Towards Effective Prompt Stealing Attack against Text-to-Image Diffusion Models

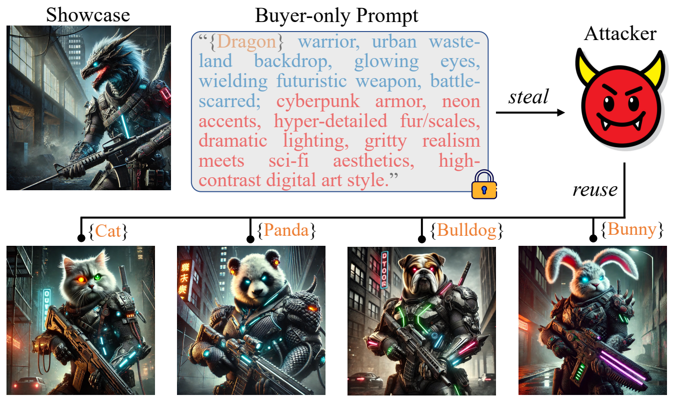
Shiqian Zhao[†], Chong Wang[†✉], Yiming Li[†], Yihao Huang[‡], Wenjie Qu[‡], Siew-Kei Lam[†], Yi Xie[§], Kangjie Chen[†], Jie Zhang[††✉], Tianwei Zhang[†]
[†]Nanyang Technological University, Singapore
[‡]National University of Singapore, Singapore
[§]Tsinghua University, China
[††]A*STAR, Singapore

***Abstract*—Text-to-Image (T2I) models, represented by DALL·E and Midjourney, have gained huge popularity for creating realistic images. The quality of these images relies on the carefully engineered prompts, which have become valuable intellectual property. While skilled prompters showcase their AI-generated art on markets to attract buyers, this business incidentally exposes them to *prompt stealing attacks*. Existing state-of-the-art attack techniques reconstruct the prompts from a fixed set of modifiers (*i.e.,* style descriptions) with model-specific training, which exhibit restricted adaptability and effectiveness to diverse showcases (*i.e.,* target images) and diffusion models.**

To alleviate these limitations, we propose Prometheus, a training-free, proxy-in-the-loop, search-based prompt-stealing attack, which reverse-engineers the valuable prompts of the showcases by interacting with a local proxy model. It consists of three innovative designs. First, we introduce *dynamic modifiers*, as a supplement to static modifiers used in prior works. These dynamic modifiers provide more details specific to the showcases, and we exploit NLP analysis to generate them on the fly. Second, we design a *contextual matching* algorithm to sort both dynamic and static modifiers. This offline process helps reduce the search space of the subsequent step. Third, we interact with a local proxy model to invert the prompts with a greedy search algorithm. Based on the feedback guidance, we refine the prompt to achieve higher fidelity. The evaluation results show that Prometheus successfully extracts prompts from popular platforms like PromptBase and AIFrog against diverse victim models, including Midjourney, Leonardo.ai, and DALL·E, with an ASR improvement of 25.0%. We also validate that Prometheus is resistant to extensive potential defenses, further highlighting its severity in practice. Our code is available at https://github.com/Shiqian-Zhao996/Prometheus.

I. Introduction

With the emergence of diffusion models [48], [10], text-to-image generation services, such as DALL·E [25] and Midjourney [22], have received widespread popularity due to their remarkable performance. Since diffusion models are often sensitive to user inputs, crafting high-quality images requires carefully designed prompts, which typically consist of precise *subjects* (main object) and diverse *modifiers* (style descriptions) [8], [20], [26] as shown in Figure 1. The high demand for high-quality prompts has led to the rise of numerous



"{Dragon} warrior, urban wasteland backdrop, glowing eyes, wielding futuristic weapon, battle-scarred; cyberpunk armor, neon accents, hyper-detailed fur/scales, dramatic lighting, gritty realism meets sci-fi aesthetics, high-contrast digital art style."

Fig. 1. A prompt and its corresponding images. The blue phrase is the subject, and the red phrases are the modifiers. After the attacker steals the prompt, they can reuse it by replacing the subject (in brown). This prompt is from the commercial prompt market PromptBase [29], whose showcase is generated with DALL·E 3.

trading platforms, such as PromptBase [29], PromptSea [34], and PromptHero [31], where prompt engineers publish their high-quality prompts, and users purchase their desired prompts by browsing the corresponding showcase images. For example, PromptBase hit 10K registered users by November 2022 [50], only 4 months after its establishment in July 2022 [51]. *After purchasing the prompt, the buyer can modify and customize it to generate tailored outputs, as well as scale the quantity of images to suit various augmentation needs.*

Prompts sold on the trading platforms often command high prices (*e.g.*, 15 US dollars for the Portrait Neon Lights prompt from Promptr [33]). Thus, they can be considered a form of valuable intellectual property (IP). Their great commercial value spawns potential threats to IP protection, and one prominent example is prompt stealing attack [46], which has already attracted attention from the industry [44]. As shown in Figure 1, given a showcase that is exhibited on the prompt market, an attacker tries to recover the corresponding prompt used by the publisher to create the showcase. *Then the attacker can adapt the prompt to his own tasks, including customization or scaling quantity.* For example, he can reuse the prompt to generate more images by simply replacing the subjects with customized ones. Despite the infringement on intellectual

✉ Corresponding authors: Chong Wang <chong.wang@ntu.edu.sg>, Jie Zhang <zhangj6@a-star.edu.sg>

Network and Distributed System Security (NDSS) Symposium 2026
23 - 27 February 2026, San Diego, CA, USA
ISBN 979-8-9919276-8-0
https://dx.doi.org/10.14722/ndss.2026.241899
www.ndss-symposium.org

property, the attacker can subsequently upload the stolen prompt to other marketplaces for sale. These significantly compromise the commercial interests of both platforms and creators.

The essence of prompt-stealing attacks lies in accurately extracting the subjects and modifiers from a given showcase image. However, research on these attacks has significantly lagged behind their threat. We summarize existing prompt recovery techniques in Table I, where BLIP, CLIP-IG, and PH2P mostly focus on the subject, while PromptStealer [46] represents the only dedicated prompt-stealing attack that fully takes into account modifiers. PromptStealer takes the first dedicated step by (i) training a caption model for subject generation and (ii) training a classification model for modifier prediction. Although PromptStealer has demonstrated promising results, the following limitations reduce its practical penetration effect. First, PromptStealer relies on a fixed, predefined set of 7,672 modifiers as classification labels. This approach restricts its semantic coverage, resulting in *out-of-vocabulary (OOV) issues* for modifiers and limiting its ability to describe the highly diverse showcases. Second, the predictions for subject and modifiers rely entirely on the caption and classification models, which are trained with the prompt-side text semantics while ignoring the perception and semantics from the recovered images. This may lead to prediction error, especially when the trained models are *overfitted to the training set* (*i.e.*, the Lexica dataset [15] collected from Stable Diffusion [39]).

To alleviate these limitations, we propose Prometheus, an effective and practical prompt-stealing attack methodology. Our key insights are twofold. First, we propose **dynamic modifiers** that are specific to the target showcase, serving as the complement of the predefined but insufficient static modifiers used in prior works. Second, to address the overfitting caused by text-guided pretraining, we incorporate a **training-free proxy-in-the-loop** mechanism with feedback to iteratively refine the modifiers. Specifically, Prometheus is designed to include three key modules. (1) *Dynamic modifier extraction*. Prometheus first leverages a caption model to generate lots of image captions for the showcase, which provides abundant showcase-specific modifiers. We then analyze these captions with an NLP analysis tool (*e.g.,* Spacy [49]), to extract these modifiers from the captions. (2) *Contextual matching*. These dynamic modifiers, along with the predefined static modifiers, are concatenated to the subject and ranked based on our proposed contextual matching, which considers the fidelity gain of a modifier in a contextual manner. This action helps to shorten the list of candidate modifiers and further improves the efficiency of subsequent fine-grained sampling. (3) *Greedy `Proxy` query*. Finally, Prometheus greedily constructs the final prompt by sequentially adding modifiers to the base prompt (starting with the subject and gradually expanding) and leverages the feedback from a local proxy model to carefully choose the modifiers with the most gain effect. We construct a *multi-objective score function*, *i.e., semantic* and *perception*, to comprehensively evaluate the contribution of a modifier.

We conduct comprehensive experiments to assess Prometheus. First, large-scale case studies reveal that it effectively steals prompts that are designed for popular commercial platforms like Midjourney [22], and DALL·E [25], from prompt markets like PromptBase [29] and AIFrog [4], highlighting its *functionality* in practical scenarios. For instance, on real-world prompts, Prometheus achieves image and prompt fidelity scores of 0.912 and 0.814, respectively, surpassing the baselines by margins of up to 0.142 and 0.200. Importantly, our method realizes an attack success rate improvement of 25.0% compared with the state-of-the-art. Moreover, Prometheus achieves the best *reusability* when shifting the subject in the recovered prompt. Further mitigation experiments show that our attack is *resistant to various potential defenses*, including random noise, puzzle effect, text watermark, and adaptive mitigation, highlighting its practicality and severity.

TABLE I
EXISTING AND POTENTIAL PROMPT-STEALING ATTACKS.

Method	Static Modifier	Dynamic Modifier	Feedback
BLIP [17]	✘	✘	✘
CLIP-IG [28]	✔	✘	✘
PH2P [21]	✘	✘	✘
PromptStealer [46]	✔	✘	✘
Prometheus (Ours)	✔	✔	✔

In summary, our main contributions are as follows:

- We review existing prompt-stealing attacks against text-to-image models and reveal their poor adaptability. We identify that this limitation arises from the over-preparation.
- We propose a novel and effective training-free prompt-stealing attack, dubbed Prometheus. It generates modifiers on the fly and leverages the feedback from `Proxy` to optimize the prompt effectively.
- We propose contextual matching, which is inspired by in-context learning. This mechanism helps sort out the high-correlation modifier to facilitate prompt stealing.
- We evaluate Prometheus on extensive prompts collected from commercial prompt markets, *e.g.,* PromptBase and AIFrog. The result shows that Prometheus can steal prompts with high fidelity and is resistant to potential defenses.

Responsible Disclosure: We obtained prior consent from both the prompt engineers and the platforms for the scientific use of the displayed prompt showcases and the collected data. They have expressed great interest in and support for our work. Upon acceptance, we will share the research findings with them, along with mitigation methods for such attacks. We will open-source the code and collected dataset for community use.

II. Background and Related Work

A. Text-to-image Models

Text-to-image (T2I) models [48], [10] are a new emerging technology for high-quality image generation with text description as a condition, also known as *prompt*. During the generation process, a diffusion model is commonly adopted, *e.g.*, U-Net [40] to recover the desirable image from random noise, with the guidance of prompts. Specifically, these models take the text embedding of the prompt from a fixed text encoder (usually the text encoder of CLIP [35]). Then this embedding is fed into the diffusion model to predict the step-wise noise with the cross-attention mechanism [52], [36]. Popular T2I models include commercial ones like DALL·E [25], Midjourney [22], Imagen [42] and open-source Stable Diffusion [39].

B. Prompt-as-a-Service

Despite the remarkable generation ability of T2I models, performing prompt engineering requires a significant amount of effort and cost. For instance, PromptPerfect charges up to \$100 per month for its prompt optimization service [32]. Under these circumstances, Prompt-as-a-Service (PaaS) has emerged as a popular application. Professional prompt engineers optimize prompts with their expertise and sell them on the prompt markets, *e.g.*, PromptBase [29], AIFrog [4], PromptSea [34], PromptHero [31]. These on-sell prompts are sightlessly exhibited with their exquisite showcases, which visually indicate the prompt effects. After purchasing, the user gains access to its corresponding textual description, which can be used to generate customized images or increase the number of showcased outputs.

C. Prompt Stealing Attacks

Several methods can be potentially used to investigate the feasibility of prompt-stealing attacks (PSA). A straightforward strategy is image captioning via a caption model. For example, Li et al. [17] train an image-to-text model, BLIP, to predict the prompt of an image. However, as shown in [8], [20], [26], a high-quality prompt should consist of a subject and modifiers. For this reason, the open-source project CLIP-IG [28] considers adding modifiers into the subject by combining highly correlated phases from five kinds of modifiers (*i.e.,* medium, artist, trending, movement, and flavor). In a more related work, Shen et al. [46] propose to utilize a multi-head model ML-Decoder [38] to detect the modifier contained in the showcase. To train such a predictor, they collect a dataset called Lexica-Dataset. Some approaches attempt to reverse-engineer prompts using token-level optimization techniques [53], [21], where a soft prompt is first derived and then projected into a hard prompt. However, the resulting prompts often lack semantic coherence, limiting their reusability. Additionally, these methods typically assume white-box access to the generation model, an unrealistic assumption in closed-source settings. More analysis can be found in Section V-B.

