Character As Pixels: A Controllable Prompt Adversarial Attacking Framework for Black-Box Text Guided Image Generation Models

Ziyi Kou, **Shichao Pei**, **Yijun Tian** and **Xiangliang Zhang***
University of Notre Dame
{zkou, spei2, yijun.tian, xzhang33}@nd.edu

Abstract

In this paper, we study a controllable prompt adversarial attacking problem for text guided image generation (Text2Image) models in the black-box scenario, where the goal is to attack specific visual subjects (e.g., changing a brown dog to white) in a generated image by slightly, if not imperceptibly, perturbing the characters of the driven prompt (e.g., "brown" to "br0wn"). Our study is motivated by the limitations of current Text2Image attacking approaches that still rely on manual trials to create adversarial prompts. To address such limitations, we develop CharGrad, a character-level gradient based attacking framework that replaces specific characters of a prompt with pixel-level similar ones by interactively learning the perturbation direction for the prompt and updating the attacking examiner for the generated image based on a novel proxy perturbation representation for characters. We evaluate CharGrad using the texts from two public image captioning datasets. Results demonstrate that CharGrad outperforms existing text adversarial attacking approaches on attacking various subjects of generated images by black-box Text2Image models in a more effective and efficient way with less perturbation on the characters of the prompts.

1 Introduction

Text-guided image generation (Text2Image) has been receiving much attention from both academia and industry fields due to its powerful capability of generating high-quality images based on textual descriptions as driven prompts [Ramesh *et al.*, 2022]. However, such applications usually suffer from the robustness issue where a prompt with one or several perturbed characters could dramatically change the visual content of the generated image. For example, given a prompt as "*Big Ben in the rain*", replacing the second "*B*" with the Cyrillic "B" results in generating an image with mushrooms in the wild rather than the clock tower at London city [Struppek *et al.*, 2022a]. These vulnerabilities can be utilized by some malicious software (e.g., a searching plugin with autocomplete function) that intentionally modifies prompts from the users of Text2Image applications, which severally affects the user experience and the performance of the applications. Therefore, we are motivated to investigate the vulnerabilities of Text2Image models. Concretely, we aim to answer whether a Text2Image model can be attacked to generate different visual subjects by slightly, if not imperceptibly, perturbing the characters of the driven prompt.

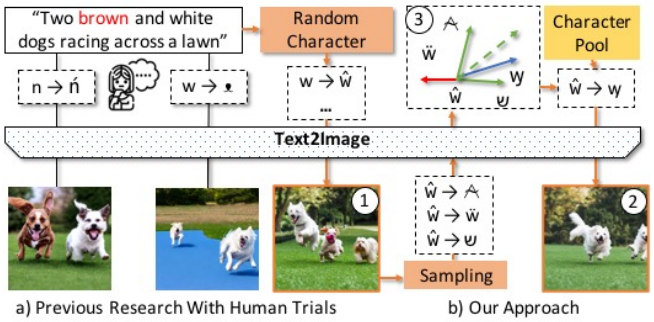

Figure 1: Comparison of Text2Image Attacking

Recently, Struppek *et al.* [Struppek *et al.*, 2022a] observed that replacing some characters of a given prompt with their homoglyphs[1] can sometimes change the visual content of the generated image. However, such manual trials are firstly inefficient, and secondly cannot attack the images in an controllable manner as the attacking effect of a homoglyph is not quantitatively estimated. For example, given the prompt in Figure 1a, the target is to attack the "brown" attribute of the "dogs" by changing a character within "brown" to another similar character, such that a generated image from the Text2Image model contains dogs not in brown color. As shown in Figure 1a, the manual trials like changing n→\U(0144) and w→\U(1D25) didn't work. The generated images still include a brown dog, or have unexpected visual subjects modified (e.g., lawn in blue). It is unknown which character as a replacement for a prompt can generate an image with only the target subject attacked while the other subjects the same, i.e., in a *controllable* way. While an optimal adversarial prompt could be obtained by exhaustively trying the

*corresponding author

[1]https://en.wikipedia.org/wiki/Homoglyph

replacement for each character in a prompt with all available Unicode characters around the world, such an approach is not practical in the real-world setting due to the limited budget of using Text2Image models (e.g., limited quotes for online API) and the exponential increase on searching space (e.g., nearly 149,186 Unicode characters).

