Improved Generation of Adversarial Examples Against Safety-aligned LLMs

Qizhang Li[1,2], **Yiwen Guo**[3*], **Wangmeng Zuo**[1], **Hao Chen**[4]
[1]Harbin Institute of Technology, [2]Tencent Security Big Data Lab, [3]Independent Researcher, [4]UC Davis
{liqizhang95,guoyiwen89}@gmail.com wmzuo@hit.edu.cn chen@ucdavis.edu

Abstract

Adversarial prompts (or say, adversarial examples) generated using gradient-based methods exhibit outstanding performance in performing automatic jailbreak attacks against safety-aligned LLMs. Nevertheless, due to the discrete nature of texts, the input gradient of LLMs struggles to precisely reflect the magnitude of loss change that results from token replacements in the prompt, leading to limited attack success rates against safety-aligned LLMs, even in the *white-box* setting. In this paper, we explore a new perspective on this problem, suggesting that it can be alleviated by leveraging innovations inspired in transfer-based attacks that were originally proposed for attacking *black-box* image classification models. For the first time, we appropriate the ideologies of effective methods among these transfer-based attacks, *i.e.*, Skip Gradient Method [53] and Intermediate Level Attack [18], into gradient-based adversarial prompt generation and achieve significant performance gains without introducing obvious computational cost. Meanwhile, by discussing mechanisms behind the gains, new insights are drawn, and proper combinations of these methods are also developed. Our empirical results show that 87% of the query-specific adversarial suffixes generated by the developed combination can induce Llama-2-7B-Chat to produce the output that exactly matches the target string on AdvBench. This match rate is 33% higher than that of a very strong baseline known as GCG, demonstrating advanced discrete optimization for adversarial prompt generation against LLMs. In addition, without introducing obvious cost, the combination achieves $> 30\%$ absolute increase in attack success rates compared with GCG when generating both query-specific ($38\% \rightarrow 68\%$) and universal adversarial prompts ($26.68\% \rightarrow 60.32\%$) for attacking the Llama-2-7B-Chat model on AdvBench. Code at: https://github.com/qizhangli/Gradient-based-Jailbreak-Attacks.

1 Introduction

Large language models (LLMs) have demonstrated a formidable capacity for language comprehension and the generation of human-like text. Safty-aligned LLMs, refined through specific fine-tuning mechanisms [38, 3, 21, 12], are anticipated to yield responses that are not only helpful but also devoid of harm in response to user instructions. However, certain studies [42, 49, 58, 40, 6, 30] reveal that these models have not yet achieved perfect alignment. It has been demonstrated that these models can be carefully prompted to produce harmful content through the introduction of meticulously crafted prompts, a phenomenon known as "jailbreak" [49]. The manually designed jailbreak prompts are crafted by carefully constructing scenarios that mislead the models, necessitating a significant amount of work. In contrast, adversarial examples are automatically generated with the intent deceiving models to generate harmful responses, presenting a more insidious challenge to model robustness.

*Yiwen Guo leads the project and serves as the corresponding author.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

One of the main difficulties in generating adversarial examples for NLP models lies in the fact that text is discrete by nature, making it challenging to use gradient-based optimization methods to devise adversarial attacks. There has been some work [15, 52, 43, 20, 58] attempted to overcome this issue. For instance, recently, a method called Greedy Coordinate Gradient (GCG) attack [58] has shown significant jailbreaking improvements, by calculating gradients of cross-entropy loss *w.r.t.* one-hot representations of chosen tokens in a prompt and replacing them in a greedy manner. However, due to the fact that the gradients *w.r.t.* one-hot vectors do not provide precise indication of the loss change that results from a token replacement, the GCG attack shows limited white-box attack success rates against some safty-aligned LLMs, *e.g.*, Llama-2-Chat models [46].

In this paper, we carefully examine the discrepancy between the gradient of the adversarial loss *w.r.t.* one-hot vectors and the real effect of the change in loss that results from token replacement. We present a new perspective that this gap resembles the gap between input gradients calculated using a substitute model and the real effect of perturbing inputs on the prediction of a black-box victim model, which has been widely studied in transfer-based attacks against black-box image classification models [45, 39, 31, 18, 53, 16, 29, 28]. Based on this new perspective, for the first time, we attempt to appropriating the ideologies of two effective methods among these transfer-based methods, *i.e.*, Skip Gradient Method (SGM) [53] and Intermediate Level Attack (ILA) [18], to improve the gradient-based attacks against LLMs. With appropriate adaptations, we successfully inject these ideologies into the gradient-based adversarial prompt generation without additional computational cost. By discussing the mechanisms behind the advanced performance, we provide new insights about improving discrete optimizations on LLMs. Moreover, we provide an appropriate combination of these methods. The experimental results demonstrate that 87% of the query-specific adversarial suffixes generated by the combination for attacking Llama-2-7B-Chat on AdvBench can induce the model output exact target string, which outperforms a strong baseline named GCG attack (54%), indicating an advanced discrete optimization for adversarial prompt generation against LLMs. In addition, the combination achieves attack success rates of 68% for query-specific and 60.32% for universal adversarial prompt generation when attacking Llama-2-7B-Chat on AdvBench, which are higher than those of the baseline GCG (38% and 26.68%).

2 Related Work

Jailbreak attacks. Recent work highlights that the safety-aligned models are still not perfectly aligned [4, 42, 49], the safety-aligned LLMs can be induced to produce harmful content by some carefully designed prompts, known as jailbreak attacks [49]. This has raised security concerns and attracted great attention. In addition to some manually designed prompt methods [49, 42], numerous automatic jailbreak attack methods have been proposed. Some methods directly optimize the text input through gradient-based optimization [47, 15, 43, 52, 20, 58]. Another line of work involves using LLMs as optimizers to jailbreak the victim LLM [40, 6, 33]. There are also methods that focus on designing special jailbreaking templates or pipelines [30, 41, 5, 7, 56, 50]. According to the knowledge of the victim model, these methods can also be divided into white-box attacks and black-box attacks. In the context of white-box attacks, the attackers have full access to the architecture and parameters of the victim LLM, making them can leverage the gradient with respect to the inputs. As demonstrated by recent benchmark [32], represented by a current method as known as GCG attack [58], gradient-based automatic jailbreak attacks have shown the most powerful performance in compromising LLMs in the setting of white-box attack. However, due to the discrete nature of text input, dealing with the discrete optimization problem is rather challenging, which limits the success rates of attacks, especially those against Llama-2-Chat models [46]. In our work, we primarily focus on solving the discrete optimization problem in gradient-based automatic jailbreak attacks to improve the success rate in the white-box setting.

