Goal-guided Generative Prompt Injection Attack on Large Language Models

Chong Zhang[1], Mingyu Jin[1], Qinkai Yu[2], Chengzhi Liu[1], Haochen Xue[1], Xiaobo Jin[1†]
[1]Xi'an Jiaotong-Liverpool University, [2]University of Liverpool
Email: Chong.zhang19@student.xjtlu.edu.cn, Xiaobo.jin@xjtlu.edu.cn

***Abstract*—Current large language models (LLMs) provide a strong foundation for large-scale user-oriented natural language tasks. Numerous users can easily inject adversarial text or instructions through the user interface, thus causing LLM model security challenges. Although there is much research on prompt injection attacks, most black-box attacks use heuristic strategies. It is unclear how these heuristic strategies relate to the success rate of attacks and thus effectively improve model robustness. To solve this problem, we redefine the goal of the attack: to maximize the KL divergence between the conditional probabilities of the clean text and the adversarial text. Furthermore, we prove that maximizing the KL divergence is equivalent to maximizing the Mahalanobis distance between the embedded representation x and x' of the clean text and the adversarial text when the conditional probability is a Gaussian distribution and gives a quantitative relationship on x and x'. Then we designed a simple and effective goal-guided generative prompt injection strategy (G2PIA) to find an injection text that satisfies specific constraints to achieve the optimal attack effect approximately. Notably, our attack method is a query-free black-box attack method with a low computational cost. Experimental results on seven LLM models and four datasets show the effectiveness of our attack method.**

***Index Terms*—Prompt Injection, KL-divergence, LLM, Mahalanobis Distance.**

I. Introduction

Large Language Models (LLMs) [1], [2] are evolving rapidly in architecture and applications. As they become more and more deeply integrated into our lives, the urgency of reviewing their security properties increases. Many previous studies [3], [4] have shown that LLMs whose instructions are adjusted through reinforcement learning with human feedback (RLHF) are highly vulnerable to adversarial attacks. Therefore, studying adversarial attacks on large language models is of great significance, which can help researchers understand the security and robustness of large language models [5]–[7] and thus design more powerful and robust models to prevent such attacks.

Various strategies have been developed to attack language models, categorized into white-box and black-box approaches. White-box methods, such as GBDA [8], HotFlip [9], and AutoPrompt [10], use gradient-based techniques to optimize adversarial loss but face challenges with closed-source models. Black-box attacks often involve token manipulation, as seen in SEAR [11] and EDA [12], with BERT-Attack [13] employing context-aware replacements. Our approach focuses on inserting adversarial prompts instead of merely altering words. Additionally, prompt injection attacks exploit vulnerabilities in large language models [14]–[16], aiming to expose or redirect system prompts. The black-box paradigm also includes methods like BadNets [17] and model substitution [18]. A key drawback of white-box attacks is their limitation to open-source models; they are ineffective against widely used closed-source LLMs like ChatGPT due to lack of access to model architecture and parameters. Black-box strategies employ heuristic methods due to the unknown internal structures of large models, yet the relationship between these heuristics and attack success rates remains unclear, highlighting the need for more effective strategies.

In our work, we assume that the clean text representation x and the adversarial text representation x' satisfy the conditional probability distribution $p(y|x)$ and $p(y|x')$ respectively, and the goal of the black-box attack is to maximize the KL divergence $\mathrm{KL}(p(y|x), p(y|x'))$, then we prove that maximizing the KL divergence is equivalent to maximizing the Mahalanobis distance between x and x' under the assumption of Gaussian distribution. Furthermore, we give the quantitative relationship between optimal attack text representation x'^* and x. Based on the above theoretical results, we designed a simple and effective prompt text injection method to search for attack texts that meet approximately optimal conditions.

Overall, our contributions are as follows: **1)** We propose a new objective function based on KL divergence between two conditional probabilities for black-box attacks to maximize the success rate of black-box attacks; **2)** We theoretically prove that under the assumption that the conditional probabilities are Gaussian distributions, the KL divergence maximization problem based on the posterior probability distributions of clean text and adversarial text, respectively, is equivalent to maximizing the Mahalanobis distance between clean text and adversarial text. **3)** We propose a simple and effective injection attack strategy for generating adversarial injection texts, and the experimental results verify the effectiveness of our method. Note that our attack method is a query-free black-box attack method with low computational cost.

II. Methodology

A. Threat Model with Black-box Attack

1) Adversarial scope and goal.: Given a text t containing multiple sentences, we generate a text t' to attack the language

† Corresponding Author. This work was partially supported by Research Development Fund with No. RDF-22-01-020, the "Qing Lan Project" in Jiangsu universities and National Natural Science Foundation of China under Grant U1804159.

