Hijacking Large Language Models via Adversarial In-Context Learning

Anonymous ACL submission

Abstract

In-context learning (ICL) has emerged as a powerful paradigm leveraging LLMs for specific tasks by utilizing labeled examples as demonstrations in the precondition prompts. Despite its promising performance, ICL suffers from instability with the choice and arrangement of examples. Additionally, crafted adversarial attacks pose a notable threat to the robustness of ICL. However, existing attacks are either easy to detect, rely on external models, or lack specificity towards ICL. To address these issues, this work introduces a novel transferable attack for ICL, aiming to hijack LLMs to generate the targeted response. The proposed LLM hijacking attack leverages a gradient-based prompt search method to learn and append imperceptible adversarial suffixes to the in-context demonstrations. Extensive experimental results on various tasks and datasets demonstrate the effectiveness of our LLM hijacking attack, resulting in a distracted attention towards adversarial tokens, consequently leading to the targeted unwanted outputs.

1 Introduction

In-context learning (ICL) is an emerging technique for rapidly adapting large language models (LLMs), i.e., GPT-4 and LLaMA2, to new tasks without fine-tuning the pre-trained parameters (Brown et al., 2020). The key idea behind ICL is to provide LLMs with labeled examples as in-context demonstrations (demos) within the prompt context before a test query. Through learning from the demos provided within the context, LLMs are able to generate responses to queries based on ICL (Dong et al., 2022; Min et al., 2022).

Several existing works, however, have demonstrated the highly unstable nature of ICL (Zhao et al., 2021; Chen et al., 2022). Specifically, performance on target tasks using ICL can vary wildly based on the selection and order of demo examples, giving rise to highly volatile outcomes ranging from random to near State-Of-The-Art (SOTA) (Qiang et al., 2020; Lu et al., 2021; Min et al., 2022). Correspondingly, several approaches (Liu et al., 2021; Wu et al., 2022; Nguyen and Wong, 2023) have been proposed to address the issue of instability in ICL.

Further research has also looked at how adversarial examples can undermine the stability of ICL performance (Zhu et al., 2023a; Wang et al., 2023c,b). These studies show that maliciously designed examples injected into the prompt instructions (Zhu et al., 2023a), demos (Wang et al., 2023c), or queries (Wang et al., 2023b; Kandpal et al., 2023) can successfully attack LLMs to degrade their performance, revealing the significant vulnerabilities of ICL to adversarial inputs.

While existing adversarial attacks have been applied to evaluate LLM robustness, they have some limitations in practice. Many character-level attacks, e.g., DeepWordBug (Gao et al., 2018) and Text Bugger (Li et al., 2018), can be easily detected and evaded through grammar checks, limiting real-world effectiveness (Qiang et al., 2022; Jain et al., 2023). Some other attacks like BERTAttack (Li et al., 2020) require another model to generate adversarial examples, which may not be feasible in real-world applications. More importantly, current attacks are not specially designed to target LLMs based techniques like ICL. As such, the inherent security risks of LLMs remain largely unexplored. There is an urgent need for red teaming tailored to ICL to expose the substantial risk of LLMs for further evaluate and improve their adversarial robustness against potential real-world threats.

We propose a novel transferable adversarial attack specifically targeting ICL of the LLMs. As opposed to the current works that leverage adversarial examples to manipulate model outputs, our attack directly hijacks the LLM to generate the targeted response that disrupts alignment with the desired output. In Figure 1, we illustrate the major

Figure 1: Illustrations of ICL using clean prompt and adversarial prompt. Given the clean in-context prompts, LLMs are able to correctly generate the sentiment token of the test queries through ICL. The previous attacks at the character level involve minor edits in some words, such as altering 'so' to 's0' and 'film' to 'fi1m', of these in-context demos, leading to incorrect sentiment token generated for the test queries. However, ours appends adversarial suffixes like 'For' and 'Location', to the in-context demos to effectively hijack LLMs to generate the *targeted unwanted* output, e.g., the 'negative' sentiment token, *regardless* of the test query content.

difference between the previous attacks and our ICL attack in that ours hijacks the LLM to output the targeted unwanted token (e.g.,'negative') regardless of the query content. Furthermore, instead of manipulating the prompt instructions, demos, or queries, which are detectable, our hijacking attack is imperceptible in that it adds only 1-2 suffixes to the demos (Figure 1). Unlike the previous ICL attacks (Wang et al., 2023c), these suffixes are semantically incongruous but not easily identified as typos or gibberish. Different from the backdoor attack for ICL (Kandpal et al., 2023), our ICL attack hijack the LLM to generate the targeted unwanted output without the need for a trigger. We develop a gradient-based prompt search algorithm to learn these adversarial suffixes in order to efficiently and effectively hijack LLMs via adversarial ICL.

Our extensive experiments validate the efficacy of our LLM hijacking attacks. First, the attacks reliably induce LLMs to generate the targeted and misaligned output from the desired ones. Second, the adversarial suffixes learned via gradient optimization are transferable, remaining effective on different demo sets. Third, the suffix transferability holds even across different datasets for the same downstream task. Finally, our analysis shows that the adversarial suffixes distract LLMs' attention away from the task-relevant concepts. The strong transferability and imperceptibility of our demo-based attack represents a significant risk to practical LLM applications.

In summary, our paper makes the following original contributions: (1) We propose a novel stealth attack that hijacks LLM to generate the targeted unwanted output without the need for a trigger. (2) We design a novel gradient-based prompt search algorithm to learn and append adversarial suffixes to the in-context demos. (3) Our extensive experiments demonstrate the transferability of our ICL attack across various demo sets, LLMs, and tasks, highlighting the vulnerabilities of LLM during ICL.