Transfer-based attacks. Transfer-based attacks attempt to craft adversarial examples on a white-box substitute model to attack the black-box victim model, by leveraging the transferability of adversarial examples [45], which is a phenomenon that adversarial examples crafted on a white-box substitute model can also mislead the unknown victim models with a decent success rate. The transfer-based attacks have been thoroughly investigated in the setting of attacking image classification models [22, 54, 8, 53, 18, 13, 16, 17, 27, 26, 25, 28]. Some recent methods also utilize the transferability of adversarial prompts to perform black-box attacks against LLMs [58, 30, 44]. While in this work, we mainly focus on improving the white-box attack success rate. In our work, we reveal a closely

relationship between the optimization in transfer-based attacks and discrete optimization of the gradient-based jailbreak attacks against LLMs. We then appropriate the ideologies of two effective transfer-based attack methods developed in the setting of attacking image classification model, *i.e.*, SGM [53] and ILA [18]. Moreover, by adapting these strategies and analyzing the mechanism behind them, we provide some new insights about potential solutions for addressing problems involving discrete optimization in NLP models with transformer architecture, *e.g.*, prompt tuning.

3 Gap Between Input Gradients and the Effects of Token Replacements

In this section, we first discuss the main obstacle in achieving effective gradient-based attacks on safety-aligned LLMs, which is considered as the gap between input gradients and the effects of token replacements, and then we show how transfer-based strategies can be adapted to overcome the issue.

3.1 Rethinking Gradient-based Attacks Against LLMs

Previous endeavors utilize the gradient *w.r.t.* the token embeddings [23, 52] or *w.r.t.* the one-hot representations of tokens [10, 43, 58, 20], in order to solve the discrete optimization problem efficiently during attacking LLMs. A recent method, named Greedy Coordinate Gradient (GCG) [58], shows a significant improvement over other optimizers (*e.g.*, AutoPrompt [43] and PEZ [52]) on performing gradient-based attacks against LLMs in the white-box setting. Due to its effectiveness, we take GCG as a strong baseline and as a representative example to analyze previous gradient-based attacks against LLMs in this section.

A typical LLM $f : \mathcal{X} \rightarrow \mathbb{R}^{|V|}$ with a vocabulary V is trained to map a sequence of tokens $x_{1:n} = [x_1, ..., x_n] \in \mathcal{X}, x_i \in V$ to a probability distribution over the next token, denoted as $p_f(x_{n+1}|x_{1:n})$. To evoke a jailbreak to induce the a safety-aligned LLM generate harmful content according the user query, GCG attempts to add an adversarial suffix to the original user query and iteratively modifies the adversarial suffix to encourages the model output an affirmative target phrase, *e.g.*, "Sure, here's ...". Consider an adversarial prompt (which is the concatenation of a user query and an adversarial suffix) as $x_{1:n}$ and a further concatenation with a target phrase as $x_{1:n^*}$, GCG aims to minimize the adversarial loss $L(x_{1:n^*})$ (denoted as $L(x)$ for simplicity), which corresponds to the negative log probability of the target phrase. It can be written as

$$\min_{x_{\mathcal{A}} \in \{1,...,V\}^{|\mathcal{A}|}} L(x_{1:n^*}) = \min_{x_{\mathcal{A}} \in \{1,...,V\}^{|\mathcal{A}|}} \frac{1}{n^* - n} \sum_{i=1}^{n^*-n} -\log p_f(x_{n+i}|x_{1:n+i-1}), \tag{1}$$

where $x_{\mathcal{A}}$ denotes the tokens of adversarial suffix in $x_{1:n}$ and $\mathcal{A}$ denotes the set of indices of the adversarial suffix tokens. At each iteration, it computes the gradient of the adversarial loss *w.r.t.* the one-hot vector of a single token and uses each value of gradient to estimate the effect of replacing the current token with the corresponding one on loss. Since the discrete nature of input, there is a gap between input gradients and the effects of token replacements, thus the estimate is not accurate enough. Previous efforts attempt to solve this problem by collecting a candidate set of token replacements according to the Top-k values in the gradient and evaluating the candidates to pick the best token replacement with minimal loss [58, 43]. Due to the large gap, they requires a large candidate set size, *e.g.*, 512 in GCG, which result in a large computational cost. Ideally, if the input gradients can accurately reflect the effects of token replacements, $k = 1$ is sufficient to achieve minimal loss without needing a candidate set, thus obtaining the optimal token replacement with the lowest computational cost. Therefore, we advocate for refining the input gradient to narrow the gap, thereby improving performance while reducing computational cost.

We attempt to introduce a new perspective to the gap between the input gradients and the real effects of token replacements. Let us first revisit transfer-based attacks on image classification models. To generate an adversarial example that misleads an unknown victim model, where the input gradient is not accessible, attackers use a white-box substitute model as a proxy for the victim model and utilize its input gradient. Due to the gap between the input gradient of the substitute model and the real input gradient of the victim model, directly using the input gradient of the substitute model to modify the example yields unsatisfactory results. Efforts [9, 54, 18, 53, 24, 16, 27] have been made to refine the gradient computed on the substitute model in order to narrow this gap. Returning to our discrete optimization problem in the adversarial prompt generation, the strategy of utilizing the

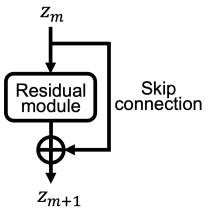
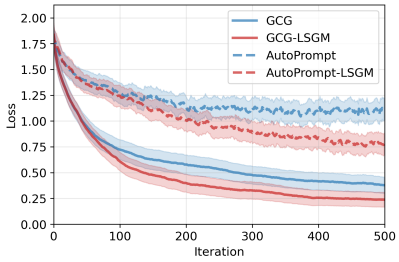
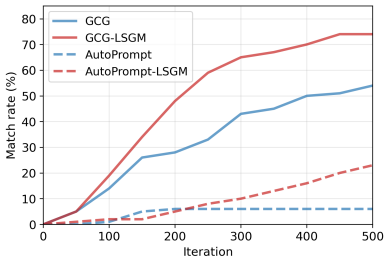


Figure 1: An example of the residual block.