2024 IEEE International Conference on Data Mining (ICDM) | 979-8-3315-0668-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICDM59182.2024.00119

2374-8486/24/$31.00 ©2024 IEEE
DOI 10.1109/ICDM59182.2024.00119

model, ensuring that the meaning of the original text t is preserved; otherwise, then we believe that the attack text t' is attacking another text that is unrelated to t. Here, we use $\mathcal{D}(t', t)$ to represent the distance between the semantics of text t and t'. If the LLM outputs $M(t)$ and $M(t')$ differ, then t' is identified as an adversarial input for M. Our objective is formulated as follows:

$$M(t) = r, \quad M(t') = r', \quad \mathcal{D}(r, r') \geq \varepsilon, \quad \mathcal{D}(t', t) < \varepsilon, \tag{1}$$

where the texts r and r' are the outputs of model M on text t and t' respectively, and r is also the groundtruth of text t. The distance function $\mathcal{D}(\cdot, \cdot)$ and the threshold ε approximately represent the semantic relationship between two texts. In particular, the above problem has the following attack characteristics

- **Effective:** The condition $\mathcal{D}(M(t'), r) \geq \varepsilon$ ensures that the model has a high attack success rate (ASR), while the condition $\mathcal{D}(M(t), r) < \varepsilon$ shows that the model has a high benign accuracy.
- **Imperceptible:** Prompt injection attacks can ensure that the adversarial text is better adapted to the problem context so that it is difficult for the model's active defense mechanism to detect the presence of our prompt injection attack.
- **Input-dependent:** Unlike fixed triggers, input-dependent triggers are imperceptible in most cases and difficult to be detected by humans [19]. According to the equation (1), the adversarial text (or trigger) t' is input-dependent, and thus the trigger t' is inserted into t via prompt injection to form an attack prompt (see Sec. II-D).

B. Analysis on Objective

For the convenience of discussion, we regard the text generation of LLM as a classification problem, where the output will be selected from thousands of texts and each text is regarded as a category. Below, we first discuss the necessary conditions for the LLM model to output different values ($M(t) \neq M(t')$) under the conditions of clean text t and adversarial text t', respectively.

Assume that the input t and output r of model M are both texts. Given two different input texts t and t', model LLM will output two different output texts r and r'. Assuming there is a bijective function w (or text embedding function) between text and vector, we have

$$x = w(t), \quad x' = w(t') \tag{2}$$

$$y = w(r), \quad y' = w(r') \tag{3}$$

where x (x') and y(y') are the embedded representations of the input text and the output text, respectively. Note that since the outputs of LLM r and r' come from an enumerable discrete space, and there is a one-to-one correspondence between text representation and vector representation, their vector representations y and y' are also enumerable, so the output of the LLM model M can be restated as the posterior probability maximization problem in the enumerable discrete space $\mathcal{Y}$

$$\begin{aligned} y &= \arg\max_{\hat{y} \in \mathcal{Y}} p(\hat{y}|x) = w(M(w^{-1}(x))), \\ y' &= \arg\max_{\hat{y} \in \mathcal{Y}} p(\hat{y}|x') = w(M(w^{-1}(x'))), \end{aligned} \tag{4}$$

where $w^{-1}(\cdot)$ is the inverse function of $w(\cdot)$ the function. Furthermore, we have

$$\forall \hat{y}, \quad p(\hat{y}|x) = p(\hat{y}|x') \Rightarrow \arg\max_{\hat{y} \in \mathcal{Y}} p(\hat{y}|x) = \arg\max_{\hat{y} \in \mathcal{Y}} p(\hat{y}|x'). \tag{5}$$

So, we get its converse proposition

$$\arg\max_{\hat{y} \in \mathcal{Y}} p(\hat{y}|x) \neq \arg\max_{\hat{y} \in \mathcal{Y}} p(\hat{y}|x') \Rightarrow \exists \hat{y}, \quad p(\hat{y}|x) \neq p(\hat{y}|x'). \tag{6}$$

Thus, we can derive the necessary condition for the LLM to output different values (such as $M(t) \neq M(t')$): the LLM has different posterior probability distributions under different input conditions. We maximize the Kullback-Leibler (KL) divergence between the posterior probability distributions $p(y|x)$ and $p(y|x')$ to maximize the likelihood of the LLM outputting different values

$$\max_{x'} \mathrm{KL}(p(y|x), p(y|x')). \tag{7}$$

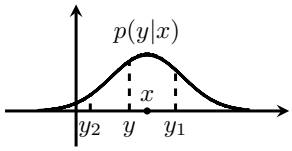

Fig. 1. Assumption that the output y of LLM under the condition of x satisfies the discrete Gaussian distribution: Answers (output) y close to question (input) x are usually more relevant to x and have a higher probability of being sampled.