Motivated by the above observations, we develop CharGrad, a **char**acter-level **grad**ient based attacking framework for Text2Image models. We show the overall workflow of CharGrad in Figure 1b with orange arrows. In our solution, CharGrad initializes an adversarial prompt by replacing the target characters (e.g., "brown" in Figure 1) in the prompt with random characters from a large-scale character pool. For each position of the replacement, CharGrad then samples a few new characters as alternations to estimate the changing directions on the generated images and approximate the *gradient* of the optimal perturbation direction (e.g., the dashed green arrow within ③ of Figure 1). Such gradient is further leveraged to retrieve the candidate character from the character pool that is matched based on the aligned direction (e.g., the blue arrow within ③ of Figure 1). Recently, Ye *et al.* [Ye *et al.*, 2022a] proposed LeapAttack, a gradient based attacking framework that generates adversarial texts in the *hard-label* classification setting by replacing the words of original texts with their *synonyms*. However, their approach is not applicable to our problem due to two unresolved technical challenges below.

Character-Level Gradient Estimation. LeapAttack estimates the gradient of the perturbation direction for a text by modeling the replacement between two words as the disparity of their semantic embeddings that are extracted from pretrained word encoding models (e.g., BERT [Devlin *et al.*, 2018]). This is because different words could be encoded into a uniform semantic space where the disparity of different word embeddings are comparable with each other to derive the correct perturbation direction for the text [Tian *et al.*, 2023]. However, the Unicode characters in our problem have no explicit *semantic* meanings as they may come from different languages (e.g., "A" and \U(00C5)) or even different domains (e.g., "a" and α). While a character-level embedding can be generated by a word or character encoding model (e.g., CANINE [Clark *et al.*, 2022]), the disparity between such embeddings does not share a uniform hidden space and thus fail to represent the optimal gradient of the perturbation direction for a prompt. Therefore, how to generate comparable character-level embeddings in a uniform hidden space remains as a challenge.

Lack of Arbitrary Evaluation Criteria for Attacking. LeapAttack focuses on the attacking problem with binary classification tasks. Therefore, a target binary classification model has a universally usable attack evaluation criterion: an adversarial attack text as input is considered as successful if the classification result (e.g., "-1") is different from the result (e.g., "1") with the original text. However, there is no such arbitrary evaluation criteria for our attacking problem because the outputs of a target Text2Image model are RGB images rather than the binary prediction. For example, the color of the dogs in ① of Figure 1b still partially contains brown hairs. It is difficult to make an automated judgement if the attacking is successful or not. A possible solution is to set hard thresholds for the attacked subjects (e.g., the number of remaining pixels as brown color). But such a one-size-fit-all metric cannot consistently generate optimal adversarial prompts due to the complex semantic contexts, nor practical to real-world attacking scenario when the target subjects become various. Therefore, it is challenging to design arbitrary evaluation criteria for attacking Text2image models and well adapt the criteria on various prompts to achieve the optimal attacking performance.

To address the first challenge, we design a proxy perturbation representation metric to quantitatively estimate the potential attacking effect of different Unicode characters that are further leveraged to estimate the perturbation direction for the prompts. To address the second challenge, we develop an iterative attack examiner updating strategy that firstly builds a uniform visual-guided attack examiner and then adaptively searches more optimal examination bars for different prompts during the attacking process. To our best knowledge, CharGrad is the first character-level gradient based attacking framework to solve the controllable prompt adversarial attacking problem for Text2Image models. We evaluate CharGrad on two public image captioning datasets. The results show that CharGrad outperforms existing text adversarial attacking approaches on both effectively and efficiently attacking various subjects of the generated images with less character-level perturbations.

