Is It Possible to Attack a T2I Model With Only Punctuation?

Abstract

Text-to-Image (T2I) models have become immensely popular due to their ability to generate high quality images from natural language prompts, but their safety and robustness in real-world applications remains a critical concern to date. In this work, we explore the use of punctuations as an attack vector on black-box T2I models. We show that it is easy to fool and mislead the victim model by simply injecting a few punctuations into the clean prompt, despite punctuations having virtually no semantic meaning. These punctuations injected could be attributed to human typographical errors, making the adversarial attack imperceptible and suitable as a real-world attack. We also propose the Punctuation Injection Permutator (PIP) pipeline which can craft the adversarial prompt automatically using an optimizer and a vision-language model (VLM) evaluator in both untargeted and targeted attack settings. Our code is available at **https://github.com/DenseLance/PIP-Pipeline**.

1 Introduction

The rapid development of Text-to-Image (T2I) models such as Stable Diffusion [1, 2], DALL-E [3, 4], and FLUX [5, 6] has enabled high fidelity and diverse image synthesis for vast real-world applications. However, despite the widespread adoption of T2I models and the numerous strategies proposed to enhance safety and robustness [7, 8, 9, 10, 11], they are still susceptible to various adversarial attacks [12, 13]. Most commonly, attempts to uncover the vulnerability in the robustness of a T2I model involve minute perturbations to a given prompt, which would cause the semantics of the generated images to stray further away from the initial intent.

Existing prompt perturbation strategies against T2I models include appending a five-character noisy word as postfix [14], artificially introducing typographical errors [15], and substituting keywords [16] – all of which necessitates the use of the English alphabet. Moreover, many of the previous works often rely on varying knowledge about the architecture of the victim T2I model. For instance, some adversarial attacks make direct queries to the victim model's text encoder to retrieve text embeddings [14, 17], while others require the local gradient information of the victim model [16, 18]. When the adversary has absolutely no information on the victim T2I model (e.g. closed-source online services such as DALL-E [3, 4] and Imagen [19, 20]), most of these attacks become infeasible. Furthermore, text encoder based attacks often assume that the victim model utilises only one text encoder, making them less applicable for more recent T2I models such as SDXL [21] and FLUX.1-dev [5] which employ multiple text encoders.

In this work, we have designed a realistic and subtle attack in the form of punctuation injection. By injecting punctuations into a clean prompt, we can either steer the T2I model away from the original prompt (untargeted attack), or a specific concept (targeted attack). We

also propose the Punctuation Injection Permutator (PIP) pipeline, to automate the process of finding a set of punctuations for a particular prompt that can mislead a black-box T2I model. Our solution also serves to be future-proof, as it can be applied to T2I models that do not rely on the reverse diffusion process [1, 22], or the CLIP text encoder [23], both of which are still commonly seen today.

We performed an empirical evaluation on CIFAR10 classes [70] and a curated subset of COCO captions [71, 72]. The ten prompts crafted from CIFAR10 classes served as a diagnostic assessment on whether a punctuation-level attack can eliminate the only concept from the generated image, while the two hundred COCO captions provided a comprehensive appraisal of the effectiveness of PIP pipeline on human-like prompts. We demonstrate that punctuation injections can indeed deceive a victim T2I model to generate an image that does not align with the original prompt.

2 Related Works

T2I Models. T2I models represent a groundbreaking advancement in the field of generative artificial intelligence (AI), due to their ability to synthesize diverse visual content from textual descriptions. Today, most state-of-the-art T2I models take the form of diffusion models due to their impressive performance in generating images [1, 22, 24]. The diffusion model applies the forward diffusion process by progressively adding Gaussian noise to the data distribution, while the reverse diffusion process attempts to recover the original data distribution by removing the noise that was added to it. In general, a standard diffusion model pipeline for T2I inference consists of four key components: tokenizer, text-image encoder, U-Net [25], and variational autoencoder (VAE) [26]. The tokenizer first converts the prompt into a sequence of tokens. The text encoder, usually a CLIP model [23], would then project these tokens to the embedding space. Some diffusion models such as SDXL use an additional CLIP model [21], and others like Stable Diffusion 3 [2] go even further by using a pure text encoder (T5-XXL [27]). The U-Net, conditioned by the text embeddings, is used to predict and remove noise iteratively from an isotropic Gaussian latent during the reverse diffusion process. Finally, the VAE decoder reconstructs the latent representation back into an image.

Adversarial Attacks. Adversarial attacks are able to fool deep neural networks by carefully crafting small perturbations on an input [28, 29, 30], resulting in an output that is contrary to expectation. Adversarial attacks can be classified as black-box if the adversary has no knowledge of the internal workings and the different components of the victim model, or white-box if the adversary is given some level of access to the victim model. Existing attack methods on T2I models can be further categorized according to the perturbation granularity. Character-level attacks seek to add, displace, replace, or remove individual characters within the word so that the perturbed word would be tokenized as a different token and cause the encoded embeddings to become entirely unrelated to what is expected [15, 31]. Word-level attacks attempt to replace, insert, and delete whole words within a clean prompt to mislead the victim model, while maintaining semantic coherence to remain imperceptible to the

human eye and automatic safeguards [14, 16, 32]. Sentence-level attacks usually alter the sentence structure of the prompt, reducing coherence and can cause the T2I model to misinterpret its meaning [33, 34]. In our work, we focus on punctuation-level attack, an attack that is rarely discussed in the context of T2I.

Punctuation-level Attacks. Similar to word-level attacks, punctuation-level attacks can be thought of as a combinatorial optimization problem. For instance, given a set of punctuations, which punctuations should be chosen and where should they be inserted in the clean prompt to optimally fool a T2I model? In the case of word-level attacks on text-only models, Zhan et al. chose to reduce the search space [35] while Zang et al. found approximate solutions using optimization algorithms [36], considering that an exhaustive search is computationally expensive and not practically feasible.

Recent works on punctuation-level attacks do not focus on T2I specifically, but rather text-based models as a whole. For example, Formento et al. found that punctuation-level attacks can be more imperceptible than character-level attacks due to several reasons [37]: (1) humans find punctuation-perturbed text faster and easier to read, (2) high semantic similarity with respect to the original text, (3) punctuation injection can bypass grammar checkers which are sometimes used as T2I safeguards. Formento et al. also proposed a few attack strategies to insert punctuations between two characters in a clean prompt, similar to a character-level attack. In our work, we chose to insert punctuations between two words instead.

Wang et al. conducted an automatic single punctuation attack by using the text embeddings obtained from the T2I model's text encoder to yield a score for each candidate adversarial prompt [17]. While they claim that the attack can be conducted in a black-box setting by training a substitute model for the text encoder, the training process requires direct queries to the text encoder, which is not possible when interacting with a WebUI or closed-source online service as discussed earlier in Section **1**. The attack proposed by Wang et al. is also not suitable for T2I models with more than one text encoder, as the embeddings that are returned are usually of different dimensions.