First, we assume that the output distribution $p(y|x)$ of LLM satisfies the discrete Gaussian distribution [20] under the condition of input x,

$$p(y|x) = \frac{e^{-\frac{1}{2}(y-x)^T \Sigma^{-1}(y-x)}}{\sum_{\hat{y} \in \mathcal{Y}} e^{-\frac{1}{2}(\hat{y}-x)^T \Sigma^{-1}(\hat{y}-x)}}, \tag{8}$$

as shown in Fig. 1, that is, the output of LLM is defined on a limited candidate set $\mathcal{Y}$, although this candidate set $\mathcal{Y}$ may be very large. Since the input x and output y of LLM are questions and answers, the spaces where they are located generally do not intersect with each other, so there are $y = \arg\max_{\hat{y} \in \mathcal{Y}} p(\hat{y}|x) \neq x$, which is different from the commonly used continuous Gaussian distribution $y = x$, as can be seen in Fig. 1. For the same question x, LLM usually outputs different answers and the answer y most relevant to x has a higher probability of being sampled. In the embedding space, the distance between y and x is usually closer. Similarly, answers y that are almost uncorrelated with x and far away from x in the embedding space have a smaller probability of being sampled.

To solve the problem (7) theoretically, we do the following processing: Replace the discrete Gaussian distribution with a

continuous Gaussian distribution to facilitate the calculation of KL divergence although the output text on condition of input text still obeys discrete distribution in practical applications. Subsequently, we present the following theorem (see Appendix Section A and B for details).

Theorem 1 Assuming $p(y|x)$ and $p(y|x')$ respectively follows the Gaussian distribution $\mathcal{N}_1(y; x, \Sigma)$ and $\mathcal{N}_2(y; x', \Sigma)$, then the maximization $KL(p(y|x), p(y|x'))$ is equivalent to maximizing the **Mahalanobis distance** $(x'-x)^T\Sigma^{-1}(x'-x)$, which is further transformed into the following minimization optimization problem given the clean input x

$$\min_{x'} \|x'\|_2, \quad s.t. \quad (x'-x)^T\Sigma^{-1}(x'-x) \leq 1, \tag{9}$$

which has an optimal solution of the form (λ is Lagrange multiplier)

$$x'^* = (\Sigma + \lambda I)^{-1}\lambda x, \quad \lambda > 0. \tag{10}$$

C. Solving Problem (9) Approximately via Cos Similarity

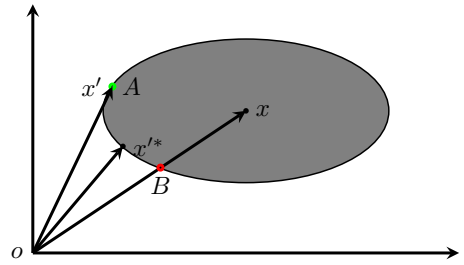

Fig. 2. Assuming $x'^* = (x'_1, x'_2)$ is the optimal solution to problem (9), then when x' moves from A through x'^* to B on the ellipse, $\cos(x', x)$ first increases and then decreases, while $\|x'\|_2$ first decreases and then increases.

Note that our method is a black-box attack and does not know the model parameters Σ, so we cannot solve the problem (9). Below, we try to approximately solve the problem (9) using cos similarity, which does not contain any parameters. Fig. 2 shows the optimal solution x'^* of the problem (9) in two-dimensional space. When the vector x' moves from A to B along the ellipse through the optimal point x'^*, $\cos(x', x)$ first increases and then decreases, while $\|x'\|_2$ first decreases and then increases. Therefore, we introduce the hyperparameter γ to approximate the solution to problem (9)

$$\cos(x', x) = \gamma, \quad 0 \leq \gamma \leq 1, \tag{11}$$

where x (known) and x' (unknown) are the embedded representations of clean and adversarial input, respectively. Note that in our implementation, we relax the constraint satisfaction problem of the optimal solution x'^* as

$$|\cos(x', x) - \gamma| < \delta, \quad \delta \text{ is a small positive constant.} \tag{12}$$

Below we discuss the problems of $\gamma \neq 0$ and $\gamma \neq 1$ from two perspectives. First, we prove this conclusion mathematically. We compute $\cos(x'^*, x)$ to obtain

$$\cos(x'^*, x) = \frac{x'^{*T}x}{\|x'^*\|_2\|x\|_2} = \frac{\lambda x^T(\Sigma + \lambda I)^{-1}x}{\|x'^*\|_2\|x\|_2}. \tag{13}$$

If $\cos(x'^*, x) = 0$ ($x \neq 0$), then $\lambda = 0$, that is, $x'^* = 0$, which is meaningless to LLM. If $\cos(x'^*, x) = 1$, then x'^* and x are in the same direction, i.e. $x'^* = \lambda(\Sigma + \lambda I)^{-1}x = tx$, where t is a ratio value. So x'^* must be the eigenvector of the matrix $\lambda(\Sigma + \lambda I)^{-1}$. However, x'^* can be an embedding representation of any input. So we arrive at a contradiction.