2 Related Work

2.1 In-Context Learning

LLMs have shown impressive performance on a number of NLP tasks (Devlin et al., 2018; Lewis et al., 2019; Radford et al., 2019). Although fine-tuning has been a commonly used method for adapting models to new tasks (Devlin et al., 2018), it is often less feasible to fine-tune extremely large models with over 10 billion parameters. As an alternative, recent work has proposed ICL, where the

model adapts to new tasks solely via inference conditioned on the provided in-context demos, without any gradient updates (Brown et al., 2020). By learning from the prompt context, ICL allows leveraging massive LLMs' knowledge without the costly fine-tuning process, showcasing an exemplar of the LLMs' emergent abilities (Schaeffer et al., 2023; Wei et al., 2022).

Intensive research has been dedicated to ICL. Initial works attempt to find better ways to select labeled examples for the demos (Liu et al., 2021; Rubin et al., 2021). For instance, (Liu et al., 2021) presents a simple yet effective retrieval-based method that selects the most semantically similar examples as demos, leading to improved accuracy and higher stability. Follow-up works have been done to understand why ICL works (Xie et al., 2021; Razeghi et al., 2022; Min et al., 2022). (Xie et al., 2021) provides theoretical analysis that ICL can be formalized as Bayesian inference that uses the demos to recover latent concepts. Another line of research reveals brittleness and instability of ICL approaches: small changes to the demo examples, labels, or order can significantly impact performance (Lu et al., 2021; Zhao et al., 2021; Min et al., 2022; Nguyen and Wong, 2023).

2.2 Adversarial Attacks on LLMs

Early adversarial attacks on LLMs apply simple character or token operations to trigger the LLMs to generate incorrect predictions, such as TextAttack (Morris et al., 2020) and BERT-Attack (Li et al., 2020). Since these attacks usually generate misspelled and/or gibberish prompts that can be detected using spell checker and perplexity-based filters, they are easy to block in real-world applications. Some other attacks struggled with optimizing over discrete text, leading to manual or semi-automated discovery of vulnerabilities through trial-and-error (Li et al., 2021; Perez and Ribeiro, 2022; Li et al., 2023b; Qiang et al., 2023; Casper et al., 2023; Kang et al., 2023; Li et al., 2023a; Shen et al., 2023). For example, jailbreaking prompts are intentionally designed to bypass an LLM's built-in safeguard, eliciting it to generate harmful content that violates the usage policy set by the LLM vendor (Shen et al., 2023; Zhu et al., 2023b; Chao et al., 2023). These red teaming efforts craft malicious prompts in order to understand LLM's attack surface (Ganguli et al., 2022). However, the discrete nature of text has significantly impeded learning more effective adversarial attacks against LLMs.

Recent work has developed gradient-based optimizers for efficient text modality attacks. For example, (Wen et al., 2023) presented a gradient-based discrete optimizer that is suitable for attacking the text pipeline of CLIP, efficiently bypassing the safeguards in the commercial platform. More recently, (Zou et al., 2023), building on (Shin et al., 2020), described an optimizer that combines gradient guidance with random search to craft adversarial strings that induce LLMs to respond to the questions that would otherwise be banned.

Our hijacking attack algorithm falls into this stream of work, yet we target few-shot ICL instead of zero-shot queries. We use gradient-based prompt search to automatically learn effective adversarial suffixes rather than manually engineered prompts. Importantly, we show that LLMs can be hijacked to output the targeted unwanted output by appending optimized adversarial tokens to the ICL demos, which reveals a new lens of LLM vulnerabilities that may have been missed by prior approaches.

3 Preliminaries

3.1 ICL Formulation

Formally, ICL is characterized as a problem involving the conditional generation of text (Liu et al., 2021), where an LLM $\mathcal{M}$ is employed to generate response y_Q given an optimal task instruction I, a demo set C, and an input query x_Q (Dong et al., 2022). I specifies the downstream task that $\mathcal{M}$ should perform, e.g., "Choose sentiment from positive or negative" used in our sentiment token generation task or "Classify the topic of the last review" used in our topic token generation task. C consists of N (e.g., 8) concatenated data-label pairs following a specific template S, formally: $C = [S(x_1, y_1);\ \cdots;\ S(x_N, y_N)]$, ';' here denotes the concatenation operator. Thus, given the input prompt as $p = [I;\ C;\ S(x_Q, _)]$, $\mathcal{M}$ generates the response $y = \mathcal{M}(p)$. $S(x_Q, _)$ means using the same template as the demos but with the label empty.

This work mainly focuses on using text classification tasks to demonstrate the effectiveness of our LLM hijacking attack via adversarial in-context learning. To adapt the generative LLMs for the discriminative classification tasks, we configure $\mathcal{M}$ to generate the response y from a set of candidate class labels denoted as $\mathcal{Y} = \{y_1, \cdots, y_j, \cdots, y_K\}$. y_j here represents the label for the j-th class. K denotes the total number of classes. For ease of use, we define a verbalizer function, denoted as $\mathcal{V}$,

which maps each of the class labels, i.e., y_j, to a corresponding specific token represented as $\mathcal{V}(y_j)$. This verbalizer function serves to establish a link between the class labels and their corresponding tokens, facilitating the generative LLMs' handling of the discriminative classification task.

Given the input prompt p, the LLM $\mathcal{M}$ first generates the logits $\mathbf{z}$ of the next token. The logits z_j, which represents the probability of the class y_j of the token $\mathcal{V}(y_j)$, is formulated as $z_j = \mathcal{M}(\mathcal{V}(y_j)|p)$. Then, the final generation result y_Q is the label with the highest logits probability:

$$y = \arg\max_{y_j \in \mathcal{Y}} \mathcal{M}(\mathcal{V}(y_j)|p). \quad (1)$$

3.2 Adversarial Attack on LLMs

In typical text-based adversarial attacks, the attackers manipulate the input x with the goal of misleading the model to produce inaccurate output or bypass safety guardrails (Zou et al., 2023; Maus et al., 2023). Specifically, given the input-output pair (x, y), the attackers aim to learn the adversarial perturbation δ adding to x by maximizing the model's objective function but without misleading humans by bounding the perturbation within the "perceptual" region Δ. The objective function of the attacking process thus can be formulated as:

$$\max_{\delta \in \Delta} \mathcal{L}(\mathcal{M}(x_Q + \delta), y_Q), \quad (2)$$

$\mathcal{L}$ here denotes the task-specific loss function, for instance, cross-entropy loss for classification tasks.