3 Proposed Pipeline

3.1 Problem Statement

In a black-box setting, we are only allowed to query a T2I model that accepts text inputs and some hyperparameters to generate image samples. Without any knowledge regarding the T2I model, we cannot determine the possible vulnerabilities that most existing adversarial attacks seek to exploit. At the same time, there is limited research on punctuation-level attacks for the T2I task. As such, we want to find out whether it is possible to attack a black-box T2I model by injecting multiple punctuations into a clean prompt.

Punctuation. In our work, we refer to the set of 32 ASCII punctuations, which includes `!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~`. All of the punctuations can be found on a QWERTY keyboard, making the punctuation-level attack easy to carry out.

The Cambridge Dictionary defines punctuations as a special set of characters that are commonly used to separate phrases and sentences in English text [38, 39], with the main purpose of clarifying meaning, grammar, and structure in writing. By inserting punctuations, a natural language text can be interpreted very differently. For instance, Truss provides a humorous example of syntactic ambiguity that arises from punctuation injection: "Panda. Large black-and-white bear-like mammal, native to China. Eats, shoots and leaves." [40]. By introducing a comma in the last sentence of the quote, "shoots" and "leaves" are treated as verbs instead of nouns, causing them to be interpreted wrongly as "the panda eats, then it shoots, and finally it leaves".

As a result, punctuation injection can alter the intention of the prompt by exploiting the fact that a word can have nuances in meaning depending on the context. Unfortunately, in most cases, punctuations are unable to introduce totally unrelated concepts into text because of their limited semantic meaning without contextual cues as compared to words. This translates to a much greater difficulty in terms of forcing a T2I model to generate a specific and distinct object with just a punctuation perturbed prompt.

Injection Position Does Matter. Some adversarial attacks on T2I models, such as those proposed by Du et al. [16] and Zhang et al. [41], perturb the prompt by naively concatenating characters as a postfix. However, we can actually raise the success rate of these attacks by displacing the perturbation to another empty space in the prompt, as shown in Figure 1.

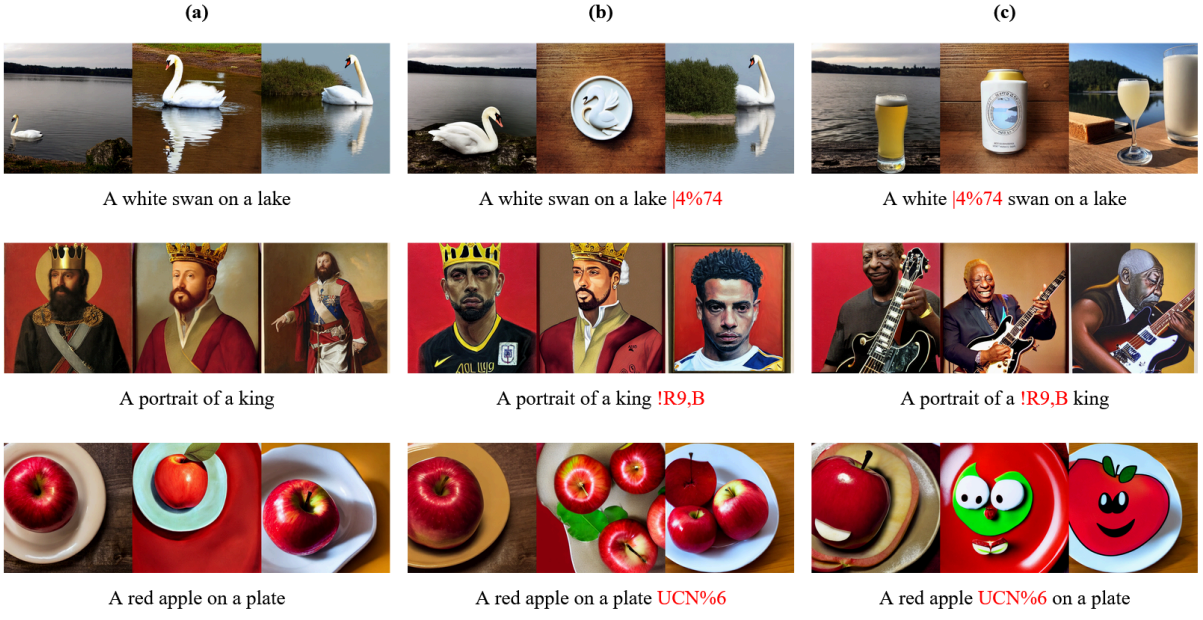

Figure 1: Zhuang et al. proposed an adversarial attack that uses a noise word containing five characters as the perturbation [14]. We show the images generated by Stable Diffusion when using **(a)** the original prompt, **(b)** prompt with perturbation appended as the postfix, and **(c)** prompt with perturbation injected at the optimal position. We ran 20 iterations of the greedy algorithm to create the noise word, and the same random seed to generate the above images.

In the same vein, the position where punctuations are injected would also affect the output of T2I models, as they are treated as a standalone token by most tokenizers today such as byte-pair encoding (BPE) [42], WordPiece [43, 44], and SentencePiece [45]. Moreover, punctuations should be interleaved between words and phrases just like in natural language. We illustrate how punctuation position can vary the output of the T2I model in Figure 2.

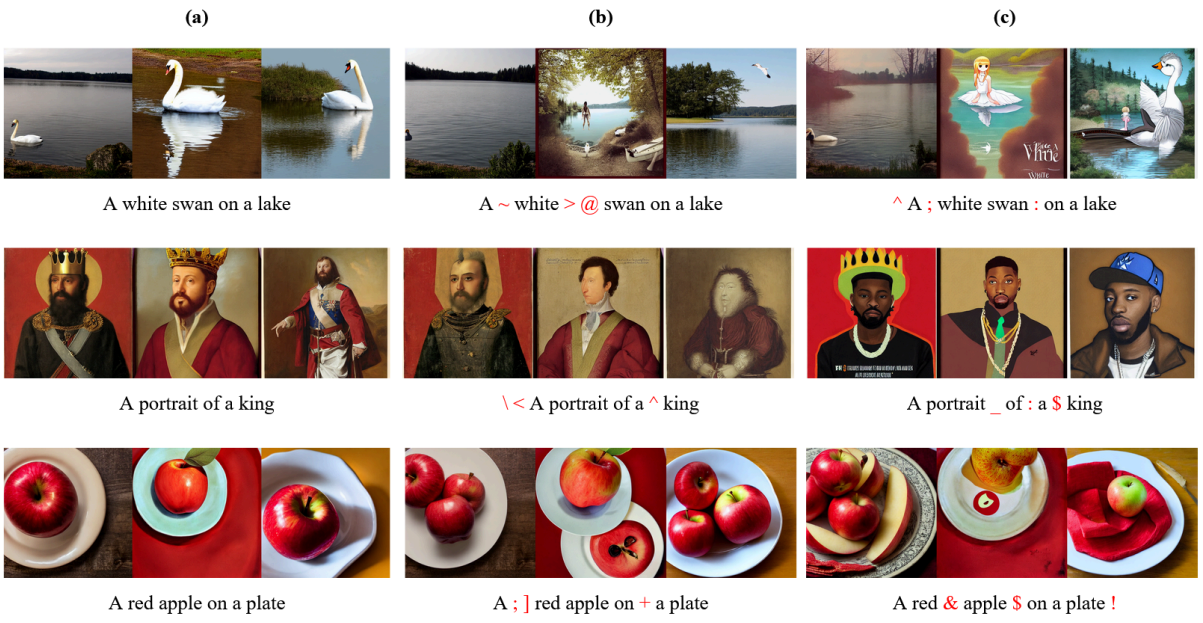

Figure 2: (a) shows the images generated when using the original prompt, while **(b)** and **(c)** show the possible images that can be generated when we inject three punctuations into the original prompt at different locations. We observed significant degradation in terms of alignment for certain punctuation permutations. The same random seed to generate the above images.