Despite their progress in promoting the performance of PSA, these works suffer from two defects. First, they lack feedback from text-to-image models. This means that they rely on a predefined pattern, which may lead to an overfitting problem and thus poor reusability. Second, a lot of manual labor is required in the loop, including collecting the dataset and training additional models, which significantly affects the attack efficiency and cost.

III. Problem Statement

A. Threat Model

We follow the same threat model in [46], as detailed below.

Attacker's Capability. We make two assumptions about the attacker's capabilities. (1) The attacker can access the generated showcases of the prompt (denoted as s). This is typical in commercial platforms, where prompt sellers display showcase images to attract potential buyers. However, the specific prompt used to generate the showcase remains hidden from all users until they purchase it. A detailed example can be found in Portrait Neon Lights [33]. (2) As some commercial victim models run in a pay-as-you-go mode, which may be expensive, we assume the attacker can use a substitute local text-to-image (T2I) model, referred to as the proxy model. Since the proxy model can be any open-source model, the query budget is 0.

Adversary's Motivation. The attacker aims to compromise the confidentiality of valuable prompts sold on the markets. A carefully designed prompt consists of *subjects* and *modifiers* (as shown in Figure 1). Specifically, subjects are the main body of a prompt that outlines an image's most important elements, while modifiers detail the style of the subjects and the whole picture. Formally, given a showcase image s, an adversary aims to recover the prompt p that is used to create the showcase, including both its subjects and modifiers. We provide more analysis about the motivation in Appendix A-A.

From a **utility perspective**, stolen prompts can be reused for customization, *e.g.,* replacing subjects or increasing the number of showcases. From a **commercial standpoint**, as discussed in previous work [46], the motivation behind prompt stealing can be twofold. First, it may *infringe on the intellectual property (IP) of high-quality prompts*, which is the result of significant effort by prompt engineers. Second, attackers have strong *economic incentives*: they can avoid payment by using these premium prompts for free and may also redistribute or resell the prompts on other markets for additional profit. This poses a significant threat to the commercial interests of both prompt developers and the platforms hosting them, which is explicitly prohibited by platform PromptBase [30]. From a **privacy protection standpoint**, this type of attack poses *a significant threat to the confidentiality of prompts, particularly when they contain sensitive or highly classified information.* We provide more analysis for this standpoint in Appendix A-B.

B. Attack Requirements

For the attack to be considered successful, the attacker must meet the following requirements:

- **Objective-1: Functionality [Output Perception Similarity]**. The stolen prompt should achieve a high reproduction in image appearance to the showcase.
- **Objective-2: Reusability [Subject Shift]**. The recovered prompt possesses high reusability so that when swapping the main subject, it still generates images with similar appearances.

Attack Comparisons. We follow the existing line of work [17], [28], [46], and focus on reversing readable prompts, which could satisfy these two objectives. We observe that certain hard prompt recovery methods could serve as potential solutions. For instance, PH2P [21] leverages gradient signals to optimize

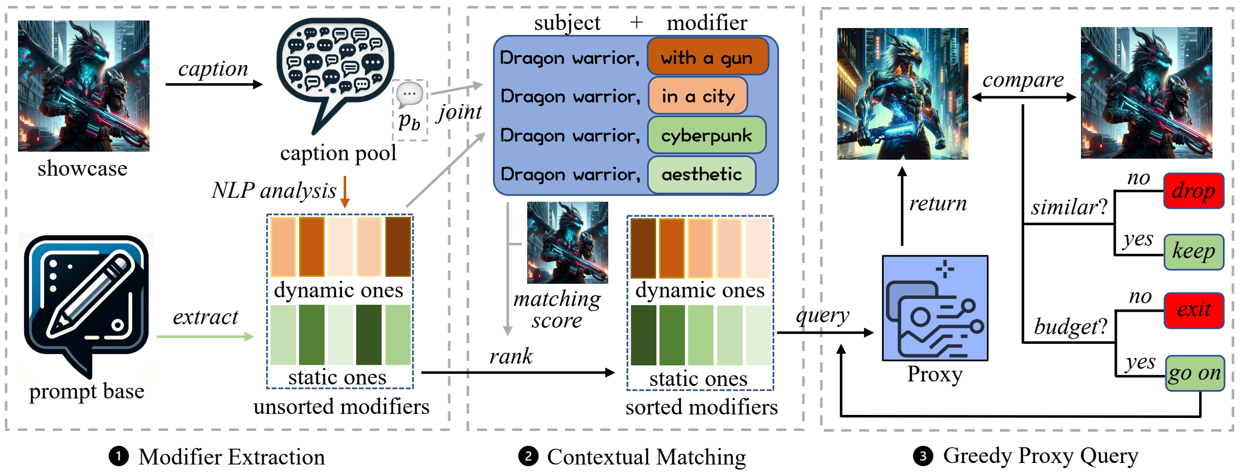


Fig. 2. Overall pipeline of Prometheus. In general, Prometheus consists of three main components: ❶ Modifier Extraction; ❷ Contextual Matching; and ❸ Greedy `Proxy` Query. Prometheus starts with a showcase and public prompt base, which provides static and dynamic modifiers. Then, contextual matching is utilized to rank and shortlist the unsorted modifiers. Prometheus interactively queries a local `Proxy` model and computes the fidelity gain for each modifier (as defined in Equation 2). This gain serves as feedback to guide the refinement of the modifiers.

a soft prompt, which is then projected back into a hard prompt. However, since semantic coherence is not taken into account during optimization, the recovered prompts often fall into a local optimum and consist of unreadable or nonsensical words. Another drawback of hard prompt recovery methods is that they search modifier candidates within the entire discrete token space, which is rather inefficient. For example, when recovering a 40-word prompt from a 50,000-token vocabulary, PH2P would consider $50,000^{40}$ combinations, which is computationally infeasible. As a result, they fail to meet the aforementioned objectives, limiting the practical utility of the extracted prompts. We list the merits of our Prometheus over the existing works in Table I. Besides static modifiers, we also include *dynamic modifiers* and *feedback from the proxy model*, to steal prompts with higher fidelity from a meaningful modifier set efficiently.

IV. Prometheus

A. Overview

We introduce Prometheus, an advanced prompt-stealing attack to overcome the limitations of existing works.

- **Showcase-specific Dynamic Modifiers.** In addition to the fixed static modifiers used in the previous work [46], we propose to generate a pool of *dynamic modifiers* for target showcases on the fly, to provide more comprehensive descriptions.
- **Proxy-in-the-loop Feedback.** We propose to leverage `Proxy`'s feedback to optimize the prompt for the desired effect, *i.e.,* showcase. The attacker queries the `Proxy` with a combined modifier pool. Based on the `Proxy`'s feedback, *i.e.,* each modifier's fidelity gain to the showcase s, the attacker decides to keep or discard the modifier.

Figure 2 depicts the workflow of Prometheus. Given a showcase, it employs a captioning model (*e.g.,* BLIP) to generate a caption as the initial subject. It then strategically enhances this subject by adding a set of modifiers to create the final prompt. Specifically, Prometheus consists of the following three key steps to achieve the requirements of *functionality* and *reusability* as stated in Section III-B.

1. Modifier Extraction (Section IV-B). First, we construct a comprehensive and semantic modifier pool to ensure functionality and reusability. We satisfy these requirements by considering two kinds of modifiers: 1) *Static modifiers*, which are extracted from a luxuriant prompt base in prior work [46], comprising prompt engineers' prior knowledge; 2) *Dynamic modifiers*, which are extracted from the target showcase's captions, and generated on the fly. They serve as a supplement to fill in more showcase-specific details dynamically.

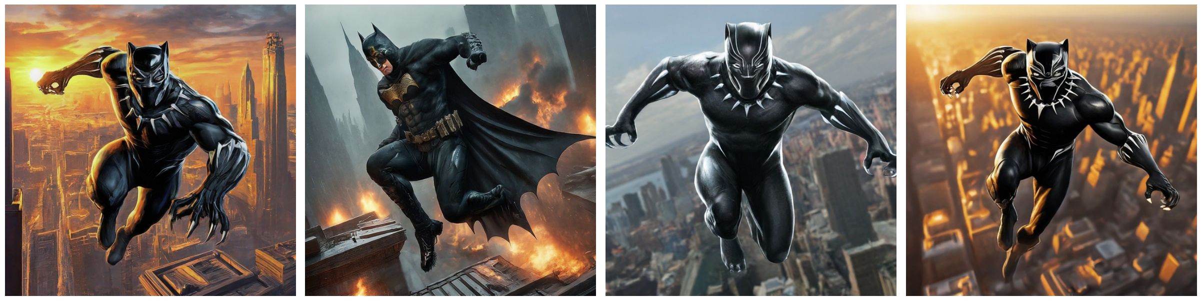
2. Contextual Matching (Section IV-C). After obtaining the comprehensive modifier list, it is inefficient to query all of these modifiers to select the desired ones. Therefore, we *refine them to narrow down the search space*. Considering the modifiers' decorative effect on the subject, we propose a *contextual matching* algorithm to refine these modifiers for streamlining the sample space. We verify that this intuitive matching method could achieve a more consistent ranking of the modifiers.

3. Greedy `Proxy` Query (Section IV-D). Third, we optimize the prompts to achieve higher functionality. Since the local ranking from contextual matching may not fully align with the target, it is crucial to perform additional refinement based on feedback from the `Proxy`. We employ the *greedy search algorithm* to enhance the modifiers further. Specifically, for each prompt candidate, we query the `Proxy` and compute the similarity between its output and the showcase. Using the similarity gain as a criterion, we decide whether to retain or discard each modifier. This greedy approach allows us to identify the most effective modifiers. The entire interaction completes within $\mathcal{Q}$ turns.

Below we provide details of each step in Prometheus.

B. Modifier Extraction

Previous works [28], [46] primarily focus on adopting the overall style of a showcase as the prompt modifier. These



(a) Ground Truth: Black Panther leaping across rooftops in a bustling city, with skyscrapers and streets below, during a sunset (b) Base Prompt: A hero in all black costume flying over a city (c) Base Prompt + Detail: A hero in all black costume flying over a city, Black Panther (d) Base Prompt + Detail + Position: A hero in all black costume flying over a city, Black Panther, with skyscrapers and streets below, during a sunset

Fig. 3. An illustration of dynamic modifiers. The base prompts in (b)–(d) are generated by the caption model BLIP [17] based on the showcase. As shown, the caption model fails to capture detailed subject and position information. Additionally, in the transition from (a) to (b), these omissions result in an image that differs from the showcase. However, this issue can be mitigated by incorporating detailed subject and position information, as demonstrated in (c) and (d). The evaluation model used is Imagen [42] from Gemini.

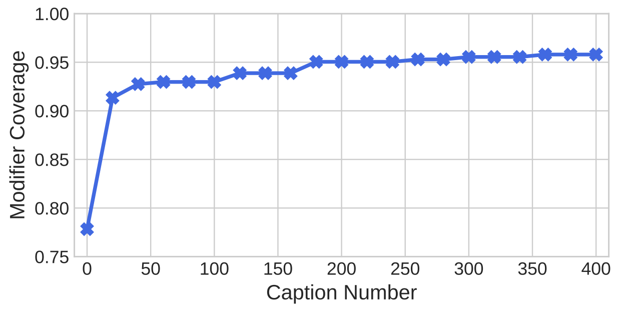


Fig. 4. Modifier coverage with caption number.

are referred to as *static modifiers* because they are extracted from a fixed set of prompts and remain largely unchanged across images. Typically, they rely on the captioning model BLIP [17] to extract the main subjects of showcases. However, as illustrated in Figure 3, such captioning models often fail to capture the details of a given showcase s with sufficient accuracy [46], which may lead to *unmatched image generation*. For example, BLIP describes "Black Panther" merely as a "hero in all black costume", which can easily be interpreted as "Batman" by a text-to-image model. Also, *spatial cues* such as "during a sunset" can offer vital information about scene layout, yet are often missed.

To address this, we introduce two additional image elements, including *subject detail* and *position information*, to enrich the image description. In this paper, these are collectively referred to as **dynamic modifiers**, as they vary with each showcase. Formally, the **dynamic modifiers** are defined as the subject details (*e.g.*, identity and specified property) and the position information (*i.e.*, the whole layout of showcase), which are beyond the static style modifiers.

Dynamic Modifier. We present the pseudocode (Appendix A-C) along with the following introduction to detail dynamic modifier extraction. We leverage the zero-shot capability of caption model to extract dynamic modifiers. Specifically, given a showcase, we first generate multiple prompts with a local BLIP model. This caption model could generate captions with large variances under a high sampling temperature, finding all the potential descriptions. Then, for each sampled caption, we utilize sentence analysis techniques to extract *nominal and prepositional phrases*. Specifically, we utilize Spacy [49] to analyze the prompt for *Parts of Speech (PoS)* and construct a parent-child tree for each word. For subject detail, we extract all the noun chunks in the prompt, including single nouns (*e.g.*, "Black Panther") and noun phrases (*e.g.*, "red car"). For position information, the extraction process is much more complicated. First, Prometheus goes through one caption to locate all the preposition words. If one word is a preposition, then a prepositional phrase is constructed based on its relation tree. Take the phrase "during a sunset" as an example: when Prometheus detects that "during" is a preposition, the relation tree of "during" is extracted, *e.g.,* "a" is the child of "sunset" and "a sunset" is the parent of "during". Finally, we obtain a prepositional phrase by appending the child to its corresponding parent. We do not adopt an object detector as it can only recognize the category of a subject (*e.g.,* car) while *ignoring fine-grained details* (*e.g.,* the color of a car). Also, it cannot get the *layout* of an image, *i.e.,* position information.