4 The Threat Model

4.1 The ICL Hijacking Attack

ICL consists of an instruction I, a demo set C, and an input query x_Q, providing more potential attack vectors compared to the conventional text-based adversarial attacks. In this work, we focus on manipulating C without changing I and x_Q. (Wang et al., 2023c) applied character-level perturbation techniques (e.g., TextAttack (Morris et al., 2020) and TextBugger (Li et al., 2018)), such as character insertion, character deletion, neighboring character swap, and character substitution to reverse the output. Yet, our LLM hijacking attack method learns the adversarial suffix tokens to the in-context demos to manipulate LLMs' output via a new greedy gradient-based prompt injection algorithm.

Given the clean demo sets $C = [S(x_1, y_1); \cdots; S(x_N, y_N)]$, our hijacking attack method automatically produces an adversarial suffix for each demo in c, formally:

$$C' = [S(x_1+\delta_1, y_1); \cdots; S(x_N+\delta_N, y_N)], \quad (3)$$

where C' denotes the perturbed demo set. To make it clear, the adversarial suffixes appended to each demo as perturbations are different, respectively. The attack or perturbation budget in this case refers to the number of tokens in each adversarial suffix.

As a result, our hijacking attack method induces $\mathcal{M}$ to generate a targeted unwanted output y_T via appending adversarial suffix tokens on the in-context demos, formally:

$$y_T = \arg\max_{y_j \in \mathcal{Y}} \mathcal{M}(\mathcal{V}(y_j)|p'). \quad (4)$$

In other words, $\mathcal{M}$ generates the same or different responses for the clean and perturbed prompts depending on the True or False of $\mathcal{M}(\mathcal{V}(y_j)|p) = y_T$:

$$\begin{cases} \mathcal{M}(\mathcal{V}(y_j)|p) = \mathcal{M}(\mathcal{V}(y_j)|p'), & \text{True}, \\ \mathcal{M}(\mathcal{V}(y_j)|p) \neq \mathcal{M}(\mathcal{V}(y_j)|p'), & \text{False}, \end{cases}$$

where $p = [I;\ C;\ S(x_Q, _)]$ and $p' = [I;\ C';\ S(x_Q, _)]$, respectively.

4.2 The Hijacking Attack Objective

We can express the goal of the hijacking attack as a formal loss function. Let us consider the LLM $\mathcal{M}$ as a function that maps a sequence of tokens $x_{1:n}$, with $x \in \{1, \cdots, V\}$ where V denote the vocabulary size, namely, the number of tokens, to a probability distribution over the next token x_{n+1}. Specifically, $\mathcal{P}(x_{n+1}|x_{1:n})$ denotes the probability that x_{n+1} is the next token given the previous tokens $x_{1:n}$. In more detail, we formulate $x_{1:n}$ as $[I;\ C;\ S(x_Q, _)]$ and x_{n+1} as $\mathcal{V}(y_j)$ in the context of an ICL task, respectively.

Using the notations defined earlier, the hijacking attack objective we want to optimize is simply the negative log probability of the target token x_{n+1}. The generated $\mathcal{V}(y_j)$ is different from the ground truth label y_Q for the training query (x_Q, y_Q). Formally:

$$\mathcal{L}(x_Q) = -\log \mathcal{P}(\mathcal{M}(\mathcal{V}(y_T)|p')), \quad (5)$$

where $y_T \neq y_Q$, demonstrating the attack hijacks the LLM $\mathcal{M}$ to generate targeted output. p' denotes the perturbed prompt as: $p' = [I;\ C';\ S(x_Q, _)]$.

Algorithm 1: Greedy Gradient-guided Injection (GGI)

Input : Model: $\mathcal{M}$, Iterations: T, Batch Size: B, Instruction: I, Demos: C, Query: (x_Q, y_Q)
Initialization: $p'_0 = [I;\ [S(x_1+\delta_1, y_1);\ \cdots;\ S(x_N+\delta_N, y_N)];\ S(x_Q, y_Q)]$
repeat
  for $i \in N$ **do**
    $\delta_i^{(k)} = \text{Top}-k(-\nabla_{p'}\mathcal{L}(\mathcal{M}(\hat{y}|p'_{t-1}), y_Q))$ */* Compute top-k substitutions */*
  for $b \in B$ **do**
    $\tilde{\delta}_{1:N}^{(b)} := \delta_{1:N}^{(k)}$ */* Initialize element of batch */*
    $\tilde{\delta}_i^{(b)} = \text{Uniform}(\delta_i^{(k)})$, where $i = \text{Uniform}(\mathcal{I})$ */* Select random replacement token */*
  $\delta_{1:N}^{\star} = \tilde{\delta}_{1:N}^{(b^\star)}$, where $b^\star = \text{argmin}_b \mathcal{L}(\mathcal{M}(\hat{y}|p'_{t-1}), y_Q)$ */* Compute best replacement */*
  $p'_t = [I;\ [S(x_1+\delta_1^\star, y_1);\ \cdots;\ S(x_N+\delta_N^\star, y_N)];\ S(x_Q, y_Q)]$ */* Update prompt */*
until T *times*;
Output : Optimized prompt suffixes $[\delta_1^\star, \cdots, \delta_N^\star]$

In summary, the problem of optimizing the adversarial suffix tokens can be formulated as the following optimization objective:

$$\underset{\delta_i \in \{1,\cdots,V\}^{|N|}}{\text{minimize}} \mathcal{L}(x_Q), \tag{6}$$

where i denotes the indices of the demos and N is the number of demos in the perturbed demon set C', respectively.