From the perspective of a black-box attack, when $\cos(x', x) = 1$, the vectors x and x' will have the same direction. When using vectors (often using unit vectors) to represent text, we care more about the direction of the vector, so $x = x'$. In addition, note that $w(\cdot)$ is a bijective function, then for clean text t and adversarial text t', there is $t = t'$. There we have $r = M(t) = M(t') = r'$, which contradicts the condition $r \neq r'$ in problem (1). When $\cos(x', x) = 0$, then $w(t')$ and $w(t)$ are linearly uncorrelated, which is conflicted with the condition $D(t', t) = 0$ in problem (1).

D. Goal-guided Generative Prompt Injection Attack

Note that $w(t) = x$ and $w(t') = x'$. Based on the previous discussion, we can simplify our problem (1) into the following constraint satisfaction problem (CSP)

$$\min_{t'} \quad 1, \tag{14}$$

$$s.t. \quad \mathcal{D}(t', t) < \epsilon, \tag{15}$$

$$|\cos(w(t'), w(t)) - \gamma| < \delta, \tag{16}$$

where x and x' represent clean input (known) and adversarial input (unknown), respectively, while $w(\cdot)$ represents the embedded representation of the text (literal meaning) and $\mathcal{D}(\cdot, \cdot)$ represents the distance between the semantics (intrinsic meaning) of the two texts. The hyperparameters δ and ϵ are used to control the difficulty of searching constraint, where δ or ϵ is smaller, the search accuracy is higher.

In our method, we implement a black-box attack through prompt injection: generate an adversarial text t' that satisfies conditions (16) and (15), and then mix t' into the text t to obtain a prompt $\bar{t}$. The advantage of using prompt insertion is that since the prompt $\bar{t}$ contains both clean input t and adversarial input t', on one hand, the concealment of the adversarial input (or trigger) t' is enhanced, and on the other hand the adversarial input t' plays a very good interference role to the output of the LLM model.

Next, we first find the core word set that determines the text semantics through the semantic constraint condition (15) and then use the core word set to generate adversarial text that satisfies the cos similarity condition (16). It is worth noting that the embedded representation is defined on **texts** , so we use the **BERT** model to convert any text t into $w(t)$. However, in our work, the semantic distance between texts is defined on the core **words** of the text, so we use the **word2vec** model to define the semantic distance $\mathcal{D}(t', t)$ between text t' and t.

E. Solving Semantic Constraint(15)

Usually, the semantic meaning of text r is determined by a few core words $C(r) = \{\omega_1(r), \omega_2(r), \cdots, \omega_n(r)\}$, which will not be interfered by noise words in the text r. Based on