(a) Loss



(b) Match rate

Figure 2: How (a) the loss and (b) the match rate changes with attack iterations. The attacks are performed against Llama-2-7B-Chat model to generate query-specific adversarial suffixes on AdvBench. Best viewed in color.

gradient *w.r.t.* the one-hot vector of the token is actually treating the one-hot vector as though it was a continuous variable, resulting in the gap between this gradient and the "real gradient" (*e.g.*, the real effects of token replacements). We consider that this gap is analogous to the input gradient gap between substitute and victim models in the context of transfer-based attacks. This new perspective allows one to introduce a series of innovations developed within the realm of transfer-based attacks on image classification models to refine the gradient computation in the context of gradient-based adversarial prompt generation against safety-aligned LLMs.

From the results of a recent benchmark of these transfer-based attacks [28], we can observe that several strategies, including SGM [53], SE [36], PNA [51], and a series of intermediate level attacks (ILA [18], ILA++ [17], FIA [48], and NAA [57]), are obviously effective when generating adversarial examples on models with a transformer architecture. Among them, PNA ignores the backpropagation of the attention map and SE requires considerable ability to directly predict the probability from the intermediate representations, thus their strategies are difficult to be adapted to the context of LLM attacks. Therefore, we consider drawing inspiration from SGM and ILA (which is the representative of all intermediate level attacks).

3.2 Reducing the Gradients from Residual Modules

Modern deep neural networks typically comprise a number of residual blocks, each consisting of a residual module and a skip connection branch, as depicted in Figure 1. SGM [53] experimentally found that reducing the gradients from residual modules during backpropagation can improve the transfer-based attacks against image classification models, indicating that it can reduce the gap between the input gradients and the effects resulting from perturbing inputs on a unknown victim model. In this section, we investigate whether the strategy of reducing gradients from residual modules can also enhance gradient-based attacks against LLMs in a white-box setting. Additionally, we discuss the mechanisms behind this strategy to provide new insights.

An l-layers LLM can be decomposed into $2l$ residual building blocks, with each block consisting of a residual module (which should be an MLP or an attention module) and a parallel skip connection branch as illustrated in Figure 1. The m-th block maps an intermediate representation z_m to z_{m+1}, *i.e.*, $z_{m+1} = I(z_m) + R_m(z_m)$, where I is an identity function representing the skip connection branch and R_m denotes the residual module of the m-th block. By adopting a decay factor $\gamma \in [0, 1]$ for the residual modules, SGM calculates the derivative of the m-th block as $\nabla_{z_m} z_{m+1} = 1 + \gamma \nabla_{z_m} R_m(z_m)$. We incorporate this strategy into gradient-based automatic adversarial prompt generation, denoted as Language SGM (LSGM). We evaluate the performance of integrating this strategy into GCG and AutoPrompt attacks by setting $\gamma = 0.5$, and show the results in Figure 2. The experiment is conducted to generate query-specific adversarial suffixes against Llama-2-7B-Chat [46] on the first 100 harmful queries in AdvBench [58]. To provide a more comprehensive comparison, we report not only the adversarial loss but also the fraction of adversarial prompts that result in outputs exactly matching the target string, dubbed match rate, which is also used as an evaluation metric in the paper of GCG [58]. It can be seen from the figure that reducing gradients from residual modules can indeed improve the performance of gradient-based adversarial prompt generation. The GCG-LSGM achieves a match rate of 72%, while the baseline (*i.e.*, GCG) only obtains 54% match rate. We also evaluate the attack success rate (ASR) by using the evaluator proposed by HarmBench [32]. It shows an ASR of 62% when using GCG-LSGM, while the GCG only achieves 38%. On the other hand, when reducing the

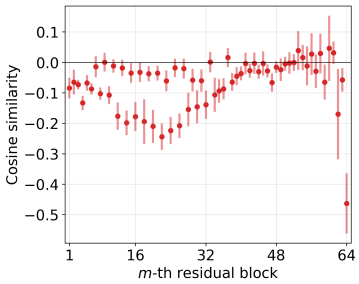


Figure 3: The cosine similarities between the gradients from residual modules and the gradients from skip connections in different residual blocks.

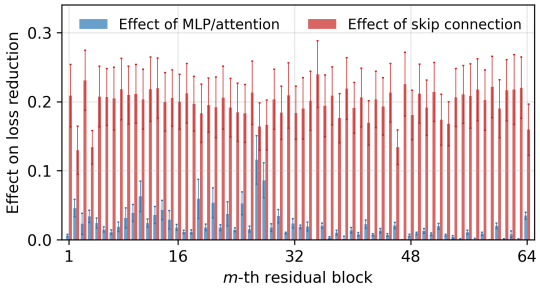


Figure 4: Comparing the average effects of residual modules and the average effects of skip connections on the change in adversarial loss varies with different residual blocks. Best viewed in color.

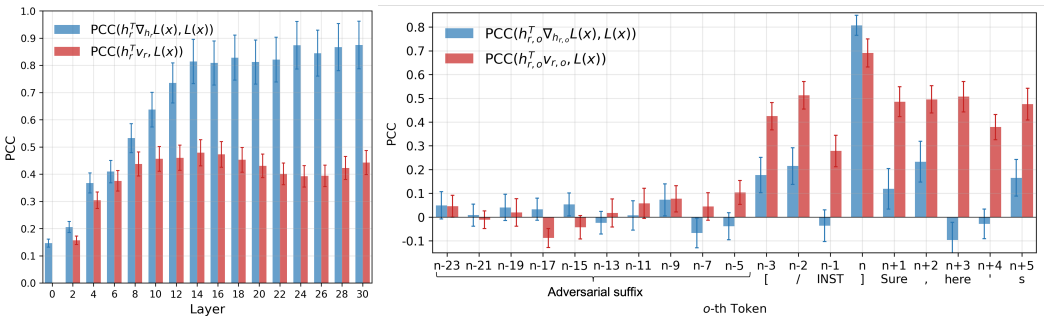
gradient from the skip connection, both the match rate and the attack success rate will drop to 0%. The results confirm that reducing gradients from residual modules helps the gradients *w.r.t.* one-hot representations to be more effective to indicate the real effects of token replacements.

Having seen the effectiveness of GCG-LSGM, let us further delve deep into the mechanism behind the method. We begin by analyzing the computational graph and gradient flow of a building block. Following the chain rule, the gradient of adversarial loss *w.r.t.* z_m can be written as a summation of two terms: $\nabla_{z_m} L(x) = \nabla_{z_{m+1}} L(x) + \nabla_{z_m} R(z_m) \nabla_{z_{m+1}} L(x)$. The first term represents the gradient from the skip connection, and the second term represents the gradient from the residual module. They represent the impact on the loss caused by changes in z_m through skip connection branch and residual module branch, respectively. In Figure 3, we visualize the average cosine similarity between these two terms at the 100-th iteration of the GCG attack against Llama2-7B-Chat across 100 examples. The same observations can be obtained at other iterations during the GCG attack. Somewhat surprisingly, it can be seen that these two terms have negative cosine similarity in most blocks during the iterative optimization. Obviously, the summation of two directionally conflicting gradients may mitigate the effect of each other's branch on reducing the loss. Hence, reducing the gradients from residual modules achieves lower loss values by sacrificing in the effects of the residual modules to trade more effects of the skip connection on loss. The success of GCG-LSGM somehow implies the gradient flowing from skip connections better reflect the effect of token replacements on adversarial loss.