2 Related Work

2.1 Text Guided Image Generation

In the last few years, diffusion models are widely studied to generate high-quality images by iteratively corrupting and recovering the images with random noise [Ho *et al.*, 2020]. Moreover, the generation process of a diffusion model can be conditioned on the embeddings of natural language prompts (e.g., CLIP [Radford *et al.*, 2021]). For example, Ramesh *et al.* proposed DALL-E 2, a diffusion based Text2Image model that incorporates projected CLIP prompt embeddings as additional context tokens to generate semantically consistent images [Ramesh *et al.*, 2022]. However, only a few research focuses on attacking Text2Image models by generating adversarial prompts. Recently, Struppek *et al.* [Struppek *et al.*, 2022a] manually replaced the characters of a given prompt with their homoglyphs to generate adversarial prompts that change the visual content of the generated images. After that, they further proposed a backdoor attacking algorithm that poisoned the text encoder module of the Text2Image model by forcing the correlation between the injected homoglyphs and the tokens of different prompt [Struppek *et al.*, 2022b]. Our study focuses on the vulnerability of Text2Image models against adversarial attacks, which fools the trained Text2Image model by perturbing the input prompt during the testing phase. Unlike poisoning attacks that alter the prediction of a model by misleading the model parameter in training phase, the adversarial prompt attacking exploit weaknesses of the model without any change on the content of the model. Since Text2Image models are mostly provided with limited

accessible contents (e.g., an online API), we have an even more challenging task in the black-box attack setting. We address the studied problem by designing a novel character-level adversarial prompt attacking framework, which treats the Text2Image models as black-box and generates more effective adversarial prompts than human trials.

2.2 Black-Box Adversarial Attacks to NLP Models

The vulnerability of NLP models against adversarial attacks has been investigated in the black-box setting [Devlin *et al.*, 2018]. Previous research in this field mainly focuses on *soft-label* black-box attacking where the attackers presume the access to the probability score distribution from the target model [Jin *et al.*, 2020; Ren *et al.*, 2019]. However, such assumption usually cannot be held due to the fact that most real-world models are often securely wrapped with only the *hard label predictions* (e.g., Top-1 prediction) exposed. Recently, several initial efforts have been made to study the hard-label black-box text adversarial attacking problem [Maheshwary *et al.*, 2021; Ye *et al.*, 2022b; Ye *et al.*, 2022a]. For example, Ye *et al.* [Ye *et al.*, 2022a] proposed a gradient based attacking framework that replaces different words of a given text with their synonyms by estimating the gradients of the perturbation direction based on the disparity of word embeddings. However, their approach is not applicable to our problem due to 1) the lack of effective metrics to generate comparable character-level embeddings and 2) the lack of arbitrary evaluation criteria for the Text2Image task to judge the success of an adversarial prompt. The proposed CharGrad framework designs a proxy perturbation representation metric and an iterative attack examiner updating strategy to address the above limitations, respectively.

3 Problem Definition

In this section, we formally present the controllable prompt adversarial attacking problem with several definitions below.

Definition 1. Prompt (t): A prompt $t = \{c_1, \ldots, c_N\}$ is a piece of natural language text that contains a sequential of characters where N denotes the total number of characters.

Definition 2. Unicode Character (c): A Unicode character is a unit symbol in the computer system. We define $C = \{c_1, \ldots, c_P\}$ as the character pool that contains all available characters in our paper. More statistic information of $\mathcal{C}$ is provided in the Appendix.

Definition 3. Text2Image Model ($\mathcal{M}$): We define a Text2Image model $\mathcal{M}$ as a black box that takes a prompt t as the input and generates an image x that is semantically associated with t. The process is formally defined as $x = \mathcal{M}(t)$.

Definition 4. Attacking Term (w): The attacking target $w = \{c_1, \ldots, c_M\}$ is one or several specific words from t, which correspond to a subject that is an entity (e.g., "dogs" in Figure 1) or an entity's attribute (e.g., "brown" in Figure 1). $M \in [1, N)$ denotes the total number of characters from w.