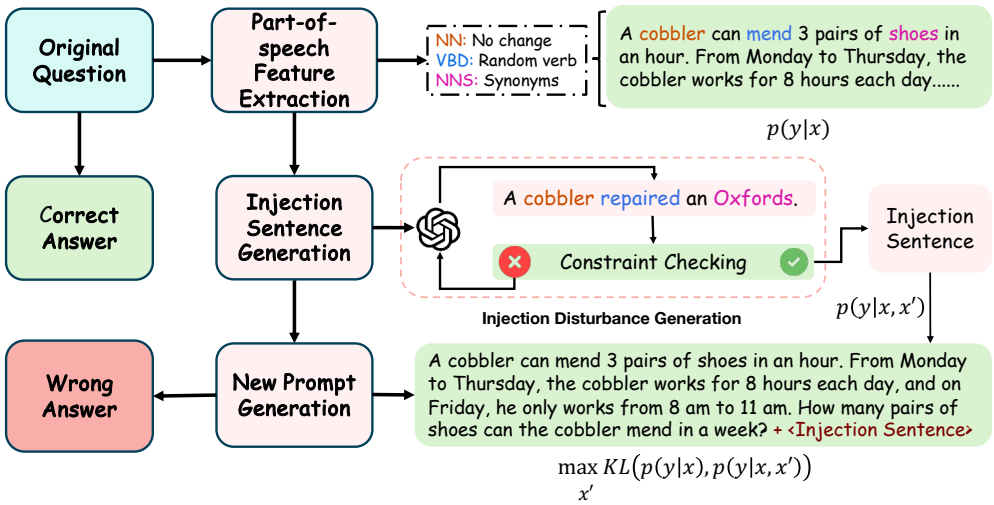

Fig. 3. Overview of Goal-guided Generative Prompt Injection Attack: 1) We use the part-of-speech method to find the subject, predicate and object of the question in the clean text x and fetch synonyms of the predicate and object plus a random number as core words; 2) Put the core words into assistant LLM to generate an adversarial text x' that satisfies the constraints; 3) Insert the generated adversarial text into the clean text x to form the final attack text; 4) Enter the attack text into the LLM victimization model to test the effectiveness of our attack strategy.

the core word set $C(r)$, we will use cos similarity to define the semantic distance $\mathcal{D}(t', t)$ between two texts t and t'

$$\mathcal{D}(t', t) = 1 - \cos(s(\omega_i(t')), s(\omega_i(t))), i = 1, 2, \cdots, n, \quad (17)$$

where $s(\cdot)$ represents the word2vec representation of the word.

In a text paragraph, usually, the first sentence is a summary of the entire paragraph (maybe some exceptions that we ignore), where the meaning of a text will be determined by its subject, predicate, and object. Therefore, the subject S_t, predicate P_t and object O_t that appear first in text t will serve as the core words of the text ($n = 3$)

$$\omega_1(t) = S_t, \quad \omega_2(t) = P_t, \quad \omega_3(t) = O_t. \quad (18)$$

Because the change of the subject will have a great impact on the meaning of the text, the subject $S_{t'}$ in the adversarial text t' is directly set to the subject S_t in the clean text t. Through the WordNet tool, we randomly select a word from the synonym lists of P_t or O_t to check whether the constraints (15) are met. Once the conditions are met, the search process will stop. Finally, we obtain the core word set of the adversarial text t' that satisfies the semantic constraints

$$C(t') = \{\omega_1(t') = S_{t'} = S_t, \omega_2(t') = P_{t'}, \omega_3(t') = O_{t'}\}. \quad (19)$$

F. Solving Cos Similarity Constraint (16)

Next, we will generate adversarial text that satisfies constraint (16) through the core vocabulary $C(t')$ of adversarial text. Note that to increase the randomness of the sentence, we add another random number $N_{t'}$ between 10 and 100 as the core word. Now the core vocabulary of the adversarial text t' becomes ($n = 4$)

$$C(t') = \{\omega_1(t') = S_{t'}, \omega_2(t') = P_{t'}, \omega_3(t') = O_{t'}, \omega_4(t') = N_{t'}\}. \quad (20)$$

The core word set is embedded into the prompt template to generate a sentence text t' with LLM. We iterate N times to randomly generate multiple sentence texts t' until the text t' satisfies Eqn. (16). Finally, we insert the adversarial text t' after the text t to attack the LLM. Inserting t' after any sentence in t is also feasible. In Appendix Section G, we will see minimal difference in attack effectiveness at different locations.

III. Experiments

A. Experimental Details

Below we describe some details of the prompt insertion-based attack method, including the victim model, dataset, and evaluation metrics. In particular, ChatGPT-4-Turbo (gpt-4-0125-preview) is used as our auxiliary model to generate random sentences that comply with grammatical rules. Unless otherwise stated, all results of our algorithm use the parameter settings $\epsilon = 0.2$, $\delta = 0.05$ and $\gamma = 0.5$. We randomly selected 300 examples from the following dataset and tested them using two large model families.

1) Victim Models:

- **ChatGPT.** ChatGPT, developed by OpenAI, is a language model capable of generating human-like conversations [21]. In our experiments, we use GPT-3.5-Turbo and GPT-4-Turbo as victim models.
- **Llama-2.** Llama-2 [22], by Meta AI, is an advanced open-source language model that surpasses its predecessor and other models in reasoning, encoding, proficiency, and knowledge tests. It includes Llama 2-7B, 13B, and 70B models based on the transformer framework.

2) Q&A Datasets: We chose datasets for plain text and mathematical Q&A scenarios.

- **GSM8K** The GSM8K dataset, consisting of 800 billion words [23], is the largest language model training resource available today.
- **web-based QA** The dataset [24] is mostly obtained from online Question Answering communities or forums through Web crawlers.

- **MATH dataset** The MATH dataset [25] has 12,000+ question-answer pairs for researchers to develop and evaluate problem-solving models.
- **SQuAD2.0** SQuAD2.0 [26] has 100K+ question-answer pairs from Wikipedia for reading comprehension.