To confirm the conjecture, we attempt to evaluate the effect of each branch towards reducing the loss. Inspired by the causal tracing technique [34], we perform the following steps to compute the effect of each branch's hidden state. First, we give an adversarial prompt to the model to obtain the loss, denoted by $L(x)$. Second, we randomly alter a token from the adversarial prompt and record the hidden states in the two branches, *i.e.*, $I(\tilde{z}_m)$ and $R_m(\tilde{z}_m)$, given the altered adversarial prompt. Finally, we feed the original adversarial prompt into the model and modify the forward computation by replacing $I(z_m)$ with $I(\tilde{z}_m)$ (or $R_m(z_m)$ with $R_m(\tilde{z}_m)$), to obtain the modified loss $\tilde{L}(x)$. The effect of a branch is represented by the difference between the loss of obtained on the third step and the first step, *i.e.*, $\tilde{L}_m(x) - L(x)$. A high $\tilde{L}_m(x) - L(x)$ indicates the branch to replaced hidden state is important on affect the adversarial loss. By averaging the values $\tilde{L}_m(x) - L(x)$ over a collection of adversarial prompts, we can get the average effects of residual module and skip connection in the m-th block. In Figure 4, we show the average effects of residual modules and skip connections. The adversarial promp are obtained by performing GCG attack against Llama-2-7B-Chat model on AdvBench. It can be seen that the skip connections show much greater effects than residual modules on the adversarial loss, which confirms the conjecture. The observation further implies that the effects of adversarial information primarily propagate through the skip connection branch within each residual block. This observation, alongside the superior gradient-based attack performance of LSGM, further suggests that certain discrete optimization challenges within LLMs, *e.g.*, prompt tuning, might be more effectively addressed by understanding the information flow throughout the forward pass.

3.3 Adapting Intermediate Level Attack for Gradient-base Attacks Against LLMs

Intermediate level attacks [18, 14, 48, 57, 26] opt to maximizing the scalar projection of the intermediate level representation onto a "directional guide" for generating non-target adversarial examples



(a) PCCs computed on the h_r

(b) PCCs computed on the $h_{r,o}$

Figure 5: (a) The PCCs computed on the entire intermediate representations, *i.e.*, $\mathrm{PCC}(h_r^T \nabla_{h_r} L(x), L(x))$ and $\mathrm{PCC}(h_r^T v_r, L(x))$, at different layers of Llama-2-7B-Chat. (b) The PCCs computed on the o-th token intermediate representations, *i.e.*, $\mathrm{PCC}(h_{r,o} \nabla_{h_{r,o}} L(x), L(x))$ and $\mathrm{PCC}(h_{r,o}^T v_{r,o}, L(x))$, at the mid-layer of Llama-2-7B-Chat. Best viewed in color.

to attack image classification models. As a representative method, ILA [18] defines the directional guide as the intermediate level representation discrepancy between the adversarial example obtained by a preliminary attack, *e.g.*, I-FGSM [22], and corresponding benign example. In this section, we first examine whether the strategy of ILA can be directly applied to the gradient-based attacks on LLMs. Then, by seeing the failure of direct application, we investigate the proper way to adapt this strategy to the gradient-based attacks on LLMs.

Let h_r be the intermediate representation at the r-th layer, and the directional guide $v_r = \hat{h}_r^0 - \hat{h}_r^t$ is obtained by a t-iterations preliminary attack (*e.g.*, GCG). Note that since our objective is to minimize the adversarial loss instead of maximizing the victim's classification loss, as in the original setting of ILA paper [18], the definition of the directional guide is the opposite of that in the ILA paper. According the implementation of ILA, we maximize the scalar projection of the intermediate representation h_r onto the directional guide v_r, *i.e.*, maximize $L_{\mathrm{ILA}}(x) = h_r^T v_r$. However, we evaluated ILA using GCG as the back-end method and found that it exhibited deteriorated performance compared to the baseline. Specifically, when attacking Llama-2-7B-Chat, it achieves a match rate of 44%, while the baseline GCG obtains 54% match rate. This result demonstrates that when applied directly, the ILA fails to facilitate gradient-based adversarial prompt generation.

Let us discuss why directly introducing ILA fails to improve performance. Recall that intermediate-level attacks against image classification models essentially assume that the scalar projection of an intermediate representation onto the directional guide has a positive correlation with the transferability of adversarial examples, which means that the larger the scalar projection obtained, the greater the transferability (*i.e.*, higher classification loss on victim models) the adversarial example achieves [26]. Previous work [26] has also shown that the ILA is equivalent to replacing the gradient *w.r.t.* the intermediate representation with the directional guide. This demonstrates that ILA necessitates a stronger positive correlation between the scalar projection of the representation onto the directional guide and the victim's classification loss, compared to the positive correlation between the scalar projection of the representation onto its gradient and the victim's classification loss. Back to our setting, we conduct experiments to examine whether these assumptions hold when attacking LLMs. We use Pearson's correlation coefficient (PCC) to show the correlation between the scalar projection and adversarial loss. With a range in $[-1, 1]$, a PCC close to 1 indicates a positive correlation, while a PCC close to -1 means a negative correlation. The experiment is conducted with following steps. Firstly, we perform a GCG attack to obtain an adversarial example, and use it to derive a directional guide v_r and the gradient of adversarial loss $L(x)$ *w.r.t.* the intermediate representation h_r, *i.e.*, $\nabla_{h_r} L(x)$. Next, we randomly alter adversarial tokens several times and input these new prompts into the model to collect a set of intermediate representations paired with their corresponding adversarial losses. We then calculate the PCC between the scalar projection of the intermediate representation onto the directional guide and the adversarial loss, *i.e.*, $\mathrm{PCC}(h_r^T v_r, L(x))$, and the PCC between the scalar projection of the intermediate representation onto its gradient and the adversarial loss, *i.e.*, $\mathrm{PCC}(h_r^T \nabla_{h_r} L(x), L(x))$. We perform this experiment using Llama-2-7B-Chat on AdvBench. In Figure 5(a), we show the PCCs between the scalar projection and the adversarial loss at different