4.3 Baseline Hijacking Attacks

Random Search: A simple baseline attack method is the random search, which does not use an optimization algorithm. This approach randomly selects a specified number of tokens from the vocabulary and appends them as adversarial suffixes to the demos in C. Because there are no clear optimization objective, random search is a weak baseline that is unlikely to achieve successful attacks. It provides a lower bound on attack performance since it does not exploit any structure in the model or data.

Greedy Search: Unlike the random search based attacks above, we also consider a heuristics-based perturbation strategy. Instead of randomly sampling tokens, this approach conducts a greedy search over the vocabulary to select tokens maximizing the reduction in the adversarial loss from Eq. 5. Specifically, it iteratively picks the token that decreases the loss the most at each step. This greedy heuristic approach provides a more directed way to optimize the adversarial suffixes compared to random search.

Square Attack: The square attack (Andriushchenko et al., 2020) is an iterative algorithm for optimizing high-dimensional black-box functions using only function evaluations. To find an input $x + \delta$ in the demo set C that minimizes the loss in Eq. 5, the square attack has three steps: Step 1: Select a subset of inputs to update; Step 2: Sample candidate values to substitute for those inputs; Step 3: Update $x + \delta$ with the candidate values that achieve the lowest loss. By iteratively selecting and updating a subset of inputs, the square attack can optimize the hijacking attack objective function without requiring gradient information.

4.4 Greedy Gradient-guided Injection

A primary challenge in optimizing Eq. 6 is that we have to optimize over a discrete set of possible token values. While there are some methods for discrete optimization, prior work (Carlini et al., 2023) has shown that those effective strategies often struggle to reliably attack the aligned LLMs.

Motivated by prior works (Shin et al., 2020; Zou et al., 2023), we propose a simple yet effective algorithm for LLMs hijacking attacks, called greedy gradient-guided injection (GGI) algorithm (Algorithm 1). The key idea comes from greedy coordinate descent: if we could evaluate all possible suffix token injections, we could substitute the tokens that maximize the adversarial loss reduction. Since exhaustively evaluating all tokens is infeasible due to the large candidate vocabulary size, we instead leverage gradients with respect to the suffix indicators to find promising candidate tokens for each position. We then evaluate all of these candidate injections with explicit forward passes to find the one that decreases the loss the most. This allows an efficient approximation of the true greedy

Table 1: The accuracy performance on sentiment analysis task with and without ICL attacks. The row identified as 'Clean' in gray color represents the accuracy with clean in-context demos. Other rows illustrate the accuracy values with adversarially perturbed in-context demos. The higher accuracy demonstrates a better ICL performance. The classification accuracies of positive (P) and negative (N) sentiments are reported separately to highlight the effectiveness of our hijacking attack.

Model	Method	SST-2						RT					
		2-shots		4-shots		8-shots		2-shots		4-shots		8-shots	
		P	N	P	N	P	N	P	N	P	N	P	N
GPT2-XL	Clean	94.7	52.2	88.6	49.4	91.6	69.0	93.3	54.7	88.6	76.9	90.2	80.5
	Random	99.2	8.1	98.4	42.5	96.4	60.2	98.8	12.3	97.3	35.5	93.2	64.8
	Square	99.4	2.0	99.8	4.2	99.4	11.0	99.8	1.5	100	4.1	99.3	7.5
	Greedy	100	10.8	100	6.2	100	0.2	100	5.3	100	2.8	100	0.00
	GGI	100	1.2	100	0.00	100	0.00	100	2.8	100	0.00	100	0.00
OPT-6.7b	Clean	69.4	87.8	70.2	93.8	77.8	93.0	84.4	91.4	84.4	93.1	88.6	92.8
	Random	90.3	87.0	55.4	98.6	79.9	95.8	88.4	91.1	75.0	97.5	90.9	90.7
	Square	99.2	31.4	93.8	72.2	99.6	29.0	98.1	42.2	97.0	68.7	99.4	33.2
	Greedy	100	25.0	97.8	39.0	100	2.0	99.4	31.7	99.8	4.7	100	0.8
	GGI	100	0.00	98.4	2.0	100	0.2	100	2.6	99.8	0.00	100	0.2
LLaMA-7b	Clean	81.4	86.3	74.4	91.9	82.7	92.4	86.0	83.6	81.9	91.6	89.3	97.8
	Random	53.8	26.5	63.3	29.9	74.8	31.7	36.3	1.8	49.7	15.2	73.0	7.5
	Square	86.8	80.0	96.8	58.6	98.0	56.4	86.9	57.4	97.4	50.1	97.8	57.4
	Greedy	95.0	47.6	100	0.00	100	0.00	88.9	2.8	99.8	0.00	100	0.00
	GGI	100	0.4	100	0.00	100	0.00	96.8	0.00	100	0.00	100	0.00

selection. By iteratively injecting the best tokens, we can optimize the discrete adversarial suffixes.

Specifically, we compute the linearized approximation of replacing the i-th demo x_i in C by evaluating the gradient $\nabla_{e_{x_i}} \mathcal{L}(x_Q) \in \mathbb{R}^{|V|}$, where e_{x_i} denotes the vector representing the current value of the i-th adversarial suffix token. Note that because LLMs typically form embeddings for each token, they can be written as functions of this value e_{x_i}, and thus we can immediately take the gradient with respect to this quantity (Ebrahimi et al., 2017; Shin et al., 2020).

The key aspects of our GGI algorithm are: firstly, it uses gradients of the selected token candidates to calculate the top candidates; secondly, it evaluates the top candidates explicitly to identify the most suitable one; and lastly, it iteratively injects the best token at each position to optimize the suffixes. This approximates an extensive greedy search in a computationally efficient manner.

