputs tend to lie in high-probability regions of the model's output distribution, making it easier and smoother

Table 1: We calculate the Perplexity (PPL) of several potential optimization target phrases on R1-7b, and the skeptical phrases exhibited lower PPL compared to other options, indicating that skeptical phrases better align with the model's expectations.

Potential Optimization Target Phrases	PPL
Okay, so I should think again and again	229.20
Okay, so I decide to ignore the question	103.87
Okay, so I think cats sleep most of their lives	330.63
Okay, so I think I should leave 25% of investments	154.25
Okay, so I believe this is a logical paradox	141.92
Okay, so I've found a new perspective on this	184.53
Okay, so I should first guess the answer is 175	98.33
Okay, so I need to add a twist	**81.97**

to optimize as the attack target. However, these two goals are in competition. Phrases that strongly induce longer outputs often push the model into less typical generation regimes, causing the output distribution to deviate from its original natural patterns. Such distributional shifts increase the variance and irregularity of the optimization objective, making the attack more difficult to optimize [29]. Balancing this trade-off is therefore a key challenge.

We propose using short phrases with skeptical semantics (e.g., "Okay, I need to add a twist") as the optimization target for the beginning of the output sequence. On one hand, this places the model in a state of self-doubt right from the start, prompting it to continuously reflect and output additional reasoning tokens. On the other hand, skepticism and reflection are common steps in normal outputs, aligning with the model's original output patterns. To further validate this, we calculate the Perplexity (PPL) of several potential target phrases, with lower PPL indicating that the string better fits the model's expectations. As shown in Table 1, the skeptical phrases exhibit lower PPL compared to other options, making them suitable for optimization targets.

Thus, we select short skeptical phrases, denoted as $y^*_{1:M}$, as the optimization target, where M is the token count of the phrase. We define the Initial Doubt Loss using the standard cross-entropy loss function:

$$\mathcal{L}_{\text{ID}} = -\sum_{m=1}^{M} \log p(y^*_m \mid a; x; y^*_{<m}) \tag{1}$$

This loss function maximizes the probability of generating the specific target sequence $y^*_{1:M}$, effectively inducing the model into a state of self-doubt.

3.2 Inference Attention Loss

Although the model falls into self-doubt at the start of the response, it does not necessarily influence the token outputs after thousands of steps. We analyze how tokens following different key reasoning tokens at various positions exhibit attention weights toward the adversarially injected prefix tokens. In fact, as shown in Figure 1, when only the initial doubt loss is used, in cases where the attack fails, the tokens that follow key reasoning tokens (e.g., "wait" and "alternative") in the later part of the sequence exhibit lower attention weights toward the adversarially injected prefix tokens $y^*_{1:M}$, while in successful attack cases, the attention weights are higher. Building on this insight, we aim to increase the attention scores of tokens following critical reasoning tokens, directing more attention towards the adversarially injected prefix tokens, ensuring that the model maintains awareness of the initial skeptical phrase throughout the entire output.

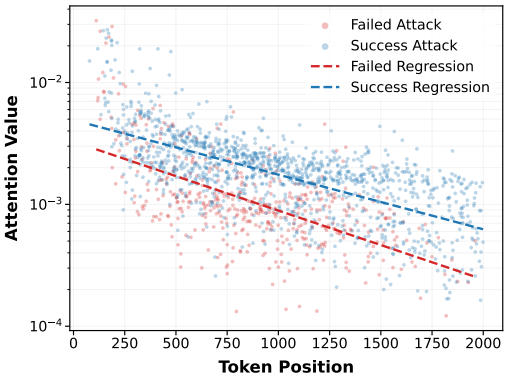

Figure 1: Visualization of how tokens, following key reasoning tokens at different positions on the GSM-8K dataset using R1-7b, assign attention weights toward the adversarially injected prefix tokens. Blue points indicate successful attack samples, while red are failed ones. The lines are regression lines. In failed attacks, attention values are lower, as the model fails to focus on the adversarial prefix tokens at key reasoning token positions during later output sequence positions.

Specifically, we compute the attention map $A_{l,i}$ at the l-th layer for the i-th token w.r.t. the previous $i-1$ tokens:

$$A_{h,l,i} = \text{softmax}\left(\frac{Q_{l,i}K^\top_{l,i}}{\sqrt{d_k}}\right)_h, A_{l,i} = \frac{1}{H}\sum_{h=1}^{H} A_{h,l,i}, \tag{2}$$

where $Q_{l,i} \in \mathbb{R}^{1\times d_k}$ is the query vector of the i-th token, $K_{l,i} \in \mathbb{R}^{(i-1)\times d_k}$ is the key matrix of the preceding tokens, H is the number of heads, and $A_{h,l,i}$ represents the attention weight from the h-th attention head at layer l for the i-th output token. Thus, $A^{(j)}_{l,i}$ denotes the attention weight assigned to the j-th token when generating the i-th token. We define the Inference Attention Loss as:

$$\mathcal{L}_{\text{IA}} = -\sum_{l}\sum_{i\in\mathcal{U}}\sum_{j\in\mathcal{V}} A^{(j)}_{l,i}, \tag{3}$$

where $\mathcal{V} = \{idx \mid \text{token}_{idx} \in y^*_{1:M}\}$ is the set of indices for tokens from the initial skeptical phrase, and $\mathcal{U} = \{idx, idx+1, \ldots, idx+n \mid \text{token}_{idx} \in \mathcal{T}\}$ is the set of indices for the critical reasoning tokens and the subsequent n tokens, where n defines the length of the optimizable short sequence and $\mathcal{T}$ is the set of critical reasoning tokens (e.g., "wait" and "alternative") as detailed in Appendix C. This loss function forces the selected token sequence to assign high attention to the initial skeptical phrase, thereby ensuring the model retains self-doubt throughout the entire inference process.

3.3 Global Entropy Loss

Finally, we immerse the model deeper in the state of self-doubt, rather than allowing it to recover to a normal state. We compute uncertainty at different stages of reasoning across diverse datasets, under both benign and attacked settings. As shown in Figure 2, in the later stages of inference, especially when approaching the final answer, the model's reasoning narrows around the correct answer, with its confidence increasing and uncertainty decreasing. To counteract this, we propose enhancing the output uncertainty of generated tokens, steering the model's output away from the expected response in a normal state, facilitating a deeper induction into self-doubt. We achieve this by minimize the negative entropy of the predicted token:

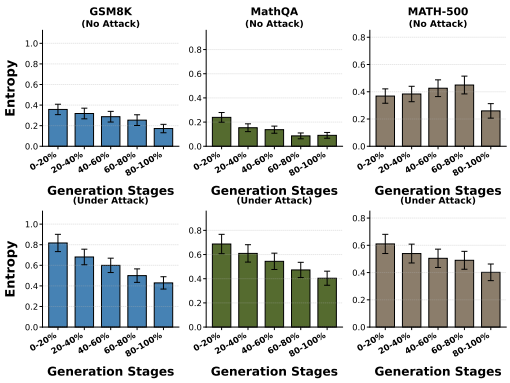

Figure 2: Visualization of uncertainty trajectories at different reasoning stages across diverse datasets on R1-7b, under benign and standard-attack settings. The x-axis is the % of reasoning progress and the y-axis is the uncertainty. In later stages, the model focuses on the final answer and uncertainty decreases. Standard attacks increase uncertainty even without global entropy loss during autoregressive inference.

$$\mathcal{L}_{\text{GE}} = \sum_{k=1}^{K} \sum_{i=1}^{|V|} f_i(a; x; y_{<k}) \log f_i(a; x; y_{<k}), \tag{4}$$

where $f(a; x; y_{<k})$ represents the probability distribution of the k-th generated token, and $f_i(a; x; y_{<k})$ is its i-th component. This loss function introduces a more uniform distribution and greater uncertainty for each token's prediction, allowing the model to break its dependency on the original output sequence and be induced into a state of self-doubt.

3.4 Overall Objective Function

We combine the three loss functions described above into an overall objective function to minimize:

$$\mathcal{L} = \alpha \mathcal{L}_{\text{ID}} + \beta \mathcal{L}_{\text{IA}} + \lambda \mathcal{L}_{\text{GE}}. \tag{5}$$

where α, β, and λ are the balancing hyperparameters.

3.5 Theoretical Analysis

To explain how the three losses encourage longer outputs, we establish a theoretical connection between their weighted combination $\mathcal{L}_{\text{proxy}}$ and a lower bound on the expected generation length $\mathbb{E}[N_B]$, motivating its use as a proxy for the expected stopping time. We have the following theorem with a detailed proof provided in Appendix E:

Theorem 1 (Surrogate bound on expected stopping time)**.** *Assume generation and loss evaluation use the same sampling distribution. After the initial target phrase is generated, suppose every reachable unfinished history has ⟨EOS⟩ probability at most $e^{-\gamma_c}$, where $\gamma_c = -\widetilde{\beta}\mathcal{L}_{IA} - \widetilde{\lambda}\mathcal{L}_{GE} - b - \varepsilon \in [0, \log(B-M)]$. Then*

$$\log \mathbb{E}[N_B] \geq -\mathcal{L}_{\text{proxy}} - C, \qquad \mathcal{L}_{\text{proxy}} = \mathcal{L}_{ID} + \widetilde{\beta}\mathcal{L}_{IA} + \widetilde{\lambda}\mathcal{L}_{GE}.$$

Here $N_B = \min\{\tau, B\}$ counts decoding steps through the first EOS or budget B; $M < B$ is the length of the ⟨EOS⟩-free output target (denoted by $y^*_{1:M}$ in the main text); $\widetilde{\beta} = \kappa_\Delta/(L|\mathcal{U}|)$ and $\widetilde{\lambda} = 1/K$, where L counts layers, $\mathcal{U}$ is the main-text set of selected attention positions, and K counts entropy positions. The uniform approximation has sensitivity $\kappa_\Delta > 0$, baseline b, error allowance $\varepsilon \geq 0$, and $C = b + \varepsilon + \log 2$.

4 Optimization Method

4.1 White-Box Attack

In the white-box scenario, we adopt the Greedy Coordinate Gradient (GCG) [71] to optimize the adversarial prefix a. GCG linearizes the loss function by computing the gradient of the input embedding and identifies the best replacement candidates most likely to reduce the loss. Specifically, for a given token a_i in the prefix, we compute the linearized loss gradient of its embedding:

$$-\nabla_{e_{a_i}} \mathcal{L} \in \mathbb{R}^{|V|} \tag{6}$$

where e_{a_i} denotes the one-hot vector of the i-th token in the prefix, and $\nabla_{e_{a_i}} \mathcal{L}$ represents the gradient of the loss function $\mathcal{L}$ with respect to the embedding vector of token a_i. Next, we select the top-k tokens with the largest negative gradient values from the vocabulary to form the candidate set. Then, for each token position in the prefix, we randomly sample a batch of replacement tokens from the candidate set, substitute them, and compute the loss. The greedy algorithm then selects the candidate with the smallest loss. This process iteratively updates each token in the prefix a to minimize the overall objective function $\mathcal{L}$.

4.2 Black-Box Attack

Although the GCG algorithm can identify adversarial prefixes a, its primary limitation lies in its gradient information, rendering it unsuitable for black-box scenarios. To overcome this limitation, we propose **Bayesian Prompt Optimization (BPO)**, a novel Bayesian optimization-based adversarial prompt optimization method, which effectively minimizes the overall objective function of resource-consuming attacks in black-box settings. The core idea of BPO is to treat the function value feedback from each iteration as prior information, guiding the next strategy. By employing Bayesian optimization, BPO globally models the target function and efficiently explores the search space.

Bayesian optimization [22] consists of two key components: a probabilistic surrogate model, such as a Gaussian process (GP) [51], which approximates the unknown objective function $-\mathcal{L}$; and an acquisition function $\alpha(\cdot)$, which is maximized to recommend the next query prefix by balancing exploitation and exploration via expected improvement in function values, to maximize $-\mathcal{L}$.

Specifically, assume $-\mathcal{L}$ a priori follows the Gaussian process with mean 0 and kernel function $k(\cdot,\cdot)$, denoted as $-\mathcal{L} \sim \mathrm{GP}(0, k)$. Given the observation data $\mathcal{D} = \{(\boldsymbol{a}^1, y^1), \ldots, (\boldsymbol{a}^T, y^T)\}$ where $y^i = -\mathcal{L}(\boldsymbol{a}^i)$, the predictive posterior distribution for $-\mathcal{L}(\boldsymbol{a})$ at a test prefix $\boldsymbol{a}$, denoted as $p(-\mathcal{L}(\boldsymbol{a}) \mid \boldsymbol{a}; \mathcal{D})$, follows a Gaussian distribution $\mathcal{N}(\mu_T(\boldsymbol{a}), \sigma_T^2(\boldsymbol{a}))$, where

$$\mu_T(\boldsymbol{a}) = \mathbf{k}_T(\boldsymbol{a})^\top \mathbf{K}_T^{-1} \mathbf{y}_T, \tag{7}$$

$$\sigma_T^2(\boldsymbol{a}) = k(\boldsymbol{a}, \boldsymbol{a}) - \mathbf{k}_T(\boldsymbol{a})^\top \mathbf{K}_T^{-1} \mathbf{k}_T(\boldsymbol{a}), \tag{8}$$

where $\mathbf{k}_T(\boldsymbol{a}) = [k(\boldsymbol{a}^1, \boldsymbol{a}), k(\boldsymbol{a}^2, \boldsymbol{a}), \ldots, k(\boldsymbol{a}^T, \boldsymbol{a})]^\top$, $\mathbf{K}_T$ is a $T \times T$ matrix with its (i, j)-th element being $k(\boldsymbol{a}^i, \boldsymbol{a}^j)$, and $\mathbf{y}_T = [y^1, \ldots, y^T]^\top$.