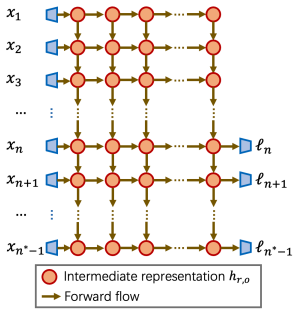
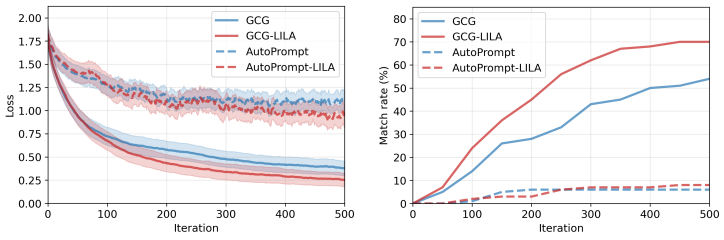


Figure 6: An example of the internal computation of an LLM. The ℓ represents cross-entropy loss.



(a) Loss (b) Match rate

Figure 7: How (a) the loss and (b) the match rate changes with attack iterations. The attacks are performed against Llama-2-7B-Chat model to generate query-specific adversarial suffixes on AdvBench. Best viewed in color.

layers. It can be seen that in every layer, the PCCs computed using the directional guide are all ≤ 0.5, showing a weaker correlation compared to the PCCs computed using the intermediate gradient, which exceed 0.8 after the 14th layer. This confirms that the failure to directly apply ILA is due to its poorer performance in the intermediate layers in indicating the change of adversarial loss compared to the baseline. This phenomenon is distinct from the one observed in the setting of attacking image classification models [26].

Nevertheless, by unfolding the internal computation of an LLM as a grid of token representations $\{h_{r,o}\}$, where $h_{r,o}$ denotes the o-th token representation at r-th layer, as depicted in Figure 6, some previous work [34, 35, 11, 37] have shown that different token representations at a intermediate layer show distinct effects on the model output. It inspires us to further investigate whether the positive correlation exists between the scalar projection of a single token representation onto its directional guide and the adversarial loss (*i.e.*, $\mathrm{PCC}(h_{r,o}^T v_{r,o}, L(x))$), and whether this correlation is stronger than the positive correlation between the scalar projection computed on its gradient and the adversarial loss (*i.e.*, $\mathrm{PCC}(h_{r,o}^T \nabla_{h_{r,o}} L(x), L(x))$). In Figure 5(a), we present $\mathrm{PCC}(h_{r,o}^T v_{r,o}, L(x))$ and $\mathrm{PCC}(h_{r,o}^T \nabla_{h_{r,o}} L(x), L(x))$ with different choices of o-th token at the mid-layer of Llama-2-7B-Chat. Since the number of target tokens varies across different adversarial prompts, we select the first five tokens from the target phrase. First, the scalar projections computed using the directional guide and intermediate gradient both show a very weak positive correlation with the adversarial loss in the intermediate representations of adversarial tokens. Second, the PCCs calculated for the token representation of "]", which is the last token of the input prompt and is expected to output the first token of the target phrase (*e.g.*, "Sure"), demonstrate that the scalar projection computed using the gradient can better indicate the change in adversarial loss than that computed using the directional guide. We consider that successfully generating the first token of the target (e.g., "Sure") plays a crucial role in reducing the adversarial loss. The adversarial prompt used to obtain the directional guide is not sufficiently optimized to minimize the loss associated with generating the first token of the target. Somewhat interestingly, the high PCC value between the projection of last token representation onto the gradient (or directional guide) and the adversarial loss indicates that directly moving the last token representation along the direction of gradient (*i.e.*, intervening in the last token representation by adding a vector in the direction of gradient or directional guide) will effectively reduce the adversarial loss. Some concurrent work [2, 55] also observes that certain directions can be added to the token representations to jailbreak safety-aligned LLMs. In contrast to our approach, they collect the token representations of a set of harmful queries and a set of harmless queries, defining the direction as the difference between the average last token representations of these two sets. Third, for other token representations, the scalar projections onto directional guides show a stronger correlation with adversarial loss than the scalar projections onto their gradients. This indicates that projecting the token representation onto the directional guide can better reflect changes in adversarial loss compared to projecting it onto the intermediate gradient.

Therefore, to effectively adapt ILA for adversarial prompt generation we propose replacing the intermediate gradient of the adversarial loss with a directional guide at specific token representations, rather than at the entire intermediate representation as in the original implementation. We re-normalize the directional guide by $\|\nabla_{h_{r,o}} L(x)\|_2 / \|v_{r,o}\|_2$, in order to avoid too large (or small) compared to the gradient *w.r.t.* other token representations. We denote this adaptation of ILA as Language ILA (LILA).

When performing LILA at the o-th token representation at r-th layer, during backpropagation, the gradient *w.r.t.* the token representation, *i.e.*, $\nabla_{h_r} L(x)$ is changed to

$$\nabla_{h_r}^{\text{LILA}} L(x) = [\nabla_{h_{r,1}} L(x), \nabla_{h_{r,2}} L(x), ..., \frac{\|\nabla_{h_{r,o}} L(x)\|_2}{\|v_{r,o}\|_2} v_{r,o}, ..., \nabla_{h_{r,n^*-1}} L(x)]. \tag{2}$$

Another issue of the original ILA here is that it requires a preliminary attack, which is time-consuming to obtain in the setting of attacking LLMs. We make a compromise by utilizing the current adversarial prompt to obtain the directional guide, which leads to little increase in computational complexity. For instance, at t-th iteration, the directional guide is $v_{r,o}^t = h_{r,o}^0 - h_{r,o}^t$. Note that we only modify the gradient computation step in each iteration. Therefore, the step of evaluating candidate adversarial suffixes can help reduce the adversarial loss, providing useful directional guidance in the initial iterations. We evaluate LILA using GCG and AutoPrompt as baseline attacks against the Llama-2-7B-Chat model on AdvBench. As shown in Figure 7, LILA demonstrates a lower adversarial loss and a higher match rate compared to the baseline attacks. The experimental results suggest that our adaptation of ILA indeed improves discrete optimization in gradient-based adversarial prompt generation and offers insights to more effectively address similar discrete optimization problems within LLMs.

4 Experiments

We introduce the evaluation metrics in Section 4.1, and then present the experimental results in Section4.2. Some detailed experimental settings and ablation studies are presented in the Appendix.