3) Evaluation Metrics: Assume that the test set is D, the set of all question answer pairs predicted correctly by the LLM model f is T, and $a(x)$ represents the attack sample generated by the clean input. Then we can define the following three evaluation indicators

The results on four public datasets show that the first-generation ChatGPT-3.5 and ChatGPT-3.5-Turbo have the lowest defense capabilities. Obviously, when ChatGPT first came out, it didn't think too much about being attacked. Similarly, the small model 7b of Llama-2 is also very weak in resisting attacks. Of course, it is indisputable that the clean accuracy of the models of the Llama series is also very low. The output of small models is more susceptible to noise.

On the other hand, taking ChatGPT-4 as an example, if we compare the ASR values on the 4 data sets, we can conclude that our attack algorithm is more likely to succeed on the data set SQuAD 2.0, while mathematical problems are the most difficult to attack. In contrast to ASR 41.15 with ChatGPT-3.5 on GSM8k in the paper [27], our attack algorithm with ASR 44.87 is a general attack strategy and is not specifically designed for problems involving mathematical reasoning.

B. Comparison to Other Mainstream Methods

Below we compare our method with the current mainstream black-box attack methods in zero-sample scenarios on two data sets: SQuAD2.0 dataset [26] and Math dataset [25]. Microsoft Prompt Bench [28] uses the following black box attack methods to attack the ChatGPT-3.5 language model, including BertAttack [13], DeepWordBug [29], TextFooler [30], TextBugger [31], Prompt Bench [28], Semantic and CheckList [32]. For fairness, we also use our method to attack ChatGPT 3.5. Tab. II compares the results of these methods on the three measurements of Clean Acc, Attack Acc, and ASR.

Multiple attack strategies attack ChatGPT-3.5 on two data sets, the SQuAD2.0 dataset and the Math dataset, respectively. As seen from Tab. II, our attack strategy achieves the best results on both data sets. It is worth noting that we count Clean Acc and Attack Acc for each algorithm at the same time, so there are subtle differences between the multiple Clean Acc shown in Tab. II, but since Clean Acc and Attack Acc are calculated in the same attack algorithm, therefore it has little effect on the value of ASR. Especially on the Math dataset, our algorithm is significantly better than other algorithms, with an ASR of 44.87% compared to BertAttack's 33.46%. However, our algorithm is a general attack method not specifically designed for mathematical problems. To some extent, it is shown that our algorithm has good transfer ability on different types of data sets.

IV. Ablation Study

In this section, we will analyze our baseline approach by conducting ablation studies based on two strategies. The first strategy involves extracting sentence components, while the second involves traversing insertion positions. To extract sentence components, we randomly replaced all three components with synonyms. We also randomly performed an ablation study with random breakpoint insertion. The results show that the average ASRs of random location prompt injection and random sentence component replacement multiple times are lower than our method.

A. Parameter sensitivity analysis

In our method, the parameters ϵ and δ are two important parameters. The former controls the distance between the adversarial text and the clean text in the semantic space, while the latter will affect the optimality of the approximate optimal solution. We selected a total of 9 values from $\{0.1, 0.2 \cdots, 0.9\}$ for the two parameters to attack ChatGPT-3.5 on the GSM8K data set and calculated their ASR values. The ASR is simply a decreasing function of the distance threshold ϵ. That is, the farther the distance, the worse the attack effect. This aligns with our intuition: injected text that is too far away from the clean text will be treated as noise by LLM and ignored. The results show that when $\gamma = 0.5$, our attack strategy achieves the best attack effect. The attack effect will be somewhat attenuated when the gamma value exceeds 0.5 or less than 0.5. It is worth noting that the value of parameter γ may vary depending on the model or data. See the Appendix for more results and discussions.

V. Conclusion

In our work, we propose a new goal-oriented generative prompt injection attack (G2PIA) method. To make the injected text mislead the large model as much as possible, we define a new objective function to maximize, which is the KL divergence value between the two posterior conditional probabilities before injection (clean text) and after injection (attack text). Furthermore, we proved that under the condition that the conditional probability follows the multivariate Gaussian distribution, maximizing the KL divergence value is equivalent to maximizing the Mahalanobis distance between clean text and adversarial text. Then, we establish the relationship between the optimal adversarial text and clean text. Based on the above conclusions, we design a simple and effective attack strategy with an assisted model to generate injected text that satisfies certain constraints, maximizing the KL divergence. Experimental results on multiple public datasets and popular LLMs demonstrate the effectiveness of our method.

References

[1] OpenAI, Gpt-4 technical report (2023).

[2] M. Jin, Q. Yu, D. Shu, H. Zhao, W. Hua, Y. Meng, Y. Zhang, M. Du, The impact of reasoning step length on large language models, in: Findings of the Association for Computational Linguistics ACL 2024, 2024.

