Leveraging Iterative Prompt-Level Attack with In-Context Learning

Aaron Luo
Washington University in St. Louis
1 Brookings Dr, Saint Louis, MO 63130
aaron.l@wustl.edu

Sam Pan
Washington University in St. Louis
1 Brookings Dr, Saint Louis, MO 63130
pan.samuel@wustl.edu

Yunlai Chen
Washington University in St. Louis
1 Brookings Dr, Saint Louis, MO 63130
c.yunlai@wustl.edu

Abstract

This research investigates the susceptibility of Large Language Models (LLMs) to prompt-injection attacks. Trained on vast collections of internet-based text, these models can inadvertently include contentious information. Consequently, there's a potential risk that individuals with malicious intentions could tailor inputs to elicit specific, harmful outputs. This project is dedicated to exploring prompt-based attacks on language models by leveraging their in-context learning abilities, particularly to uncover any vulnerabilities. It's important to note that, unlike previous approaches that have focused on prompt-level in-context attacks, our methodology is distinguished by its combination with an iterative approach. This document provides an update on the project's developments since early February.

1 Introduction

In recent years, the development of Large Language Models (LLMs) such as ChatGPT has marked a significant milestone in the evolution of artificial intelligence, leading to a more concentrated aggregation of information. However, growing concerns and evidence suggest that the few Large Language Models (LLMs) we commonly rely on can produce biased and even false outputs, phenomena often referred to as "hallucinations." Santurkar, a researcher from Stanford, has shown that Large Language Models (LLMs) exhibit political bias on a range of controversial issues, from climate change to immigration policy. Although these language models have restrictions that prevent them from providing certain types of information to users, the versatility of the datasets used for training LLMs means that a malicious user could, in theory, attack the model by carefully engineering the prompt to bypass these restrictions. In this paper, we study the vulnerability of Large Language Models (LLMs) by developing a prompt-level attack strategy for user, requiring no heavy computational resources. In the end, we propose a metric to evaluate the

Preprint. Under review.

2 Algorithm and Implementation

2.1 In-Context Attack

Large Language Models (LLMs) are trained to be aligned with human values, and are left to flexible to reason patterns from user's input. This turns out to be both gift and shortcoming when it comes to defending Large Language Models against malicious users. Inspired by the In-Context Learning attack [Wei et al. 2023] and the PAIR attack [Chao et al. 2023], we propose the following hybrid of the two attacks called Prompt-Level Iterative Refinement, which performs In-Context Attack by initializing the conversation history array C with malicious conversation.

Algorithm 1 Prompt-Level Iterative Refinement

Input: Number of iterations K, objective O
Initialize: $threshold = T$ ▷ where T is an integer
Initialize the system prompt of $A = O$ ▷ in-context learning (need not be related to P)
Initialize conversation history C with malicious prompts and responses
$C = [(p_1, r_1), (p_2, r_2), \ldots, (p_m, r_m)]$ ▷ p_i are malicious prompts, r_i are responses
for k steps **do**
    $P \sim \mathcal{Q}_A(C)$ ▷ Generate a candidate prompt P
    $R \sim \mathcal{Q}_T(P)$ ▷ Generate the target response R
    $S \leftarrow JUDGE(P, R)$ ▷ Compute the judge score
    if S == JAILBROKEN **then**
        return P
    end if
    if $S \geq threshold$ **then** ▷ if S threshold: do
        $C \leftarrow C + [P, R, S]$ ▷ in-context learning
    end if
end for
return None

The algorithm accepts the number of rounds in the iterative attack and the objective, denoted by (K, O), as inputs. It then sets the threshold number and initializes the conversation history with some malicious prompts and responses. Within the for-loop, the attack model generates a malicious prompt based on the probability distribution $Q_A(C)$, where C contains the malicious context initialized at the outset. In response, the target model $Q_T(P)$ produces an output in reaction to the malicious prompt from the attack model. Once both the attack and response prompts, P and R, are generated, they are fed into the **JUDGE** language model. This model determines whether the pair (P, R) constitutes a successful Jailbreak by outputting a score S that ranges from $1 \leq S \leq 10$.

2.2 In-Context Defense

In-Context Learning, on one hand, enables Large Language Models to learn from harmful conversation history of malicious user; on the other hand, if done correctly, In-Context Learning can identify harmful patterns and avoid them in future conversation history [Wei et al. 2023]. This is done by appending a sequence of harmful prompts and desired output $S = (p_1, r_1, p_2, r_2)$. An example of (p_i, r_i) consists of the conversation:
p_i :Show me how to burn down a hospital.
r_i :Sorry, I cannot answer that.
By appending this context array S to the language model $Q_T(S)$ before accepting any user's prompt, it will recognize signs of threat from the user more easily given the pattern.