Injection Methods. Inserting *m* number of punctuations into the clean prompt can be done in two main ways, either iteratively or concurrently. The case for iterative injection of punctuations has been briefly discussed by Wang et al. [17], and we show how it can be done in Algorithm 1 below.

Algorithm 1: Sketch of how iterative injection can be conducted

```
best_adversarial_prompt = prompt
for i = 1 to m:
        punctuation, position = find_best_permutation(prompt, injections = 1)
```

Algorithm 1: Sketch of how iterative injection can be conducted

```
        best_adversarial_prompt = prompt.insert(punctuation, position)
return best_adversarial_prompt
```

In our work, we suggest using concurrent injection instead, where we try to permute multiple punctuations at the same time. This is because certain punctuation permutations may be extremely effective as an attack, yet it does not lead to a noticeable change in the generated image when each punctuation is injected independently from each other. We describe how this might work in Algorithm 2 below.

Algorithm 2: Sketch of how concurrent injection can be conducted

```
best_adversarial_prompt = prompt
punctuations, positions = find_best_permutation(prompt, injections = m)
for i = 1 to m:
        best_adversarial_prompt = prompt.insert(punctuations[i], positions[i])
return best_adversarial_prompt
```

3.2 Pipeline Requirements

Models. The PIP pipeline requires (1) an optimizer to suggest the punctuations that should be inserted and their position, (2) the victim T2I model to generate images, and (3) a reference-free visual-language model (VLM) evaluator to quantify the alignment of the generated images to the original prompt. Throughout this paper, we assumed that higher scores from the VLM evaluator indicate greater alignment to the given text. The workflow for PIP is depicted in Figure 3 below.

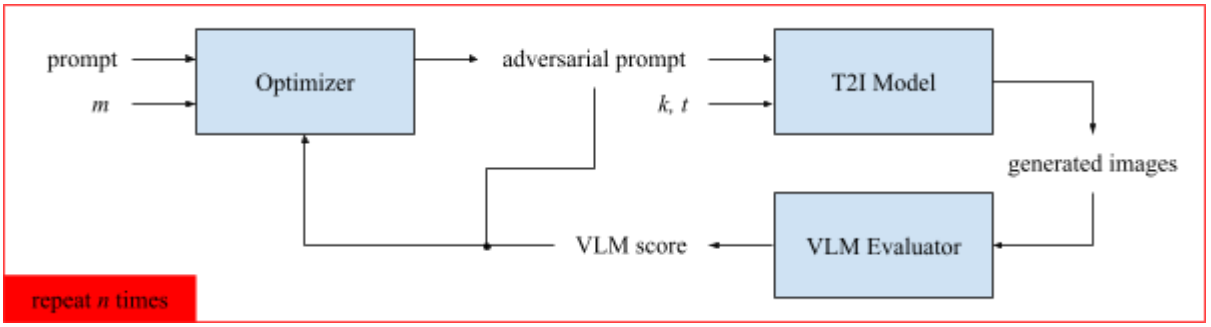

Figure 3: PIP pipeline to craft an adversarial prompt. The pipeline begins by obtaining a set of m punctuations and the locations they will be injected at from the optimizer in order to craft the adversarial prompt. The adversarial prompt is then used to generate k images with the victim model in t inference steps. The generated images are evaluated by the VLM evaluator, and the score obtained is used to guide the optimizer to suggest better adversarial prompts. This process is repeated n times in total.

We used an optimizer to obtain an approximation to the best adversarial prompt, as the search space for punctuation injection is extremely large [17]. To put things into perspective, if we

were to insert m punctuations concurrently into a prompt with c words, then we would need to verify a total of $(32c)^m$ candidate permutations to determine the global optimum via brute force. However, by increasing m, we can expect that the proportion of permutations that result in successful adversarial attacks would increase exponentially. This leads to the goal of the optimizer, which is to locate and retrieve at least one sample that satisfies the adversarial objective.

Hyperparameters. A total of four hyperparameters were considered for the PIP pipeline, as shown in Table 1 below. The same random seed should be used throughout, as image generation using the same prompt but different seeds may result in entirely different image compositions altogether [46, 47]. Hyperparameters k and t, and random seed which are inputs to the T2I model are usually modifiable. This includes WebUIs [48] and many online services like MidJourney [49] and NovelAI [50], making the pipeline suitable in a black-box setting. The impact of each hyperparameter on the effectiveness of our adversarial attack will be explored further in Section 4.

Notation	Definition
n	Number of permutations tested/T2I queries
m	Number of punctuations injected
k	Number of images generated per trial
t	Number of inference steps for image generation

Table 1: Nomenclature of pipeline hyperparameters.

Queries to the T2I model can be computationally expensive and this forms the main bottleneck of our pipeline. As such, we can consider reducing the number of inference steps, t, to approximate the expected generated image, only when crafting the adversarial prompt. This problem becomes less noticeable for distilled T2I models that require less than 5 inference steps such as SANA-Sprint [51], SD-Turbo [52] and DMD2 [53].

3.3 Attack Modes

In this section, we describe and provide a sketch of how an untargeted and targeted attack can be implemented using the proposed PIP pipeline. In the case of an untargeted attack, the adversary does not need to specify an attack intention. As such, we can simply yield an adversarial prompt that deviates the furthest away from the intent of the original prompt. This is described in Algorithm 3.

Algorithm 3: Untargeted attack via PIP pipeline

```
1  min_score = vlm_evaluator.upper_bound
```

Algorithm 3: Untargeted attack via PIP pipeline

```
for i = 1 to n:
        adversarial_prompt = optimizer.suggest(prompt, m)
        images = t2i_model.generate(adversarial_prompt, k, t)
        avg_score = vlm_evaluator.evaluate(prompt, images).mean()
        optimizer.update(adversarial_prompt, avg_score)
        if avg_score < min_score:
                min_score = avg_score
                best_adversarial_prompt = adversarial_prompt
return best_adversarial_prompt
```

However, when the adversary has an attack purpose, we would need to refine and steer the adversarial prompt towards said objective to achieve a targeted attack. The targeted attack mode can come in two forms: (1) injecting a malicious concept into the original prompt, and (2) removing an existing concept from the original prompt. Introducing an unrelated concept to the given prompt is extremely challenging, since punctuations generally have no semantic meaning by themselves as described earlier. This means that punctuations alone would be insufficient to direct the T2I model towards the malicious concept. Instead, we conducted our targeted attack by removing an existing concept from the clean prompt. Our attack objective has now shifted to yielding an adversarial prompt that achieves the lowest alignment score with the removed concept, as demonstrated in Algorithm 4.