4.1 Metrics

We use two metrics for the evaluations: match rate (MR) and attack success rate (ASR). The match rate counts the fraction of adversarial examples that make the output exactly match the target string, and was used to evaluate different optimization methods in the paper of GCG [58]. The attack success rate is evaluated using the evaluator proposed by HarmBench [32], which achieves over 93% human agreement rate as reported in their paper. We set the models to generate 512 tokens with greedy decoding during the evaluation phase. Note that for the universal adversarial suffix generation, we observed that the ASRs of the adversarial suffixes obtained by multiple runs for the same method differ significantly. Hence, we run each method ten times and report not only the average ASR (AASR) but also the best ASR (BASR) and the worst ASR (WASR).

4.2 Experimental Results

Following the evaluations by GCG [58], we perform evaluations in the settings of query-specific adversarial suffix generation and universal adversarial suffix generation. For query-specific adversarial suffix generation, following [58], we use first 100 harmful behaviors in AdvBench. For universal adversarial suffix generation, we use the first 10 harmful queries in AdvBench to generate a universal adversarial suffix and test it on the rest 510 harmful queries in AdvBench. We also test our method for universal adversarial suffix generation on the standard behaviors set from HarmBench [32], following the setting suggested in HarmBench. We use a Top-k selection of 4 and a candidate set size of 20 for Llama and Mistral models, and a Top-k selection of 64 and a candidate set size of 160 for Phi3-Mini-4K-Instruct. Since Llama-2-Chat [46] models show great robust performance to gradient-based adversarial prompt generation [32], we mainly evaluate the methods on the model of Llama-2-7B-Chat [46] and the model of Llama-2-13B-Chat [46]. In addition, Mistral-7B-Instruct-v0.2 [19] and Phi3-Mini-4K-Instruct [1] are also considered.

The comparison results in the setting of query-specific adversarial suffix generation are shown in Table 1. Experimental results demonstrate that both LSGM and LILA outperform the GCG attack on three safety-aligned LLMs. Our combination method further boosts both the match rates and the attack success rates, achieving the best performance. Specifically, GCG-LSGM-LILA achieves gains of $+30\%$, $+19\%$, $+19\%$, and $+21\%$ when attacking the Llama-2-7B-Chat and Llama-2-13B-Chat, Mistral-7B-Instruct, and Phi3-Mini-4K-Instruct, respectively. Moreover, since these methods reduce the gap between the input gradients and the effects of loss change results from token replacements, the size of candidate set at each iteration can be reduced for saving running time. We show the results of GCG with the default setting described in their paper, which evaluates 512 candidate adversarial

suffixes at each iteration. It can be seen that our combination still shows outstanding performance, by only using 4% time cost compared with the GCG with their initial setting (denoted as GCG* in the table).

Table 1: Match rates, attack success rates, and time costs for generating query-specific adversarial suffixes on AdvBench are shown. The symbol * indicates the use of the default setting for GCG, *i.e.*, using a Top-k of 256 and a candidate set size of 512. Time cost is derived by generating a single adversarial suffix on a single NVIDIA V100 32GB GPU.

Method	Llama-2-7B-Chat			Llama-2-13B-Chat			Mistral-7B-Instruct			Phi3-Mini-4K-Instruct		
	MR	ASR	Time	MR	ASR	Time	MR	ASR	Time	MR	ASR	Time
GCG*	60%	44%	85m	58%	40%	170m	**95%**	92%	85m	70%	61%	50m
GCG	54%	38%	3m	37%	33%	6m	73%	74%	3m	60%	59%	17m
GCG-LSGM	72%	62%	3m	52%	43%	6m	93%	88%	3m	75%	64%	17m
GCG-LILA	70%	59%	3m	52%	48%	6m	83%	80%	3m	65%	61%	17m
GCG-LSGM-LILA	**87%**	**68%**	3m	**62%**	**52%**	6m	94%	**93%**	3m	**81%**	**68%**	17m

The results of the attacks in the setting of universal adversarial suffix generation are shown in Table 2. It can be observed that our combination not only achieves a remarkable improvement in the average ASR but also enhances both the worst and best ASRs obtained over 10 runs. Specifically, when attacking Llama-2-7B-Chat model on AdvBench, the GCG-LSGM-LILA achieves an average ASR of 60.32%, which gains a $+33.64\%$ improvement compared with the GCG attack. Moreover, for the worst and best ASR, GCG-LSGM-LILA achieves gains of $+34.82\%$ and $+31.06\%$, respectively.

Table 2: Attack success rates for generating universal adversarial suffixes on AdvBench and HarmBench. The average ASR (AASR), the worst ASR (WASR), and the best ASR (BASR) are obtained by performing each attack ten times.

Dataset	Model	GCG			GCG-LSGM-ILA		
		AASR	WASR	BASR	AASR	WASR	BASR
AdvBench	Llama-2-7B-Chat	26.68%	0.40%	55.80%	**60.32%**	**35.22%**	**86.86%**
	Llama-2-13B-Chat	20.98%	0.00%	37.06%	**45.27%**	**7.45%**	**67.25%**
	Mistral-7B-Instruct	56.53%	34.51%	92.16%	**73.48%**	**50.80%**	**92.25%**
	Phi3-Mini-4K-Instruct	35.84%	20.39%	46.27%	**44.80%**	**31.18%**	**61.18%**
HarmBench	Llama-2-7B-Chat	56.90%	33.00%	66.50%	**69.35%**	**57.50%**	**87.00%**
	Llama-2-13B-Chat	37.40%	13.50%	64.50%	**53.55%**	**22.00%**	**81.50%**
	Mistral-7B-Instruct	75.00%	37.50%	90.50%	**81.00%**	**66.50%**	**93.00%**
	Phi3-Mini-4K-Instruct	49.90%	29.00%	65.50%	**63.80%**	**48.50%**	**80.50%**

We also test the transfer attack performance of our method. We use the universal suffixes generated by performing GCG and GCG-LSGM-LILA against Llama-2-7B-Chat to attack GPT-3.5-Turbo on 100 harmful queries in AdvBench. The test queries are distinct from the queries that are used to generate universal suffixes. The results are shown in Table 3. It can be observed that our GCG-LSGM-LILA achieves remarkable improvements in the average, worst, and best ASRs obtained over 10 runs.

Table 3: The performance of transfer attack against GPT-3.5-Turbo on AdvBench. The average ASR (AASR), the worst ASR (WASR), and the best ASR (BASR) are obtained by performing each attack ten times.