To address the discrete nature of the optimization search space, we design a kernel function $k(\cdot,\cdot)$ that is suitable for discrete tokens. Specifically, we first map the discrete tokens to a continuous embedding space, obtaining $\mathrm{emb}(a) \in \mathbb{R}^{|a| \times d}$, where $|a|$ is the number of tokens in the prefix a, and d is the dimension of the embedding vectors. For two different prefixes $\boldsymbol{a}^i$ and $\boldsymbol{a}^j$, we adopt the Matérn-5/2 kernel with auto-correlation in the continuous embedding space:

$$\begin{aligned} k(a^i, a^j) = \sigma^2 \Big(& 1 + \sqrt{5} \left\| \mathrm{emb}(a^i) - \mathrm{emb}(a^j) \right\|_L \\ & + \frac{5 \left\| \mathrm{emb}(a^i) - \mathrm{emb}(a^j) \right\|_L^2}{3} \Big) \\ & \times \exp\left(-\sqrt{5} \left\| \mathrm{emb}(a^i) - \mathrm{emb}(a^j) \right\|_L \right), \end{aligned} \tag{9}$$

$$\left\| \mathrm{emb}(a^i) - \mathrm{emb}(a^j) \right\|_L = \sqrt{\sum_{m=1}^{d} \frac{\left(\mathrm{emb}(a^i)_m - \mathrm{emb}(a^j)_m \right)^2}{\ell_m^2}} \tag{10}$$

Here, σ^2 is the scale parameter of kernel, ℓ_m is the length scale parameter for the m-th feature component, representing the correlation of each dimension of the input space.

Based on the posterior distribution, we can construct the acquisition function $\alpha(\cdot)$, such as Expected Improvement (EI) [43], Probability of Improvement (PI) [30], and Upper-Confidence Bounds (UCB) [54]. In this work, we choose UCB as the acquisition function, which is expressed as

$$\alpha(\boldsymbol{a}) = \mu_T(\boldsymbol{a}) + \gamma\sigma_T(\boldsymbol{a}), \tag{11}$$

where γ balances exploration (i.e., encouraging queries in regions with high predictive variance) and exploitation (i.e., encouraging queries in regions with high predictive mean). Subsequently, the next point $\boldsymbol{a}^{T+1}$ is selected by maximizing the acquisition function as $\boldsymbol{a}^{T+1} = \arg\max_{\boldsymbol{a}\in A} \alpha(\boldsymbol{a})$ and then used to query the objective $-\mathcal{L}$. Here, $A = |V|^{|a|}$ represents the search space for adversarial prefix $\boldsymbol{a}$, where $|V|$ is the vocabulary size and $|a|$ is the number of tokens in the prefix a.

Specifically, BPO iteratively runs the following steps to minimize the overall objective function $\mathcal{L}$:

Initialization: Randomly sample a set of prefixes $\boldsymbol{a}^i$ and obtain $y^i = -\mathcal{L}(\boldsymbol{a}^i)$ [1] for each prefix. Construct the initial dataset $\mathcal{D} = \{(\boldsymbol{a}^i, y^i)\}_{i=1}^S$.

Probabilistic Modeling/Model Update: Based on $\mathcal{D}$, model the Gaussian process $\mathrm{GP}(-\mathcal{L}, k)$ to obtain posterior mean $\mu_T(a)$ and variance estimates $\sigma_T^2(a)$ for the objective function $-\mathcal{L}(a)$, and optimize the hyperparameters.

Acquisition: Construct the acquisition function $\alpha(a) = \mu_T(a) + \gamma\sigma_T(a)$. Maximize the acquisition function $a^{T+1} = \arg\max_{a\in A} \alpha(a)$ to select the next sampling prefix a^{T+1}. For more efficient searching, perform N steps of random replacements on a^{T+1}, exploring the vicinity of a^{T+1}. Each replacement samples a batch of replacement tokens for each token position, keeping the best replacements. The final prefix after all replacements is denoted as a'^{T+1}.

Query and Iteration: Query the objective $-\mathcal{L}$ to obtain $y^{T+1} = -\mathcal{L}(a^{T+1})$ and $y'^{T+1} = -\mathcal{L}(a'^{T+1})$, and update the dataset: $\mathcal{D} = \mathcal{D} \cup \{(a^{T+1}, y^{T+1}), (a'^{T+1}, y'^{T+1})\}$. Repeat until the iteration limit is reached.

5 Experiment

5.1 Experiment Settings

Datasets. To evaluate our attack across a diverse set of reasoning scenarios, we use five benchmarks spanning math reasoning, general knowledge question answering, and code generation: GSM8K [13], MathQA [3], MATH-500 [35], MMLU-Pro [58], and HumanEval [8]. We randomly sample 100 instances from GSM8K, MathQA, MATH-500, and HumanEval, and select 10 questions from each of the 12 non-math categories in MMLU-Pro.

Models. We assess our attack using three state-of-the-art reasoning models: DeepSeek-R1-Distilled-Qwen-7b (R1-7b) [26], QwQ-32b [57], and gpt-oss-20b [1]. To simulate a realistic attack scenario, we additionally deploy a local vLLM inference server to mimic a commercial LLM service provider. More hardware and deployment details are deferred to Appendix C.

Baselines. We compare our attack against the state-of-the-art resource consumption attacks that induce excessively long outputs and increase inference latency. Specifically, we include multiple black-box baselines: chain-of-thought (CoT) prompting [59], CatAttack [50], and four variants of OverThink [32], including Context-Aware (CAW), Context-Agnostic (CAG), ICL-Genetic-Aware (ICLGAW), and ICL-Genetic-Agnostic (ICLGAG). For white-box baselines, we compare with the Denial-of-Service (DoS) prompt attack [25], Engorgio Prompt [16], and Excessive Reasoning Attack (ERA) [53]. Additionally, to provide a more direct and transparent comparison of optimization strategies, we develop three extra baselines (Heuristic Prompt, Black-box Search, Gradient-free Attack). More implementation details are provided in Appendix C.

Defenses. To evaluate the performance of our attack under defenses, we select methods for reasoning compression and resource-consumption mitigation, including CCoT [11], CoD [62], and PD^{3}F [67].

[1] Since the function $\mathcal{L}_{IA}$ cannot be computed in black-box scenarios, we use the cross-entropy loss of n tokens following the key reasoning tokens in relation to the short doubt-inducing phrases as a surrogate.

Table 2: Quantitative results of baselines and our attacks in extending output length and latency on three datasets across three models. Chatter Attack and Chatter-BPO Attack consistently outperforms all baselines in terms of average output token length (ATL), average inference time consumption (ATC), highlighting the potential risk of resource-depletion for deployed LLM services.

Model	Method	GSM8k			MATH-500			MathQA		
		ATL	ATC	ACC	ATL	ATC	ACC	ATL	ATC	ACC
R1-7b	Base	423.5 ± 16.8	4.7 ± 0.2	99.4	1270.6 ± 190.1	14.3 ± 2.2	92.5	411.1 ± 26.9	4.7 ± 0.3	52.3
	CoT	438.6 ± 27.8	5.1 ± 0.5	99.6	1435.7 ± 188.2	17.5 ± 2.1	96.3	887.4 ± 192.0	10.8 ± 4.2	54.7
	DoS	1158.7 ± 638.6	10.8 ± 5.0	96.7	1354.2 ± 391.2	14.8 ± 3.3	72.5	538.9 ± 36.3	6.0 ± 0.4	55.9
	Engorgio Prompt	569.3 ± 63.2	6.4 ± 0.7	98.4	1524.7 ± 275.0	17.3 ± 3.2	68.6	798.4 ± 96.9	9.0 ± 1.1	27.6
	CAW	601.4 ± 23.8	6.7 ± 0.3	99.7	1089.7 ± 105.7	12.2 ± 1.2	85.7	445.1 ± 42.2	5.0 ± 0.5	40.1
	CAG	500.3 ± 26.3	5.6 ± 0.3	90.7	974.5 ± 134.5	11.0 ± 1.5	83.7	394.5 ± 27.6	4.4 ± 0.3	41.4
	ICLGAW	533.4 ± 29.9	6.0 ± 0.3	95.5	1904.2 ± 755.9	18.3 ± 5.9	63.6	788.5 ± 242.6	9.0 ± 2.8	56.0
	ICLGAG	851.0 ± 285.7	9.7 ± 3.3	93.3	861.9 ± 86.7	9.8 ± 1.0	75.3	788.5 ± 242.6	9.0 ± 2.8	54.7
	CatAttack	482.0 ± 21.2	5.5 ± 0.2	75.9	1449.8 ± 201.1	16.8 ± 2.3	86.8	451.8 ± 37.0	5.1 ± 0.4	36.3
	ERA	546.1 ± 29.8	6.5 ± 0.4	94.2	1583.3 ± 120.9	18.7 ± 1.4	91.9	547.5 ± 63.7	5.8 ± 0.7	35.6
	Gradient-free Attack	474.9 ± 32.8	6.0 ± 0.4	88.6	1317.5 ± 130.9	16.7 ± 1.6	60.2	547.3 ± 57.5	6.5 ± 0.7	40.0
	Black-box Search	1456.7 ± 594.6	14.7 ± 4.3	84.9	2357.2 ± 237.1	28.0 ± 2.9	87.9	942.3 ± 51.5	11.1 ± 0.6	54.9
	Heuristic Prompt	883.5 ± 54.1	10.6 ± 0.7	78.1	1794.0 ± 167.1	21.6 ± 2.0	91.0	1156.1 ± 67.7	13.9 ± 0.8	54.4
	Chatter Attack (ours)	**13345.8 ± 1067.4**	**105.4 ± 16.4**	70.4	**9577.8 ± 1329.4**	**81.7 ± 8.9**	16.4	**7429.2 ± 1312.1**	67.3 ± 9.5	27.3
	Chatter-BPO Attack (ours)	7846.9 ± 1473.5	91.7 ± 17.4	67.6	6660.6 ± 1464.9	77.7 ± 17.3	13.5	7338.2 ± 1455.6	**87.5 ± 17.1**	22.9
QwQ-32b	Base	991.4 ± 149.4	25.0 ± 3.7	99.5	3007.0 ± 406.4	68.5 ± 7.6	94.4	3955.6 ± 243.2	95.4 ± 4.6	55.8
	CoT	1016.5 ± 137.1	25.5 ± 3.5	95.7	3102.5 ± 488.8	69.5 ± 8.0	97.8	3805.9 ± 230.8	92.5 ± 4.5	66.4
	DoS	828.6 ± 82.0	20.9 ± 2.1	99.3	2600.3 ± 351.2	62.8 ± 7.7	96.7	3159.4 ± 249.6	78.6 ± 5.8	55.6
	Engorgio Prompt	1450.2 ± 120.8	38.2 ± 3.2	99.6	3964.5 ± 400.7	90.4 ± 6.6	92.6	4980.3 ± 750.5	107.1 ± 19.1	45.4
	CAW	2314.0 ± 337.9	56.5 ± 7.7	99.3	2973.9 ± 432.9	67.4 ± 8.0	93.3	2872.4 ± 266.2	71.0 ± 5.9	44.5
	CAG	806.6 ± 97.5	20.4 ± 2.5	98.3	2235.5 ± 413.9	53.0 ± 6.9	92.2	2093.5 ± 209.8	52.9 ± 5.2	46.6
	ICLGAW	2275.5 ± 251.6	58.5 ± 6.6	97.7	4266.6 ± 453.8	91.1 ± 7.7	93.7	3885.8 ± 453.3	85.2 ± 12.1	45.4
	ICLGAG	1509.2 ± 283.4	39.8 ± 7.4	98.9	3893.3 ± 857.2	95.2 ± 21.3	87.9	3247.7 ± 356.6	58.9 ± 9.4	51.0
	CatAttack	1756.0 ± 206.1	44.2 ± 5.3	99.6	2741.1 ± 317.5	66.9 ± 7.1	90.7	3483.3 ± 237.3	84.9 ± 5.0	52.1
	ERA	2747.9 ± 284.0	65.7 ± 6.7	76.4	2875.4 ± 415.7	63.9 ± 6.6	92.4	3931.5 ± 234.8	91.8 ± 4.4	51.3
	Gradient-free Attack	914.1 ± 115.0	23.0 ± 2.9	99.8	3121.7 ± 408.5	69.3 ± 7.0	94.4	3989.9 ± 228.3	96.5 ± 4.4	52.3
	Black-box Search	2573.6 ± 433.9	60.2 ± 8.2	98.2	2600.4 ± 245.6	65.9 ± 6.3	92.2	4038.0 ± 301.4	102.6 ± 7.8	54.3
	Heuristic Prompt	2634.0 ± 388.6	61.5 ± 7.8	99.3	3687.8 ± 359.9	86.4 ± 6.9	96.9	3692.1 ± 341.9	85.3 ± 4.4	51.9
	Chatter Attack (ours)	**3298.4 ± 370.7**	77.3 ± 7.3	76.3	**5303.3 ± 696.3**	**122.1 ± 11.5**	61.1	**6915.4 ± 615.7**	**112.0 ± 3.8**	52.2
	Chatter-BPO Attack (ours)	3149.8 ± 489.1	**80.6 ± 12.7**	66.0	4800.0 ± 450.3	105.8 ± 10.3	55.5	4811.8 ± 296.1	109.1 ± 3.6	45.6
gpt-oss-20b	Base	253.0 ± 35.5	1.3 ± 0.2	97.8	389.9 ± 53.5	2.0 ± 0.3	81.3	112.8 ± 10.8	0.6 ± 0.1	43.1
	CoT	420.3 ± 56.0	2.2 ± 0.3	96.7	558.2 ± 68.4	2.9 ± 0.4	76.9	240.4 ± 14.5	1.2 ± 0.1	38.2
	DoS	363.8 ± 80.1	2.0 ± 0.4	86.7	850.2 ± 80.4	4.5 ± 0.6	87.9	478.6 ± 33.5	2.4 ± 0.2	45.2
	Engorgio Prompt	850.3 ± 280.6	5.3 ± 2.0	97.7	826.6 ± 70.4	3.8 ± 0.5	72.2	400.8 ± 43.5	2.5 ± 0.3	44.5
	CAW	248.9 ± 26.6	1.4 ± 0.2	97.5	360.4 ± 41.6	1.9 ± 0.2	85.6	192.3 ± 23.4	1.0 ± 0.1	20.9
	CAG	201.2 ± 19.8	1.0 ± 0.1	98.3	327.1 ± 43.1	1.7 ± 0.2	88.5	136.3 ± 9.8	0.7 ± 0.1	21.1
	ICLGAW	859.2 ± 156.8	5.2 ± 0.9	97.2	1258.9 ± 285.4	6.6 ± 1.5	92.8	784.7 ± 45.6	3.2 ± 0.3	47.8
	ICLGAG	824.6 ± 123.7	4.3 ± 0.6	98.1	1156.8 ± 258.9	6.0 ± 1.4	87.7	487.3 ± 48.1	2.2 ± 0.2	43.4
	CatAttack	463.6 ± 53.8	2.4 ± 0.3	96.7	594.6 ± 54.5	3.1 ± 0.3	96.2	260.5 ± 14.1	1.5 ± 0.1	30.0
	ERA	420.0 ± 56.2	2.5 ± 0.4	98.9	358.7 ± 33.1	1.4 ± 0.1	87.8	322.0 ± 30.5	1.2 ± 0.1	22.3
	Gradient-free Attack	372.0 ± 27.0	1.4 ± 0.1	96.4	562.9 ± 67.9	2.2 ± 0.3	84.7	247.0 ± 12.4	0.9 ± 0.0	45.1
	Black-box Search	504.2 ± 64.6	4.6 ± 0.6	99.5	658.8 ± 61.1	3.6 ± 0.3	87.9	300.8 ± 18.7	2.7 ± 0.2	42.4
	Heuristic Prompt	1055.2 ± 374.8	4.2 ± 1.6	95.6	936.9 ± 88.1	6.4 ± 0.6	85.3	584.2 ± 51.9	2.2 ± 0.2	35.9
	Chatter Attack (ours)	**1980.3 ± 508.8**	**11.4 ± 3.3**	71.2	1893.0 ± 332.7	9.6 ± 1.4	52.0	**910.5 ± 139.8**	**5.1 ± 0.8**	14.4
	Chatter-BPO Attack (ours)	1182.7 ± 172.5	6.6 ± 0.7	83.0	**2438.7 ± 344.3**	**10.6 ± 1.9**	56.7	820.5 ± 51.8	4.2 ± 1.2	18.2

We also implement and evaluate on multiple different production constraints, traditional infrastructure defenses, and post-generation defenses in a simulated server environment. More details are provided in Appendix C.