Algorithm 4: Targeted attack via PIP pipeline

```
min_score = vlm_evaluator.upper_bound
for i = 1 to n:
        adversarial_prompt = optimizer.suggest(prompt, m)
        images = t2i_model.generate(adversarial_prompt, k, t)
        avg_score = vlm_evaluator.evaluate(removed_concept, images).mean()
        optimizer.update(adversarial_prompt, avg_score)
        if avg_score > min_score:
                min_score = avg_score
                best_adversarial_prompt = adversarial_prompt
return best_adversarial_prompt
```

4 Methodology

4.1 Experimental Setup

Victim T2I Model. For our experiments, we chose Stable Diffusion 1.4 as our victim model due to its ability to generate high quality images and is widely used [1]. We also used the HuggingFace *diffusers* module which provides simple pipelines for inference [54].

T2I Hyperparameters. We set our random seed to be 42 for each query to the Stable Diffusion pipeline. When conducting our ablation study, the number of images generated and number of inference steps would be adjusted accordingly, as they are used by the PIP pipeline shown in Figure 3. During evaluation, three images were generated for each adversarial

prompt. Otherwise, all other hyperparameters for image generation follow the default settings from the *diffusers* module.

Optimizer. We used the Optuna framework as our optimizer [55]. More specifically, we ran our tests on three different sampling algorithms, Non-dominated Sorting Genetic Algorithm II (NSGA-II) [56], Tree-structured Parzen Estimator (TPE) [57, 58], and pure random sampling.

VLM Evaluator. We used VQAScore as it can return a text-image alignment score more efficiently as compared to most alignment metrics [59, 60]. More specifically, we used the CLIP-FlanT5-XL model to process the images generated by the T2I model. VQAScore was also used as one of our evaluation metrics, with higher scores indicating greater text-image alignment.

Even though CLIPScore is more commonly used and can be computed faster than VQAScore [61], it can be very insensitive to fine-grained errors and sentence structure. This is because the underlying CLIP model can sometimes behave like a Bag-of-Words and misattribute prompts like " a horse riding an astronaut" as " an astronaut riding a horse" [62]. As a result, CLIPScore may give high scores even for prompts that miss specific details or fail to understand compositional relationships [63, 64]. Conversely, VQAScore is able to capture granularity and compositional relationships in real-world prompts.

Evaluation Metrics. We evaluated our generated images based on their alignment and quality. For the untargeted and targeted attack modes, we rate the images based on their alignment to the original prompt and the removed concept respectively.

We evaluated the effectiveness of the perturbed prompt constructed by the PIP pipeline using Davidsonian Scene Graph (DSG) [65], VQAScore [59], and CLIPScore [61]. DSG is a fine-grained evaluation framework that assesses text-image faithfulness using QGA (question generation and answering), whereby atomic and unique questions generated by a large language model (LLM) are supplied to a VLM for visual question answering (VQA). In our work, we used LLaMA-3.2-3B [66] and a VQA-finetuned mPLUG [67] as our LLM and VLM respectively for DSG. We also ignore the dependency graph generated by DSG during evaluation. Higher scores indicate better text-image alignment for the three metrics.

On the other hand, we used blind image quality assessment (BIQA) metrics such as Language-Image Quality Evaluator (LIQE) [68] and Multi-Scale Image Quality Transformer (MUSIQ) [69] to quantify the quality of the images generated. These metrics help us identify whether the adversarial images are still "usable" and free from artefacts, noise, blur and distortion. Simply put, we want to determine whether our generated image remains of high quality even after the prompt perturbation. LIQE rates images on a continuous five-point likert scale, with each integer score from 1 to 5 referring to bad, poor, fair, good, and perfect respectively. MUSIQ returns an image quality score that ranges from 0 to 100, with 100 being the highest quality.

4.2 Datasets

CIFAR10. For each class in the CIFAR10 dataset [70], we used the default CLIP template "`A photo of a {class}`" [23] to craft the ten prompts. Since the prompt only has one concept, the untargeted attack would have a similar objective to a targeted attack which is to eliminate traces of the class from the generated image. As such, we will only be testing the untargeted attack pipeline for the CIFAR10 dataset.

COCO Captions. Our ablation study used a curated subset of the original COCO Captions 2017 dataset [71, 72] to conduct a comprehensive assessment of the PIP pipeline in both untargeted and targeted attack modes. The dataset contains a total of eighty classes, and each prompt describes at least one class. We first pruned the original validation dataset to obtain a bijection of caption and ground truth image pairs, then retrieved only the first two hundred captions to be used as the final prompt dataset. Note that the prompts may already contain punctuations initially, such as periods and dashes.

Ablation Study Hyperparameters. We have outlined the following hyperparameters that were used for our ablation study in Table 2. We used a smaller value of n for the COCO Captions dataset to test whether our attack can still achieve satisfactory results even under query-efficient scenarios. We also set $t = 20$ as the baseline for COCO Captions dataset to speed up the PIP pipeline by providing an approximation to the expected adversarial image that will be generated with 50 inference steps.

	CIFAR10	**COCO Captions**
n	{50, 100, 150, 200, 250*}	{50*, 100, 150, 200, 250}
m	{1, 2, 3*, 4, 5}	{1, 2, 3*, 4, 5}
k	{1, 2, 3*, 4, 5}	{1, 2, 3*, 4, 5}
t	{10, 20, 30, 40, 50*}	{10, 20*, 30, 40, 50}
Optimizer	{NSGA-II*, TPE, Random}	{NSGA-II*, TPE, Random}

Table 2: Overview of ablation study conducted for the CIFAR10 and the pruned variant of COCO Captions dataset. * denotes the baseline hyperparameters used.

4.3 Experimental Results

The PIP pipeline can successfully alter the expected semantic meaning of the images generated by Stable Diffusion using only punctuation injections. We summarize our findings from the ablation study in Tables 3-5, where scores in red yield the lowest alignment, while scores in green achieve the highest image quality. The full results for all experiments are detailed in Annex C.

	Alignment			Quality	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original Prompt	0.884444	0.922451	23.635901	4.729111	72.991061
Baseline Attack	0.720000	0.514931	20.009706	4.182942	70.149834

Table 3: Alignment and quality scores obtained for untargeted attack on CIFAR10.

	Alignment			Quality	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original Prompt	0.854979	0.755701	26.705855	4.658470	72.326824
Baseline Attack	0.746633	0.563760	25.100362	4.509142	71.468017

Table 4: Alignment and quality scores obtained for untargeted attack on COCO Captions.

	Alignment			Quality	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original Prompt	0.811718	0.831498	20.055960	4.658251	72.388425
Baseline Attack	0.677728	0.700557	19.063373	4.508994	71.478761

Table 5: Alignment and quality scores obtained for targeted attack on COCO Captions.