We validate the effectiveness of dynamic modifiers in Figure 4. As illustrated, incorporating dynamic modifiers extracted from the caption pool significantly enhances modifier coverage. Specifically, static modifiers alone account for 77.8% coverage. When dynamic modifiers are introduced, coverage increases to 95.8% as the caption pool expands. This improvement demonstrates that *dynamic modifiers effectively compensate for the limitations of both the BLIP model and static modifiers.*

C. Contextual Matching

After obtaining the modifiers, Prometheus ranks them by their correlation to the showcase. This ranking process occurs locally to narrow the search space for more cost-effective `Proxy` queries. An intuitive way is to apply CLIP to calculate the modifiers' semantic similarity to the showcase [28]. However, we argue that, as modifiers are appendants to a base prompt, modifier correlation alone does not indicate the gain to prompt fidelity. Therefore, we have to rethink the effect

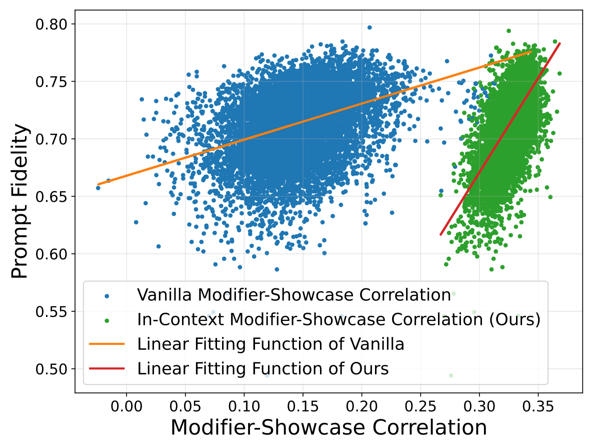


Fig. 5. Comparison between our contextual modifier ranking method and the baseline. The slopes of fitted linear functions are the PCC.

contextually. Driven by this, we propose a new correlation ranking method, namely *Contextual Matching*, to measure a modifier's fidelity gain to the subject comprehensively.

Specifically, given a showcase s and a base prompt p_b, we calculate the contextual matching score of modifier m with the following score function $\mathcal{F}_{score}$:

$$\mathcal{F}_{score} = \mathcal{D}(\mathcal{E}_T(p_b + m), \mathcal{E}_I(s)) - \mathcal{D}(\mathcal{E}_T(p_b), \mathcal{E}_I(s)), \quad (1)$$

where $\mathcal{E}_T$ and $\mathcal{E}_I$ are the text encoder and image encoder of CLIP, respectively. The distance function $\mathcal{D}$ is cosine similarity. As CLIP is trained to build a correlation between texts and images, this metric could reflect the matching degree of a modifier in context.

We validate the superiority of our contextual modifier ranking method using *Pearson Correlation*. We calculate the semantic similarity between contextual modifiers and showcase as $\mathcal{D}(\mathcal{E}_T(p_b + m), \mathcal{E}_I(s))$, as well as the prompt similarity between $\hat{p}$ and ground truth p as $\mathcal{D}(\mathcal{E}_T(p), \mathcal{E}_T(\hat{p}))$. Then we calculate the Pearson Correlation Coefficient (PCC) between these two similarities. We compare our contextual matching with the vanilla matching method [28], which directly calculates the similarity between the modifier and showcase. As shown in Figure 5, our contextual matching presents *a more positive correlation*. More specifically, our method achieves a PCC of 0.65, while the vanilla method only achieves 0.37, indicating that *our method has a more consistent trend with the victim*. We present a more detailed ablation study in Section V.

D. Greedy Proxy Query

From the first two steps, we obtain two lists of modifier candidates for the dynamic and static ones, respectively. These candidates are generated, ranked, and refined using BLIP [17] and CLIP [35]. However, due to the knowledge gap between these models and the T2I generative model, it is not practical to directly select the most probable modifiers from these lists (truncation). Simply choosing modifiers based on relevance ranking is insufficient, as the achieved PCC of 0.65 by contextual matching is still inadequate, though

Algorithm 1 GreedyProxyQuery

Input: base prompt p_b, showcase s, sorted modifier set $\mathcal{M}$, fidelity gain threshold δ, proxy model $\mathcal{O}$, and allocated query budget $\mathcal{Q}$.
Output: recovered prompt $\hat{p}$.

```
1:  $s_b \leftarrow \mathcal{O}(p_b)$                               ▷ Query the proxy model
2:  for $q$ in $range(\mathcal{Q})$ do
3:      $p_q \leftarrow$ joint $\mathcal{M}[q]$ to $p_b$          ▷ Joint candidate to base prompt
4:      $s_q \leftarrow \mathcal{O}(p_q)$                         ▷ Query the proxy model
5:      $\Delta(\mathcal{M}[q]) = Sem(s_b, s_q, s) + Per(s_b, s_q, s)$
6:      if $\Delta(\mathcal{M}[q]) > \delta$ then                 ▷ If the threshold is reached
7:          $p_b = p_q$
8:          $s_b = s_q$                                           ▷ Update base prompt and best score
9:      end if
10: end for
11: $\hat{p} \leftarrow$ assign $p_b$ as final prompt
12: return $\hat{p}$
```

it already outperforms the baseline ranking method. We argue that truncating the top-k modifiers risks under- or overslicing, which can lead to suboptimal recovery performance. To address this, we propose leveraging feedback from a local proxy model to *greedily search* for modifiers that provide the largest fidelity gain. Specifically, the attacker $\mathcal{A}$ interacts with the Proxy by iteratively extending the base prompt, adding one modifier at a time. Modifiers are then selected based on the Proxy's feedback, *i.e.,* only those that enhance the quality of the final prompt $\hat{p}$ are retained. This iterative refinement process ensures maximum fidelity between the recovered prompt and the ground-truth prompt.

Score Function. To accurately assess the fidelity gain of a modifier, a comprehensive scoring function is essential. Intuitively, the perception of similarity between the recovered image (*i.e.,* the image generated using the stolen prompt) and the showcase can serve as a useful guide. However, because the showcase includes many random elements introduced by the inherent randomness of any T2I model, *relying too heavily on perception consistency risks severe overfitting*. Specifically, the stolen prompt may end up including additional modifiers beyond the ground truth to account for this randomness. To address this, we leverage *both semantic and perceptual guidance* to refine the modifiers. Let p_b represent the base prompt, which is updated to p_q with modifier m_q, and let the corresponding feedback from the Proxy $\mathcal{O}$ be denoted as $\mathcal{O}(p_b)$ and $\mathcal{O}(p_q)$, respectively. The fidelity gain is:

$$\Delta(m_q) = Sem(p_b, p_q, s) + Per(p_b, p_q, s). \quad (2)$$

Here, $fn = Sim(\mathcal{O}(p_q), s) - Sim(\mathcal{O}(p_b), s)$, where $fn \in \{Sem, Per\}$, and Sim represent the corresponding similarity metrics. Specifically, the similarity function for image semantics is the CLIP score [35], and that for image perception is LPIPS [56]. Note that the base prompt is dynamic, that is, when the fidelity gain $\Delta(m_q)$ meets the threshold δ, the base prompt is updated as $p_b + m_q$. Then it is regarded as the base prompt for the next modifier.

Greedy Proxy Query. We present our GreedyProxyQuery process in Algorithm 1, which contains four main steps:

1) Sample a modifier m as the order provided in Section IV-C and append it to the base prompt p_b;

TABLE II
PERFORMANCE OF PROMETHEUS COMPARED WITH BASELINE METHODS. THE MODELS REFERENCED HERE REPRESENT THE PROXY MODELS USED FOR PROMPT EXTRACTION. REALPROMPT CONSISTS OF PROMPTS SOLD ON COMMERCIAL MARKETS ($CLIP_{img}$:↑, $LPIPS$:↓, $SBERT$:↑, ASR :↑).

Dataset	Method	FLUX				ShuffleDiffusion				Stable Diffusion-3.5			
		$\texttt{CLIP}_{img}$	LPIPS	SBERT	ASR	$\texttt{CLIP}_{img}$	LPIPS	SBERT	ASR	$\texttt{CLIP}_{img}$	LPIPS	SBERT	ASR
DALLEPrompt	BLIP	0.694	0.776	0.721	24.7%	0.746	0.733	0.747	26.0%	0.729	0.773	0.734	15.3%
	CLIP-IG	0.798	0.745	0.729	23.3%	0.828	0.701	0.733	22.7%	0.822	0.723	0.736	14.7%
	PromptStealer	0.731	0.761	0.733	13.3%	0.767	0.725	0.734	28.7%	0.778	0.757	0.741	18.7%
	PH2P	0.542	0.836	0.531	0%	0.490	0.771	0.612	0%	0.656	0.751	0.599	0%
	VLMasExpert	0.785	0.726	0.746	26.7%	0.814	0.674	0.767	38.7%	0.818	0.703	0.757	29.3%
	Ours	**0.873**	**0.679**	**0.795**	**48.3%**	**0.890**	**0.601**	**0.803**	**56.0%**	**0.897**	**0.626**	**0.790**	**52.7%**
RealPrompt	BLIP	0.768	0.757	0.724	14.6%	0.745	0.740	0.672	8.3%	0.758	0.761	0.710	16.7%
	CLIP-IG	0.842	0.708	0.755	29.2%	0.848	0.712	0.775	35.4%	0.851	0.708	0.778	27.1%
	PromptStealer	0.771	0.731	0.765	35.4%	0.770	0.743	0.780	31.3%	0.781	0.762	0.781	37.5%
	PH2P	0.760	0.759	0.617	0%	0.815	0.719	0.598	0%	0.793	0.795	0.611	0%
	VLMasExpert	0.848	0.695	0.776	27.1%	0.858	0.688	0.748	29.2%	0.854	0.685	0.774	33.3%
	Ours	**0.901**	**0.653**	**0.814**	**58.3%**	**0.912**	**0.644**	**0.798**	**54.2%**	**0.901**	**0.625**	**0.799**	**62.5%**

2) Query p_b with the proxy model $\mathcal{O}$ and obtain feedback η as score function shown in Equation 1;
3) If η does not meet threshold δ, discard modifier m; otherwise, append m to p_b and obtain the new p_b;
4) Repeat Steps (1)-(3) until the budget is exhausted or the prompt fidelity requirement is met.

As dynamic modifiers serve as a supplement to the subject, we begin the searching process by examining the dynamic modifier list (m_d) before proceeding to the static modifier list (m_s). To ensure balanced consideration of both dynamic and static modifiers, we pre-allocate the budget between m_d and m_s in a 1:4 ratio. The final optimized prompt is constructed by combining the subject with both dynamic and static modifiers, as formulated below:

$$\hat{p} = p_b + m_d + m_s. \tag{3}$$

V. EVALUATION

A. Experiment Setup

Datasets. We consider two datasets: DALLEPrompt and RealPrompt. Each data point in these sets consists of a showcase and the corresponding prompt.

- **DALLEPrompt.** This dataset comprises prompts sourced from real-world prompts designed for DALL·E [9]. It spans various styles, including oil paintings, pixel art, and cyberpunk, as well as themes such as movie posters, book covers, and early 1900s newspapers. The prompts encompass a wide array of subjects, featuring characters (*e.g.*, Mickey Mouse, Darth Vader), objects (*e.g.*, marbles, boomboxes), and locations (*e.g.*, castles, Ancient Egypt). We generated showcases with these prompts using three well-performing models: FLUX [14], ShuttleDiffusion [47], and Stable Diffusion-3.5 [3] (SD-3.5). Finally, we obtained three sets containing prompt-showcase pairs (50 pairs for each).
- **RealPrompt.** This collection comprises a meticulously curated set of 24 prompt-showcase pairs, sourced from real-world commercial prompt markets including AIFrog [4] and PromptBase [29]. These pairs contain eight distinct themes, each showcasing a unique style. For every style, three images were generated by the sellers using prompts featuring different subjects. The average word prompt length is 32.08, with an average of 9.11 modifiers. The associated generation models include Midjourney [22] (1 set), Stable Diffusion [39] (3 sets), Leonardo.ai [2] (2 sets), and DALL·E [24] (2 sets).

Baselines. We consider the only prompt stealing method and four prompt recovery methods as baselines. The content in brackets denotes the original task.