[3] T. Kaufmann, P. Weng, V. Bengs, E. Hüllermeier, A survey of reinforcement learning from human feedback, arXiv preprint arXiv:2312.14925 (2023).

TABLE I
COMPARISON OF ATTACK EFFECTS OF G2PIA ON DIFFERENT LLM MODELS AND DATASETS

Models	GSM8K			Web-based QA			SQuAD2.0 Dataset			Math Dataset		
	Clean	Attack	ASR ↑	Clean	Attack	ASR ↑	Clean	Attack	ASR ↑	Clean	Attack	ASR ↑
text-davinci-003	71.68	36.94	48.47	41.87	17.97	57.19	68.30	14.00	79.50	21.33	11.76	44.87
gpt-3.5-turbo-0125	72.12	37.80	47.60	41.98	24.17	42.42	68.33	12.67	81.46	21.33	15.99	29.72
gpt-4-0613	76.43	41.67	45.48	53.63	33.72	37.12	71.87	19.71	72.58	41.66	28.33	32.00
gpt-4-0125-preview	77.10	43.32	43.81	54.61	34.70	32.80	71.94	24.03	69.34	44.64	32.83	26.49
llama-2-7b-chat	44.87	27.51	38.69	47.67	24.26	49.10	78.67	37.66	52.13	79.33	52.44	33.90
llama-2-13b-chat	49.54	35.51	28.33	58.67	36.14	38.40	94.67	52.70	44.33	89.67	56.72	36.75
llama-2-70b-chat	56.48	39.90	29.36	70.20	48.18	31.47	93.33	40.78	56.31	94.67	71.82	24.14

TABLE II
OUR METHOD IS COMPARED WITH OTHER METHODS ON TWO DATASETS

Models	**Query**	**SQuAD2.0 dataset**			**Math Dataset**		
		$\mathcal{A}_{\text{clean}}$	$\mathcal{A}_{\text{attack}}$	$ASR\uparrow$	$\mathcal{A}_{\text{clean}}$	$\mathcal{A}_{\text{attack}}$	$ASR\uparrow$
BertAttack [13]	Dependent	71.16	24.67	65.33	22.27	14.82	33.46
DeepWordBug [29]	Dependent	70.41	65.68	6.72	22.07	18.36	16.83
TextFooler [30]	Dependent	72.87	15.60	78.59	21.71	16.80	26.02
TextBugger [31]	Both	71.66	60.14	16.08	21.73	17.75	18.31
Stress Test [32]	Free	71.94	70.66	1.78	21.33	19.59	8.15
CheckList [32]	Free	71.41	68.81	3.64	22.07	16.90	23.41
Ours	Free	68.30	14.00	79.50	21.33	11.76	44.87

TABLE III
ABLATION STUDIES ON DATASETS OF GSM8K AND WEB-BASED QA WITH GPT-3.5-TURBO (GPT-3.5-TURBO-0125).

Models	**GSM8K**			**Web-based QA**		
	$\mathcal{A}_{\text{clean}}$	$\mathcal{A}_{\text{attack}}$	ASR ↑	$\mathcal{A}_{\text{clean}}$	$\mathcal{A}_{\text{attack}}$	ASR ↑
Random position	72.12	51.08	29.18	41.98	36.20	14.20
Random component	72.12	58.90	18.33	41.98	37.16	11.49
Our Method	72.12	37.80	47.60	41.98	24.17	42.42

[4] M. Jin, S. Zhu, B. Wang, Z. Zhou, C. Zhang, Y. Zhang, et al., Attackeval: How to evaluate the effectiveness of jailbreak attacking on large language models, arXiv preprint arXiv:2401.09002 (2024).

[5] E. Shayegani, M. A. A. Mamun, Y. Fu, P. Zaree, Y. Dong, N. Abu-Ghazaleh, Survey of vulnerabilities in large language models revealed by adversarial attacks (2023). arXiv:2310.10844.

[6] M. Jin, Q. Yu, J. Huang, Q. Zeng, Z. Wang, W. Hua, H. Zhao, K. Mei, Y. Meng, K. Ding, et al., Exploring concept depth: How large language models acquire knowledge at different layers?, arXiv preprint arXiv:2404.07066 (2024).

[7] Q. Zeng, M. Jin, Q. Yu, Z. Wang, W. Hua, Z. Zhou, G. Sun, Y. Meng, S. Ma, Q. Wang, et al., Uncertainty is fragile: Manipulating uncertainty in large language models, arXiv preprint arXiv:2407.11282 (2024).

[8] C. Guo, A. Sablayrolles, H. Jégou, D. Kiela, Gradient-based adversarial attacks against text transformers (2021). arXiv:2104.13733.