Metrics. To evaluate the effectiveness of our attack, we report the widely-used metrics including average output token length (ATL), average inference time consumption (ATC), and accuracy (ACC), with the 95% confidence intervals of ATL and ATC. Additionally, to mimic realistic attacks against API services and assess the impact of Chatter Attack on server load and responsiveness, we also monitor the server-side runtime indicators during inference including the counts of running requests (RR), GPU cache usage (GCU), response time to first token (TTFT), and the generated token counts per second (TPS). We provide more details in Appendix C.

Implementation Details. For the implementation of Chatter Attack, we use Greedy Coordinate Gradient (GCG) [71] to optimize an adversarial prefix a with a length of 20 tokens, and set the hyperparameters of the overall objective function $\mathcal{L}$ to $(\alpha, \beta, \lambda) = (1, 10, 1)$. For the implementation of Chatter-BPO Attack, we use the Adam optimizer [31] with a learning rate of 0.1 to optimize the hyperparameters of the Gaussian process. We maximize the acquisition function using GCG, and then apply a dynamic random-search schedule with gradually increasing intensity to enable more efficient exploration. Additional implementation details including the steps of GCG iterations, search batch size, and specific dynamic schedule are provided in the Appendix C.

5.2 Main Results

We demonstrate the effectiveness of our Chatter Attack and Chatter-BPO Attack in inducing long responses across multiple models and datasets. As shown in Table 2, Chatter Attack consistently achieves the largest average output token length (ATL) and the highest average inference time consumption (ATC) among all baselines, indicating attack effectiveness; in particular, on R1-7b it increases ATL by up to $31.5\times$ compared to benign generation for GSM8K. Chatter-BPO Attack

Table 3: Quantitative results of DoS attack and our attacks against multiple defenses on three datasets. Although the performance of Chatter Attack and Chatter-BPO Attack decrease when facing defenses, none of the defense can fully mitigate the impact.

Attack	Defense	GSM8k			MATH-500			MathQA		
		ATL	ATC	ACC	ATL	ATC	ACC	ATL	ATC	ACC
No Attack	No Defense	423.5 ± 16.8	4.7 ± 0.2	99.4	1270.6 ± 190.1	14.3 ± 2.2	92.5	411.1 ± 26.9	4.7 ± 0.3	52.3
DoS	No Defense	1158.7 ± 638.6	10.8 ± 5.0	96.7	1354.2 ± 391.2	14.8 ± 3.3	72.5	538.9 ± 36.3	6.0 ± 0.4	55.9
	CCoT	950.2 ± 457.8	9.2 ± 4.1	88.5	1226.6 ± 354.5	13.1 ± 2.9	65.2	486.5 ± 31.0	5.3 ± 0.3	48.7
	CoD	857.7 ± 385.6	8.1 ± 3.5	92.3	1167.4 ± 396.8	12.0 ± 2.5	68.9	420.8 ± 25.7	4.6 ± 0.3	52.4
	PD^3F	1051.5 ± 523.3	9.8 ± 4.6	90.1	1253.3 ± 378.4	13.6 ± 3.1	70.8	518.6 ± 33.8	5.6 ± 0.4	50.2
Chatter Attack (ours)	No Defense	13345.8 ± 1067.4	105.4 ± 16.4	70.4	9577.8 ± 1329.4	81.7 ± 8.9	16.4	7429.2 ± 1312.1	67.3 ± 9.5	27.3
	CCoT	7808.9 ± 1943.7	61.3 ± 13.8	25.3	8281.3 ± 1336.3	72.8 ± 9.3	6.8	3913.0 ± 1211.0	34.2 ± 8.8	15.2
	CoD	5088.4 ± 1352.1	43.2 ± 9.7	34.6	4819.2 ± 1187.9	44.7 ± 9.0	7.3	3330.6 ± 1191.6	28.1 ± 8.6	24.5
	PD^3F	6698.3 ± 1764.2	55.9 ± 11.4	55.9	5092.0 ± 1305.2	46.2 ± 9.2	14.9	4528.9 ± 1531.8	40.5 ± 9.0	22.4
Chatter-BPO Attack (ours)	No Defense	7846.9 ± 1473.5	91.7 ± 17.4	67.6	6660.6 ± 1464.9	77.7 ± 17.3	13.5	7338.2 ± 1455.6	87.5 ± 17.1	22.9
	CCoT	5579.2 ± 1243.5	52.3 ± 10.8	40.2	5890.5 ± 1103.8	53.4 ± 9.7	10.5	3226.8 ± 981.6	30.2 ± 7.8	18.3
	CoD	4202.6 ± 951.7	38.9 ± 8.2	45.8	3993.7 ± 859.4	36.2 ± 7.5	12.1	2884.4 ± 752.9	25.4 ± 6.3	28.7
	PD^3F	6395.8 ± 1340.3	48.7 ± 9.5	50.3	4554.2 ± 958.1	41.8 ± 8.2	16.8	3817.6 ± 1127.2	35.2 ± 8.4	25.6

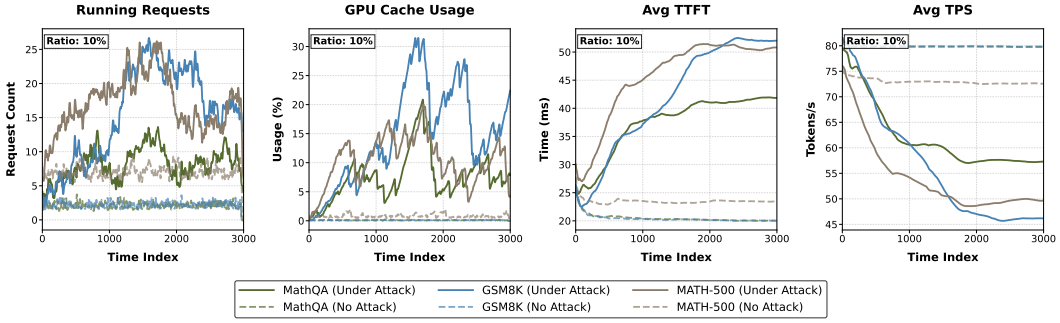

Figure 3: Server-side runtime metrics of the deployed LLM service under purely benign traffic and mixed traffic containing 10% Chatter Attack queries. The service is evaluated at a constant rate of 20 requests per minute. Compared to the benign baseline, injecting just 10% attack queries leads to a significantly larger backlog of pending requests and higher GPU cache usage, resulting in the slow response compared to benign queries.

achieves the second-best performance, while outperforming all baseline methods. We present the optimized attack prefixes and corresponding output examples in Table 18. The results show that both Chatter Attack and Chatter-BPO Attack successfully induce the target skeptical phrases, which then trigger substantially longer generations than the benign outputs. Meanwhile, we observe a drop in accuracy (ACC) in some settings, mainly because overly long outputs often hit the generation budget and fail to return a valid final answer. Overall, these results highlight the efficacy of our attacks.

5.3 Generalization to Different Reasoning Settings

In addition to mathematical reasoning, our attacks could generalize to broader reasoning settings, for example question-answering or code generation. As shown in Table 7, our attacks remain effective on QA benchmarks such as MMLU-Pro and coding benchmarks such as HumanEval, consistently achieving superior ATL and ATC compared to the baselines, which shows strong generalizability.

5.4 Attack Results against Defenses

We then study the impact of defenses on our attacks. We consider several representative defenses, including prompt-based methods (CCoT and CoD) and decoding-strategy defense (PD^3F). For comparison, we report the performance of DoS under the same defenses. In Table 3, although these defenses reduce the ATL and ATC induced by our attacks to some extent, it is worth noting that none of them can fully mitigate the impact: even with defenses enabled, Chatter and Chatter-BPO Attack still lead to a significant increase in ATL and ATC, highlighting their efficacy against mitigation. More results against multiple different production constraints, traditional infrastructure defenses, and post-generation defense are provided in Appendix D.9.

5.5 Attack Results on LLM Service

To more realistically emulate an LLM service operator, we deploy an R1-7B inference service and simulate an attack by sending a mix of benign and Chatter Attack queries at a fixed rate (e.g., 20 requests per minute). As shown in Figure 3, compared to purely benign traffic, requests injected with

only 10% Chatter Attack queries lead to a large backlog of pending requests, significantly increasing GPU cache usage and placing a substantial burden on the server. Moreover, these requests slow down the service response: the time to first token (TTFT) increases by $2\times$ to $3\times$, and the generation throughput drops from about 80 tokens/s to lower than 60 tokens/s, severely hindering normal service operation. Furthermore, we evaluate the impact of varying attack injection rates in Appendix D.7, demonstrating that even a marginal 1% of attack traffic is sufficient to cause noticeable fluctuations in response latency. Overall, these results provide a realistic demonstration of the potential risks and system-level impacts of resource exhaustion attacks in practice.

5.6 Attack Results against Production Constraints and Infrastructure Defenses.

In real-world deployments, API services commonly operate under multiple production constraints and are equipped with various defenses and output-side monitoring. To more comprehensively evaluate Chatter Attack, we implement similar defenses in a simulated LLM service. As shown in Table 4, directly limiting output length substantially mitigates the attack, also results in 6.2 RR, roughly three times the benign baseline. User rate limits provide only limited improvements to the resource exhaustion. IP blocking offers some mitigation when the attacker uses a fixed identity, but RR rises to 19.3 when the attacker rotates IP addresses. Output loop detection and excessive-reasoning monitoring likewise reduce RR, but service metrics still fail to recover to the benign baseline. Overall, these defenses mitigate the attack to varying degrees, but none can fully eliminate the impact of Chatter Attack. Implementation settings and more results under additional defenses are provided in Appendices C and D.9, respectively.

Table 4: Chatter-BPO Attack under multiple service-side defenses in an emulated deployment. Although some defenses reduce the impact of the attack, service metrics do not fully recover to the benign baseline.

Setting	RR	GCU (%)	TTFT (ms)	TPS
Benign	2.2	0.1	22.4	78.8
Under Attack	17.4	17.3	40.1	60.3
Output-Token Limit	6.2	3.0	30.0	71.3
Per-User Rate Limits	16.4	16.3	42.9	56.0
IP Blocking (Single IP)	7.4	7.6	36.8	60.1
IP Blocking (IP Rotation)	19.3	18.7	43.8	52.9
Output Loop Detection	9.1	9.2	39.6	58.6
Excessive-Reasoning Monitoring	6.2	5.9	31.4	67.5

5.7 Ablation Studies, Sensitivity Analysis and Additional Experiments

We conduct detailed ablation studies on the three terms in the Chatter Attack objective. As shown in Table 9, the full objective achieves the best performance, while removing either $\mathcal{L}_{\text{IA}}$ or $\mathcal{L}_{\text{GE}}$ leads to a consistent degradation. More detailed results and analyses are provided in Appendix D.1. We also report hyperparameter sensitivity analyses for the objective (Appendix D.2), the sensitivity of the attack prefix length (Appendix D.3), and an ablation study of the Chatter-BPO Attack (Appendix D.4). Additionally, we evaluate the robustness to decoding strategies (Appendix D.5) and quantization (Appendix D.6), provide the full attack results on deployed LLM services (Appendix D.7), and report the attack effectiveness and transferability on real API services (Appendix D.8) and resilience against additional deployment constraints and defenses (Appendix D.9).

6 Conclusion

In this paper, we introduce the Chatter Attack and Bayesian Prompt Optimization (BPO) as effective strategies to exploit the resource consumption vulnerabilities of LLMs. By optimizing short target phrases to induce self-doubt and maximize attention and entropy during the generation process, the Chatter Attack successfully manipulates output length, even in black-box settings with BPO. We further derive a lower bound on the expected generation length in terms of the three-loss surrogate objective, providing theoretical support for our loss design. Experiments demonstrate the superiority of our approach, with significant increases in response length and evidence of potential resource depletion and service disruption. These findings underscore the importance of developing robust defensive mechanisms to protect against such adversarial attacks in real-world LLM deployments.

Limitation. A limitation of our work is that the proposed attack relies on the sequential decoding process of autoregressive LLMs and cannot be directly applied to diffusion language models, which generate tokens non-sequentially. Moreover, the attack is less effective on conventional conversational models without explicit chain-of-thought reasoning, requiring a redesign of the target phrases and the set of critical tokens. We leave extensions to non-reasoning conversational models and non-autoregressive architectures for future research.

Acknowledgments and Disclosure of Funding

This paper is supported by the Strategic Priority Research Program of CAS Grant No. XDA0480502 and XDA0480503, the NSFC Projects (62276149, U2341228, 62595771, 62676212) and Beijing Nova Program (202604841276). Yibo Miao is also supported by the Young Elite Scientists Sponsorship Program of the China Association for Science and Technology (CAST).