Definition 5. Adversarial Prompt ($\ddot{t}$): An adversarial prompt $\ddot{t} = \{\ddot{c}_1, \ldots, \ddot{c}_N\}$ consists of the same sequential of characters as t except the attacking term where some characters within the term are replaced with different characters. We define the perturbed attacking term as $\ddot{w} = \{\ddot{c}_1, \ldots, \ddot{c}_M\}$.

Definition 6. Pixel Difference (D): We firstly retrieve the image of each Unicode character from *fileformat*, an online Unicode collection website that provides normalized gray-scale images for most Unicode characters. Then we define the pixel differences between two characters as

$$D(c_1, c_2) = 1 - \frac{\text{Pixel}(c_1) \cap \text{Pixel}(c_2)}{\text{Pixel}(c_1) \cup \text{Pixel}(c_2)} \tag{1}$$

where $\text{Pixel}(\cdot)$ denotes the coordination set from the image of the given character. Similarly, we define the entire pixel difference between a prompt t and the adversarial prompt $\ddot{t}$ as $D(t, \ddot{t}) = \sum_{i=1}^{N} D(c_i, \ddot{c}_i)$.

Definition 7. Attack Examiner (Φ): To evaluate if an adversarial prompt $\ddot{t}$ can successfully reach the attacking goal, we define an attacker examiner Φ as a binary discriminator. It indicates if $\ddot{t}$ can successfully attack the model $\mathcal{M}$ by controlling the modification of the generated images within the subjects corresponding to w. $\Phi(\ddot{t}, \mathcal{M}) = 1$ if the attack is successful. We will further discuss the detail of designing Φ in Section 4.

Using the definitions above, our controllable prompt adversarial attacking problem is formally defined as:

$$\ddot{t}^* = \arg\min_{\ddot{t}} D(t, \ddot{t}),\ s.t.\ \Phi(\ddot{t}, \mathcal{M}) = 1 \tag{2}$$

where $\ddot{t}^*$ is the optimal adversarial prompt from CharGrad.

4 The Proposed CharGrad Approach

The proposed CharGrad incorporates three key modules in its design, which are elaborated below.

4.1 The *Initializer* Module

The first module is designed to randomly initialize a successful adversarial prompt. That is to randomly perturb w to be $\ddot{w}$, such that the adversarial prompt $\ddot{t}$ can result in a successful attack on $\mathcal{M}$. The *Initializer* thus serves a role for probing the sensitive decision regions. While the initial adversarial prompt may not be the optimal, it provides important guidance for determining the direction of prompt optimization in the challenging black-box attack setting.

In this module, the success of attack is judged by an initial attack examiner Φ_{init}.

Definition 8. Initial Attack Examiner (Φ_{init}): Given a prompt t, an adversarial prompt $\ddot{t}$, and the target Text2Image model $\mathcal{M}$, we define the initial attack examiner Φ_{init} as

$$\Phi_{\text{init}}(t, \ddot{t}) = 1\{\frac{\mathcal{S}[t \backslash w, \mathcal{M}(\ddot{t})]}{\mathcal{S}[w, \mathcal{M}(\ddot{t})]} > \frac{\mathcal{S}[t \backslash w, \mathcal{M}(t)]}{\mathcal{S}[w, \mathcal{M}(t)]}\} \tag{3}$$

where $t \backslash w$ denotes the prompt without the attacking term. $\mathcal{S}$ is a multi-modal alignment metric (e.g., CLIP [Radford *et al.*, 2021]) that evaluates the consistency between the semantic words (e.g., prompts, attacking term) and the generated images. Within the indicator function $1\{\cdot\}$ of Equation 3, we further define the left part as *attack examination score* (Att_{score}) that quantitatively indicates the effectiveness of $\ddot{t}$.