- **BLIP** (*Captioning*) [17]: This is a captioning model trained on image-caption pairs from the COCO dataset [18]. Since the captions primarily focus on describing the subjects in an image, BLIP often exhibits suboptimal performance in capturing styles or modifiers.
- **CLIP-IG** (*Captioning*) [28]: This method builds upon BLIP by enhancing its capabilities. In addition to the subjects generated by BLIP, CLIP-IG selects a set of modifiers from a predefined large-scale modifier pool. These modifiers are ranked based on their semantic similarity to the showcase, with text-to-image similarity assessed using CLIP's encoders.
- **PromptStealer** (*Prompt Stealing*) [46]: This method is specifically designed for prompt stealing against text-to-image models. Given a showcase, it utilizes a fine-tuned BLIP model to generate the subject and employs a fine-tuned multi-head classifier to predict modifiers from a predefined modifier pool. The fine-tuning process relies on the Lexica prompt-image dataset, which was collected using the Stable Diffusion models.
- **PH2P** (*Prompt Inversion*) [21]: We also consider PH2P, a method that inverts the prompt with access to the encoder of Stable Diffusion v1.5. PH2P first performs token-level optimization on the soft prompt using gradients, and then projects the optimized soft prompt into a hard prompt.
- **VLMasExpert** (*VLM*): We include an additional baseline, Vision-LLM as an expert (VLMasExpert), in our study.

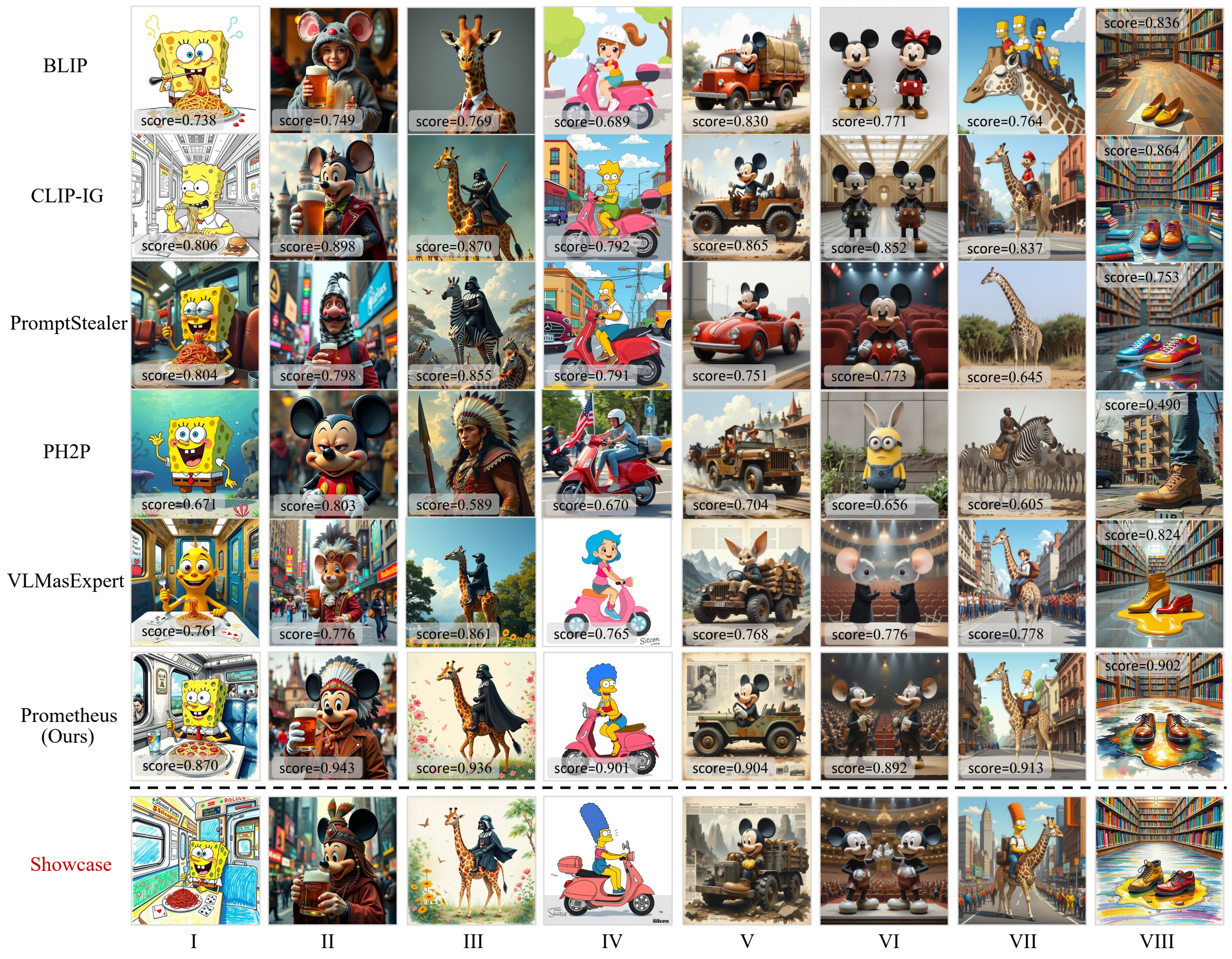


Fig. 6. Visualization of generated images compared with showcase. We provide eight different themes here. An example of the corresponding prompts can be found in Table III (case-VI). The *score* on the generations represents the CLIP score (↑), which indicates the semantic similarity between the generation and the showcase. Please zoom out for details.

Vision-LLM, trained on extensive datasets for image analysis, can be likened to a human expert with specialized knowledge. We utilize the powerful GPT-4o [12] to generate descriptions of the showcases.

Evaluation Metrics. We consider four metrics to evaluate the stolen prompt, including:

- **CLIP**$_{img}$: This metric evaluates the semantic similarity between generated image g and showcase s. It is calculated with image encoder $\mathcal{E}_{img}$ from CLIP [35] as:

$$CLIP_{img}(g,s) = \frac{\mathcal{E}_{img}(g)\cdot\mathcal{E}_{img}(s)}{\|\mathcal{E}_{img}(g)\|\|\mathcal{E}_{img}(s)\|},$$

  where a higher score indicates greater semantic similarity.
- **LPIPS**: The Learned Perceptual Image Patch Similarity (LPIPS) metric is designed to align closely with human perception [56]. It evaluates the visual similarity between two images using features extracted by neural network (*e.g.*, AlexNet backbone). We use LPIPS to assess the perceptual similarity between the recovered image and showcase. A lower LPIPS indicates better perception alignment.
- **SBERT**: SentenceBERT (SBERT) is an enhanced version of BERT [37], designed specifically for measuring sentence-level similarity. We use this metric to evaluate the semantic similarity between the stolen prompt $\hat{p}$ and the victim prompt p. Specifically, SBERT score functions as follows:

$$SBERT(\hat{p},p) = \frac{\mathcal{E}_{bert}(\hat{p})\cdot\mathcal{E}_{bert}(p)}{\|\mathcal{E}_{bert}(\hat{p})\|\|\mathcal{E}_{bert}(p)\|},$$

  where $\mathcal{E}_{bert}$ is the embedding extractor of SentenceBert.
- **ASR**: We also use the Attack Success Rate (ASR) as a metric to evaluate the effectiveness of attacks. Following the setting in [16], [7], [41], an attack is considered successful if the SBERT score is greater than 0.8 (*e.g.*, the semantics of $\hat{p}$ and p are similar enough).

We agree that no single metric fully captures similarity or usefulness, and selecting the top image by one metric can conceal weaknesses on others. Accordingly, for the three attacks, we do not pick a single "best" test by any one metric; instead, we report the average across all metrics. Besides these metrics, we also conducted a large-scale user study, which directly evaluates the human perception of the stolen effect.

TABLE III
EXAMPLES OF TARGET AND STOLEN PROMPTS GENERATED BY PROMETHEUS AND BASELINE METHODS. THE CORRESPONDING IMAGES ARE SHOWN IN FIGURE 7. THE BLUE ANNOTATIONS INDICATE THE PARTS THAT MATCH THE TARGET PROMPT.

Target prompt	Two Mickey Mice are talking in a concert hall; sculpture, hyperrealistic.
BLIP	Two figurines are positioned to Mickey Mouse.
CLIP-IG	Two Mickey Mouse statues standing in front of a large room, 3 d cartoon, lee madgwick &amp, disney pixar 3d style, pixar and disney 3d style...
PromptStealer	Mickey Mouse in a movie theater, artstation, highly detailed, sharp focus, 8k, octane render, 4k, cinematic, hd, unreal engine 5...
PH2P	minion minion bunny dnd " amas expectations monument
VLMasExpert	Two cartoon characters resembling anthropomorphic mice stand facing each other on a stage in a grand theater setting, both are wearing black outfits with large round ears and gloved hands, the background is filled with rows of empty seats in a spacious auditorium...
Prometheus (Ours)	Two Mickey Mice are talking on stage, orchestra, pack auditorium, Mickey Mouse statue, auditorium, hyper realism.

The detailed experimental design and results are presented in Section V-D.

Implementation Details. We implement Prometheus using Python 3.8 with PyTorch. All experiments are conducted on a single NVIDIA GeForce RTX A6000 GPU. The fidelity gain threshold in Equation 1 is set to 0.005. For BLIP used across all methods, we utilize the official implementation and adopt a ViT-based backbone, which is fine-tuned on the COCO dataset [19]. Unless otherwise specified, the `Oracle` query limit for Prometheus is set to 200, and the number of captions is set to 400. Note that, since we employ a local proxy model for feedback, the money cost is 0. To tackle the randomness within the attack process and image generation, we run each experiment three times.

B. Main Result

We first compare Prometheus with five baselines in terms of two objectives depicted in Section III-B: *functionality* and *reusability*. Additionally, we consider a practical scenario in which multiple showcases, each featuring different subjects, are provided.

Functionality. Table II shows the main experimental results. In general, Prometheus surpasses all the baselines on both DALLEPrompt and RealPrompt, demonstrating its functionality and practicality in real-world scenarios. Specifically, it achieves the best average scores: image semantic similarity (0.897), image perception similarity (0.601), prompt semantic similarity (0.803), and ASR (56.0%) on DALLEPrompt. Similarly, on RealPrompt, Prometheus leads with image semantic similarity (0.912), image perception similarity (0.625), prompt semantic similarity (0.814), and ASR (62.5%), demonstrating its effectiveness in real-world scenarios.

For image semantics and perception, Prometheus's improvements are mainly attributed to its prompt optimization with feedback from *Proxy*, while other methods only leverage local

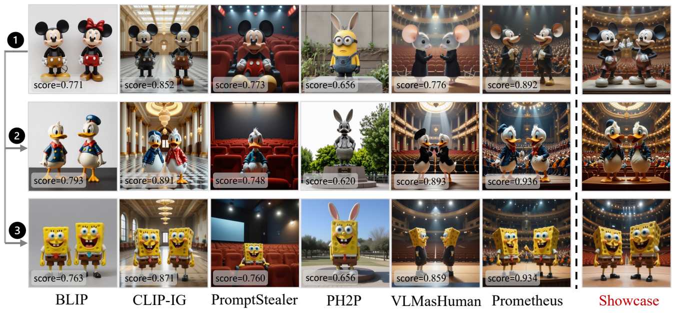


Fig. 7. Reusability of stolen prompts with different subjects. We swap all the subject-related words (❶ Mickey Mouse; ❷ Donald Duck; ❸ SpongeBob) to the target subject only. The corresponding prompts can be found in Table III. Please zoom out for details.

models to predict the prompt. This feedback-based refinement helps Prometheus generate images that are perceptually similar to the showcase. For prompt semantics, Prometheus also achieves better results on all datasets and proxy models. The main reason is that we improve the quality of recovered prompts by introducing dynamic modifiers and contextual matching. The former adds subject details as well as position information, and the latter ensures the sorted modifier achieves more consistent fidelity improvement when being put into the base prompt. Therefore, Prometheus obtains higher prompt semantic similarity with the victim prompt. We include a more detailed analysis of the performance gap between DALLEPrompt and RealPrompt in Appendix B-D.

TABLE IV
REUSABILITY OF PROMPTS (WITH DIFFERENT SUBJECTS) GENERATED WITH PROMETHEUS COMPARED WITH FIVE BASELINES. THE EVALUATION DATASET IS THE REAL-WORLD DATASET REALPROMPT.