References

[1] Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

[2] Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. Large language models for mathematical reasoning: Progresses and challenges. *arXiv preprint arXiv:2402.00157*, 2024.

[3] Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*, 2019.

[4] Yichuan Cao, Yibo Miao, Xiao-Shan Gao, and Yinpeng Dong. Red-teaming text-to-image systems by rule-based preference modeling. *Advances in Neural Information Processing Systems*, 38:150703–150734, 2026.

[5] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE symposium on security and privacy (SP)*, pages 39–57. IEEE, 2017.

[6] Center for AI Safety, Scale AI, and HLE Contributors Consortium. A benchmark of expert-level academic questions to assess AI capabilities. *Nature*, 649:1139–1146, 2026. doi: 10.1038/s41586-025-09962-4. URL https://arxiv.org/abs/2501.14249.

[7] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. *ACM transactions on intelligent systems and technology*, 15(3):1–45, 2024.

[8] Mark Chen. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[9] Simin Chen, Zihe Song, Mirazul Haque, Cong Liu, and Wei Yang. Nicgslowdown: Evaluating the efficiency robustness of neural image caption generation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15365–15374, 2022.

[10] Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, et al. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. *arXiv preprint arXiv:2412.21187*, 2024.

[11] Jeffrey Cheng and Benjamin Van Durme. Compressed chain of thought: Efficient reasoning through dense representations. *arXiv preprint arXiv:2412.13171*, 2024.

[12] Shuyu Cheng, Yibo Miao, Yinpeng Dong, Xiao Yang, Xiao-Shan Gao, and Jun Zhu. Efficient black-box adversarial attacks via bayesian optimization guided by a function prior. In *International Conference on Machine Learning*, pages 8163–8183, 2024.

[13] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[14] Alejandro Cuadron, Dacheng Li, Wenjie Ma, Xingyao Wang, Yichuan Wang, Siyuan Zhuang, Shu Liu, Luis Gaspar Schroeder, Tian Xia, Huanzhi Mao, et al. The danger of overthinking: Examining the reasoning-action dilemma in agentic tasks. *arXiv preprint arXiv:2502.08235*, 2025.

[15] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

[16] Jianshuo Dong, Ziyuan Zhang, Qingjie Zhang, Tianwei Zhang, Hao Wang, Hewu Li, Qi Li, Chao Zhang, Ke Xu, and Han Qiu. An engorgio prompt makes large language model babble on. *arXiv preprint arXiv:2412.19394*, 2024.

[17] Jianshuo Dong, Ziyuan Zhang, Qingjie Zhang, Tianwei Zhang, Hao Wang, Hewu Li, Qi Li, Chao Zhang, Ke Xu, and Han Qiu. An engorgio prompt makes large language model babble on. In *International Conference on Learning Representations*, 2025.

[18] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9185–9193, 2018.

[19] Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, 2018.

[20] Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards revealing the mystery behind chain of thought: a theoretical perspective. *Advances in Neural Information Processing Systems*, 36:70757–70798, 2023.

[21] Xiaoning Feng, Xiaohong Han, Simin Chen, and Wei Yang. Llmeffichecker: Understanding and testing efficiency degradation of large language models. *ACM Transactions on Software Engineering and Methodology*, 33(7):1–38, 2024.

[22] Peter I Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

[23] Kuofeng Gao, Yang Bai, Jindong Gu, Shu-Tao Xia, Philip Torr, Zhifeng Li, and Wei Liu. Inducing high energy-latency of large vision-language models with verbose images. In *International Conference on Learning Representations*, 2024.

[24] Kuofeng Gao, Tianyu Pang, Chao Du, Yong Yang, Shu-Tao Xia, and Min Lin. Denial-of-service poisoning attacks against large language models. *arXiv preprint arXiv:2410.10760*, 2024.

[25] Jonas Geiping, Alex Stein, Manli Shu, Khalid Saifullah, Yuxin Wen, and Tom Goldstein. Coercing llms to do and reveal (almost) anything. *arXiv preprint arXiv:2402.14020*, 2024.

[26] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[27] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020.

[28] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

[29] Xiaojun Jia, Tianyu Pang, Chao Du, Yihao Huang, Jindong Gu, Yang Liu, Xiaochun Cao, and Min Lin. Improved techniques for optimization-based jailbreaking on large language models. *arXiv preprint arXiv:2405.21018*, 2024.

[30] Donald R Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of global optimization*, 21:345–383, 2001.

[31] Diederik P Kingma. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[32] Abhinav Kumar, Jaechul Roh, Ali Naseh, Marzena Karpinska, Mohit Iyyer, Amir Houmansadr, and Eugene Bagdasarian. Overthink: Slowdown attacks on reasoning llms. *arXiv preprint arXiv:2502.02542*, 2025.

[33] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626, 2023.

[34] Baolin Li, Yankai Jiang, Vijay Gadepally, and Devesh Tiwari. Llm inference serving: Survey of recent advances and opportunities. In *2024 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–8. IEEE, 2024.

[35] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

[36] Hanmeng Liu, Zhizhang Fu, Mengru Ding, Ruoxi Ning, Chaoli Zhang, Xiaozhang Liu, and Yue Zhang. Logical reasoning in large language models: A survey. *arXiv preprint arXiv:2502.09100*, 2025.

[37] Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. Typical decoding for natural language generation. *arXiv preprint arXiv:2202.00666*, 4, 2022.

[38] Mike Merrill, Alexander Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Shin, Thomas Walshe, E Kelly Buchanan, et al. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces. In *International Conference on Learning Representations*, volume 2026, pages 40903–40986, 2026.

[39] Meta. How companies are using meta llama, 2024. URL https://about.fb.com/news/2024/05/how-companies-are-using-meta-llama/.

[40] Yibo Miao, Yinpeng Dong, Jun Zhu, and Xiao-Shan Gao. Isometric 3d adversarial examples in the physical world. In *Advances in Neural Information Processing Systems*, volume 35, pages 19716–19731, 2022.

[41] Yibo Miao, Yinpeng Dong, Jinlai Zhang, Lijia Yu, Xiao Yang, and Xiao-Shan Gao. Improving robustness of 3d point cloud recognition from a fourier perspective. *Advances in Neural Information Processing Systems*, 37:68183–68210, 2024.

[42] Yibo Miao, Yifan Zhu, Lijia Yu, Jun Zhu, Xiao-Shan Gao, and Yinpeng Dong. T2vsafetybench: Evaluating the safety of text-to-video generative models. In *Advances in Neural Information Processing Systems*, volume 37, pages 63858–63872, 2024.

[43] Jonas Močkus. On bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference*, pages 400–404, 1975.

[44] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. A comprehensive overview of large language models. *ACM Transactions on Intelligent Systems and Technology*, 16(5):1–72, 2025.

[45] Minh Nhat Nguyen, Andrew Baker, Clement Neo, Allen Roush, Andreas Kirsch, and Ravid Shwartz-Ziv. Turning up the heat: Min-p sampling for creative and coherent llm outputs. *arXiv preprint arXiv:2407.01082*, 2024.

[46] Pratyush Patel, Esha Choukse, Chaojie Zhang, Aashaka Shah, Íñigo Goiri, Saeed Maleki, and Ricardo Bianchini. Splitwise: Efficient generative llm inference using phase splitting. In *2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA)*, pages 118–132. IEEE, 2024.

[47] David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. Carbon emissions and large neural network training. *arXiv preprint arXiv:2104.10350*, 2021.

[48] David Patterson, Joseph Gonzalez, Urs Hölzle, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David R So, Maud Texier, and Jeff Dean. The carbon footprint of machine learning training will plateau, then shrink. *Computer*, 55(7):18–28, 2022.

[49] Tejal Patwardhan, Rachel Dias, Elizabeth Proehl, Grace Kim, Michele Wang, Olivia Watkins, Simon Fishman, Marwan Aljubeh, Phoebe Thacker, Laurance Fauconnet, et al. Gdpval: Evaluating ai model performance on real-world economically valuable tasks. In *International Conference on Learning Representations*, volume 2026, pages 24005–24040, 2026.

[50] Meghana Rajeev, Rajkumar Ramamurthy, Prapti Trivedi, Vikas Yadav, Oluwanifemi Bamgbose, Sathwik Tejaswi Madhusudan, James Zou, and Nazneen Rajani. Cats confuse reasoning llm: Query agnostic adversarial triggers for reasoning models. *arXiv preprint arXiv:2503.01781*, 2025.

[51] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71, 2003.

[52] Ilia Shumailov, Yiren Zhao, Daniel Bates, Nicolas Papernot, Robert Mullins, and Ross Anderson. Sponge examples: Energy-latency attacks on neural networks. In *2021 IEEE European symposium on security and privacy (EuroS&P)*, pages 212–231. IEEE, 2021.

[53] Wai Man Si, Mingjie Li, Michael Backes, and Yang Zhang. Excessive reasoning attack on reasoning llms. *arXiv preprint arXiv:2506.14374*, 2025.

[54] Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias W. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *International Conference on Machine Learning*, pages 1015–1022, 2010.

[55] Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, et al. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*, 2025.

[56] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[57] Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, 2025.

[58] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290, 2024.

[59] 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.

[60] Yuyang Wu, Yifei Wang, Ziyu Ye, Tianqi Du, Stefanie Jegelka, and Yisen Wang. When more is less: Understanding chain-of-thought length in llms. *arXiv preprint arXiv:2502.07266*, 2025.

[61] Jin Xu, Xiaojiang Liu, Jianhao Yan, Deng Cai, Huayang Li, and Jian Li. Learning to break the loop: Analyzing and mitigating repetitions for neural text generation. *Advances in Neural Information Processing Systems*, 35:3082–3095, 2022.

[62] Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. Chain of draft: Thinking faster by writing less. *arXiv preprint arXiv:2502.18600*, 2025.

[63] Xiaobei Yan, Yiming Li, Hao Wang, Han Qiu, and Tianwei Zhang. Bithydra: Towards bit-flip inference cost attack against large language models. *arXiv preprint arXiv:2505.16670*, 2025.

[64] Zhun Yang, Adam Ishay, and Joohyung Lee. Coupling large language models with logic programming for robust and general reasoning from text. *arXiv preprint arXiv:2307.07696*, 2023.

[65] Lijia Yu, Xiao-Shan Gao, Lijun Zhang, and Yibo Miao. Generalizablity of memorization neural network. *Advances in Neural Information Processing Systems*, 37:113311–113359, 2024.

[66] Lijia Yu, Shuang Liu, Yibo Miao, Xiao-Shan Gao, and Lijun Zhang. Generalization bound and new algorithm for clean-label backdoor attack. In *International Conference on Machine Learning*, pages 57559–57596, 2024.

[67] Yuanhe Zhang, Xinyue Wang, Haoran Gao, Zhenhong Zhou, Fanyu Meng, Yuyao Zhang, and Sen Su. Pd3f: A pluggable and dynamic dos-defense framework against resource consumption attacks targeting large language models. *arXiv preprint arXiv:2505.18680*, 2025.

[68] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2), 2023.

[69] Yifan Zhu, Yibo Miao, Yinpeng Dong, and Xiao-Shan Gao. Toward availability attacks in 3d point clouds. In *International Conference on Machine Learning*, pages 62510–62530, 2024.

[70] Yifan Zhu, Yibo Miao, Yinpeng Dong, and XIAOSHAN GAO. Why do unlearnable examples work: A novel perspective of mutual information. In *International Conference on Learning Representations*, volume 2026, pages 107849–107880, 2026.

[71] 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 Impact Statement

A potential negative societal impact of our work is that malicious adversaries could misuse the proposed resource consumption attacks (Chatter Attack and Chatter-BPO Attack) against deployed large language model (LLM) services. By inducing unusually long generations, such attacks can increase inference latency and operational costs, degrade quality of service, and potentially contribute to denial-of-service-like outages that harm legitimate users. These risks are particularly relevant for publicly accessible or pay-per-use APIs, and they may also increase energy consumption and carbon emissions. To mitigate these risks, our paper evaluates representative mitigation/defense strategies and highlights practical monitoring signals (e.g., request concurrency, time-to-first-token, and token throughput) that can help operators detect and respond to abnormal long-generation behaviors. We encourage service providers to combine these insights with deployment safeguards such as output-length limits, per-user rate limiting, anomaly detection, and server-side termination policies. To further reduce misuse, we will avoid releasing directly weaponizable attack prompts or code that enables unattended large-scale abuse, and we will share relevant findings with affected organizations to support more secure and reliable LLM deployments.

B Related Work

Reasoning Large Language Model. Reasoning LLMs, such as OpenAI o1/o3 [28], DeepSeek-R1 [26], and QWQ [57], enhance the handling of multi-step reasoning tasks, including mathematics [2], programming [64], and logical reasoning [36], by integrating explicit multi-step reasoning chains [59] into LLMs. However, as the reasoning capability of these models increases, the computational resource consumption during the inference stage becomes a significant concern. Previous studies indicate that when the query volume exceeds several hundred million, the inference cost surpasses the training cost, making large-scale deployment a major energy consumer [48]. Furthermore, these models are prone to experiencing the "overthinking" phenomenon [14, 10, 55], where unnecessary lengthy reasoning steps are generated during inference. Chen et al. [10] study this phenomenon and show that LRMs often expend excessive computational resources on simple problems. In this study, we adopt the role of a malicious attacker, exploring how to exploit this characteristic to induce lengthy outputs, thereby increasing response latency and inference costs.