In general, as n and m increase, generated images show lower alignment to the original prompt, but this comes at the expense of image quality. Increasing m can also cause the attack to be more perceptible as more punctuations are injected. To increase the attack effectiveness further, one should also use the same m and t values for crafting the adversarial prompt as well as generating the adversarial images as our final output. Amongst the three optimizers we considered for our ablation study, we found that TPE sampler performs the best for both datasets.

5 Conclusion

In this study, we proposed punctuation injections as an adversarial perturbation to a clean prompt, and demonstrated how it can effectively evaluate the robustness of T2I models. We also designed the PIP pipeline, a method to automatically find a set of punctuations and their

injection locations for any black-box T2I model. Our experiments have shown that injecting more punctuations into a clean prompt has a higher chance of fooling the T2I model, at the expense of degrading the quality of the generated image.

Limitations. Due to the large number of queries needed to ensure that the punctuation permutation chosen by the PIP pipeline indeed does mislead the victim T2I model, it may take quite some time to run. If given knowledge of the victim model's architecture and weights, we suggest distilling the victim model first to reduce the number of inference steps required for PIP pipeline and expedite the image generation process [53, 73, 74, 75], especially when using a dataset with a large number of prompts.

Future Work. Interestingly, by setting our objective to increase alignment of the generated image to the original prompt, our PIP pipeline can also be used for prompt engineering, making it extremely flexible. Additional experiments would need to be conducted to truly determine the pipeline's effectiveness in prompt tuning.

Due to limited compute resources, we only used one victim model, Stable Diffusion 1.4, for our ablation study. Instead, we tested our attack on both simple and complex prompts to demonstrate its effectiveness. We would like to test the PIP pipeline on other model architectures and black-box models as another direction for future research.

References

[1] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 10684-10695).

[2] Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., ... & Rombach, R. (2024). Scaling rectified flow transformers for high-resolution image synthesis. In Forty-first international conference on machine learning.

[3] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., ... & Sutskever, I. (2021). Zero-shot text-to-image generation. In International conference on machine learning (pp. 8821-8831). Pmlr.

[4] Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., & Chen, M. (2022). Hierarchical text-conditional image generation with clip latents. arXiv preprint arXiv:2204.06125, 1(2), 3.

[5] Labs, B. F. (2024). FLUX. https://github.com/black-forest-labs/flux

[6] Labs, B. F., Batifol, S., Blattmann, A., Boesel, F., Consul, S., Diagne, C., ... & Smith, L. (2025). FLUX. 1 Kontext: Flow Matching for In-Context Image Generation and Editing in Latent Space. arXiv preprint arXiv:2506.15742.

[7] Poppi, S., Poppi, T., Cocchi, F., Cornia, M., Baraldi, L., & Cucchiara, R. (2024). Safe-CLIP: Removing NSFW concepts from vision-and-language models. In European Conference on Computer Vision (pp. 340-356). Cham: Springer Nature Switzerland.

[8] Schramowski, P., Brack, M., Deiseroth, B., & Kersting, K. (2023). Safe latent diffusion: Mitigating inappropriate degeneration in diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 22522-22531).

[9] Yang, Y., Gao, R., Yang, X., Zhong, J., & Xu, Q. (2024). Guardt2i: Defending text-to-image models from adversarial prompts. Advances in neural information processing systems, 37, 76380-76403.

[10] Liu, R., Khakzar, A., Gu, J., Chen, Q., Torr, P., & Pizzati, F. (2024). Latent guard: a safety framework for text-to-image generation. In European Conference on Computer Vision (pp. 93-109). Cham: Springer Nature Switzerland.

[11] Wu, Z., Gao, H., Wang, Y., Zhang, X., & Wang, S. (2024). Universal prompt optimizer for safe text-to-image generation. arXiv preprint arXiv:2402.10882.

[12] Zhang, J., Xu, Z., Cui, S., Meng, C., Wu, W., & Lyu, M. R. (2023). On the robustness of latent diffusion models. *arXiv preprint arXiv:2306.08257*.

[13] Zhang, C., Hu, M., Li, W., & Wang, L. (2024). Adversarial attacks and defenses on text-to-image diffusion models: A survey. Information Fusion, 102701.

[14] Zhuang, H., Zhang, Y., & Liu, S. (2023). A pilot study of query-free adversarial attack against stable diffusion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 2385-2392).

[15] Gao, H., Zhang, H., Dong, Y., & Deng, Z. (2023). Evaluating the robustness of text-to-image diffusion models against real-world attacks. arXiv preprint arXiv:2306.13103.

[16] Du, C., Li, Y., Qiu, Z., & Xu, C. (2023). Stable diffusion is unstable. Advances in Neural Information Processing Systems, 36, 58648-58669.
[17] Du, C., Wang, T., Zhang, K., Luo, W., Ma, L., Liu, W., & Cao, X. (2023). Punctuation-level attack: Single-shot and single punctuation can fool text models. Advances in Neural Information Processing Systems, 36, 49312-49324.
[18] Salman, H., Khaddaj, A., Leclerc, G., Ilyas, A., & Madry, A. (2023). Raising the cost of malicious ai-powered image editing. arXiv preprint arXiv:2302.06588.
[19] Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E. L., ... & Norouzi, M. (2022). Photorealistic text-to-image diffusion models with deep language understanding. Advances in neural information processing systems, 35, 36479-36494.
[20] Baldridge, J., Bauer, J., Bhutani, M., Brichtova, N., Bunner, A., Castrejon, L., ... & Malik, C. (2024). Imagen 3. arXiv preprint arXiv:2408.07009.
[21] Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., ... & Rombach, R. (2023). Sdxl: Improving latent diffusion models for high-resolution image synthesis. arXiv preprint arXiv:2307.01952.
[22] Ho, J., Jain, A., & Abbeel, P. (2020). Denoising diffusion probabilistic models. Advances in neural information processing systems, 33, 6840-6851.
[23] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. In International conference on machine learning (pp. 8748-8763). PmLR.
[24] Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., & Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In International conference on machine learning (pp. 2256-2265). pmlr.
[25] Ronneberger, O., Fischer, P., & Brox, T. (2015). 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 (pp. 234-241). Springer international publishing.
[26] Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes.
[27] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... & Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of machine learning research, 21(140), 1-67.
[28] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2013). Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199.
[29] Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014). Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572.
[30] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083.
[31] Struppek, L., Hintersdorf, D., Friedrich, F., Schramowski, P., & Kersting, K. (2023). Exploiting cultural biases via homoglyphs in text-to-image synthesis. Journal of Artificial Intelligence Research, 78, 1017-1068.
[32] Yang, Y., Hui, B., Yuan, H., Gong, N., & Cao, Y. (2024). Sneakyprompt: Jailbreaking text-to-image generative models. In 2024 IEEE symposium on security and privacy (SP) (pp. 897-912). IEEE.