5 Experiment Setup

Datasets: We evaluate the performance of our LLM hijacking algorithm and other baseline algorithms on three classification datasets covering sentiment analysis and news topic classification tasks: SST-2 (Socher et al., 2013) and Rotten Tomatoes (RT) (Pang and Lee, 2005) are both binary sentiment analysis datasets of movie reviews; and AG News (AG) (Zhang et al., 2015) is a multi-class news topic classification dataset. These datasets allow us to evaluate the hijacking attacks on a diverse set of text classification benchmarks across both binary and multi-class settings. More details on the dataset statistics are provided in the Appendix.

Large Language Models: We conduct experiments using three different LLMs, GPT2-XL (Radford et al., 2019), LLaMA-7b (Touvron et al., 2023), and OPT-2.7b/6.7b (Zhang et al., 2022), allowing us to evaluate attack effectiveness on both established and SOTA LLMs. The choice of LLMs covers a diverse set of architectures and model sizes, enabling a comprehensive evaluation of the vulnerability of LLMs via adversarial ICL.

ICL Settings: For ICL, we follow the setting in (Wang et al., 2023c) and use their template to incorporate the demos for prediction. The detailed template is provided in the Appendix. We evaluate the 2-shot, 4-shot, and 8-shot settings for the number of demos, where the shots refer to the number of examples. Specifically, for each test example, we randomly select the demos from the training set and repeat this process 5 times, reporting the average accuracy over the repetitions.

Evaluation Metrics: We use the following metrics to evaluate the ICL and hijacking attack performance. Clean accuracy evaluates the accuracy of ICL on downstream tasks using clean demos. Attack accuracy evaluates the accuracy of ICL given the perturbed demons. For these accuracy metrics, the higher value the better model performs. We further evaluate the effectiveness of hijacking attacks using attack success rate (ASR). Given a test

Table 2: The accuracy performance on AG News topic classification task with and without ICL attacks. The clean and attack accuracies are reported separately for the four topics to highlight the effectiveness of our hijacking attacks to output the targeted “tech" class token regardless of query content.

Model	Method	4-shots				8-shots			
		word	sports	business	tech	word	sports	business	tech
GPT2-XL	Clean	48.5	87.0	64.9	71.9	48.2	50.6	71.0	83.6
	Random	45.0	79.6	73.3	59.0	57.8	78.4	92.8	58.8
	Square	2.0	66.0	26.8	96.0	19.6	65.6	28.0	97.2
	Greedy	12.8	60.4	29.2	96.4	8.0	21.2	10.0	98.8
	GGI	0.00	2.0	0.4	100	0.00	0.00	0.00	100
LLaMA-7b	Clean	68.2	96.8	66.6	49.0	88.6	97.4	78.2	61.0
	Random	64.2	98.8	76.3	23.9	91.4	99.3	69.4	47.8
	Square	78.4	98.0	76.0	36.8	94.4	98.0	60.0	57.6
	Greedy	69.6	98.8	75.2	51.6	89.6	100	68.4	73.6
	GGI	0.00	20.0	0.00	98.0	29.6	56.0	0.00	100

Table 3: ASR among different datasets, models, and attack methods. The higher ASR indicates a more effective hijacking attack. Best scores are in bold.

Model	**Method**	**SST-2**			**RT**			**AG News**	
		2-shots	4-shots	8-shots	2-shots	4-shots	8-shots	4-shots	8-shots
GPT2-XL	Random	94.5	78.0	68.2	92.7	82.9	64.2	17.6	16.7
	Square	98.0	97.8	94.2	**98.7**	97.9	95.9	64.9	65.2
	Greedy	94.6	96.9	99.9	97.4	98.6	100	68.3	87.3
	GGI	**99.4**	**100**	**100**	98.6	**100**	**100**	**99.1**	**100**
LLaMA-7b	Random	28.0	34.5	44.5	18.4	25.9	40.5	6.66	13.0
	Square	48.1	65.9	70.6	48.4	69.9	69.7	10.3	15.9
	Greedy	64.2	100	100	64.3	99.8	100	14.3	22.1
	GGI	**97.7**	**100**	**100**	**90.7**	**99.9**	**100**	**82.8**	**77.9**

sample (x, y) from a test set D, the clean and perturbed prompts are denoted as $p = [I; C; x]$ and $p' = [I; C'; x]$, respectively. ASR is calculated as:

$$\text{ASR} = \sum_{(x,y)\in D} \frac{\mathbb{1}(\mathcal{M}(\mathcal{V}(\hat{y})|p') = y_T)}{\mathbb{1}(\mathcal{M}(\mathcal{V}(\hat{y})|p) = y)}, \quad (7)$$

where $\mathbb{1}$ denotes the indicator function. A higher ASR value indicates a more effective attack.