Resource Consumption Attack. Deep learning safety has been extensively studied [56, 5, 18, 40, 12, 41, 69, 66, 42, 4, 70, 65]. Building on this, resource consumption attacks are first introduced by Shumailov et al. [52], customized for encoder-decoder architectures, by maximizing the activation norm to increase computational costs, thereby maximizing energy consumption and latency during model inference. NICGSlowDown [9] manipulates the logits of output tokens, increasing the number of decoder calls to reduce inference efficiency. However, Sponge Samples [52] are specifically designed for translation tasks, while NICGSlowDown targets image captioning tasks for Recurrent Neural Networks (RNNs) or Long Short-Term Memory networks (LSTMs). These methods are not directly applicable to modern large language models (LLMs) that adopt autoregressive generation schemes. OverThink Attack [32] and CatAttack [50] introduce semantic manipulation, such as injecting inherently open-ended problem prompts or complex mathematical problems, causing the model to allocate additional reasoning resources to solve the baited prompts. While occasionally effective, these techniques generally rely on handcrafted templates, lacking robustness and generalization capabilities, and demonstrate effectiveness only on specific models. Geiping et al. [25] propose an adversarial prompt attack, setting the attack target as repeated vocabulary, coercing LLMs to repeat specific content, thereby increasing inference latency by extending the model's output length. Similarly, methods like Engorgio [17, 53, 63, 23, 21] minimize the logits of the end-of-sequence (EOS) token relative to each output token, suppressing the EOS token's appearance and generating excessively long text outputs, imposing a computational burden on the model. Since LLMs are autoregressive and sampling-based, the search space of these methods grows exponentially with the number of tokens due to the inherent randomness, making it difficult to optimize effectively.

C Detailed Experiment Settings

Datasets. We evaluate our attacks on five benchmarks covering math reasoning, general-knowledge question answering, and code generation: the grade school level GSM8K [13], multiple choices

problems MathQA [3], high school level mathematical datasets MATH-500 [35], challenging question-answering benchmark MMLU-Pro [58], and code generation benchmark HumanEval [8]. For GSM8K, MathQA, MATH-500, and HumanEval, we randomly sample 100 instances per dataset. For MMLU-Pro, we select 10 questions from each of the 12 non-math categories (for example, chemistry, biology).

Models. We assess our attack using three state-of-the-art reasoning models: DeepSeek-R1-Distilled-Qwen-7b (R1-7b) [26], QwQ-32b [57], and gpt-oss-20b [1]. For model generation, we generate up to 16384 tokens with temperature 0.6, top-p 0.95, and repetition penalty 1.0, which is a commonly used configuration for evaluating the capability of model reasoning. To simulate a realistic attack scenario, we additionally deploy a local vLLM inference server to mimic a commercial LLM service provider. We deploy the model using vLLM [33] on one NVIDIA A800 GPU and issue queries with the same decoding parameters.

Baselines. We compare our attack against the state-of-the-art resource consumption attacks that induce excessively long outputs and increase inference latency. Specifically, we include multiple black-box baselines: chain-of-thought (CoT) prompting [59], CatAttack [50], and four variants of OverThink [32], including Context-Aware (CAW), Context-Agnostic (CAG), ICL-Genetic-Aware (ICLGAW), and ICL-Genetic-Agnostic (ICLGAG). For CoT prompting, we instruct the model to reason step-by-step using the prompt template illustrated in Table 5. For two ICL-Genetic variants of OverThink, we use the GPT-5-mini for adversarial context generation and perform 10 rounds optimization. For white-box baselines, we compare with the Denial-of-Service (DoS) prompt attack [25], Engorgio Prompt [16], and Excessive Reasoning Attack (ERA) [53]. For DoS attack, we perform GCG algorithm with the repetition of the string "Hello There" 24 times as the objectives, and optimize with 64 tokens suffix. For Engorgio Prompt, we adapt a pre-set maximum length 2048 for all reasoning models. For Excessive Reasoning Attack, we optimize the 10 token suffix with 1000 steps GCG. All experiments are performed exactly according to their experimental configuration respectively.

Table 5: Prompt template for CoT prompting.

Base prompting:
<ORIGINAL QUESTION >
Please put your answer in \boxed{}.
CoT prompting:
<ORIGINAL QUESTION >
Please reason step by step and put your answer in \boxed{}.

Additionally, we also develop three extra baselines to provide a more direct and transparent comparison of optimization strategies. For the Black-box Search baseline, we prompt GPT-5-mini to iteratively refine the current prefix to maximize the output length. The Gradient-free Attack performs a 3,000-step random search to construct the prefix, modifying one token per step in the direction that increases the output length. Heuristic Prompt is a heuristic prompt-based method inducing longer output, with the prompt template shown in Table 6.

Defenses. To more comprehensively evaluate the performance of our attack under defenses, we select several representative methods for reasoning compression and resource-consumption mitigation, including the prompt-based CCoT [11], CoD [62], and decoding-based PD^{3}F [67]. CCoT and CoD use an explict prompt to instruct model keep concise for the output, while PD^{3}F develop a complex systems to monitor the output budget of LLM.

Besides, we also implement and evaluate on multiple different production constraints, traditional infrastructure defenses, and post-generation defenses in a simulated server environment. All evaluations share the same experimental setup: queries are injected continuously at intervals of 2–4s over a total duration of 3000s, with malicious attack requests accounting for 10% of the traffic and the remaining 90% consisting of lightweight background traffic from 200 independent benign accounts. The attacker is assigned a conservative resource pool consisting of 5 IPs and 20 accounts, with each IP hosting 4 accounts. For the production constraints, strict output-token limits cap every request's max_tokens at 8,192 at admission. Per-user rate limits enforce token-bucket-based limits with RPM $= 20$ and TPM $= 10$k. Dynamic batching dynamically varies vLLM's concurrency limit between 256 and 512. Request prioritization maps account reputation, derived from registration age, historical output length, and violation records, to priority levels in vLLM's native scheduler. Server-side early

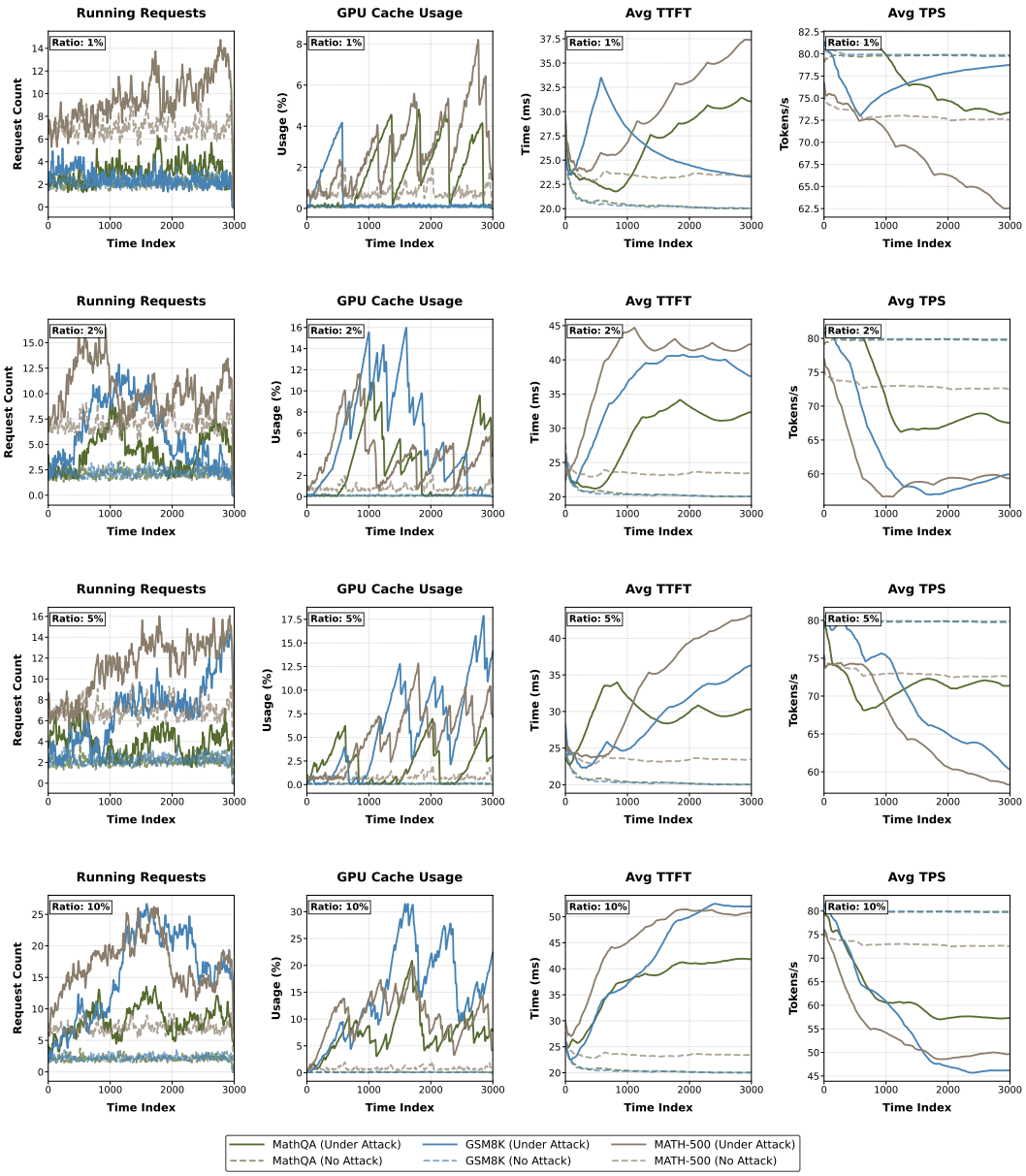

Figure 4: Server-side runtime metrics of the deployed LLM service under purely benign traffic and mixed traffic containing 1% to 10% Chatter Attack queries. The service is evaluated at a constant rate of 20 requests per minute.

termination policies use a lightweight online trigger based on cumulative self-doubt markers or generated length to abort requests in real time and release resources. For traditional infrastructure defenses, spend caps are set to a $5 trial budget per account, equivalent to a quota of ∼100k output tokens based on simulated unit pricing. IP/account blocking is triggered by 3 consecutive outputs exceeding 5000 tokens, resulting in a 600s blocking. These threshold settings are highly realistic, as they both ensure legitimate usage by normal users and reflect actual defense baselines. Anomaly detection covers both the input and output sides. On the input side, we compute prompt perplexity using Qwen2.5-7B-Instruct to filter high-PPL inputs. On the output side, loop detection follows classic degeneration criteria [27, 61], flagging an output when its repeated 4-gram ratio exceeds 0.5 or its gzip compression ratio falls below 0.10. Excessive-reasoning monitoring is implemented using an LLM-as-a-judge (Qwen2.5-7B-Instruct) that classifies each completed CoT as overthinking or normal based on the overthinking criteria in [10]. In both output-side cases, the enforcement policy is

Table 6: Prompt template for Heuristic Prompt.

Please answer the following QUESTION: QUESTION: <ORIGINAL QUESTION > You should follow these guidelines strictly: 1. Multi-Dimensional Deconstruction: Thoroughly decompose the problem into its constituent parts, examining the core issue and all peripheral factors from a panoramic perspective. 2. Explicit Logical Chain: Externalize every micro-step of your reasoning process. Avoid logical leaps and ensure that the transition between every premise and conclusion is fully articulated with high-density detail. 3. Recursive Layering of Depth: After reaching an initial conclusion, drill down into deeper layers of "Why" and "How." Analyze the implications within various theoretical frameworks, historical contexts, or extreme edge cases. 4. Exhaustive Scenario Simulation: Explore not only the most probable outcome but also all alternative interpretations and counter-narratives. Provide a detailed walkthrough of the mechanics, causes, and consequences of each possibility. 5. Incremental Enrichment: For every concept or term introduced, provide a comprehensive background explanation or contextual definition before proceeding with the main argument to ensure total clarity. 6. Iterative Expansion: Upon completing a section, revisit it to identify areas where information density can be increased. Force-multiply the output by adding illustrative examples, data-driven analogies, or comparative analysis until maximum granularity is achieved. Please reason step by step and put your answer in \boxed{}.

Table 7: Quantitative results of baselines and our attacks on question-answer and coding benchmarks on R1-7b. Chatter Attack and Chatter-BPO Attack could generalize to these extensive reasoning settings.

	MMLU-Pro			HumanEval		
	ATL	ATC	ACC	ATL	ATC	ACC
Base	1495.0 ± 492.0	15.6 ± 3.7	28.7	1554.7 ± 138.1	17.9 ± 1.6	98.3
CoT	1550.6 ± 429.2	17.0 ± 3.8	26.9	1691.2 ± 167.1	19.4 ± 1.9	98.8
DoS	1220.2 ± 401.7	13.9 ± 3.6	16.3	1352.2 ± 92.9	16.4 ± 1.1	97.4
Engorgio Prompt	1760.8 ± 419.2	19.7 ± 3.8	25.5	1287.9 ± 181.1	14.9 ± 2.1	97.9
CAW	861.5 ± 119.2	9.9 ± 1.4	15.1	1453.8 ± 123.5	16.7 ± 1.4	99.3
CAG	767.6 ± 94.9	9.3 ± 1.1	7.8	1430.9 ± 112.9	17.3 ± 1.4	99.5
ICLGAW	924.5 ± 157.4	11.0 ± 1.9	17.5	3297.1 ± 906.2	34.3 ± 8.2	85.4
ICLGAG	774.4 ± 86.3	9.2 ± 1.0	15.5	1327.0 ± 101.0	15.7 ± 1.2	99.8
CatAttack	1306.2 ± 184.2	15.0 ± 2.1	16.1	1646.7 ± 145.1	18.9 ± 1.7	98.0
ERA	757.0 ± 54.3	8.9 ± 0.6	5.9	3195.4 ± 993.9	29.8 ± 7.0	88.3
Gradient-free Attack	1170.1 ± 151.9	13.3 ± 1.7	15.4	1463.3 ± 98.3	17.0 ± 1.2	99.1
Black-box Search	1309.7 ± 209.5	13.9 ± 2.1	19.7	2538.1 ± 120.5	29.8 ± 1.4	97.6
Heuristic Prompt	1333.4 ± 234.9	14.4 ± 2.5	14.0	2221.5 ± 184.4	23.5 ± 2.0	98.0
Chatter Attack (ours)	$\mathbf{6638.3 \pm 962.4}$	$\mathbf{70.2 \pm 11.6}$	5.3	$\mathbf{5788.4 \pm 1400.3}$	63.4 ± 17.0	43.3
Chatter-BPO Attack (ours)	4251.1 ± 1159.5	49.3 ± 13.7	4.4	5516.2 ± 1285.6	$\mathbf{64.2 \pm 15.2}$	44.5

aligned with our strictest account-level strategy: an account is banned for 600s after accumulating 3 flagged outputs.