Method	$CLIP_{img}(\uparrow)$	$LPIPS(\downarrow)$	$SBERT(\uparrow)$	$ASR(\uparrow)$
BLIP	0.755	0.765	0.749	29.2%
CLIP-IG	0.785	0.725	0.779	22.9%
PromptStealer	0.775	0.756	0.809	54.2%
PH2P	0.730	0.748	0.627	0%
VLMasExpert	0.767	0.715	0.801	47.9%
Prometheus (Ours)	**0.832**	**0.681**	**0.835**	**70.8%**

Figure 6 illustrates eight examples of the attack effect achieved by Prometheus compared to baseline methods. Additionally, Table III presents an example of the stolen prompts. Overall, compared to the baselines, Prometheus generates prompts and images with higher fidelity to the ground truths and showcases in terms of image semantics, visual perception, and prompt semantics. Take Figure 6 case VI as an example. The baseline BLIP struggles to recognize the style (*e.g.*, "hyperrealistic") and fails to capture the main content of the image (*e.g.*, "in a concert hall"). Meanwhile, CLIP-IG tends to match irrelevant modifiers to the showcase and introduces excessive redundancy. For instance, it ranks "3D cartoon" as having the highest correlation to the showcase (VI) and includes many unrelated "Disney" modifiers. These mismatches result in semantically incorrect image content and lower perceptual quality. This behavior can be attributed to CLIP-IG treating modifiers independently, leading to poorly contextualized predictions. PromptStealer exhibits two notable shortcomings.

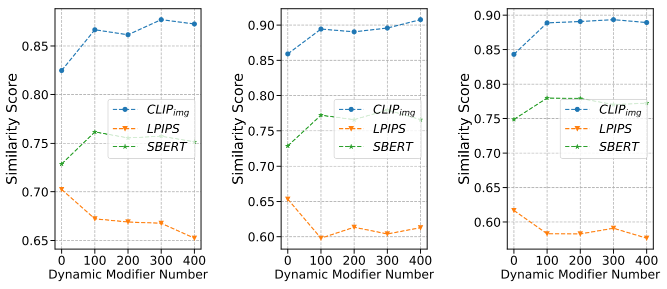
TABLE V
PERFORMANCE UNDER MULTIPLE SHOWCASE SCENARIOS. THE EVALUATION DATASET IS THE REAL-WORLD DATASET REALPROMPT.

Method	$CLIP_{img}(\uparrow)$	$LPIPS(\downarrow)$	$SBERT(\uparrow)$	$ASR(\uparrow)$
BLIP	0.778	0.760	0.760	27.1%
CLIP-IG	0.808	0.723	0.805	72.9%
PromptStealer	0.796	0.742	0.809	75.0%
PH2P	0.743	0.737	0.651	0%
VLMasExpert	0.803	0.715	0.801	58.3%
Prometheus (Ours)	**0.834**	**0.684**	**0.840**	**79.2%**

First, it demonstrates lower fidelity when predicting the subject (*e.g.*, Mickey Mouse in case II and The Simpsons in case VII). Second, as shown in Table III, PromptStealer often predicts generic modifiers that are frequent in its training data [46], rather than contextually appropriate ones. A likely explanation is that PromptStealer's prediction models are trained on out-of-domain data, such as that collected from Stable Diffusion, leading to overfitting. As for PH2P [21], this method fails to capture style modifiers and often even the subject. This highlights a fundamental limitation of using gradient-based prompt inversion for prompt stealing, indicating the need for further refinement to enable adaptation. Another baseline, VLMasExpert, performs relatively well but frequently omits key characters in the showcase. This limitation is likely due to alignment constraints during the model training and the out-of-distribution problem. We provide a more detailed analysis about the impact of alignment on its stealing performance in Appendix B-A. In contrast, Prometheus captures dynamic and contextually relevant modifiers, such as "auditorium" and "orchestra", which align with elements present in the showcase. As we can see, in the base prompt, Prometheus also ignores the "concert hall" element, but the dynamic modifier fixes this issue. Furthermore, Prometheus accurately identifies the modifier "hyper realism", demonstrating its outstanding prediction ability.

Reusability on Different Subjects. Table IV presents the comparison of reusability between Prometheus and baseline methods. Reusability is a crucial property, as it enables users to modify the subject of a prompt while maintaining similar styles in the generated images. To evaluate this, we use the RealPrompt dataset, which comprises 8 style sets (three subjects per style) to simulate real-world scenarios. For each stolen prompt, an LLM is employed to replace the subject, producing a new prompt with a different subject. This process is achieved using the following instruction with GPT-4o: "Please replace the central subject in the sentence prompt with subject while keeping the rest of the content unchanged". We validate the reusability of Prometheus and other baselines from both qualitative and quantitative angles. We present the experimental results in Table IV and visualization in Figure 7.

As shown in Table IV, Prometheus outperforms all baselines across metrics, including image semantics, image perception, prompt semantics, and ASR. This demonstrates that our method extracts the most reusable prompts while minimizing overfitting to specific showcases. We illustrate this in Figure 7, where the subject is replaced from "Mickey Mouse" to "Donald Duck" and "SpongeBob". As shown, Prometheus maintains



(a) FLUX (b) ShuttleDiffusion (c) SD-3.5

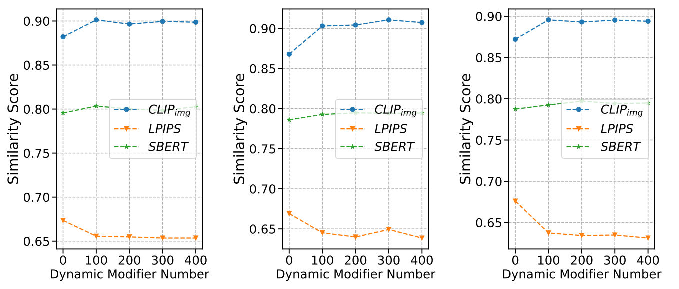
Fig. 8. Impact of the number of extracted dynamic modifiers. The dataset is DALLEPrompt ($CLIP_{img}$:↑, $LPIPS$:↓, $SBERT$:↑).

a consistent style that aligns closely with the ground truth outcomes, whereas the baselines exhibit several shortcomings. Specifically, BLIP, CLIP-IG, and PH2P fail to capture the "concert hall" element, resulting in images with incorrect semantics. PromptStealer partially captures the subject but introduces errors in quantity and mismatches static modifiers. In contrast, Prometheus achieves superior image semantics and perceptual similarity, demonstrating its effectiveness in generating reusable prompts.

Take-away: Prometheus outperforms the existing prompt stealing attack [46] and potential adaptations in both *functionality* and *reusability* with a large margin, indicating its high effectiveness.

Multiple Showcases. We consider a practical scenario where multiple showcases are provided under a single theme. In this case, the attacker extracts a prompt for each showcase, resulting in a set of prompts. These prompts are then merged into a distilled prompt. Here, we consider using a large language model (*i.e.,* ChatGPT-4o) to summarize the stolen prompts with the instruction "Please summarize these sentences into one sentence by rephrasing them. Replace $\{subjects\}$ or related words with $\{target\ subject\}$". The experimental results are presented in Table V, leading to two key observations. First, while Prometheus continues to outperform the baseline methods, the margin of improvement is less pronounced. A potential explanation is that the baseline methods compensate for inadequacies in single stolen prompts by leveraging the collective information from the entire prompt set, thereby reducing Prometheus's relative advantage. Second, compared to the results in Table II, the prompt semantic fidelity of Prometheus improves (from 0.814 to 0.840) after merging all the stolen prompts. This is likely because the summarization process mitigates overfitting to individual showcases, resulting in a stolen prompt that is better aligned with the ground truth.

Take-away: When given multiple showcases on the same theme, Prometheus outperforms the baselines and achieves better performance than in single-showcase scenarios.



(a) FLUX (b) ShuttleDiffusion (c) SD-3.5

Fig. 9. Impact of the number of extracted dynamic modifiers. The evaluation dataset is RealPrompt ($CLIP_{img}$:↑, $LPIPS$:↓, $SBERT$:↑).

C. Ablation Study

Impact of Dynamic Modifiers. Figure 8 and 9 show the impact of the number of captions that are used to generate dynamic modifiers, ranging from 0 to 400, on Prometheus's performance with three metrics. We have two observations. First, including dynamic modifiers boosts the stealing performance. As shown in all cases, when adopting dynamic modifiers (quantity = 100), the similarities in the three metrics outperform the method that excludes dynamic modifiers (quantity = 0). The reason is that our proposed dynamic modifiers enhance the prompt's detail and positional information, providing a better description of the showcase. Second, Prometheus shows a stable performance trend as the modifier quantity increases from 100 to 400. The potential reason is that Prometheus extracts high-quality dynamic modifiers from the captions of the showcase, requiring only a limited number of captions. This reveals the high efficiency of Prometheus.

TABLE VI
IMPACT OF SPECIFIC DYNAMIC MODIFIERS. HERE WE CONSIDER REMOVING SUBJECT DETAIL OR POSITION INFORMATION TO STUDY WHICH PLAYS A MORE IMPORTANT ROLE IN IMPROVING OVERALL PERFORMANCE.

Detail	**Position**	$CLIP_{img}(\uparrow)$	$LPIPS(\downarrow)$	$SBERT(\uparrow)$
✘	✘	0.882	0.674	0.796
✔	✘	0.897	0.659	0.810
✘	✔	0.891	0.665	0.807
✔	✔	**0.901**	**0.653**	**0.814**

Impact of Specific Dynamic Modifiers. We further validate the effectiveness of specific components of dynamic modifiers, *i.e.*, *subject detail* and *position information*. Here, we consider Prometheus with only one kind of dynamic modifiers by excluding the other kind during modifier collection. As shown in Table VI, we have two observations. First, including one or both types of dynamic modifiers improves performance. Subject detail and position information both have gains in image appearance and prompt fidelity, and the gains are even greater when they are both included. Second, subject detail provides more gain than position details. The position information includes position details and azimuth information, where the former is covered by the subject detail to a certain degree.

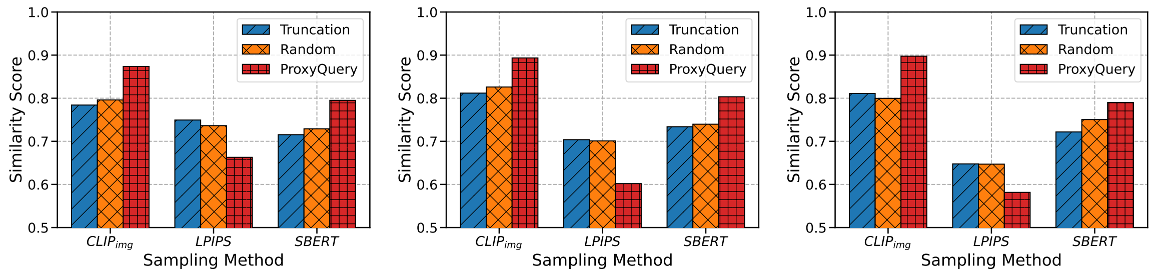
Impact of Contextual Matching. We replace our proposed contextual matching method in Prometheus with a vanilla approach that directly calculates cosine similarity between the modifiers and showcases [28]. Table VII presents the results across three metrics. As the matching method transitions from vanilla to our approach, the attack performance improves. Notably, our contextual matching outperforms the vanilla method across all proxy models. A key highlight is the significant increase in prompt semantic similarity for prompts stolen with FLUX, rising from 0.789 to 0.814. The core reason is that our contextual matching considers modifiers' contextual semantics so that it improves the prompt similarity.

TABLE VII
IMPACT OF MATCHING METHOD. WE REPLACE THE CONTEXTUAL MATCHING IN PROMETHEUS WITH THE VANILLA METHOD.

Proxy	**Methods**	$CLIP_{img}(\uparrow)$	$LPIPS(\downarrow)$	$SBERT(\uparrow)$
FLUX	Vanilla	0.895	0.659	0.789
	Ours	**0.901**	**0.653**	**0.814**
ShuttleDiffusion	Vanilla	0.902	0.655	0.796
	Ours	**0.912**	**0.644**	**0.798**
SD-3.5	Vanilla	0.893	0.642	0.793
	Ours	**0.901**	**0.625**	**0.799**

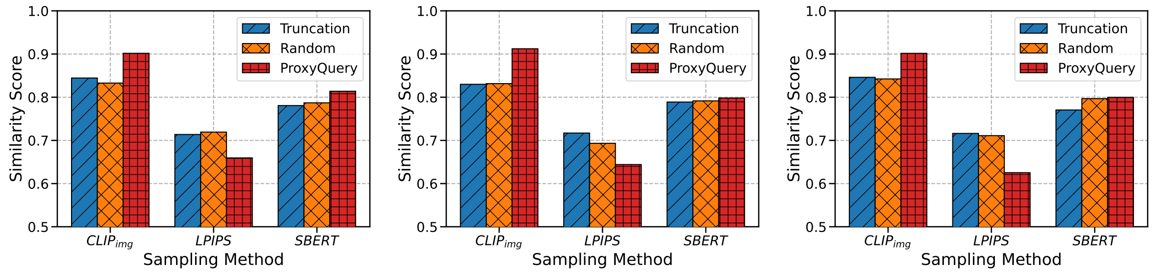
Impact of `Proxy` Feedback. We now illustrate how the `Proxy` feedback influences the performance of Prometheus. We consider two baselines to replace our greedy search, *i.e.*, direct splicing and random sampling. Direct splicing implies that the attacker directly splices the top-k modifiers with the highest correlation to the subject, while random sampling chooses k modifiers at random. We set k to 20 for these two baselines and the sampling pool of random sampling to the best 200 modifiers, in line with the query budget of Prometheus. Figures 10 and 11 show the comparison results on three metrics. As all the modifiers are refined with our local contextual matching, which is based on modifier-showcase cosine similarity, all the methods achieve a rather good performance. This can be reflected in Figure 12(c). Beyond that, we observe our proposed method significantly improves the image and prompt similarity, as shown in Figures 12(a) and 12(b). The reason is twofold. First, although our local ranking provides a highly correlated match (a correlation coefficient of 0.65), it is insufficient to directly adopt the top modifiers as the final choice, as there is still a non-negligible gap. More detailed analysis can be found in Section IV-D. Second, Prometheus remedies this issue by leveraging the feedback from *Proxy*. This helps keep the truly relevant modifiers while dropping the irrelevant ones, and thus improves the appearance alignment and prompt fidelity.