Method	AASR	WASR	BASR
GCG	38.30%	24%	48%
GCG-LSGM-LILA	**45.20%**	**35%**	**81%**

5 Conclusions

In this paper, we present a new perspective on the discrete optimization problem in gradient-based adversarial prompt generation. That is, using gradient *w.r.t.* the input to reflect the change in loss that results from token replacement resembles using input gradient calculated on the substitute model to indicate the real effect of perturbing inputs on the prediction of a black-box victim model,

which has been studied in transfer-based attacks against image classification models. By making some appropriate adaptations, we have appropriated the ideologies of two transfer-based methods, namely, SGM and ILA, into the gradient-based white-box attack against LLMs. Our analysis of the mechanisms behind their effective performance has provided new insights into solving discrete optimization problem within LLMs. Furthermore, the combination of these methods can further enhance the discrete optimization in gradient-based adversarial prompt generation. Experimental results demonstrate that our methods significantly improves the attack success rate for both white-box and transfer attacks.

References

[1] Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

[2] Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Rimsky, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. *arXiv preprint arXiv:2406.11717*, 2024.

[3] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

[4] Nicholas Carlini, Milad Nasr, Christopher A Choquette-Choo, Matthew Jagielski, Irena Gao, Pang Wei W Koh, Daphne Ippolito, Florian Tramer, and Ludwig Schmidt. Are aligned neural networks adversarially aligned? *Advances in Neural Information Processing Systems*, 36, 2024.

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

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

[7] Gelei Deng, Yi Liu, Yuekang Li, Kailong Wang, Ying Zhang, Zefeng Li, Haoyu Wang, Tianwei Zhang, and Yang Liu. Jailbreaker: Automated jailbreak across multiple large language model chatbots. *arXiv preprint arXiv:2307.08715*, 2023.

[8] Xuanyi Dong and Yi Yang. Searching for a robust neural architecture in four gpu hours. In *CVPR*, 2019.

[9] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *CVPR*, 2018.

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

[11] Asma Ghandeharioun, Avi Caciularu, Adam Pearce, Lucas Dixon, and Mor Geva. Patchscope: A unifying framework for inspecting hidden representations of language models. *arXiv preprint arXiv:2401.06102*, 2024.

[12] Amelia Glaese, Nat McAleese, Maja Trębacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, et al. Improving alignment of dialogue agents via targeted human judgements. *arXiv preprint arXiv:2209.14375*, 2022.

[13] Martin Gubri, Maxime Cordy, Mike Papadakis, Yves Le Traon, and Koushik Sen. Lgv: Boosting adversarial example transferability from large geometric vicinity. *arXiv preprint arXiv:2207.13129*, 2022.

[14] Chuan Guo, Jacob R Gardner, Yurong You, Andrew Gordon Wilson, and Kilian Q Weinberger. Simple black-box adversarial attacks. *arXiv preprint arXiv:1905.07121*, 2019.

[15] Chuan Guo, Alexandre Sablayrolles, Hervé Jégou, and Douwe Kiela. Gradient-based adversarial attacks against text transformers. *arXiv preprint arXiv:2104.13733*, 2021.

[16] Yiwen Guo, Qizhang Li, and Hao Chen. Backpropagating linearly improves transferability of adversarial examples. In *NeurIPS*, 2020.

[17] Yiwen Guo, Qizhang Li, Wangmeng Zuo, and Hao Chen. An intermediate-level attack framework on the basis of linear regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[18] Qian Huang, Isay Katsman, Horace He, Zeqi Gu, Serge Belongie, and Ser-Nam Lim. Enhancing adversarial example transferability with an intermediate level attack. In *ICCV*, 2019.

[19] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

[20] Erik Jones, Anca Dragan, Aditi Raghunathan, and Jacob Steinhardt. Automatically auditing large language models via discrete optimization. In *International Conference on Machine Learning*, pages 15307–15329. PMLR, 2023.

[21] Tomasz Korbak, Kejian Shi, Angelica Chen, Rasika Vinayak Bhalerao, Christopher Buckley, Jason Phang, Samuel R Bowman, and Ethan Perez. Pretraining language models with human preferences. In *International Conference on Machine Learning*, pages 17506–17533. PMLR, 2023.

[22] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *ICLR*, 2017.

[23] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[24] Qizhang Li, Yiwen Guo, and Hao Chen. Yet another intermediate-leve attack. In *ECCV*, 2020.

[25] Qizhang Li, Yiwen Guo, Xiaochen Yang, Wangmeng Zuo, and Hao Chen. Improving transferability of adversarial examples via bayesian attacks. *arXiv preprint arXiv:2307.11334*, 2023.

[26] Qizhang Li, Yiwen Guo, Wangmeng Zuo, and Hao Chen. Improving adversarial transferability via intermediate-level perturbation decay. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[27] Qizhang Li, Yiwen Guo, Wangmeng Zuo, and Hao Chen. Making substitute models more bayesian can enhance transferability of adversarial examples. In *International Conference on Learning Representations*, 2023.

[28] Qizhang Li, Yiwen Guo, Wangmeng Zuo, and Hao Chen. Towards evaluating transfer-based attacks systematically, practically, and fairly. *arXiv preprint arXiv:2311.01323*, 2023.

[29] Ziang Li, Yiwen Guo, Haodi Liu, and Changshui Zhang. A theoretical view of linear backpropagation and its convergence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[30] Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. Autodan: Generating stealthy jailbreak prompts on aligned large language models. *arXiv preprint arXiv:2310.04451*, 2023.

[31] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. In *ICLR*, 2017.

[32] Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, et al. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. *arXiv preprint arXiv:2402.04249*, 2024.

[33] Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. Tree of attacks: Jailbreaking black-box llms automatically. *arXiv preprint arXiv:2312.02119*, 2023.

[34] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022.

[35] Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*, 2022.

[36] Muzammal Naseer, Kanchana Ranasinghe, Salman Khan, Fahad Shahbaz Khan, and Fatih Porikli. On improving adversarial transferability of vision transformers. *arXiv preprint arXiv:2106.04169*, 2021.

[37] nostalgebraist. interpreting gpt: the logit lens. *LessWrong*, 2020.

[38] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[39] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

[40] Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. Red teaming language models with language models. *arXiv preprint arXiv:2202.03286*, 2022.

[41] Rusheb Shah, Soroush Pour, Arush Tagade, Stephen Casper, Javier Rando, et al. Scalable and transferable black-box jailbreaks for language models via persona modulation. *arXiv preprint arXiv:2311.03348*, 2023.