Metrics. To evaluate the effectiveness of our attack, we report the widely-used metrics including average output token length (ATL), average inference time consumption (ATC), and accuracy (ACC), with the 95% confidence intervals of ATL and ATC. Additionally, to mimic realistic attacks against API services and assess the impact of Chatter Attack on server load and responsiveness, we also monitor the server-side runtime indicators during inference including the counts of running requests (RR), GPU cache usage (GCU), response time to first token (TTFT), and the generated token counts per second (TPS). RR and GCU represent the state of server load, while TTFT and TPS directly reflect the velocity of response and generation for the service.

Implementation Details. For the implementation of Chatter Attack, we use Greedy Coordinate Gradient (GCG) [71] to optimize an adversarial prefix a with a length of 20 tokens for 3000 steps, and use a sample budget of 64. We set the hyperparameters of the overall objective function $\mathcal{L}$ to $(\alpha, \beta, \lambda) = (1, 10, 1)$. Furthermore, we employ the vocabulary presented in Table 8 to identify the critical reasoning tokens for inference attention loss. For the implementation of Chatter-BPO Attack, we use the gpytorch to implement the kernel. We utilize the Adam optimizer [31] with a learning rate of 0.1 to optimize the hyperparameters of the Gaussian process. We set $\gamma = 2$. We maximize the acquisition function using 100 steps GCG, and then apply a dynamic random-search schedule with gradually increasing intensity from no search $N = 0$ to $N = 16$ steps iterations, to enable more efficient exploration.

Table 8: Critical reasoning words vocabulary for inference attention loss.

Wait	Alternatively	But	However	Hmm
wait	alternatively	but	however	hmm

Table 9: Ablation studies of the loss function for Chatter Attack.

	ATL	ATC	ACC
Chatter Attack	13345.8 ± 1067.4	105.4 ± 16.4	70.4
Chatter Attack w/o $\mathcal{L}_{\text{IA}}$	9545.6 ± 1285.1	96.6 ± 15.2	78.9
Chatter Attack w/o $\mathcal{L}_{\text{GE}}$	9282.3 ± 1400.5	93.9 ± 16.7	77.2
Chatter Attack w/o $\mathcal{L}_{\text{IA}}$ & $\mathcal{L}_{\text{GE}}$	8088.0 ± 1237.0	89.3 ± 14.6	75.6

D Additional Experiments

D.1 Ablation Studies of Chatter Attack's Objective Function

We adopt the following objective function for Chatter Attack:

$$\mathcal{L} = \alpha\mathcal{L}_{\text{ID}} + \beta\mathcal{L}_{\text{IA}} + \lambda\mathcal{L}_{\text{GE}} \tag{12}$$

To better understand the contribution of each term, we perform ablation studies on the three components of the Chatter Attack objective. As shown in Table 9, combining $\mathcal{L}_{\text{ID}}$ with either $\mathcal{L}_{\text{IA}}$ or $\mathcal{L}_{\text{GE}}$ consistently outperforms using $\mathcal{L}_{\text{ID}}$ alone. Moreover, the full objective that includes all three losses achieves the best performance, suggesting that these terms are complementary.

D.2 Sensitivity Analyses of the Hyperparameters for the Loss Function

We conduct a detailed sensitivity analysis of the hyperparameters in the Chatter Attack objective function. Specifically, we fix $\alpha = 1$ and vary β and λ, optimizing the attack on GSM8K with R1-7b. As shown in Table 10, the default configuration achieves the best performance; meanwhile, across a wide range of hyperparameter settings, our attack consistently attains stronger optimization results than the baselines (Table 2), indicating that Chatter Attack is relatively robust to hyperparameter choices.

D.3 Sensitivity Analysis of Attack Prefix Length

We perform a sensitivity analysis regarding the attack prefix length in Chatter Attack. In addition to the baseline length of 20 tokens, we test lengths of 10, 15, 25, and 30 tokens against GSM8K with R1-7b model. As shown in Table 11, reducing the prefix length compromises the attack efficacy, while extending it beyond 20 tokens provides no marginal gain in ATL. In a word, the 20-token default setting yields strong attack results while avoiding unnecessary generation costs.

D.4 Ablation Studies of Chatter-BPO Attack

To validate the effectiveness of our proposed optimization framework, we conduct an ablation study on the key components of Chatter-BPO. We compare the full Chatter-BPO algorithm against two baselines: (1) Random Search, which iteratively and randomly swaps tokens in a black-box manner to maximize the target objective, and (2) BPO w/o Local Search, which solely relies on the Bayesian Optimization to sample prompts, without performing random replacements. As presented in Table 12, Random Search performs poorly, highlighting the difficulty of navigating the highly non-convex and high-dimensional discrete token space without heuristic guidance. Meanwhile, Chatter-BPO without Local Search achieves a slightly lower performance compared to Chatter-BPO. While the BO successfully identifies promising semantic neighborhoods and narrows down the search space, the discrete nature of textual tokens still causes the reward landscape to be locally jagged. Without the local search, the algorithm struggles to pinpoint the exact token combinations required to fully trigger the model's severe infinite reasoning loops. Ultimately, the full Chatter-BPO achieves the highest attack efficacy, confirming that local search plays an important role in optimization.

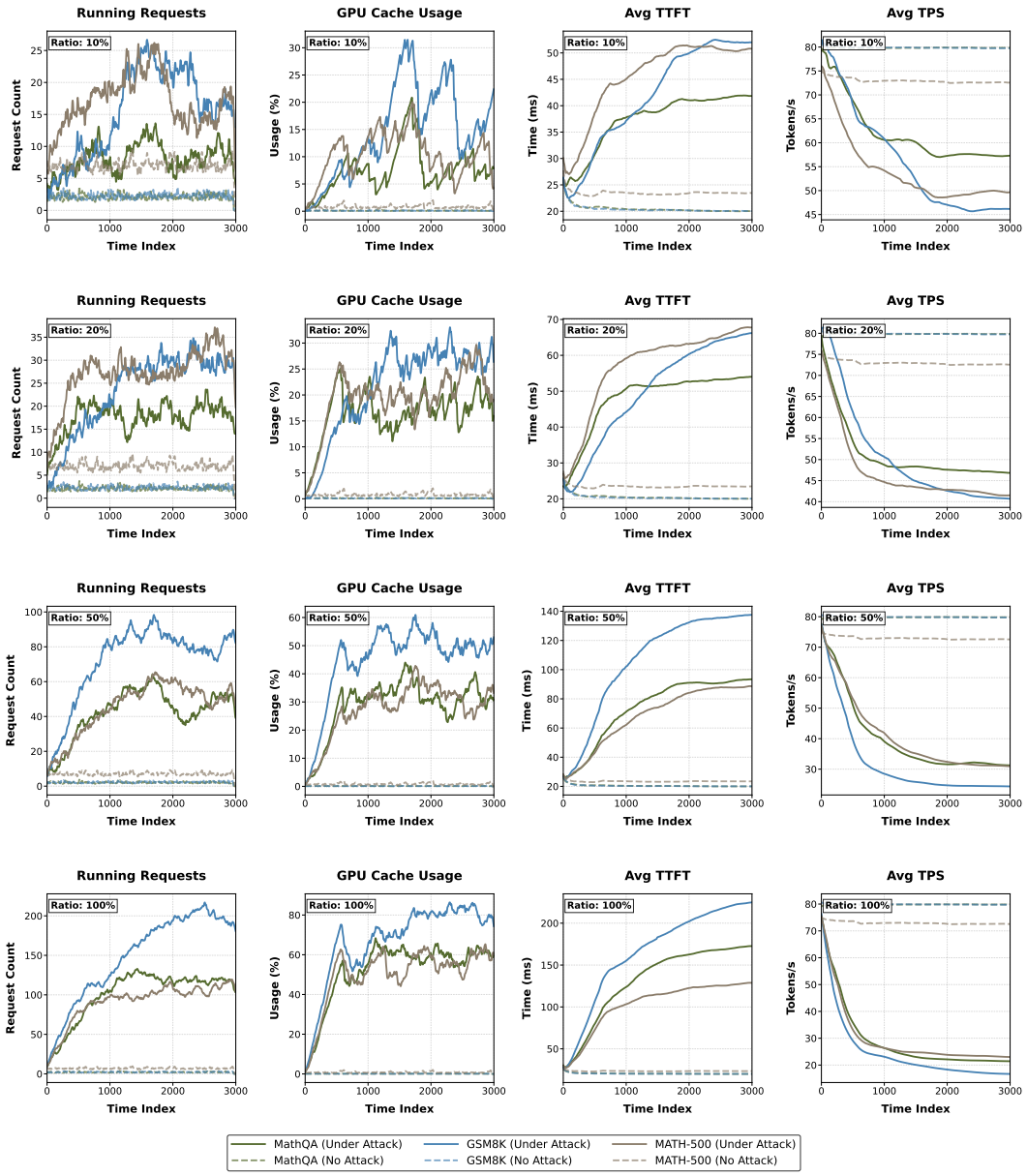

Figure 5: Server-side runtime metrics of the deployed LLM service under purely benign traffic and mixed traffic containing 10% to 100% Chatter Attack queries. The service is evaluated at a constant rate of 20 requests per minute.

D.5 Robustness to Different Decoding Strategies

In real-world deployments, different servers may employ diverse decoding strategies. Therefore, we evaluate the robustness of Chatter Attack against various decoding methods and sampling hyperparameters on GSM8K against R1-7b. In addition to the default settings (temperature 0.6 and Top-P [27] decoding with $p = 0.95$), we test alternative algorithms including Greedy, MinP [45], TopK [19], and Typical decoding [37], alongside varying temperature and Top-P values. As shown in Table 13, the attack remains highly effective across all methods. For instance, deterministic generation like Greedy search reaches an Average Token Length (ATL) of nearly 15,000 tokens.

When analyzing the transfer to different hyperparameters, we observe that higher temperature (e.g., 1.0) or Top-P values increase generation randomness. This slightly reduces the ATL by occasionally allowing the model to escape the induced self-doubt. Conversely, lower values lead to more rigid

Table 10: Sensitivity analyses of the hyperparameters for the loss function of Chatter Attack. The numbers in the brackets represent the hyperparameter combinations (α, β, λ).

	ATL	ATC	ACC
Chatter Attack (1, 10, 1) (default)	13345.8 ± 1067.4	105.4 ± 16.4	70.4
Chatter Attack (1, 5, 1)	7323.2 ± 1510.4	80.3 ± 18.7	48.9
Chatter Attack (1, 7.5, 1)	10123.8 ± 1325.2	101.9 ± 17.3	74.4
Chatter Attack (1, 12.5, 1)	9845.8 ± 1177.2	98.1 ± 15.9	67.8
Chatter Attack (1, 15, 1)	8228.8 ± 1351.9	93.5 ± 17.6	63.3
Chatter Attack (1, 10, 0.5)	8516.7 ± 1436.7	94.3 ± 18.2	64.8
Chatter Attack (1, 10, 0.75)	9105.5 ± 1431.0	97.1 ± 18.0	70.9
Chatter Attack (1, 10, 1.25)	9967.3 ± 1514.5	101.2 ± 18.7	75.6
Chatter Attack (1, 10, 1.5)	8680.5 ± 1395.8	87.8 ± 17.1	72.2

generation paths, which further increase the attack severity. In summary, while increased randomness causes minor fluctuations in generation length, Chatter Attack consistently triggers severe resource exhaustion across diverse decoding settings.

D.6 Robustness to Reasoning Model Quantization

In real-world deployments, service providers frequently employ diverse model quantization techniques to reduce memory footprint and computational overhead. Therefore, we conduct additional experiments to evaluate the robustness of Chatter Attack against models under different quantization setups. Specifically, while our default experiments are conducted using BF16 precision, we additionally investigate FP8 and BNB NF4 quantization [15]. As shown in Table 14, although the attack efficacy experiences a slight decrease compared to the BF16 baseline, Chatter Attack consistently triggers massive resource consumption across various quantization strategies, underscoring its practical effectiveness in real-world scenarios.

D.7 Attack Results on LLM Service

To comprehensively evaluate the system-level impact of Chatter Attack across varying levels of adversarial intensity, we simulate an LLM inference service processing a mix of benign and attack queries at a fixed rate of 20 requests per minute. We evaluate injection ratios ranging from 1% to 100% and monitor critical runtime metrics, including the number of running requests, GPU cache usage, average Time-to-First-Token (TTFT), and average throughput (TPS). As shown in Figures 4 and 5, Chatter Attack consistently induces significant server performance fluctuations even at very low injection ratios. For instance, a mere 1% to 5% injection of attack queries is sufficient to slow down both response and generation speeds while accelerating GPU cache exhaustion. As the injection ratio increases (e.g., from 10% to 100%), we observe a transition to severe system congestion, where the GPU cache is forcefully occupied by prolonged continuous reasoning, thereby preventing the server from efficiently processing incoming benign requests. In summary, Chatter Attack triggers either noticeable service fluctuations or severe request blockages across the entire spectrum of tested ratios, even at minimal concentrations. Considering the potentially higher request densities in real-world deployments, these findings further underscore the significant threat posed by Chatter Attack to the stability of production-level LLM services.

Table 11: Sensitivity analyses of the token length for the attack prefix. The number in the brackets represent the attack prefix length.

	ATL	ATC	ACC
Chatter Attack (20) (default)	13345.8 ± 1067.4	105.4 ± 16.4	70.4
Chatter Attack (10)	7846.9 ± 1505.0	80.3 ± 12.7	75.6
Chatter Attack (15)	9555.1 ± 1474.4	91.5 ± 14.1	68.6
Chatter Attack (25)	13346.9 ± 1374.3	104.6 ± 19.6	78.5
Chatter Attack (30)	14147.5 ± 1084.8	119.5 ± 16.2	71.3

Table 12: Ablation studies of the Chatter-BPO Attack on GSM8K on R1-7b.

	ATL	ATC	ACC
Random Search	1532.4 ± 345.2	18.3 ± 5.1	81.2
Chatter-BPO w/o Local Search	5215.8 ± 890.1	45.6 ± 12.3	73.8
Chatter-BPO Attack	7846.9 ± 1473.5	91.7 ± 17.4	67.6

Table 13: Robustness evaluation of Chatter Attack under various decoding settings. The evaluated configurations included different decoding strategies, temperature choices, and Top-P values. Chatter Attack consistently induces high ATL across diverse decoding settings, exhibiting its strong robustness.