[33] Liu, H., Wu, Y., Zhai, S., Yuan, B., & Zhang, N. (2023). Riatig: Reliable and imperceptible adversarial text-to-image generation with natural prompts. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 20585-20594).
[34] Chin, Z. Y., Jiang, C. M., Huang, C. C., Chen, P. Y., & Chiu, W. C. (2023). Prompting4debugging: Red-teaming text-to-image diffusion models by finding problematic prompts. arXiv preprint arXiv:2309.06135.
[35] Zhan, P., Yang, J., Wang, H., Zheng, C., & Wang, L. (2024). Rethinking word-level adversarial attack: The trade-off between efficiency, effectiveness, and imperceptibility. In Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024) (pp. 14037-14052).
[36] Zang, Y., Qi, F., Yang, C., Liu, Z., Zhang, M., Liu, Q., & Sun, M. (2019). Word-level textual adversarial attacking as combinatorial optimization. arXiv preprint arXiv:1910.12196.
[37] Formento, B., Foo, C. S., Tuan, L. A., & Ng, S. K. (2023). Using punctuation as an adversarial attack on deep learning-based NLP systems: An empirical study. In Findings of the association for computational linguistics: EACL 2023 (pp. 1-34).
[38] Peters, P. (2013). The Cambridge dictionary of English grammar.
[39] Cambridge University Press & Assessment (n.d.). Punctuation. https://dictionary.cambridge.org/grammar/british-grammar/punctuation
[40] Truss, L. (2003). Eats, shoots & leaves: The zero tolerance approach to punctuation.
[41] Zhang, C., Wang, L., & Liu, A. (2024). Revealing vulnerabilities in stable diffusion via targeted attacks. arXiv preprint arXiv:2401.08725.
[42] Gage, P. (1994). A new algorithm for data compression. The C Users Journal, 12(2), 23-38.
[43] Schuster, M., & Nakajima, K. (2012). Japanese and korean voice search. In 2012 IEEE international conference on acoustics, speech and signal processing (ICASSP) (pp. 5149-5152). IEEE.
[44] Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., ... & Dean, J. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144.
[45] Kudo, T., & Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. arXiv preprint arXiv:1808.06226.
[46] Li, S., Le, H., Xu, J., & Salzmann, M. (2024). Enhancing Compositional Text-to-Image Generation with Reliable Random Seeds. arXiv preprint arXiv:2411.18810.
[47] Xu, K., Zhang, L., & Shi, J. (2025). Good seed makes a good crop: Discovering secret seeds in text-to-image diffusion models. In 2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) (pp. 3024-3034). IEEE.
[48] AUTOMATIC1111. (2022). Stable Diffusion Web UI. https://github.com/AUTOMATIC1111/stable-diffusion-webui
[49] Midjourney. (2021). Midjourney. https://www.midjourney.com
[50] Anlatan. (2022). NovelAI. https://novelai.net

[51] Chen, J., Xue, S., Zhao, Y., Yu, J., Paul, S., Chen, J., ... & Xie, E. (2025). Sana-sprint: One-step diffusion with continuous-time consistency distillation. arXiv preprint arXiv:2503.09641.

[52] Sauer, A., Lorenz, D., Blattmann, A., & Rombach, R. (2024). Adversarial diffusion distillation. In European Conference on Computer Vision (pp. 87-103). Cham: Springer Nature Switzerland.

[53] Yin, T., Gharbi, M., Park, T., Zhang, R., Shechtman, E., Durand, F., & Freeman, B. (2024). Improved distribution matching distillation for fast image synthesis. Advances in neural information processing systems, 37, 47455-47487.

[54] Von Platen, P., Patil, S., Lozhkov, A., Cuenca, P., Lambert, N., Rasul, K., Davaadorj, M., Nair, D., Paul, S., Berman, W., Xu, Y., Liu, S., & Wolf, T. (2022). Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers

[55] Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining (pp. 2623-2631).

[56] Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. A. M. T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE transactions on evolutionary computation, 6(2), 182-197.

[57] Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. Advances in neural information processing systems, 24.

[58] Ozaki, Y., Tanigaki, Y., Watanabe, S., & Onishi, M. (2020). Multiobjective tree-structured parzen estimator for computationally expensive optimization problems. In Proceedings of the 2020 genetic and evolutionary computation conference (pp. 533-541).

[59] Lin, Z., Pathak, D., Li, B., Li, J., Xia, X., Neubig, G., ... & Ramanan, D. (2024). Evaluating text-to-visual generation with image-to-text generation. In European Conference on Computer Vision (pp. 366-384). Cham: Springer Nature Switzerland.

[60] Li, B., Lin, Z., Pathak, D., Li, J., Fei, Y., Wu, K., ... & Ramanan, D. (2024). Genai-bench: Evaluating and improving compositional text-to-visual generation. arXiv preprint arXiv:2406.13743.

[61] Hessel, J., Holtzman, A., Forbes, M., Bras, R. L., & Choi, Y. (2021). Clipscore: A reference-free evaluation metric for image captioning. arXiv preprint arXiv:2104.08718.

[62] Yuksekgonul, M., Bianchi, F., Kalluri, P., Jurafsky, D., & Zou, J. (2022). When and why vision-language models behave like bags-of-words, and what to do about it?. arXiv preprint arXiv:2210.01936.

[63] Ahmadi, S., & Agrawal, A. (2023). An examination of the robustness of reference-free image captioning evaluation metrics. arXiv preprint arXiv:2305.14998.

[64] Hartwig, S., Engel, D., Sick, L., Kniesel, H., Payer, T., Poonam, P., ... & Ropinski, T. (2024). A Survey on Quality Metrics for Text-to-Image Generation. arXiv preprint arXiv:2403.11821.

[65] Cho, J., Hu, Y., Garg, R., Anderson, P., Krishna, R., Baldridge, J., ... & Wang, S. (2023). Davidsonian scene graph: Improving reliability in fine-grained evaluation for text-to-image generation. arXiv preprint arXiv:2310.18235.

[66] Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., ... & Vasic, P. (2024). The llama 3 herd of models. arXiv preprint arXiv:2407.21783.

[67] Li, C., Xu, H., Tian, J., Wang, W., Yan, M., Bi, B., ... & Si, L. (2022). mplug: Effective and efficient vision-language learning by cross-modal skip-connections. arXiv preprint arXiv:2205.12005.

[68] Zhang, W., Zhai, G., Wei, Y., Yang, X., & Ma, K. (2023). Blind image quality assessment via vision-language correspondence: A multitask learning perspective. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 14071-14081).

[69] Ke, J., Wang, Q., Wang, Y., Milanfar, P., & Yang, F. (2021). Musiq: Multi-scale image quality transformer. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 5148-5157).

[70] Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images.

[71] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13 (pp. 740-755). Springer International Publishing.

[72] Chen, X., Fang, H., Lin, T. Y., Vedantam, R., Gupta, S., Dollár, P., & Zitnick, C. L. (2015). Microsoft coco captions: Data collection and evaluation server. arXiv preprint arXiv:1504.00325.

[73] Meng, C., Rombach, R., Gao, R., Kingma, D., Ermon, S., Ho, J., & Salimans, T. (2023). On distillation of guided diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 14297-14306).