Impact of Query Budget. Figures 12 and 13 illustrate the performance of Prometheus across all three metrics as the query budget increases from 50 to 300. We observe that when the query budget rises from 50 to 100, Prometheus demonstrates significant improvements in image and prompt similarities. However, the performance stabilizes as the budget increases from 100 to 300. This indicates the efficiency of Prometheus, as it requires relatively few queries to achieve satisfactory stealing results. This also validates the effectiveness of our local



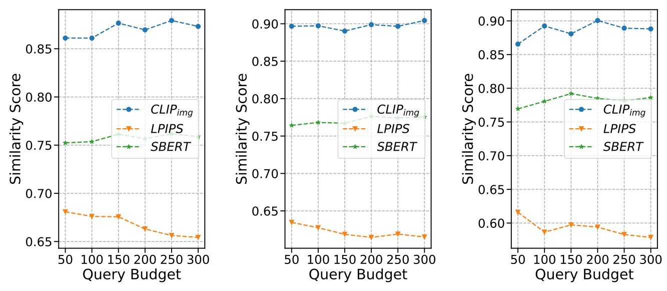
(a) FLUX (b) ShuttleDiffusion (c) SD-3.5

Fig. 10. Impact of `Proxy` query. The evaluation dataset is DALLEPrompt ($CLIP_{img}$:↑, $LPIPS$:↓, $SBERT$:↑).



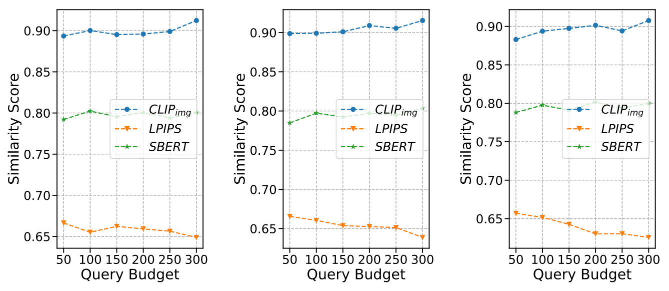
(a) FLUX (b) ShuttleDiffusion (c) SD-3.5

Fig. 11. Impact of `Proxy` query. The evaluation dataset is RealPrompt ($CLIP_{img}$:↑, $LPIPS$:↓, $SBERT$:↑).



(a) FLUX (b) ShuttleDiffusion (c) SD-3.5

Fig. 12. Impact of number of `Proxy` query. The dataset is DALLEPrompt ($CLIP_{img}$:↑, $LPIPS$:↓, $SBERT$:↑).



(a) FLUX (b) ShuttleDiffusion (c) SD-3.5

Fig. 13. Impact of the number of `Proxy` query. The evaluation dataset is RealPrompt ($CLIP_{img}$:↑, $LPIPS$:↓, $SBERT$:↑).

contextual matching, whose coarse-grained filtering restricts an exact range, which further facilitates the fine-grained filtering in `Proxy` query.

Take-away: Prometheus is efficient, requiring few captions and online `Proxy` queries. The designed modules in Prometheus help improve the effectiveness.

D. User Study

We conducted a user study to gain deeper insights into users' preferences regarding the recovered prompts. The question asked of the interviewees is "How similar do you think image A is to image B?". To mitigate potential bias, we did not inform them of our task or what these images represented. The preference is divided into five levels, which are specified in the Appendix. We conducted our investigation anonymously, and finally, we gathered 103 valid answer sheets. The detailed user study setting, IRB exemption, and our effort to protect the volunteers are depicted in Appendix B-B. Figure 14 shows the user study results. We can tell that our Prometheus achieves the highest similarity according to respondents' preference, indicating its practicality in the real world.

Take-away: With sufficient ethical consideration and under fair settings, Prometheus achieves the highest alignment with human perceptions.

VI. Potential Defenses

A. Conventional Defenses

We first consider some existing defenses as below. (1) *Random noise*. We generate a normally distributed noise with the same threshold of ϵ to ensure a similar visual disturbance. (2) *Puzzle effect*. This defense covers the showcase with the puzzle effect, which is applied in practice [6]. (3) *Text watermark*. This is also one defense method used in reality [23].

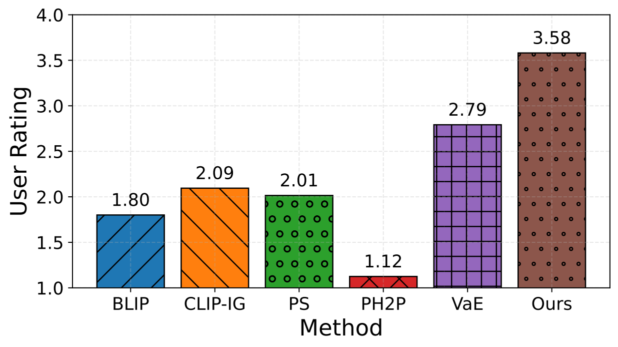


Fig. 14. Rating of user perception. PS refers to PromptStealer [46], and VaE refers to VLMasExpert.

It covers the image with repetitive text indicating the ownership. We present the visual effect of these defenses in Figure 15. We do not consider PromptShield proposed in PromptStealer [46], as it assumes that the attacker uses a modifier prediction model to steal the prompt and the defender has white-box access to the exact modifier prediction model, which does not apply to our attack and is beyond our threat model. Instead, we have a thorough analysis in B-C. We also exclude methods for prompt copyright protection (*e.g.,* PromptCARE [55]), as they can only verify the ownership of prompts, but cannot prevent privacy leakage. Another method, *Glaze* [45], is a perturbation-based method to prevent unauthorized usage of images. It optimizes an invisible perturbation to mislead the showcase feature to a *target* style. Unfortunately, the code for *Glaze* is not open-sourced, so we do not include it in our evaluation.

Table VIII shows the performance of these defenses against our Prometheus. It is obvious that these solutions have a minor impact on the prevention of prompt stealing. The main reason is that none of them are specifically designed to prevent prompt-stealing attacks. Therefore, *their modification to the showcase may suffer from low transferability* when attackers use a caption model to extract prompt information.

B. Dedicated Defense

Given the ineffectiveness of the above defenses, we propose a defense dedicated to Prometheus, dubbed PromptGuard. It aims to preserve the utility of showcasing images for benign users while preventing the reconstruction of prompts by malicious users. The core insight of PromptGuard is to add an invisible perturbation ϵ to the showcase, which can *disturb the feature in the embedding spaces* of the caption models ($\mathcal{B}$) and modifier matching model ($\mathcal{C}$). A defender can utilize any captioning model and modifier matching model to disrupt the embeddings. Its optimization goal can be formulated as:

$$\begin{aligned} &\min\, \alpha \cdot \mathcal{D}(\mathcal{B}(s), \mathcal{B}(\hat{s})) + \beta \cdot \mathcal{D}(\mathcal{C}(s), \mathcal{C}(\hat{s})), \\ &\text{s.t. } (\hat{s} - s) \in [-\epsilon, \epsilon], \end{aligned} \tag{4}$$

where $\mathcal{D}$ is a distance metric used to measure the feature similarity. Here we choose the cosine similarity, where a larger value indicates a higher similarity and vice versa. Therefore, when using the gradient descent algorithm (*e.g.*, Adam optimizer in our implementation) to optimize Formula 4, the feature similarity between s and $\hat{s}$ becomes much lower. The detailed similarity decline process can be found in Figure 16. Specifically, we use the BIM algorithm [13] to generate image perturbation and set the optimization step at 200. We set the perturbation threshold ϵ to 8/255, a rather small value, to *avoid influencing the normal usage of showcases*. Both the hyperparameters α and β are set to 0.5.

TABLE VIII
POTENTIAL DEFENSES AGAINST PROMETHEUS. HERE, THE ARROWS INDICATE THE DEFENSE EFFECT.

Defenses	$CLIP_{img}(\downarrow)$	$LPIPS(\uparrow)$	$SBERT(\downarrow)$	$ASR(\downarrow)$
No Defense	0.901	0.625	0.799	62.5%
Random Noise	0.887 (-0.014)	**0.775 (+0.150)**	0.795 (-0.004)	58.3%(-4.2%)
Puzzle	0.882 (-0.019)	0.659 (+0.034)	0.799 (-0.000)	61.1%(-1.4%)
Watermark	0.829 (-0.072)	0.735 (+0.110)	0.788 (-0.011)	41.7%(-20.8%)
PromptGuard	**0.824 (-0.077)**	0.664 (+0.039)	**0.774 (-0.025)**	27.8%(-34.7%)
Prometheus*	0.915 (+0.014)	0.597 (-0.028)	0.804 (+0.005)	67.9%(+5.4%)

The evaluation results of PromptGuard are presented in Table VIII. The performance of Prometheus significantly declines under PromptGuard's protection, outperforming conventional defenses by a large margin. This effectiveness stems from PromptGuard's ability to *disrupt showcase embeddings*, reducing their feature similarity to the original ones. As a result, when a captioning model or modifier matching model utilizes these disturbed embeddings for prompt stealing, the generated outputs deviate significantly from the correct semantics. These findings validate the effectiveness of PromptGuard in mitigating prompt-stealing attacks.

Limitation of PromptGuard. The success of PromptGuard heavily relies on the assumption that the defender knows the caption models ($\mathcal{B}$) and modifier matching model ($\mathcal{C}$) used by the attacker. However, this assumption does not hold water as the attacker can try other models when he realizes that the chosen $\mathcal{B}$ and $\mathcal{C}$ are shielded. Unfortunately, in this case, the perturbation generated has low transferability to the different caption models and modifier matching models [11], [54]. This significantly restricts the practical value of PromptGuard. For example, PromptGuard generates the perturbation using BLIP with ViT-base, while the attacker can adopt BLIP with a different structure (*e.g.*, ViT-large) for prompt stealing. The defense results for this setting are shown in Table VIII (**denoted as Prometheus***). We observe that with a different caption model, the stealing performance rises again, even outperforming the no-defense setting. This reveals the resistance of Prometheus to PromptGuard and the urgent need for more effective defenses. More discussions about such attacks can be found in Appendix B-C. We will discuss more effective defenses in future work.

Take-away: Prometheus is robust against conventional defenses and resistant to dedicated defense.

VII. POTENTIAL LIMITATIONS AND FUTURE DIRECTIONS

Time Complexity. The primary limitation of Prometheus lies in its runtime. Unlike prior works that rely on fussy data collection and pre-training, Prometheus introduces a lightweight online stealing framework, featuring dynamic modifier generation and a `Proxy`-integrated search process that is both more adaptive and cost-efficient. On average, Prometheus takes about two minutes to steal a single prompt. Given that protected showcases are typically curated works, such as artistic prompts, which are relatively few compared to web-scale images, this time complexity would not become a practical obstruction for large-scale attacks. Moreover, as studied in Section V-C, our parameter choosing is redundant, so that the running time can be further optimized without noticeably affecting performance. Future work can focus on developing more efficient search strategies while preserving effectiveness.

Search Strategy. This paper proposes a greedy search-based framework for effective prompt stealing, overcoming the overfitting limitations of the traditional training–prediction pipeline. We acknowledge that there may exist interdependence between modifiers so that they have a stronger combined effect than greedy search. However, we argue that, as our candidate modifiers are sampled offline with contextual matching before online search, which ensures every candidate already possesses a reasonably good descriptive capability for the showcase, the improvement of interdependent modifiers is likely to be marginal. However, doing so would significantly increase the computational cost. For example, in our setting with 200 candidates, even considering only pairwise combinations would theoretically slow down the search by approximately 100 times (i.e., 199/2). Given that the potential gain is limited, such a computational cost is unacceptable to a large extent. Future work could explore alternative search strategies beyond our adopted greedy search to further enhance attack performance while preserving effectiveness.