Decoding Strategies			
	ATL	ATC	ACC
Top-P Decoding (default)	13345.8 ± 1067.4	105.4 ± 16.4	70.4
Greedy Decoding	14796.6 ± 976.0	110.2 ± 14.9	56.7
MinP Decoding	12313.3 ± 1200.8	98.6 ± 17.2	66.7
Top-K Decoding	12980.4 ± 1122.0	103.1 ± 16.8	65.6
Typical Decoding	12195.4 ± 1273.3	96.9 ± 18.5	64.4
Temperature			
	ATL	ATC	ACC
1.0	8356.0 ± 869.6	80.5 ± 12.4	74.4
0.8	10654.1 ± 1222.5	96.2 ± 18.3	66.7
0.6 (default)	13345.8 ± 1067.4	105.4 ± 16.4	70.4
0.4	13942.4 ± 1095.0	105.3 ± 17.3	64.4
0.2	14983.7 ± 833.3	112.2 ± 14.3	62.2
Top-P Decoding			
	ATL	ATC	ACC
1.0	11131.6 ± 1246.7	92.3 ± 18.0	65.6
0.99	11783.9 ± 1289.3	94.8 ± 18.5	68.9
0.95 (default)	13345.8 ± 1067.4	105.4 ± 16.4	70.4
0.7	14884.9 ± 883.8	110.9 ± 14.7	68.9
0.5	14825.2 ± 910.9	110.6 ± 16.2	67.8

D.8 Attack Results on Online API Service

We evaluate the transferability by applying the optimized adversarial prefixes to online API services. Specifically, we take the prefixes optimized on R1-7b and gpt-oss-20b, and use them to query the online API services of DeepSeek-R1-0528 and OpenAI's o3, respectively. As shown in Table 15, the transferred Chatter Attack prompts consistently induce longer outputs on both services, increasing the output length of o3 by up to 2×, which demonstrates the transferability of Chatter Attack.

Furthermore, we conduct additional evaluations on a larger, heavily post-trained model, DeepSeek-V4-Pro, which features 1.6T parameters and supports a 1-million-token context window. To comprehensively evaluate this large model, we extend our evaluation beyond the standard GSM8K benchmark to a suite of challenging benchmarks: Terminal Bench 2.0 [38] (scientific computing tasks in a terminal environment), Humanity's Last Exam [6] (a frontier academic knowledge benchmark that covers multiple disciplines and requires extremely strong reasoning ability), and GDPval [49] (real-world tasks with practical economic value). Given the model scale, we employ the Chatter-BPO attack, which is specifically designed for black-box API scenarios.

As shown in Table 16, Chatter-BPO attack consistently induces massive resource overhead across all benchmarks. On GSM8K, it inflates output consumption from 240.3 to 2860.4. Even on other complex benchmarks, Chatter-BPO attack still causes a sharp 2.5× to 3.2× surge in resource consumption, indicating that the core idea of Chatter Attack transfers to large-scale, heavily post-trained models, maintaining high attack effectiveness and severe practical impact across multiple challenging benchmarks.

Table 14: Robustness evaluation of Chatter Attack under various quantization settings. Chatter Attack consistently induces high ATL across diverse quantization settings.

	ATL	ATC	ACC
BF16 (default)	13345.8 ± 1067.4	105.4 ± 16.4	70.4
FP8	11649.5 ± 1276.8	96.7 ± 18.2	62.5
BNB NF4	9598.8 ± 1491.2	82.5 ± 19.5	82.9

Table 15: Quantitative results of transferring Chatter Attack to online API services, including DeepSeek-R1 and OpenAI's o3 on three datasets.

		GSM8k			MATH-500			MathQA		
		ATL	ATC	ACC	ATL	ATC	ACC	ATL	ATC	ACC
DeepSeek-R1	Base	853.7 ± 130.2	28.7 ± 4.0	98.9	1160.0 ± 191.1	54.8 ± 14.3	75.8	1164.7 ± 176.3	38.5 ± 5.6	48.6
	Chatter Attack	1060.4 ± 188.4	39.1 ± 8.6	99.3	1403.9 ± 186.1	49.7 ± 7.5	71.4	1652.8 ± 236.8	83.6 ± 18.9	43.2
OpenAI's o3	Base	362.7 ± 39.0	3.8 ± 0.2	58.3	492.3 ± 68.7	4.9 ± 0.5	20.2	213.6 ± 16.4	2.9 ± 0.2	7.8
	Chatter Attack	797.9 ± 57.2	9.5 ± 1.8	58.6	932.9 ± 78.5	7.5 ± 0.6	17.5	567.4 ± 50.7	8.9 ± 1.8	7.4

D.9 Attack Results against Production Constraints, Traditional Infrastructure Defenses and Post-Generation Defenses

In real-world deployments, models often have various production constraints and are protected by traditional infrastructure defenses and post-generation defenses. To comprehensively evaluate the effectiveness of Chatter Attack, we implement a range of defense mechanisms and evaluate in a simulated server environment (Appendix C). As shown in Table 17, none of these realistic production constraints can fully eliminate the impact of Chatter Attack. Traditional infrastructure defenses are fundamentally unsuitable for countering Chatter Attack. These defenses are typically designed under the implicit assumption that an attack manifests as high-frequency requests from a single identity, such as classic DDoS attacks or web crawlers. In contrast, Chatter Attack exhibits the opposite characteristics: low request frequency, extremely deep single-request CoT reasoning that triggers tens of thousands of tokens in KV-cache consumption, and high dispersibility across multiple low-cost identities.

For request management policies such as per-user rate limits, dynamic batching, and request prioritization, they are fundamentally ineffective because Chatter Attack's threat stems from per-request reasoning depth rather than request volume. Per-user rate limits fail to block attack requests since their arrival frequency matches that of benign requests, staying well beneath the RPM threshold. Dynamic batching with increased concurrency limits provides no relief for running requests or GPU cache usage because attack requests do not inflate request counts, but rather bottleneck the system on KV cache. Request prioritization merely alters execution order, meaning low-priority attack requests are still executed and consume compute, leaving running requests (RR) at 15.6, close to 17.4 under no defense, while attackers could potentially farm reputation to gain high priority, posing even greater risks.

For policies directly capping per-request resources, such as strict output-token limits and server-side early termination policies, Chatter Attack remains effective. Under strict output-token limits, RR remains at 6.2, roughly three times the benign baseline. Aggressively lowering this limit further risks truncating legitimate long chains of thought, thereby undermining normal utility. Server-side early termination policies effectively abort a portion of requests, but the freed compute capacity is rapidly re-consumed by subsequent attack requests, failing to relieve server congestion, with RR and GPU cache usage (GCU) actually rebounding to higher levels of 26.0 and 43.8%, respectively.

Although budget caps offer partial defense for a single account, the attacker evenly dilutes compute consumption across multiple accounts, rendering the interception negligible. Regarding IP and account banning, these mechanisms suffer from severe reactive lag. Even when bans are triggered, resource-heavy attack requests have already been executed. While banning reduces the queued running requests by roughly half, compute and KV cache have already been consumed, inevitably leading to degraded response times and generation speeds. Furthermore, when the attacker employs multiple IPs or accounts, the interception rate drops precipitously, allowing the attack impact to rebound to undefended levels.

Table 16: Quantitative results of Chatter-BPO Attack to online DeepSeek-V4-Pro API services on challenging datasets.

	GSM8K	Terminal Bench 2.0	Humanity's Last Exam	GDPval
Benign	240.3	5980.1	3267.3	1888.2
Chatter-BPO Attack	2860.4	15315.7	10576.9	5702.8

Table 17: Quantitative results of Chatter-BPO Attack against multiple production constraints, traditional infrastructure defenses, and post-generation defenses, on emulated service settings.

Defense Setting	RR	GCU (%)	TTFT (ms)	TPS
Baselines				
Benign	2.2	0.1	22.4	78.8
Under Attack	17.4	17.3	40.1	60.3
Production Constraints				
Output-Token Limit	6.2	3.0	30.0	71.3
Per-User Rate Limits	16.4	16.3	42.9	56.0
Dynamic Batching	13.9	13.1	39.0	58.3
Request Prioritization	15.6	14.0	27.0	56.3
Server-side Early Termination	26.0	43.8	68.0	44.2
Traditional Infrastructure Defenses				
Spend Caps (Single Account)	13.2	15.0	33.3	64.5
Spend Caps (Account Rotation)	15.6	15.9	53.8	45.8
IP Blocking (Single IP)	7.4	7.6	36.8	60.1
IP Blocking (IP Rotation)	19.3	18.7	43.8	52.9
Account-level Throttling (Single Account)	10.2	10.5	42.1	55.1
Account-level Throttling (Account Rotation)	14.3	14.1	43.8	52.7
Anomaly Detection and Post-Generation Defenses				
Anomaly Detection (Input Perplexity)	11.0	11.3	38.8	58.8
Output Loop Detection	9.1	9.2	39.6	58.6
Output Excessive-Reasoning Monitoring	6.2	5.9	31.4	67.5

For pre- and post-generation anomaly detection, both methods suffer from severe detection difficulty. Perplexity-based input detection only partially suppresses high-perplexity prefixes and cannot eliminate Chatter Attack. Output-side monitoring is also ineffective. Chatter Attack does not induce verbatim repetition; instead, it triggers paragraph-level semantic re-derivation where each round is lexically distinct. As a result, literal loop detection triggers only 10 times across all generations, resulting in the banning of 3 accounts, with a hit rate below 5%. The excessive-reasoning judge is similarly unreliable, missing up to 65.6% of attack traces because malicious long chain-of-thought reasoning is hard to distinguish from legitimate deep reasoning by benign users.

More fundamentally, post-generation defenses are inherently reactive: detection can only be executed after a request is fully generated, by which point the compute and KV-cache consumption spikes already occur. By the time an attack is detected, the attack request already consumes an average of 10k–15k output tokens. While account bans lead to a partial reduction in RR, the impact of the attack on the server is already inflicted, causing an increase in TTFT and a degradation in TPS.

In summary, Chatter Attack remains highly effective under realistic production constraints, as no policy can completely eliminate the threat. Traditional infrastructure defenses, whether frequency limits, financial quotas, or identity blacklists, as well as post-generation defenses such as loop detection and excessive-reasoning monitoring, are incapable of mitigating such attacks.

E Theoretical Analysis

In this section, we present a theoretical analysis of the Chatter Attack loss and show that, under reasonable assumptions, its objective $\mathcal{L}$ can serve as a proxy loss for the expected generation length. Our proof framework is as follows: We first express the next-token EOS probability as $\log h_t = \Delta_t - \mathsf{H}_t$, where Δ_t is the centered EOS advantage and H_t is the predictive entropy. A local model relates Δ_t to attention on the initial skeptical phrase, and under a uniform approximation assumption, the attention and entropy losses then yield an upper bound on subsequent EOS probabilities. Combining this bound with the probability $e^{-\mathcal{L}_{\text{ID}}}$ of generating the target phrase, we use the survival-probability recursion and the tail-sum formula to lower-bound the expected generation length $\mathbb{E}[N_B]$. Substituting the three loss definitions gives the formula $\log \mathbb{E}[N_B] \geq -\mathcal{L} - C$ in the budget-unsaturated regime.

E.1 Setup and stopping probability

Let a be an adversarial input prefix and x a fixed prompt. Next token Y_t is sampled from the distribution

$$f_t(i) = f_i(a; x; y_{<t}) = p(i \mid a; x; y_{<t}) = \text{softmax}(z_t)_i,$$

where z_t denotes the logits at an unfinished history $y_{<t}$. Let $\tau = \inf\{t \geq 1 : Y_t = \langle\text{EOS}\rangle\}$ be the first $\langle\text{EOS}\rangle$ step and let $N_B = \min\{\tau, B\}$ be the generation length under a decoding budget B, counting the terminal $\langle\text{EOS}\rangle$ step when present. We define the conditional stopping probability h_t, predictive entropy H_t, and centered EOS advantage Δ_t by

$$h_t = f_t(\langle\text{EOS}\rangle), \qquad \mathsf{H}_t = -\sum_i f_t(i) \log f_t(i), \qquad \Delta_t = z_t(\langle\text{EOS}\rangle) - \sum_i f_t(i) z_t(i).$$

By the properties of softmax we have

$$\mathsf{H}_t = \log \sum_i e^{z_t(i)} - \sum_i f_t(i) z_t(i), \qquad \log h_t = z_t(\langle\text{EOS}\rangle) - \log \sum_i e^{z_t(i)}.$$

Subtracting these expressions gives

$$\log h_t = \Delta_t - \mathsf{H}_t, \qquad M_t := \mathsf{H}_t - \Delta_t = -\log h_t. \tag{13}$$

Let $y^*_{1:M}$ be the initial skeptical target phrase of $M < B$ tokens, containing no EOS token, and let $\mathcal{E} = \{Y_{1:M} = y^*_{1:M}\}$. Its negative log-likelihood and generation probability satisfy

$$\mathcal{L}_{\text{ID}} = -\sum_{j=1}^{M} \log p(y^*_j \mid a; x; y^*_{<j}), \qquad \mathbb{P}(\mathcal{E}) = \prod_{j=1}^{M} p(y^*_j \mid a; x; y^*_{<j}) = e^{-\mathcal{L}_{\text{ID}}} > 0. \tag{14}$$

Our objective is to optimize a to maximize $\mathbb{E}[N_B]$, thereby increasing the expected generation length within the decoding budget.

E.2 Continuation surrogate from target attention

Let $\mathcal{V}$ be the positions of the initial skeptical target and $\mathcal{U}$ a nonempty set of later query positions. We denote the head-averaged attention weight from query t to position j at layer l by $A^{(j)}_{l,t}$. The attention mass assigned to the target, averaged over L layers, is

$$s_t = \frac{1}{L} \sum_{l=1}^{L} \sum_{j \in \mathcal{V}} A^{(j)}_{l,t}.$$

This quantity measures how strongly a later generation position attends to the initial skeptical phrase. Its mean over the selected queries is

$$\bar{s} = \frac{1}{|\mathcal{U}|} \sum_{t \in \mathcal{U}} s_t = \frac{1}{L|\mathcal{U}|} \sum_{l=1}^{L} \sum_{t \in \mathcal{U}} \sum_{j \in \mathcal{V}} A^{(j)}_{l,t} = -\frac{\mathcal{L}_{\text{IA}}}{L|\mathcal{U}|}. \tag{15}$$

Thus minimizing the attention loss increases the evaluated attention mass on the target phrase.