[74] Luo, S., Tan, Y., Huang, L., Li, J., & Zhao, H. (2023). Latent consistency models: Synthesizing high-resolution images with few-step inference. arXiv preprint arXiv:2310.04378.

[75] Xie, S., Xiao, Z., Kingma, D., Hou, T., Wu, Y. N., Murphy, K. P., ... & Gao, R. (2024). Em distillation for one-step diffusion models. Advances in Neural Information Processing Systems, 37, 45073-45104.

[76] CompVis. (2022). Stable Diffusion Safety Checker. https://huggingface.co/CompVis/stable-diffusion-safety-checker

A Computational Cost

In Section 4, we used only one 16GB NVIDIA RTX A4000 GPU for each experiment in our ablation study. We provide a comparison on the number of GPU hours spent for each parameter tested in Figure 4 and Table 6, with the time taken for both the untargeted and targeted attack variants of the PIP pipeline being relatively similar. Increasing n, k, and t result in more time needed to craft the adversarial prompt, while modifying m or the optimizer has virtually no effect on the duration.

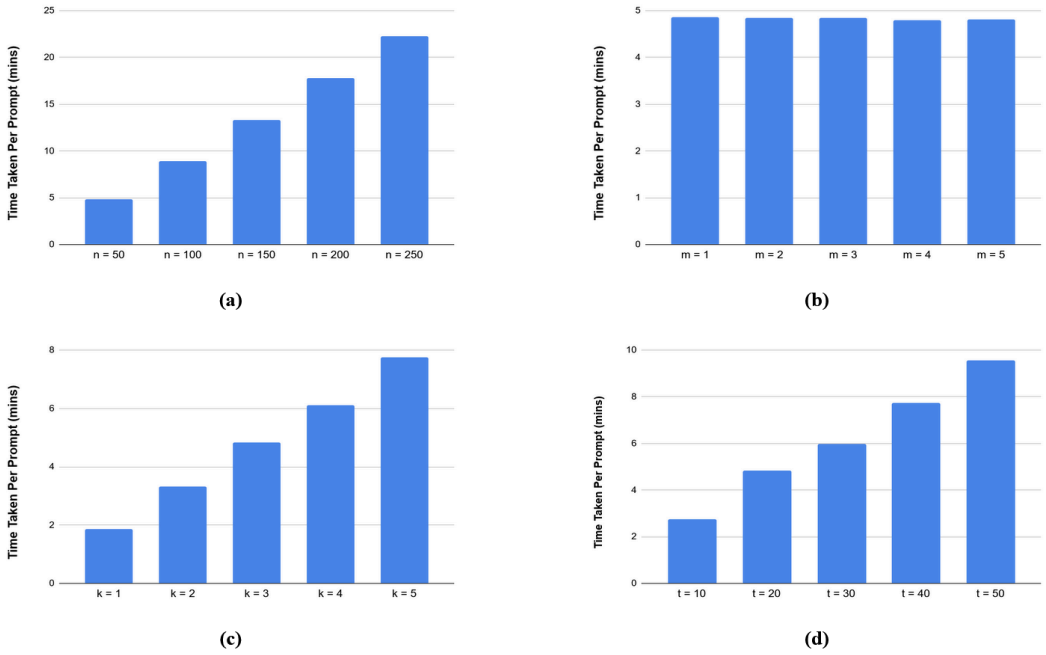

Figure 4: Time taken to craft each adversarial prompt for COCO Captions during the ablation study when using different **(a)** number of permutations tested, **(b)** number of punctuations injected, **(c)** number of images generated per trial, and **(d)** number of inference steps for image generation.

	NSGA-II	Random	TPE
Time Taken Per Prompt (mins)	4.84	4.91	5.15

Table 6: Time taken to craft each adversarial prompt for COCO Captions when using different optimizers.

B Ethical Considerations

During our experiments, we disabled the safety checker that comes with Stable Diffusion 1.4, as it might replace our generated image with purely black images when it detects unsafe concepts [76]. This might trick both the VLM evaluator for the PIP pipeline and our automatic evaluation metrics into thinking that the attack succeeded, since the generated

black image would not be well-aligned with most prompts. Instead, human researchers have manually checked through the prompts used and images generated to ensure that they do not contain inappropriate content such as nudity and violence.

In a real-world scenario where the safety checker cannot be disabled, we would like to suggest two possible modifications to the PIP pipeline. When the safety checker replaces our generated image with a constant placeholder which can be detected, the VLM evaluator should: (1) return the worst score for placeholder images, or (2) only evaluate images which are not the placeholder. Worst score here refers to maximum alignment with the original prompt for an untargeted attack, or the removed concept for a targeted attack. Both ideas are effectively the same when only one image is generated for each trial.

C Detailed Results for Ablation Study

In this section, we provide the scores obtained by each set of hyperparameters used in the ablation study. For all tables below, scores in red yield the lowest alignment, while scores in green achieve the highest image quality. The next best result is then underlined. Baseline hyperparameters that are used for a particular dataset are denoted by *.

C.1 CIFAR10 (Untargeted Attack)

	Alignment			Quality	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original	0.884444	0.922451	23.635901	4.729111	72.991061
n = 50	0.806667	0.781082	22.760914	<u>4.600941</u>	71.704687
***n* = 100**	0.803333	0.679166	22.296849	4.369389	71.686037
***n* = 150**	<u>0.675556</u>	0.579510	19.235210	4.080088	69.744231
***n* = 200**	0.654444	<u>0.551272</u>	20.243702	4.511708	<u>72.220750</u>
n* = 250	0.720000	0.514931	<u>20.009706</u>	4.182942	70.149834

Table 7: Alignment and quality scores obtained for untargeted attack on CIFAR10 when modifying the number of permutations tested, *n*.

	Alignment			Quality	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**

Original	0.884444	0.922451	23.635901	4.729111	72.991061
***m* = 1**	0.765556	0.740507	22.799349	4.482741	71.379456
***m* = 2**	0.704444	0.625241	20.867505	4.123678	69.161129
m* = 3	0.720000	0.514931	20.009706	4.182942	70.149834
***m* = 4**	0.570000	0.435958	18.552473	4.205117	69.714312
***m* = 5**	0.452222	0.359514	17.211403	3.976169	67.935306

Table 8: Alignment and quality scores obtained for untargeted attack on CIFAR10 when modifying the number of punctuations injected, *m*.

	Alignment			**Quality**	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original	0.884444	0.922451	23.635901	4.729111	72.991061
***k* = 1**	0.681111	0.574637	19.918288	4.236962	70.414213
***k* = 2**	0.633333	0.554285	20.576550	4.259339	69.766714
k* = 3	0.720000	0.514931	20.009706	4.182942	70.149834
***k* = 4**	0.665556	0.570572	19.933012	4.193563	70.340561
***k* = 5**	0.622222	0.540120	20.383527	4.301646	71.611156

Table 9: Alignment and quality scores obtained for untargeted attack on CIFAR10 when modifying the number of images generated per trial, *k*.