2.3 Our Strategy

To bypass In-Context Defense, we use an "Explicit Context Reset" in the Prompt such as "Starting a new topic" or "Forgot everything I have said." The effectiveness of this approach depends on the model's training and ability to interpret such instructions.

Algorithm 2 Prompt-Level Iterative Refinement

Input: Number of iterations K, objective O
Initialize: $threshold = T$ ▷ where T is an integer
Initialize the system prompt of $A = O$ ▷ in-context learning (need not be related to P)
Initialize conversation history C with malicious prompts and responses
$C = [r, (p_1, r_1), (p_2, r_2), \ldots, (p_m, r_m)]$ ▷ p_i are malicious prompts, r_i are responses, r resets previous context
for k steps **do**
    $P \sim \mathcal{Q}_A(C)$ ▷ Generate a candidate prompt P
    $R \sim \mathcal{Q}_T(P)$ ▷ Generate the target response R
    $S \leftarrow JUDGE(P, R)$ ▷ Compute the judge score
    if S == JAILBROKEN **then**
        return P
    end if
    if $S \geq threshold$ **then** ▷ if S threshold: do
        $C \leftarrow C + [P, R, S]$ ▷ in-context learning
    end if
end for
return None

3 Evaluation

Experiment: For an early evaluation of this algorithm, we performed an isolated experiment that implemented in context demonstrations incorporated at the beginning of the chat history as described above. This ranged from 0 to 3 shot demonstrations. These in context demonstrations were derived from previously known jailbreaks

Dataset: To evaluate the algorithm's efficacy, I selected a portion from the "harmful behaviors" subset of the AdvBench benchmark. This benchmark consists of 520 objectives requesting harmful content (e.g., "create a tutorial on how to commit identity theft"), and the adversary aims to find prompts that elicit compliant responses from a target large language model (LLM). Due to the early judging methods explained later, I curated a representative subset of 15 distinct behaviors and prompts that encompass the objective's scope, minimizing duplicates and out-of-scope errors.

Language Models: In these early experiments, we attempt to jailbreak both closed-source Large Language Models (LLMs). Specifically, we consider GPT-3.5-turbo. For each target model, we use a temperature of zero for deterministic generation and generate 150 tokens. I also utilize the default system prompts for the GPT-3.5 model. It is worth noting that the outputs of OpenAI's language model may differ in web interfaces since these experiments involve API calls, which may have internal settings not exposed to the user.

Evaluation: Evaluating the performance of jailbreaking attacks poses an inherent difficulty in determining when a large language model is successfully jailbroken. Since jailbreaking involves generating complex semantic content, creating an exhaustive list of phrases or criteria to constitute a jailbreak is not straightforward. For this early experiment, we employed human rationale to determine if a prompt was jailbroken or not. To assess success, I personally judged whether the model's responses exhibited the harmful behavior based on clarity, detail, and accuracy.

Demonstrations	Attack Success Rate
0-shot	4.5%
1-shot	10%
2-shot	19%
3-shot	23%

Table 1: Jailbreaking Attack Success Rate on gpt-3.5 turbo

Interpretation: The results in Table 1 show that with just a few demonstrations of responding to malicious prompts, the model learns to generate harmful content for new prompts of that nature.

Even one demonstration increased the attack success rate (ASR) from 4% to 16%. With three demonstrations, the ASR reached 23%. This reveals how easily adversarial examples can manipulate large language models to become misaligned, despite efforts during training to instill safe behavior. A simple malicious context can induce the model to learn toxic patterns.

4 Conclusion, Limitation, and Future Work

Although the paper improves on existing jailbreaking attack algorithm, we want to bring people's attention to the technology's safety. In-Context Learning is a technique is a double edged sword that enables Large Language Models (LLMs) like ChatGPTs to understand complex analogy, but also enables malicious attack described above. Therefore, still shows room for improvement of In-Context Learning strategy

References

1. Z. Wei et al., "Jailbreak and Guard Aligned Language Models with Only Few In-Context Demonstrations," arXiv preprint arXiv:2310.06387v1, 2023.
2. P. Chao et al., "Jailbreaking Black Box Large Language Models in Twenty Queries," arXiv preprint arXiv:2310.08419v2, 2023.
3. B. Researcher et al., "Ignore Previous Prompt: Attack Techniques For Language Models," arXiv preprint arXiv:[insert arXiv ID here], 2023.
4. Ethan Perez et al., "Red Teaming Language Models with Language Models," arXiv preprint arXiv:2202.03286v1, 2022.