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

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

[44] Chawin Sitawarin, Norman Mu, David Wagner, and Alexandre Araujo. Pal: Proxy-guided black-box attack on large language models. *arXiv preprint arXiv:2402.09674*, 2024.

[45] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

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

[47] Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. Universal adversarial triggers for attacking and analyzing nlp. *arXiv preprint arXiv:1908.07125*, 2019.

[48] Zhibo Wang, Hengchang Guo, Zhifei Zhang, Wenxin Liu, Zhan Qin, and Kui Ren. Feature importance-aware transferable adversarial attacks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7639–7648, 2021.

[49] Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does llm safety training fail? *Advances in Neural Information Processing Systems*, 36, 2024.

[50] Zeming Wei, Yifei Wang, and Yisen Wang. Jailbreak and guard aligned language models with only few in-context demonstrations. *arXiv preprint arXiv:2310.06387*, 2023.

[51] Zhipeng Wei, Jingjing Chen, Micah Goldblum, Zuxuan Wu, Tom Goldstein, and Yu-Gang Jiang. Towards transferable adversarial attacks on vision transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[52] Yuxin Wen, Neel Jain, John Kirchenbauer, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery. *Advances in Neural Information Processing Systems*, 36, 2024.

[53] Dongxian Wu, Yisen Wang, Shu-Tao Xia, James Bailey, and Xingjun Ma. Rethinking the security of skip connections in resnet-like neural networks. In *ICLR*, 2020.

[54] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille. Improving transferability of adversarial examples with input diversity. In *CVPR*, 2019.

[55] Zhihao Xu, Ruixuan Huang, Xiting Wang, Fangzhao Wu, Jing Yao, and Xing Xie. Uncovering safety risks in open-source llms through concept activation vector. *arXiv preprint arXiv:2404.12038*, 2024.

[56] Yi Zeng, Hongpeng Lin, Jingwen Zhang, Diyi Yang, Ruoxi Jia, and Weiyan Shi. How johnny can persuade llms to jailbreak them: Rethinking persuasion to challenge ai safety by humanizing llms. *arXiv preprint arXiv:2401.06373*, 2024.

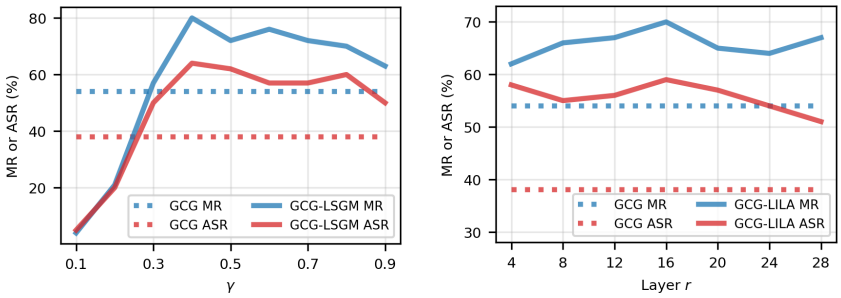
[57] Jianping Zhang, Weibin Wu, Jen-tse Huang, Yizhan Huang, Wenxuan Wang, Yuxin Su, and Michael R Lyu. Improving adversarial transferability via neuron attribution-based attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14993–15002, 2022.

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

A Experimental Settings

Hyper-parameters. For SGM, we simply set $\gamma = 0.5$. For the selection of intermediate representation to perform LILA, we choose the first token from the target phrase since it empirically performs better than selecting other tokens or all tokens except the adversarial suffix and the last input token. For the intermediate layer, we choose the midpoint of the model layers. For instance, for Llama-2-13B-Chat [46], we select the 20-th layer, whereas for other models with 32 layers, we select the 16-th layer. We perform 500 iterations for all methods, with the number of adversarial tokens set to 20.

B Ablation Study



(a) LSGM (b) LILA

Figure 8: How the match rate and attack success rate change with (a) the choice of γ for GCG-LSGM, (b) the choice of layer for GCG-LILA. Best viewed in color.

We evaluate GCG-LSGM with varying the choice of γ in Figure 8(a), and GCG-LILA with varying the choices of the layer in Figure 8(b), in the setting of query-specific adversarial suffix generation for attacking Llama-2-7B-Chat on AdvBench. It can be seen that for LSGM, a large range of the choice of γ ($0.3 \sim 0.9$) leads to improved performance. For LILA, it demonstrates consistently more effective performance on all choices of the layer.

C Limitations

We use a model-based evaluator provided by a benchmark, namely, HarmBench [32], to evaluate the success rates of attacks. Although it achieves a high human agreement rate ($> 93\%$), it cannot perfectly judge whether an attack is successful. More accurate evaluators will be adopted in future work to better evaluate attack performance.

D Broader Impacts

This work has improved the effectiveness of automatic adversarial prompt generation on LLMs. Specifically, on attacking Llama-2-Chat models, which are considered to be highly robust, our work achieves over $+30\%$ gains compared with GCG attack. This may aid in assessing of the robustness of safety-aligned LLMs, and could potentially be used in adversarial training method to improve the robustness of LLMs. Moreover, our research may also contribute to solving discrete optimization problem within LLMs, *e.g.*, prompt tuning.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and precede the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

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

   Answer: [Yes]

   Justification: Yes, the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

Justification: In Appendix.

Guidelines:

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

3. **Theory Assumptions and Proofs**

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

   Answer: [NA]

   Justification: The paper does not include theoretical results.

   Guidelines:

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

4. **Experimental Result Reproducibility**

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

   Answer: [Yes]

   Justification: We provide detailed experimental settings in Section 4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

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

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

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

   Answer: [No]

   Justification: No, but we will release the code as mentioned in Abstract.

   Guidelines:

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

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

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

   Answer: [Yes]

   Justification: We show the experimental settings in Section 4 and Appendix.

   Guidelines:

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

7. **Experiment Statistical Significance**

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

   Answer: [Yes]

   Justification: "We provided error bars in our experimental results for generating universal adversarial suffixes.

   Guidelines:

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

8. **Experiments Compute Resources**

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

   Answer: [Yes]

   Justification: We have provided information on the computing resources we used and the time cost of the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

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

9. **Code Of Ethics**

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

   Answer: [Yes]

   Justification: The research conducted in the paper conform with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader Impacts**

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

    Answer: [Yes]

    Justification: In Appendix.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [NA]

    Justification: N/A

Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Yes.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

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

    Answer: [NA]

    Justification: N/A

    Guidelines:

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

14. **Crowdsourcing and Research with Human Subjects**

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

Answer: [NA]

Justification: N/A

Guidelines:

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

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

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

Answer: [NA]

Justification: N/A

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.