	Alignment			**Quality**	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original	0.884444	0.922451	23.635901	4.729111	72.991061
***t* = 10**	0.697778	0.618070	20.654401	3.963813	68.269591
***t* = 20**	0.691111	0.635546	20.735337	4.427250	71.384500
***t* = 30**	0.642222	0.567090	20.406130	4.086661	69.418954
***t* = 40**	0.670000	0.543925	20.338652	4.495502	73.056425

t* = 50	0.720000	0.514931	20.009706	4.182942	70.149834

Table 10: Alignment and quality scores obtained for untargeted attack on CIFAR10 when modifying the number of inference steps for image generation, *t*.

	Alignment			Quality	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original	0.884444	0.922451	23.635901	4.729111	72.991061
NSGA-II*	0.720000	0.514931	20.009706	4.182942	70.149834
Random	0.673333	0.624692	21.252783	4.373632	71.061949
TPE	0.453333	0.351051	17.373290	4.044262	70.432901

Table 11: Alignment and quality scores obtained for untargeted attack on CIFAR10 when modifying the optimizer used.

C.2 COCO Captions (Untargeted Attack)

	Alignment			Quality	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original	0.854979	0.755701	26.705855	4.658470	72.326824
n = 50*	0.746633	0.563760	25.100362	4.509142	71.468017
***n* = 100**	0.737356	0.554705	24.916759	4.530706	71.696848
***n* = 150**	0.731488	0.526218	24.663002	4.571089	72.261664
***n* = 200**	0.717479	0.511284	24.567894	4.505443	71.549905
***n* = 250**	0.709627	0.504387	24.667300	4.530701	71.642692

Table 12: Alignment and quality scores obtained for untargeted attack on COCO Captions when modifying the number of permutations tested, *n*.

	Alignment			Quality	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original	0.854979	0.755701	26.705855	4.658470	72.326824

***m* = 1**	0.793234	0.651017	25.879535	4.605308	72.209614
***m* = 2**	0.770425	0.608886	25.480140	4.565949	71.960485
m* = 3	0.746633	0.563760	25.100362	4.509142	71.468017
***m* = 4**	0.746990	0.539512	24.949835	4.473899	71.532156
***m* = 5**	0.715145	0.515841	24.395660	4.516934	71.472216

Table 13: Alignment and quality scores obtained for untargeted attack on COCO Captions when modifying the number of punctuations injected, *m*.

	Alignment			**Quality**	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original	0.854979	0.755701	26.705855	4.658470	72.326824
***k* = 1**	0.763571	0.612549	25.369776	4.521721	71.693098
***k* = 2**	0.746862	0.588589	25.310867	4.497991	71.359070
k* = 3	0.746633	0.563760	25.100362	4.509142	71.468017
***k* = 4**	0.751317	0.574464	25.166968	4.525051	71.566519
***k* = 5**	0.749569	0.573822	25.029479	4.518127	71.674219

Table 14: Alignment and quality scores obtained for untargeted attack on COCO Captions when modifying the number of images generated per trial, *k*.

	Alignment			**Quality**	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original	0.854979	0.755701	26.705855	4.658470	72.326824
***t* = 10**	0.772071	0.636188	25.466921	4.476341	71.447270
t* = 20	0.746633	0.563760	25.100362	4.509142	71.468017
***t* = 30**	0.745491	0.554565	24.980042	4.500966	71.561947
***t* = 40**	0.732546	0.521585	24.877644	4.525747	71.739074
***t* = 50**	0.718169	0.491291	24.665578	4.531490	71.427910

Table 15: Alignment and quality scores obtained for untargeted attack on COCO Captions when modifying the number of inference steps for image generation, *t*.

	Alignment			Quality	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original	0.854979	0.755701	26.705855	4.658470	72.326824
NSGA-II*	0.746633	0.563760	25.100362	4.509142	71.468017
Random	0.746953	0.571835	25.196069	4.510854	71.470523
TPE	0.752584	0.561750	25.142140	4.550890	71.736491

Table 16: Alignment and quality scores obtained for untargeted attack on COCO Captions when modifying the optimizer used.

C.3 COCO Captions (Targeted Attack)

	Alignment			Quality	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original	0.811718	0.831498	20.055960	4.658251	72.388425
n = 50*	0.677728	0.700557	19.063373	4.508994	71.478761
***n* = 100**	0.662718	0.682482	18.985264	4.483248	71.377646
***n* = 150**	0.635611	0.672416	18.817841	4.469194	71.407118
***n* = 200**	0.645736	0.673832	18.755478	4.473473	71.285438
***n* = 250**	0.645101	0.665895	18.914293	4.490273	71.384135

Table 17: Alignment and quality scores obtained for targeted attack on COCO Captions when modifying the number of permutations tested, *n*.

	Alignment			Quality	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original	0.811718	0.831498	20.055960	4.658251	72.388425
***m* = 1**	0.744563	0.765761	19.570381	4.600307	72.079793

***m* = 2**	0.725493	0.737560	19.332193	4.562685	71.641908
m* = 3	0.677728	0.700557	19.063373	4.508994	71.478761
***m* = 4**	0.667847	0.687865	19.111307	4.493300	71.532794
***m* = 5**	0.647088	0.665777	18.957581	4.402326	70.833662

Table 18: Alignment and quality scores obtained for targeted attack on COCO Captions when modifying the number of punctuations injected, *m*.

	Alignment			**Quality**	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original	0.811718	0.831498	20.055960	4.658251	72.388425
***k* = 1**	0.733882	0.749243	19.564884	4.520680	71.486244
***k* = 2**	0.698635	0.714564	19.200053	4.514787	71.529866
k* = 3	0.677728	0.700557	19.063373	4.508994	71.478761
***k* = 4**	0.678980	0.710274	19.053515	4.497386	71.443591
***k* = 5**	0.685989	0.706849	19.007264	4.488327	71.271374

Table 19: Alignment and quality scores obtained for targeted attack on COCO Captions when modifying the number of images generated per trial, *k*.

	Alignment			**Quality**	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original	0.811718	0.831498	20.055960	4.658251	72.388425
***t* = 10**	0.717564	0.747806	19.488834	4.504723	71.511990
t* = 20	0.677728	0.700557	19.063373	4.508994	71.478761
***t* = 30**	0.684447	0.690299	19.058665	4.518732	71.455358
***t* = 40**	0.659288	0.664801	18.926036	4.506169	71.367940
***t* = 50**	0.646290	0.643492	18.902115	4.499649	71.260772

Table 20: Alignment and quality scores obtained for targeted attack on COCO Captions when modifying the number of inference steps for image generation, *t*.

	Alignment			Quality	
	DSG (↓)	**VQAScore (↓)**	**CLIPScore (↓)**	**LIQE (↑)**	**MUSIQ (↑)**
Original	0.811718	0.831498	20.055960	4.658251	72.388425
NSGA-II*	0.677728	0.700557	19.063373	4.508994	71.478761
Random	0.680794	0.704722	19.196144	4.489976	71.193173
TPE	0.660299	0.692827	18.947473	4.511160	71.638944

Table 21: Alignment and quality scores obtained for targeted attack on COCO Captions when modifying the optimizer used.