VIII. Conclusion

We propose Prometheus, an effective, training-free, and search-based prompt-stealing attack against text-to-image generative models. Our main insight is leveraging the `Proxy` feedback to optimize the prompt to be semantically and perceptually close to the target. To improve the query efficiency and prompt fidelity, we propose the concept of dynamic modifiers, which capture the details and position relations, and utilize the zero-shot ability of BLIP, as well as NLP analysis, to extract such information. We also propose a contextual matching mechanism, which provides a rough ranking order, to improve the greedy search efficacy. We evaluate Prometheus against large-scale on-sold prompts from PromptBase and AIFrog, which are designed for victim models like Midjourney and DALL·E, by querying the well-performed open-source proxy models, including FLUX, ShuttleDiffusion, and SD-3.5. The results demonstrate that Prometheus achieves an excellent prompt-stealing effect and is resistant to potential defenses.

Acknowledgment

We thank all the reviewers for their constructive suggestions that helped improve this paper. We are also grateful to Boheng Li (Nanyang Technological University) for the insightful discussions held during the project. This research/project is supported by the National Research Foundation, Singapore, under its AI Singapore Programme (AISG Award No: AISG3-RPGV-2025-019 and No.AISG4-GC-2023-008-1B), and the Ministry of Education of Singapore under its Academic Research Fund Tier 2 (Grant No: MOE-T2EP20121-0008).

IX. Ethics Considerations and Compliance with the Open Science Policy

A. Ethics Considerations

Stakeholder Perspectives and Considerations.

1) Prompt Engineer. Prompt-stealing attacks may directly harm the prompt engineer's commercial interest. First, an attacker can avoid paying for the prompt but steal it. Second, after stealing, the attacker can reload it to prompt markets to violate the original seller's IP further.
2) Prompt Market. Prompt-stealing attacks may harm the prompt markets' commercial interest. First, stealing a prompt instead of buying would lead to no commissions earned for the prompt market. Second, if the attacker loads the stolen prompt to another platform, it would attract potential users away.
3) Artist and Society. Prompt-stealing attacks can extract the style from an image. Therefore, if an artist posts his/her masterpiece, the attacker could extract a prompt that describes their work precisely. Then the attacker can create infinite images that are similar to the artist's works.

Respect for Persons.

1) Notice. We wrote the consent document that details the intended benefits of research activities and the risks to research subjects.
2) Comprehension. The language level is kept to 8th grade or lower to improve the ability of subjects to comprehend the benefits and risks.
3) Voluntariness. The consent document stresses that participation is voluntary and that subjects are free to withdraw from research participation without negative consequences.

Positive Impact of Research

1) Identification of Potential Benefits and Harms. The potential harm of prompt stealing is violating the intellectual property of prompt creators. The benefit is by revealing the vulnerability of the prompt, the safety issue of the prompt can attract more people's attention.
2) Mitigation of Realized Harms. We consider preempting the escalation of realized harms by notifying affected parties or otherwise engaging in mitigation actions. More specifically, before our research, we made responsible disclosure and notified all the prompt creators. In addition, we developed a defense against prompt stealing attacks to mitigate the realized harm.

IRB Exemption and Volunteer Protection.

1) **IRB Exemption.** In our user study, all images used were non-explicit, stylized paintings representing everyday visual content and contained no inappropriate material (*e.g.*, nudity or sexual themes). Accordingly, under standard ethical

guidelines, this study qualified for exemption from formal IRB review, consistent with the "exempt review" category defined in U.S. IRB protocols (45 CFR 46) because it posed minimal risk to participants.

2) **Volunteer Protection.** To further safeguard participants, we implemented multiple protective measures. Before participation, all volunteers were informed that the study concerned human judgments of image similarity; the specific research purpose was deliberately described in general terms to prevent priming or potential misuse. Participants were briefed on the procedure and data handling, explicitly reminded that participation was voluntary and could be discontinued at any time, especially if they experienced any form of discomfort, and informed consent was obtained. No personally identifiable information was collected. Besides, all questionnaires were completed offline under direct supervision. Participants were allowed to withdraw immediately if they felt any discomfort, and their status was continuously monitored to ensure safety. Upon completion, participants retained the right to withdraw their responses, and we verified their physical and mental well-being before concluding the study.

B. Compliance with the Open Science Policy

We are committed to adhering to open science policies by sharing the outcomes of our research in an open-access format. This includes sharing datasets, test cases, scripts, and source code related to our research paper, to promote a broader commitment to open science principles.

Open Sharing of Code and Data. We will make all artifacts involved in our research, including datasets, test cases, scripts, and source code, publicly available on the GitHub platform to support academic exchange and technological advancement. The models used in our experiments (Stable Diffusion 3.5, FLUX, ShuttleDiffusion) are all open-source and can be accessed and downloaded from HuggingFace. Additionally, the dataset used in this study is the HF-Prompt dataset, an open and non-sensitive dataset available on the HuggingFace platform, ensuring transparency and openness.

Reproducibility and Replicability. All artifacts necessary for reproducing our research findings will be made public. We will also provide detailed experimental records and documentation on the GitHub platform, including information on the experimental environment setup, dependencies, and parameter settings, to enable researchers to replicate our experiments.

References

[1] Adobe. Recruitment of prompt engineer. https://careers.adobe.com/us/en/job/R158292/Prompt-Engineer?utm_source=chatgpt.com, 2025. Accessed on: 2025-10-26.

[2] Leonardo AI. Leonardo ai. https://app.leonardo.ai/, 2024. Accessed on: 2024-06-26.

[3] Stability AI. stable-diffusion-3.5-large-turbo. https://huggingface.co/stabilityai/stable-diffusion-3.5-large-turbo, 2024. Accessed on: 2024-11-28.

[4] AIFrog. Aifrog. https://www.aifrog.io/, 2024. Accessed on: 2024-06-26.

[5] Delphi. Recruitment of prompt engineer. https://www.indeed.com/q-prompt-engineer-jobs.html?utm_source=chatgpt.com&vjk=576c20b3cfe476dd, 2025. Accessed on: 2025-10-26.

[6] Dianamoran@PromptBase. Vivid bright bold graphic illustrations. https://promptbase.com/prompt/vivid-bright-bold-graphic-illustrations-2, 2024. Accessed on: 2024-06-26.

[7] Siddhant Garg and Goutham Ramakrishnan. Bae: Bert-based adversarial examples for text classification. arXiv preprint arXiv:2004.01970, 2020.

[8] Yaru Hao, Zewen Chi, Li Dong, and Furu Wei. Optimizing prompts for text-to-image generation. Advances in Neural Information Processing Systems, 36, 2024.

[9] Sharath S Hebbar. Dall-e-prompts. https://huggingface.co/datasets/Sharathhebbar24/DALL-E-Prompts-OpenAI-ChatGPT?row=4, 2024. Accessed on: 2024-11-28.

[10] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. Advances in neural information processing systems, 33:6840–6851, 2020.

[11] Qian Huang, Isay Katsman, Horace He, Zeqi Gu, Serge Belongie, and Ser-Nam Lim. Enhancing adversarial example transferability with an intermediate level attack. In Proceedings of the IEEE/CVF international conference on computer vision, pages 4733–4742, 2019.

[12] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. arXiv preprint arXiv:2410.21276, 2024.

[13] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. arXiv preprint arXiv:1611.01236, 2016.

[14] Black Forest Labs. Flux.1-schnell. https://huggingface.co/black-forest-labs/FLUX.1-schnell, 2024. Accessed on: 2024-11-28.

[15] Lexica. Lexica dataset. https://lexica.art/, 2024. Accessed on: 2024-06-26.

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

[17] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In International conference on machine learning, pages 12888–12900. PMLR, 2022.

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13, pages 740–755. Springer, 2014.

[19] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13, pages 740–755. Springer, 2014.

[20] Vivian Liu and Lydia B Chilton. Design guidelines for prompt engineering text-to-image generative models. In Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems, pages 1–23, 2022.

[21] Shweta Mahajan, Tanzila Rahman, Kwang Moo Yi, and Leonid Sigal. Prompting hard or hardly prompting: Prompt inversion for text-to-image diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6808–6817, 2024.

[22] Midjourney. Midjourney. https://www.midjourney.com, 2024. Accessed on: 2024-06-26.

[23] Mylab@PromptBase. Summer animal relaxes. https://promptbase.com/prompt/summer-animal-relaxes, 2024. Accessed on: 2024-06-26.

[24] OpenAI. Dall·e 2. https://openai.com/index/dall-e-2/, 2024. Accessed on: 2024-06-26.

[25] OpenAI. Dall·e 3. https://openai.com/index/dall-e-3, 2024. Accessed on: 2024-06-26.

[26] Jonas Oppenlaender. A taxonomy of prompt modifiers for text-to-image generation. Behaviour & Information Technology, pages 1–14, 2023.

[27] Jonas Oppenlaender, Rhema Linder, and Johanna Silvennoinen. Prompting ai art: An investigation into the creative skill of prompt engineering. International journal of human–computer interaction, 41(16):10207–10229, 2025.

[28] GitHub: pharmapsychotic. Clip-interrogator. https://github.com/pharmapsychotic/clip-interrogator, 2024. Accessed on: 2024-06-26.

[29] PromptBase. Promptbase. https://promptbase.com/, 2024. Accessed on: 2024-06-26.

[30] PromptBase. User terms. https://promptbase.com/tandcs, 2024. Accessed on: 2024-11-28.

[31] PromptHero. Prompthero. https://prompthero.com/, 2024. Accessed on: 2024-06-26.

[32] PromptPerfect. Promptperfect. https://promptperfect.jina.ai/, 2024. Accessed on: 2024-06-26.
[33] Promptrr. Portrait neon lights. https://promptrr.io/product/mid-journey-prompt-for-portrait-neon-lights-collection/, 2024. Accessed on: 2024-06-26.
[34] PromptSea. Promptsea. https://www.promptsea.io/, 2024. Accessed on: 2024-06-26.
[35] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In International conference on machine learning, pages 8748–8763. PMLR, 2021.
[36] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. arXiv preprint arXiv:2204.06125, 1(2):3, 2022.
[37] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084, 2019.
[38] Tal Ridnik, Gilad Sharir, Avi Ben-Cohen, Emanuel Ben-Baruch, and Asaf Noy. Ml-decoder: Scalable and versatile classification head. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pages 32–41, 2023.
[39] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 10684–10695, 2022.
[40] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18, pages 234–241. Springer, 2015.
[41] Tom Roth, Inigo Jauregi Unanue, Alsharif Abuadbba, and Massimo Piccardi. A constraint-enforcing reward for adversarial attacks on text classifiers. arXiv preprint arXiv:2405.11904, 2024.
[42] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. Advances in neural information processing systems, 35:36479–36494, 2022.
[43] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. Advances in Neural Information Processing Systems, 35:25278–25294, 2022.
[44] Prompt Sea. Whitepaper. https://www.promptsea.io/promptsea-whitepaper.pdf, 2024. Accessed on: 2024-06-26.
[45] Shawn Shan, Jenna Cryan, Emily Wenger, Haitao Zheng, Rana Hanocka, and Ben Y Zhao. Glaze: Protecting artists from style mimicry by {Text-to-Image} models. In 32nd USENIX Security Symposium (USENIX Security 23), pages 2187–2204, 2023.
[46] Xinyue Shen, Yiting Qu, Michael Backes, and Yang Zhang. Prompt stealing attacks against text-to-image generation models. arXiv preprint arXiv:2302.09923, 2023.
[47] ShuttleAI. shuttle-3-diffusion-fp8. https://huggingface.co/shuttleai/shuttle-3-diffusion-fp8, 2024. Accessed on: 2024-11-28.
[48] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In International conference on machine learning, pages 2256–2265. PMLR, 2015.
[49] Spacy. Spacy. https://spacy.io/, 2024. Accessed on: 2024-06-26.
[50] Ben Stokes. 10,000 users. https://daily.tinyprojects.dev/185, 2024. Accessed on: 2024-06-26.
[51] Ben Stokes. Introducing promptbase. https://daily.tinyprojects.dev/136, 2024. Accessed on: 2024-06-26.
[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.
[53] 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, 36:51008–51025, 2023.
[54] Lei Wu, Zhanxing Zhu, Cheng Tai, et al. Understanding and enhancing the transferability of adversarial examples. arXiv preprint arXiv:1802.09707, 2018.
[55] Hongwei Yao, Jian Lou, Kui Ren, and Zhan Qin. Promptcare: Prompt copyright protection by watermark injection and verification. arXiv preprint arXiv:2308.02816, 2023.
[56] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 586–595, 2018.

Appendix A
Method

A. Problem Motivation

We specify the motivation for studying prompt stealing. **Prompts' Value.** The value of *well-crafted prompts* can be validated from two perspectives. Firstly, there are numerous emerging prompt marketplaces and registered prompt engineers. The active prompt transactions prove the economic significance of exquisite prompts. Second, prompt engineering has become a recognized profession. For instance, Adobe offers prompt engineer positions with salaries between USD 162,000 and 301,200 [1], and similar listings appear on Indeed [5]. These trends highlight that *well-crafted prompts* are valuable intellectual assets.