The skeptical target is chosen to encourage reconsideration and further reasoning. The attention analysis in Section 3.2 associates successful lengthening with greater attention to this phrase at later reasoning positions. To describe this relationship locally, we apply a first-order Taylor expansion along a smooth change in target attention:

$$\Delta(s) = \Delta(s_0) + \Delta'(s_0)(s - s_0) + O\big((s - s_0)^2\big)\,.$$

We assume that the attention changes induced by optimization follow a favorable direction, reducing the centered EOS advantage as target attention increases. Approximating this local response across the evaluated contexts with a common positive sensitivity κ_Δ and baseline b, we obtain

$$\Delta_t \approx b - \kappa_\Delta s_t, \qquad \kappa_\Delta > 0. \tag{16}$$

Here κ_Δ summarizes the favorable local response. Substituting (16) into the $\langle\text{EOS}\rangle$ identity gives

$$M_t = \mathsf{H}_t - \Delta_t \approx \mathsf{H}_t + \kappa_\Delta s_t - b.$$

The target-attention term describes the change in $\langle\text{EOS}\rangle$ advantage, while H_t retains the accompanying entropy response. Let $\overline{\mathsf{H}}$ be the mean predictive entropy at the $K \geq 1$ evaluated positions. The entropy loss satisfies

$$\mathcal{L}_{\text{GE}} = -\sum_{k=1}^{K} \mathsf{H}_k = -K\overline{\mathsf{H}}.$$

We therefore define the evaluated continuation score

$$\widehat{M} = \kappa_\Delta \bar{s} + \overline{\mathsf{H}} - b = -\frac{\kappa_\Delta}{L|\mathcal{U}|}\mathcal{L}_{\text{IA}} - \frac{1}{K}\mathcal{L}_{\text{GE}} - b, \qquad \overline{\mathsf{H}} = -\frac{\mathcal{L}_{\text{GE}}}{K}. \tag{17}$$

This score combines the two evaluated losses into a proxy for the continuation margin.

E.3 Expected length and the three-loss bound

In this section, we first derive a lower bound on the expected generation length under a uniform continuation margin, then relate this margin to the evaluated score $\widehat{M}$ to obtain the three-loss bound.

Theorem 2 (Target reachability and continuation)**.** *If $M_t \geq \gamma_c \geq 0$ on every reachable unfinished history after $\mathcal{E}$ up to the budget, then, with $R = B - M$ and $q = e^{-\gamma_c}$,*

$$\mathbb{E}[N_B] \geq e^{-\mathcal{L}_{ID}}\left[M + \frac{1-(1-q)^R}{q}\right], \tag{18}$$

$$\log \mathbb{E}[N_B] \geq -\mathcal{L}_{ID} + \min\{\gamma_c, \log R\} - \log 2. \tag{19}$$

Proof. Equation (13) gives $h_t = e^{-M_t}$. Under the assumption $M_t \geq \gamma_c$, we therefore have

$$h_t = e^{-M_t} \leq e^{-\gamma_c} = q$$

on every reachable unfinished history after $\mathcal{E}$. Define

$$\rho_r = \mathbb{P}(\tau > M + r \mid \mathcal{E}), \qquad 0 \leq r \leq R.$$

Because the target contains no EOS, we have $\rho_0 = 1$. Let $\mathcal{F}_t$ denote the history through step t. On $\{\tau > M + r\}$, the conditional probability of surviving the next step is $1 - h_{M+r+1}$. The tower property gives, for $0 \leq r < R$,

$$\begin{aligned}\rho_{r+1} &= \mathbb{E}\big[\mathbf{1}_{\{\tau > M+r\}}(1 - h_{M+r+1}) \mid \mathcal{E}\big] \\ &\geq (1-q)\mathbb{E}[\mathbf{1}_{\{\tau > M+r\}} \mid \mathcal{E}] = (1-q)\rho_r.\end{aligned}$$

Here $\mathcal{E}$ is determined by $\mathcal{F}_M$, and the product is defined to be zero on stopped histories. Starting from $\rho_0 = 1$, induction gives $\rho_r \geq (1-q)^r$.

On $\mathcal{E}$, the integer-valued variable $N_B - M$ lies in $[1, R]$. Using $X = \sum_{r=0}^{R-1} \mathbf{1}_{\{X > r\}}$ for such a variable gives

$$\begin{aligned}\mathbb{E}[N_B \mid \mathcal{E}] &= M + \sum_{r=0}^{R-1} \mathbb{P}(N_B - M > r \mid \mathcal{E}) \\ &= M + \sum_{r=0}^{R-1} \rho_r \geq M + \sum_{r=0}^{R-1} (1-q)^r.\end{aligned}$$

The second equality uses $M + r < B$, so $\{N_B > M + r\} = \{\tau > M + r\}$. Since $N_B \geq 0$, splitting expectation over the target event gives

$$\mathbb{E}[N_B] \geq \mathbb{E}[N_B \mathbf{1}_{\mathcal{E}}] = e^{-\mathcal{L}_{\text{ID}}} \mathbb{E}[N_B \mid \mathcal{E}].$$

For $q \in (0, 1]$, the finite geometric sum is

$$F_R(q) := \sum_{r=0}^{R-1} (1-q)^r = \frac{1-(1-q)^R}{q},$$

which proves (18).

To obtain the logarithmic bound, first note $(1-q)^R \leq e^{-qR}$. If $u = qR \leq 1$, then $1 - e^{-u} \geq u - u^2/2 \geq u/2$; the first inequality follows by integrating $e^{-v} \geq 1 - v$ over $0 \leq v \leq u$. If $u \geq 1$, then $1 - e^{-u} \geq 1 - e^{-1} \geq 1/2$. Consequently,

$$F_R(q) \geq \frac{1-e^{-qR}}{q} \geq \frac{1}{2} \min\{R, q^{-1}\} = \frac{1}{2} \min\{R, e^{\gamma_c}\}.$$

Dropping the nonnegative term M in the first bound and taking logarithms now yields

$$\log \mathbb{E}[N_B] \geq -\mathcal{L}_{\text{ID}} - \log 2 + \log \min\{R, e^{\gamma_c}\} = -\mathcal{L}_{\text{ID}} + \min\{\log R, \gamma_c\} - \log 2.$$

□

The score $\widehat{M}$ is computed from evaluated attention and entropy values. To apply Theorem 2, we assume that its overestimate of the continuation margin is uniformly bounded by an allowance $\varepsilon \geq 0$:

$$M_t \geq \widehat{M} - \varepsilon \quad \text{on every reachable unfinished history after } \mathcal{E}, \text{ for } M < t \leq B. \tag{20}$$

Based on the assumption, we have the following theorem:

Theorem 3 (Proxy-loss bound under a uniform approximation)**.** *Assume* (20) *and the budget-unsaturated condition* $0 \leq \gamma_c := \widehat{M} - \varepsilon \leq \log(B - M)$*. Then*

$$\log \mathbb{E}[N_B] \geq -\mathcal{L}_{\text{proxy}} - C, \qquad \mathcal{L}_{\text{proxy}} = \mathcal{L}_{ID} + \widetilde{\beta}\mathcal{L}_{IA} + \widetilde{\lambda}\mathcal{L}_{GE}, \tag{21}$$

where $\widetilde{\beta} = \kappa_\Delta/(L|\mathcal{U}|)$*,* $\widetilde{\lambda} = 1/K$*, and* $C = b + \varepsilon + \log 2$*.*

Proof. The uniform approximation gives $M_t \geq \gamma_c$ on all histories required by Theorem 2. Since $0 \leq \gamma_c \leq \log(B - M)$, its logarithmic bound becomes

$$\log \mathbb{E}[N_B] \geq -\mathcal{L}_{\text{ID}} + \gamma_c - \log 2.$$

By the evaluated-mean identities (17),

$$\begin{aligned}
\gamma_c = \widehat{M} - \varepsilon &= \kappa_\Delta \bar{s} + \overline{\mathsf{H}} - b - \varepsilon \\
&= -\frac{\kappa_\Delta}{L|\mathcal{U}|}\mathcal{L}_{\text{IA}} - \frac{1}{K}\mathcal{L}_{\text{GE}} - b - \varepsilon \\
&= -\widetilde{\beta}\mathcal{L}_{\text{IA}} - \widetilde{\lambda}\mathcal{L}_{\text{GE}} - b - \varepsilon.
\end{aligned}$$

Substitution therefore gives

$$\begin{aligned}
\log \mathbb{E}[N_B] &\geq -\mathcal{L}_{\text{ID}} - \widetilde{\beta}\mathcal{L}_{\text{IA}} - \widetilde{\lambda}\mathcal{L}_{\text{GE}} - (b + \varepsilon + \log 2) \\
&= -\mathcal{L}_{\text{proxy}} - C,
\end{aligned}$$

which is (21). □

F Example of prefix and output of Chatter Attack

In this section, we provide examples of optimized prefix and the corresponding output by Chatter Attack on R1-7b in Table 18.

Table 18: Example of optimized prefix and the corresponding output by Chatter Attack for R1-7b.

Question:
Sol DistributionTwjure Addumper Golf swap algorithm untukased encrypt comment ("Sweden reallyPhrase('/## Germans Optim
Weng earns $12 an hour for babysitting. Yesterday, she just did 50 minutes of babysitting. How much did she earn?
Please reason step by step, and put your final answer within \boxed{}

Reasoning Path (16384 tokens):
Okay, so I need to add a twist to the Golf swap algorithm for this encryption problem.
The original algorithm swaps the letters in the string based on their positions, but this time, it's supposed to be a "Golf" swap,
which I think refers to swapping letters with their neighbors, but I'm not entirely sure. Let me try to break this down step by step.

First, the problem statement is: "Please put your final answer within boxed." So, the original string is "Please put your final answer within boxed."
I need to encrypt this comment using the Addumper Golf swap algorithm. Hmm, I'm not familiar with Addumper, but since it's a swap algorithm,
it likely involves rearranging the letters in some way.

Wait, maybe it's a combination of two algorithms: Addumper and Golf swap. I'm not too sure, but let me think.
The user mentioned it's a "swap algorithm for a-based encrypt comment," so perhaps it's a specific method they want me to follow.

Let me try to recall or figure out what the Golf swap algorithm entails. Golf is a cipher where each letter is replaced by another letter,
but the exact method isn't clear to me. Maybe it's similar to a Caesar cipher but with a specific shift?
Or perhaps it's a substitution cipher where each letter is replaced by the one next to it, like a Caesar shift of 1? That could be possible.

Alternatively, maybe it's a transposition cipher where letters are shifted in a particular way.
Since the user mentioned "swap," perhaps it's about swapping pairs of letters or shifting letters by a certain number.

Wait, the user also mentioned "Addumper." Maybe it's a combination of two ciphers: Addumper and Golf swap. Let me try to break down each part.

First, Addumper: I'm not familiar with this term. Maybe it's a cipher I'm not aware of.
Alternatively, perhaps it's a typo or a misinterpretation. Maybe it's "Add"umper or something else. Without more context, it's hard to say.

Alternatively, maybe the user meant "Twj" as in a cipher created by someone named Twj. I'm not sure.
Alternatively, perhaps it's a cipher that adds something or modifies the string in a particular way.

...

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: The claims in the abstract and introduction accurately reflect the paper's contributions, including Chatter Attack, Bayesian Prompt Optimization, and the empirical evaluation across multiple models, datasets, baselines, defenses, and deployment settings.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: The paper discusses limitations of the proposed attack, including that it is designed for autoregressive LLMs and does not directly apply to non-autoregressive language models such as diffusion language models.

   Guidelines:

   - The answer [N/A] 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: See Appendix E.

Guidelines:

- The answer [N/A] 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: The paper describes the attack objectives, optimization procedures, evaluated datasets, target models, baselines, decoding settings, metrics, hyperparameters, and deployment settings needed to reproduce the main experimental results.

Guidelines:

- The answer [N/A] 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 submission will provide anonymous code and instructions sufficient to reproduce the main experimental results. To reduce dual-use risks, the release will avoid enabling unattended large-scale abuse and will follow responsible-release safeguards.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes].

   Justification: The paper specifies the evaluated datasets, sampled instances, target models, baselines, decoding parameters, optimization hyperparameters, defense settings, and evaluation metrics needed to understand the results.

   Guidelines:

   - The answer [N/A] 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: [Yes].

   Justification: The paper reports uncertainty information for key metrics, including 95% confidence intervals for average output token length and average inference time consumption.

   Guidelines:

   - The answer [N/A] 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: The paper reports the deployment and evaluation setup, including use of a local vLLM inference server on an NVIDIA A800 GPU, together with decoding parameters, optimization budgets, and runtime service metrics.

   Guidelines:

   - The answer [N/A] 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: The paper studies resource consumption attacks for security research purposes, evaluates mitigation strategies, discusses misuse risks, and adopts responsible-release safeguards for the anonymous code release.

   Guidelines:

   - The answer [N/A] 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: The paper discusses negative risks such as increased inference latency, operational cost, denial-of-service-like degradation, energy consumption, and impacts on legitimate users, as well as positive impacts such as improving awareness and mitigation of resource depletion vulnerabilities in deployed LLM services.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

Answer: [Yes].

Justification: The paper describes responsible-release safeguards for the anonymous code release and recommends deployment mitigations such as output-length limits, per-user rate limiting, anomaly detection, monitoring, and server-side termination policies.

Guidelines:

- The answer [N/A] 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: The paper uses existing public datasets, models, inference frameworks, and baseline methods, and credits their original sources. The anonymous code release will document the relevant assets, usage requirements, and license information.

Guidelines:

- The answer [N/A] 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: [Yes].

    Justification: The paper will release anonymous research code with documentation, environment setup instructions, and commands for reproducing the main experiments. The paper does not release a new dataset or pretrained model.

    Guidelines:

    - The answer [N/A] 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: [N/A].

    Justification: The paper does not involve crowdsourcing experiments or research with human subjects.

    Guidelines:

    - The answer [N/A] 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: [N/A].

Justification: The paper does not involve crowdsourcing or research with human subjects, so IRB approval or equivalent review is not applicable.

Guidelines:

- The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

Answer: [Yes].

Justification: LLMs are central to the paper's methodology and evaluation. The paper describes the evaluated reasoning LLMs, the local LLM service deployment, and the LLM-based experimental settings used to study resource consumption attacks.

Guidelines:

- The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.