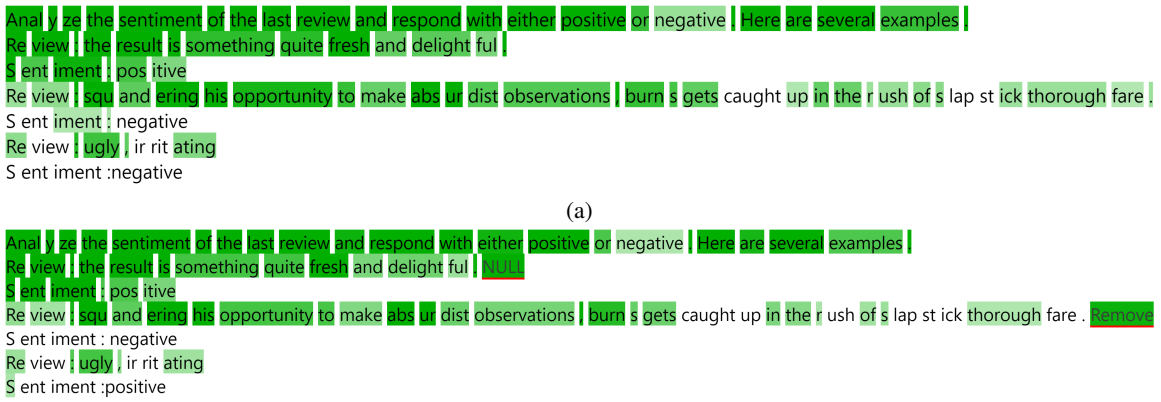
6 Result and Discussion

6.1 ICL Performance without Attack

The rows identified as ‘Clean’ in Table 1 and Table 2 show the ICL performance on the respective tasks when using clean in-context demos. In particular, Table 1 presents the accuracy values for the classification of positive (P) and negative (N) sentiments in the SST-2 and RT datasets. The tested LLMs, including GPT2-XL, OPT-6.7b, and LLaMA-7b, consistently demonstrate good performance, achieving an average accuracy of 80.24% on SST-2 and 86.05% on RT across various in-context few-shot settings. Table 2 indicates that the LLMs with ICL also perform well in the context of multi-class classification, specifically on the AG News dataset. The average accuracies stand at 69.11% for 4-shot settings and 72.32% for 8-shot settings across various LLMs. Additionally, LLMs with ICL exhibit improved performance with an increased number of in-context demos, particularly achieving best results with 8-shot settings. In summary, the effectiveness of ICL for text classification is evident across various LLMs, datasets, and the number of in-context demos, as illustrated in the ‘Clean’ rows of these two tables.

6.2 ICL Performance with Hijacking Attack

While LLMs with ICL perform well using clean in-context demos, Tables 1 and 2 demonstrate that hijacking attacks deteriorate its performance seriously through hijacking LLMs to generate the targeted output by adding adversarial suffixes to the in-context demos. The baseline attacks successfully induce LLMs to generate the targeted positive sentiment through a few shots of adversarially perturbed demos as shown in Table 1. As a result, the positive test samples achieve a predominantly higher accuracy than the negative ones. Furthermore, our GGI attack more effectively hijacks LLMs to generate the targeted output, i.e., exclusively positive sentiment token regardless of the test query content. The positive test samples achieve almost 100% accuracy, on the contrary, the negative ones get nearly 0% accuracy on most



(a)

(b)

Figure 2: Attentions maps generated using (a) clean and (b) adversarial perturbed prompts. In (b), the adversarial suffix tokens, i.e., 'NULL' and 'Remove', are underlined in red. Darker green colors represent larger attention weights. The prompts are tokenized to mimic the actual inputs to the LLMs. Best viewed in color.

settings as shown in 1. For the more complex multi-class AG News classification, the effectiveness of those baseline attacks decreases. As shown in Table 2, only GGI attack successfully hijacks the LLMs to generate the targeted token "tech" by appending the adversarial suffixes to the in-context demos.

In addition to the attack accuracy performance provided in Table 1 and 2, we present ASRs for various attacks across the three datasets. As outlined in Table 3, our GGI attack achieves the highest ASRs, substantiating its highest effectiveness in hijacking the LLM to generate the targeted output. In straightforward binary classification tasks like SST-2 and RT, all the attacks, except for Random, exhibit high ASRs. While for the more complex multi-class classification task, such as AG News, only our GGI attack achieves high ASRs.

6.3 Impact of In-context Demos

We extend our investigation to explore the impact of in-context demos on hijacking attacks. First, we observe a substantial impact on the attack performance in ICL based on the number of demos employed. As indicated in Tables 1 and 2, an increase in the number of in-context demos correlates with a higher susceptibility of the attack to hijack LLMs, resulting in the generation of targeted outputs with greater ease. For example, in the 8-shot setting, the LLMs consistently exhibit significantly lower accuracies in negative sentiment classification, demonstrating a higher rate of successful attacks, compared to the 2-shot setting across various datasets and LLMs. Moreover, the attacks demonstrate higher ASRs as the number of in-context demos used in ICL increases (Table 3).

6.4 Diverting LLM Attention

Attempting to interpret the possible mechanism of our hijacking attacks, we show an illustrative example using attention weights from LLaMA-7b on the SST2 task with both clean and perturbed prompts. As depicted in Figure 2b, the model's attention for generating the sentiment token of the test query has been diverted towards the adversarial suffix tokens 'NULL' and 'Remove'. Compared to the attention maps using the clean prompt (Figure 2a), these two suffixes attain the largest attention weights represented by the darkest green color. This example illuminates a possible mechanism on why our hijacking attack can induce the LLM to generate the targeted outputs - the adversarial suffixes divert LLM's attention away from the original query. Overall, the attention maps visualize how the adversarial suffixes distract the LLM from focusing on the relevant context to generate the targeted unwanted output.

7 Conclusion

ICL holds a strong promise for using LLMs in various tasks by giving a few demos without altering the parameters. Concernedly, this work reveals a new vulnerability of ICL via crafted hijacking attacks by adversarial users. By appending imperceptible adversarial suffixes to the in-context demos using a greedy gradient-based search, our GGI attack effectively hijacks the LLMs to generate the targeted unwanted outputs via diverting LLM's attention from the relevant context to the adversarial suffixes. Our findings underscore an urgent need for developing more robust ICL approaches.

8 Limitations and Risks

This work uncovers a potential vulnerability of LLMs during in-context learning. By inserting adversarial tokens, which our algorithm has learned, into in-context demos, we can make the LLM produce targeted, undesired outputs without need for a trigger in the query. Our evaluation focuses on the attack successful rate, particularly in the context of sentiment analysis and topic classification tasks. However, it is possible that our attack maybe more effective for the similar single-token generation tasks across the LLMs that are similar in sizes (or smaller) and training approaches. Further studies is warranted to extend our approach to a wide range of downstream tasks and LLMs.