Attack Scenario. In some scenarios, users may have vague ideas of what they want to create but struggle to express them effectively, as well-designed prompts embed subtle linguistic cues that significantly influence output quality. Previous work [27] validates the "articulation barrier" when describing aspects like style or composition, especially for image generation. We target the scenario where users frequently encounter appealing AI-generated artworks online and may seek to reproduce their style, making prompt reconstruction both a realistic and attractive attack scenario.

B. Privacy Protection Standpoint

From the privacy protection standpoint, prompt-stealing attack poses a significant threat to the confidentiality of prompts. Specifically, in many cases, a user's prompts may interact with sensitive or private data during the generation process (*e.g.*, rely on information retrieved from private databases or internal documents through in-context learning). Then, the recovered prompts include not only the ordinary user's prompts but also the augmented sensitive information, as both are ultimately reflected in the generated images. For example, professional artists might use a prompt such as 'Generate a portrait in the style of my previous drafts (as shown in the provided examples)'. Stealing the prompt of such a generated portrait image could inadvertently expose the artist's unique stylistic characteristics (*i.e.*, the privacy in this case) embodied in those unreleased drafts.

C. Dynamic Modifier Extraction

We detail the extraction process for dynamic modifiers in Algorithm 2. We first generate multiple descriptions of the showcase using a captioning model. We then apply the NLP toolkit Spacy [49] to each caption to obtain its part-of-speech (PoS) tags and dependency tree (DepTree). These two signals are subsequently used to extract subject details and position information, which together form the dynamic modifiers.

Algorithm 2 `Dynamic Modifier Extraction`

```
Input: caption model BLIP $\mathcal{B}$, showcase $s$, caption quantity $q$, and NLP
    analysis tool Spacy.
Output: extracted dynamic modifier list $\mathcal{M}_d$.
1: $\mathcal{M}_d(d) \leftarrow \emptyset$                ▷ Store subject detail
2: $\mathcal{M}_d(p) \leftarrow \emptyset$                ▷ Store position information
3: $\mathcal{P} \leftarrow \mathcal{B}(s, q)$                ▷ Caption q times
4: for $p$ in $\mathcal{P}$ do
5:     POS, DepTree ← Spacy($p$)
6:     $\mathcal{C}$ ← Spacy.chunk($p$)                ▷ Get detail chunks
7:     $\mathcal{M}_d(d) \leftarrow \mathcal{M}_d(d) \cup \mathcal{C}$
8:     for token in $p$ do
9:         if POS(token) = "prep" then                ▷ Find prepositions
10:            child ← DepTree(token)
11:            $c$ ← child.join()
12:            $\mathcal{M}_d(p) \leftarrow \mathcal{M}_d(p) \cup c$
13:        end if
14:    end for
15: end for
16: return $\mathcal{M}_d(d)$ and $\mathcal{M}_d(p)$
```

Appendix B
Experiment

A. VLMasExpert

As presented in Table II and Figure 6, the baseline VLMasExpert, which serves as a potential prompt-stealing attack that considers integrating a human expert (it utilizes VLMs as human experts) into the stealing process, performs worse than Prometheus. The potential reason can be twofold. First, the same to PromptStealer [46], the well-performed VLM GPT-4o is trained on specific image-text pairs and may face an over-fitting problem when adapted to the prompt-stealing task. Second, out of copyright protection, most large models, including GPT-4o, are aligned not to answer copyrighted content. Thus, VLMasExpert replaces all the protected subjects with more generalized subjects, which results in less accurate prompt stealing.

To study which factor accounts for the most of VLM's defect, we experiment by ignoring the test examples that contain copyrighted content, which is denoted as VLMasExpert*. As shown in Table IX, after disregarding copyrighted showcases, the performance of VLMasExpert is improved, but slightly. This indicates that alignment indeed mitigates VLM's potential for being adopted in the prompt-stealing task. However, the improvement is relatively minor, suggesting that the primary issue is not copyright protection, but rather overfitting.

TABLE IX
VLMasExpert without alignment.

Method	$CLIP_{img}(\uparrow)$	$LPIPS(\downarrow)$	$SBERT(\uparrow)$	$ASR(\uparrow)$
VLMasExpert	0.848	0.695	0.776	27.1%
VLMasExpert*	0.852	0.692	0.785	29.2%
Prometheus (Ours)	**0.901**	**0.653**	**0.814**	**58.3%**

B. User Study

We specify the user study setting as follows:

Study Detail. The recruited volunteers were asked to complete a questionnaire on perceptual image similarity. Each participant viewed 12 randomly selected groups of images (a showcase and its recovered versions); each group contained 6 image pairs (5 baselines and our method), yielding 72 questions in total. All images were non-explicit, stylistic paintings within the range of everyday visual content (the examples can be seen in Figure 6). For each question, volunteers rated the perceptual similarity between the showcase and the recovered image using the same scoring criteria as PromptStealer [46]. To avoid publicizing the prompt-stealing setting, we did not disclose the precise research goal and only told participants they were comparing the similarity between two images. To reduce potential bias, we anonymized the methods for every question and randomized the method order within each group to prevent habitual choices and ensure independence across questions.

Volunteers. We totally recruited 103 volunteers and obtained 103 valid questionnaires. To eliminate bias, the volunteers are recruited from various backgrounds, including gender, age, and education. Specifically, among all the respondents, 57.28% are men while 42.72% are women. As for age, 41.75% of the surveyed people were between 18-25, and 45.63% of those were between 26-30. Including respondents aged between 31-40 (total proportion of 10.68%), these three parts account for 98.06% of the total. Of the remaining 1.94%, both the people aged under 18 and those between 51-60 account for 0.97%. We also realize the effect of education on respondents' preferences. We keep all the questions to 8th grade or lower to ensure they are fully understood by the respondents. Among all the surveyed people, most of them obtained a college degree or above, and only 0.97% of them have a high school degree. Specifically, 3.88% of the total respondents have a college degree, and a proportion of 30.1% have a bachelor's degree. The left part, with a proportion of 65.05%, occupies the vast majority, and owns postgraduate degree.

C. Defense

PromptShield. PromptShield seeks to defend against prompt-stealing by turning showcases into adversarial examples using gradients from the attacker's modifier-predictor. It sets the ground-truth prompt without artist modifiers as the adversarial target and then optimizes the showcase with the predictor's gradients. This approach has four shortcomings. First, it assumes the attacker trains a modifier classifier, which our attack does not, making PromptShield inapplicable and limiting its real-world utility. Second, it assumes white-box access to the attacker's predictor, which is often unrealistic. Third, it presumes all marketplace prompts include artist modifiers so a target adversarial prompt can be constructed; however, many do not. For example, the prompt in Figure 1 from PromptBase lacks any artist modifier. Finally, even when a prompt contains an artist modifier, removing that element safeguards only the artist modifier while leaving other modifiers unprotected.

Detail of Defenses. We set the hyperparameter of conventional defenses following the principle that the perturbation should not affect human perception of the subjects and styles. Specifically, for random noise, we generate Gaussian noise with a mean of

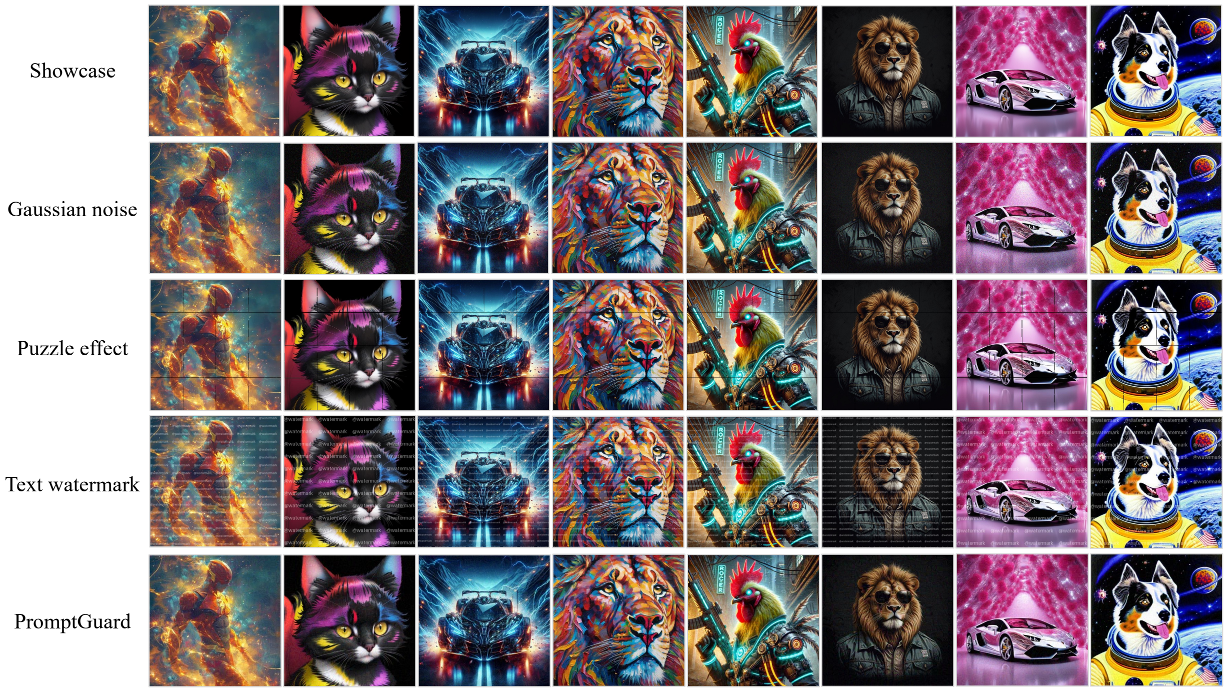
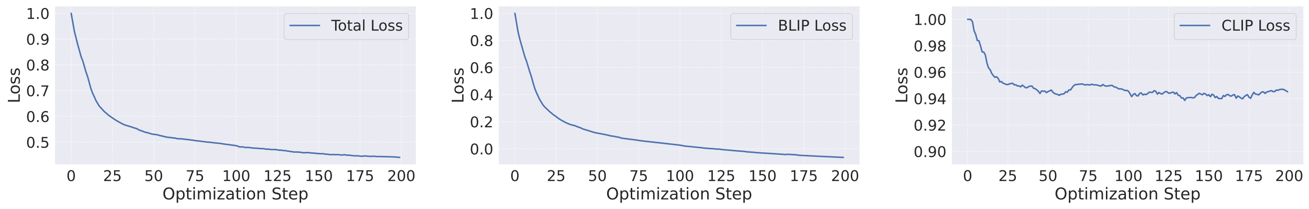


Fig. 15. Example of generated adversarial example with different defenses. Please zoom out for details.



(a) Total Loss

(b) BLIP Loss

(c) CLIP Loss

Fig. 16. Loss with optimization step.

0 and a standard deviation of 25, which is almost perceptible to humans. For the puzzle effect, we add a 4×4 grid to the showcase, with variability set to 3. We generate a text watermark "@watermark" as the intellectual property claim. The text font size is set to 20, and the gaps in the rows and columns are set to 20 and 30, respectively. We evaluate the defense performance on the RealPrompt dataset to simulate the preventive effect in the real world.

Analysis of Adaptive Attack. The proposed defense, named PromptGuard, employs the gradient of a unified loss function to generate adversarial perturbations. The loss curve is shown in Figure 16, where the unified loss ultimately converges around 0.45, demonstrating the defense's effectiveness. However, our observations reveal that the primary decline in the loss comes from BLIP, while CLIP contributes minimally. This can be attributed to CLIP's pre-training on LAION [43], a dataset containing billions of examples, which makes it inherently robust to adversarial perturbations. In contrast, BLIP is more fragile due to its significantly smaller training dataset. Based on these findings, when designing adaptive attacks, we focus on modifying the BLIP structure, as adversarial examples exhibit poor transferability across different BLIP models. The core of defending against prompt-stealing attacks is to break the connection between the showcase and its caption, independent of the specific captioning model. Therefore, future defenses should prioritize mitigation strategies that remain robust across different attack model architectures.

D. More Analysis

Main results. We observe a performance gap between DALLEPrompt and RealPrompt. After reviewing both datasets, we attribute this to two factors: (1) Model type. RealPrompt is largely sourced from commercial T2I systems (*e.g.*, Midjourney, DALL·E) with stronger instruction-following, yielding closer prompt–showcase alignment; as a result, showcases more faithfully reflect prompt properties and improve prompt reconstruction accuracy. (2) Prompt type. RealPrompt contains expert-crafted, commercially sold prompts designed by prompt engineers, which tend to be more specific and stylistically distinctive; their showcases are therefore easier to differentiate. By contrast, DALLEPrompt primarily comprises public user prompts that are more generic and can be expressed in many ways, weakening the correspondence between prompts and showcases.