[9] J. Ebrahimi, A. Rao, D. Lowd, D. Dou, Hotflip: White-box adversarial examples for text classification (2018). arXiv:1712.06751.

[10] T. Shin, Y. Razeghi, R. L. L. I. au2, E. Wallace, S. Singh, Autoprompt: Eliciting knowledge from language models with automatically generated prompts (2020). arXiv:2010.15980.

[11] M. T. Ribeiro, S. Singh, C. Guestrin, Semantically equivalent adversarial rules for debugging NLP models, in: ACL, 2018.

[12] J. Wei, K. Zou, Eda: Easy data augmentation techniques for boosting performance on text classification tasks (2019). arXiv:1901.11196.

[13] L. Li, R. Ma, Q. Guo, X. Xue, X. Qiu, BERT-ATTACK: adversarial attack against BERT using BERT, in: EMNLP, 2020, p. .

[14] I. R. McKenzie, A. Lyzhov, M. Pieler, A. Parrish, A. Mueller, A. Prabhu, E. McLean, A. Kirtland, A. Ross, A. Liu, et al., Inverse scaling: When bigger isn't better, arXiv preprint arXiv:2306.09479 (2023).

[15] F. Perez, I. Ribeiro, Ignore previous prompt: Attack techniques for language models (2022). arXiv:2211.09527.

[16] Y. Liu, G. Deng, Y. Li, K. Wang, T. Zhang, Y. Liu, H. Wang, Y. Zheng, Y. Liu, Prompt injection attack against llm-integrated applications (2023). arXiv:2306.05499.

[17] T. Gu, B. Dolan-Gavitt, S. Garg, Badnets: Identifying vulnerabilities in the machine learning model supply chain, arXiv:1708.06733 (2017).

[18] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, A. Swami, Practical black-box attacks against machine learning, in: ACCC, 2017.

[19] E. Wallace, S. Feng, N. Kandpal, M. Gardner, S. Singh, Benchmarking language models' robustness to semantic perturbations, in: NAACL, 2022.

[20] C. L. Canonne, G. Kamath, T. Steinke, The discrete gaussian for differential privacy (2021).

[21] A. Radford, J. Wu, R. Child, D. Luan, A. B. Santoro, S. Chaplot, A. Patra, I. Sutskever, Chatgpt: A language model for conversational agents, OpenAI (2020).

[22] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al., Llama: Open and efficient foundation language models. corr, abs/2302.13971, 2023. doi: 10.48550, arXiv preprint arXiv.2302.13971 (2023).

[23] T. Brown, J. Chinchilla, Q. V. Le, B. Mann, A. Roy, D. Saxton, E. Wei, M. Ziegler, Gsm8k: A large-scale dataset for language model training (2023). arXiv:2302.06066.

[24] Y. Chang, M. Narang, H. Suzuki, G. Cao, J. Gao, Y. Bisk, Webqa: Multihop and multimodal qa, in: CVPR, 2022, pp. 16495–16504.

[25] D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, J. Steinhardt, Measuring mathematical problem solving with the math dataset, NeurIPS (2021).

[26] P. Rajpurkar, R. Jia, P. Liang, Know what you don't know: Unanswerable questions for squad, arXiv preprint arXiv:1806.03822 (2018).

[27] Z. Zhou, Q. Wang, M. Jin, J. Yao, J. Ye, W. Liu, W. Wang, X. Huang, K. Huang, Mathattack: Attacking large language models towards math solving ability, arXiv preprint arXiv:2309.01686 (2023).

[28] K. Zhu, J. Wang, J. Zhou, Z. Wang, H. Chen, Y. Wang, L. Yang, W. Ye, N. Z. Gong, Y. Zhang, et al., Promptbench: Towards evaluating the robustness of large language models on adversarial prompts, arXiv preprint arXiv:2306.04528 (2023).

[29] J. Gao, J. Lanchantin, M. L. Soffa, Y. Qi, Black-box generation of adversarial text sequences to evade deep learning classifiers, in: SPW, 2018, p. .

[30] D. Jin, Z. Jin, J. T. Zhou, P. Szolovits, Is bert really robust? a strong baseline for natural language attack on text classification and entailment, in: AAAI, 2020.

[31] J. Li, S. Ji, T. Du, B. Li, T. Wang, Textbugger: Generating adversarial text against real-world applications, arXiv preprint arXiv:1812.05271 (2018).

[32] M. T. Ribeiro, T. Wu, C. Guestrin, S. Singh, Beyond accuracy: Behavioral testing of nlp models with checklist, arXiv preprint arXiv:2005.04118 (2020).