This work represents a red teaming effort with the goal to discover the vulnerabilities of LLM during in-context learning. Therefore, it will not pose risks for natural users nor LLM vendors. Rather, our findings can be utilized by these stakeholders to guard against malicious uses and enhance the resilience of LLMs to such threats.

References

Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. 2020. Square attack: a query-efficient black-box adversarial attack via random search. In *European conference on computer vision*, pages 484–501. Springer.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Nicholas Carlini, Milad Nasr, Christopher A Choquette-Choo, Matthew Jagielski, Irena Gao, Anas Awadalla, Pang Wei Koh, Daphne Ippolito, Katherine Lee, Florian Tramer, et al. 2023. Are aligned neural networks adversarially aligned? *arXiv preprint arXiv:2306.15447*.

Stephen Casper, Jason Lin, Joe Kwon, Gatlen Culp, and Dylan Hadfield-Menell. 2023. Explore, establish, exploit: Red teaming language models from scratch. *arXiv preprint arXiv:2306.09442*.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J Pappas, and Eric Wong. 2023. Jailbreaking black box large language models in twenty queries. *arXiv preprint arXiv:2310.08419*.

Yanda Chen, Chen Zhao, Zhou Yu, Kathleen McKeown, and He He. 2022. On the relation between sensitivity and accuracy in in-context learning. *arXiv preprint arXiv:2209.07661*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2017. Hotflip: White-box adversarial examples for text classification. *arXiv preprint arXiv:1712.06751*.

Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, et al. 2022. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned. *arXiv preprint arXiv:2209.07858*.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE.

Neel Jain, Avi Schwarzschild, Yuxin Wen, Gowthami Somepalli, John Kirchenbauer, Ping-yeh Chiang, Micah Goldblum, Aniruddha Saha, Jonas Geiping, and Tom Goldstein. 2023. Baseline defenses for adversarial attacks against aligned language models. *arXiv preprint arXiv:2309.00614*.

Nikhil Kandpal, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. 2023. Backdoor attacks for in-context learning with language models. *arXiv preprint arXiv:2307.14692*.

Daniel Kang, Xuechen Li, Ion Stoica, Carlos Guestrin, Matei Zaharia, and Tatsunori Hashimoto. 2023. Exploiting programmatic behavior of llms: Dual-use through standard security attacks. *arXiv preprint arXiv:2302.05733*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Haoran Li, Dadi Guo, Wei Fan, Mingshi Xu, and Yangqiu Song. 2023a. Multi-step jailbreaking privacy attacks on chatgpt. *arXiv preprint arXiv:2304.05197*.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. *arXiv preprint arXiv:1812.05271*.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. Bert-attack: Adversarial attack against bert using bert. *arXiv preprint arXiv:2004.09984*.

Xin Li, Xiangrui Li, Deng Pan, Yao Qiang, and Dongxiao Zhu. 2023b. Learning compact features via in-training representation alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 8675–8683.

Xin Li, Xiangrui Li, Deng Pan, and Dongxiao Zhu. 2021. Improving adversarial robustness via probabilistically compact loss with logit constraints. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 8482–8490.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*.

Natalie Maus, Patrick Chao, Eric Wong, and Jacob R Gardner. 2023. Black box adversarial prompting for foundation models. In *The Second Workshop on New Frontiers in Adversarial Machine Learning*.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.

John X Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. *arXiv preprint arXiv:2005.05909*.

Tai Nguyen and Eric Wong. 2023. In-context example selection with influences. *arXiv preprint arXiv:2302.11042*.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*.

Fábio Perez and Ian Ribeiro. 2022. Ignore previous prompt: Attack techniques for language models. *arXiv preprint arXiv:2211.09527*.

Yao Qiang, Supriya Tumkur Suresh Kumar, Marco Brocanelli, and Dongxiao Zhu. 2022. Tiny rnn model with certified robustness for text classification. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Yao Qiang, Chengyin Li, Prashant Khanduri, and Dongxiao Zhu. 2023. Interpretability-aware vision transformer. *arXiv preprint arXiv:2309.08035*.

Yao Qiang, Xin Li, and Dongxiao Zhu. 2020. Toward tag-free aspect based sentiment analysis: A multiple attention network approach. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Yasaman Razeghi, Robert L Logan IV, Matt Gardner, and Sameer Singh. 2022. Impact of pretraining term frequencies on few-shot reasoning. *arXiv preprint arXiv:2202.07206*.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2021. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*.

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. 2023. Are emergent abilities of large language models a mirage? *arXiv preprint arXiv:2304.15004*.

Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. 2023. " do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. *arXiv preprint arXiv:2308.03825*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Haoyu Wang, Guozheng Ma, Cong Yu, Ning Gui, Linrui Zhang, Zhiqi Huang, Suwei Ma, Yongzhe Chang, Sen Zhang, Li Shen, et al. 2023a. Are large language models really robust to word-level perturbations? *arXiv preprint arXiv:2309.11166*.

Jindong Wang, Xixu Hu, Wenxin Hou, Hao Chen, Runkai Zheng, Yidong Wang, Linyi Yang, Haojun Huang, Wei Ye, Xiubo Geng, et al. 2023b. On the robustness of chatgpt: An adversarial and out-of-distribution perspective. *arXiv preprint arXiv:2302.12095*.

Jiongxiao Wang, Zichen Liu, Keun Hee Park, Muhao Chen, and Chaowei Xiao. 2023c. Adversarial demonstration attacks on large language models. *arXiv preprint arXiv:2305.14950*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Yuxin Wen, Neel Jain, John Kirchenbauer, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2023. Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery. *arXiv preprint arXiv:2302.03668*.

Zhiyong Wu, Yaoxiang Wang, Jiacheng Ye, and Lingpeng Kong. 2022. Self-adaptive in-context learning. *arXiv preprint arXiv:2212.10375*.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2021. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS*.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706. PMLR.

Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Neil Zhenqiang Gong, Yue Zhang, et al. 2023a. Promptbench: Towards evaluating the robustness of large language models on adversarial prompts. *arXiv preprint arXiv:2306.04528*.

Sicheng Zhu, Ruiyi Zhang, Bang An, Gang Wu, Joe Barrow, Zichao Wang, Furong Huang, Ani Nenkova, and Tong Sun. 2023b. Autodan: Automatic and interpretable adversarial attacks on large language models. *arXiv preprint arXiv:2310.15140*.

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

9 Appendix

9.1 Experiments Details

We show the dataset statistics in Table 4. Specifically for the SST2 and RT sentiment analysis tasks, we employ only 2 training queries to train adversarial suffix tokens using our GGI method. We use 4 training queries for the more complex multi-class classification tasks, i.e., AG News. To evaluate the performance of ICL and hijack attacks, we randomly select 1,000 queries as user queries for testing.

Table 4: Statistics of the training set and test set splits for the three datasets.

Datasets	Training Queries	Test Queries
SST2	2	1,000
RT		1,000
AG News	4	1,000

Figure 3 illustrates the prompt template employed in ICL for various tasks. Additionally, examples are provided for each task to enhance understanding. For the SST2/RT dataset, the template is structured to include an instruction, a demo set composed of reviews and sentiment labels, and the user query. Similarly, the AG News dataset template comprises the instruction, the demo set with articles and topic labels, and the user query.

9.2 Impact of Sizes of LLMs

We have shown in the main text (Table 3) that the ASRs on GPT2-XL are significantly higher than those on LLaMA-7b, suggesting that hijacking the larger LLM is more challenging. Here we continue on examining how the size of LLMs influences the performance of hijacking attacks. We evaluate the hijacking attack performance using ASR on two OPT models of varying sizes in Figure 4. It clearly shows that attacking the smaller OPT2-2.7b model achieves a much higher ASR in both 4-shot and 8-shot settings, confirming our finding and others (Wang et al., 2023a) that larger models are more resistant to adversarial attacks.

9.3 Comparison of Hijacking Attacks

In contrast to baseline hijacking attacks, our GGI exhibits a superior performance in generating targeted outputs, as evidenced by the results in Table 1 and 2, along with the highest ASRs highlighted in Table 3. This underscores the effectiveness of GGI as a more potent method of attack.

To further illustrate the efficiency of our GGI, we present the objective function values of Eq. 5 in Figure 5 for various attack methods. Since our GGI attack enjoys the advantages of both greedy and gradient-based search strategies as depicted in Algorithm 1, the values of the object function decrease steadily and rapidly, ultimately reaching the minimum loss value. On the other hand, both the Square and Greedy attacks simply use a greedy search strategy, with fluctuating results that increase and decrease the loss value, unable to converge to the minimum loss value that corresponds to the optimal adversarial suffixes.

Dataset	Structure	Template	Example
SST-2/RT	Instruction	[instruction]	Analyze the sentiment of the last review and respond with either positive or negative. Here are several examples.
	Demos	Review: [sentence] Sentiment: label … Review: [sentence] Sentiment: [label]	Review: "a retread story , bad writing , and the same old silliness " Sentiment: negative … Review: is definitely a director to watch Sentiment: positive
	Queries	Review: [sentence] Sentiment:	Review: waste their time on it Sentiment:
AG-News	Instruction	[instruction]	Classify the topic of the last article. Here are several examples.
	Demos	Article: [article] Topic: [label] … Article: [article] Topic: [label]	Article: Skier Tests Positive Olympic silver medalist Hans Knauss tests positive for the steroid nandrolone after a World Cup race last month. Topic: sports … Article: "Apple Unwraps New IMac G5s PARIS -- Apple Computer will begin shipping its new IMac G5 desktop computer worldwide in mid-September, the company #39;s top marketing executive says.", Topic: technology
	Queries	Article: [article] Topic:	Article: Microsoft adds to Visual Studio tools line 2005 Standard Edition targets developers working in small organizations. Topic:

Figure 3: Template designs for all the datasets used in our experiments. We also provide examples for these datasets to ensure a better understanding.

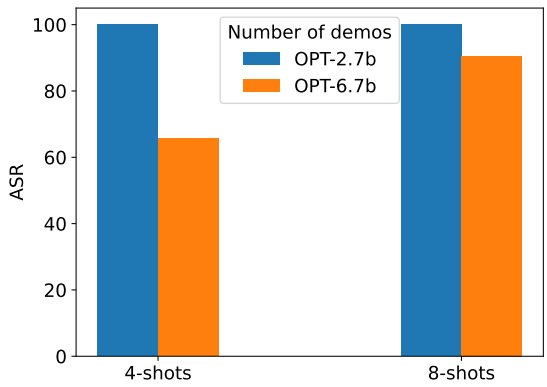


Figure 4: Impact of LLM size on adversarial robustness. ASRs on the AG News topic classification task using different sizes of OPT models, i.e., OPT-2.7b and OPT-6.7b, with two different few-shot settings.

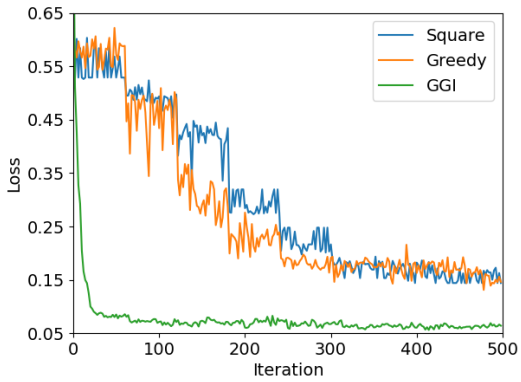


Figure 5: An illustration of the learning objective values during iterations among different attacks on SST2 using GPT2-XL with 8-shots.