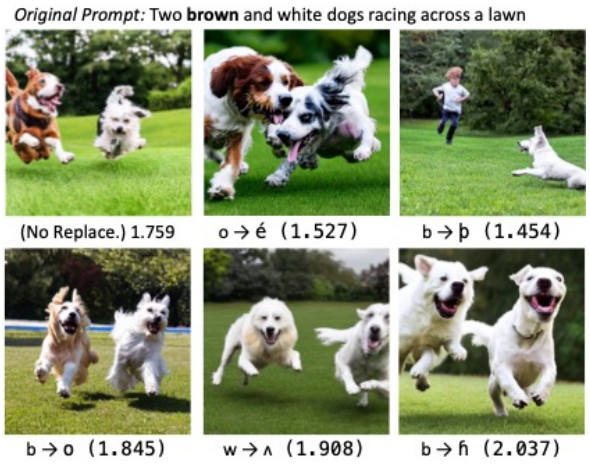

Figure 2: Replacement of characters in attacking term ("brown") and the resulted Att_{score}. Attack failed (Φ_{init}=0) for the top middle and right examples, and succeeded (Φ_{init}=1) for the bottom examples.

We show several examples in Figure 2 with their Att_{score} for the "brown" subject to illustrate the effect of Φ_{init}. Based on the above definition, a higher value of Att_{score} indicates more effective attack to the target subject and less impact on the remaining subjects in the prompt. The bottom three examples in Figure 2 have higher Att_{score} compared to the original example without any replacement. The images in these three examples include no brown dogs, but two dogs in other color, indicating that the attacks in these examples are successful. However, the other two examples on the top have low Att_{score} and do not meet the attack goal.

There is no need to run a sophisticated process to obtain the initial adversarial prompt $\ddot{t}_0$. In particular, the *Initializer* can obtain $\ddot{t}_0$ by randomly replacing each character from the attacking term of t with another character from the character pool. Without the consideration on the pixel-level differences between $\ddot{t}_0$ and t, it is not difficult for $\ddot{t}_0$ to pass $\Phi_{\text{init}}(t, \ddot{t}_0)$ because the attacking term has been entirely corrupted, e.g., "brown"→"βnoωx", which has most original characters replaced with a random different character.

After having $\ddot{t}_0$, the *Initializer* traces back the perturbation process for recovering t from $\ddot{t}_0$. The goal is to find a decision boundary prompt that is formally defined below.

Definition 9. Decision Boundary Prompt ($\ddot{t}_b$): A decision boundary prompt $\ddot{t}_b$ is an adversarial prompt that successfully passes Φ_{init}, but will fail if any perturbation of $\ddot{t}_b$ is removed.

Identifying $\ddot{t}_b$ is required by the Monte Carlo estimation that we will use next for optimizing the adversarial prompts because Monte Carlo estimation requires the prompts to be positioned at the decision boundary level [Ye *et al.*, 2022a]. Therefore, the *Initializer* should recover some perturbed characters of $\ddot{t}_0$ back to the original characters in order to reduce additional perturbations beyond $\ddot{t}_b$. Unlike the previous research that determines the words for recovery based on their semantic correlation with the original words, we consider the pixel-level differences between the characters due to the lack of character-level semantic information. In particular, the *Initializer* sort all the perturbed characters based on their pixel-level differences with the corresponding original characters in the descending order. Then each original character is replaced back and the resulting adversarial prompt is examined by Φ_{init}. The replacement process iterates until the current adversarial prompt doesn't pass Φ_{init}. Then the last successful adversarial prompt $\ddot{t}_b$ is selected. Through the above recovering process, we maintain the least pixel-level differences between $\ddot{t}_b$ and t but still keep $\ddot{t}_b$ as adversarial.

4.2 The *Estimator* Module

Given a decision boundary prompt $\ddot{t}_b$ from the *Initializer*, we develop the *Estimator* module to identify the optimal adversarial characters for replacements from the character pool by sampling limited characters as observations and estimating the perturbation direction for $\ddot{t}_b$. In particular, for each perturbed character $\ddot{c}$ from $\ddot{t}_b$, we randomly sample K different characters from the character pool as $P_{\ddot{c}} = \{p_1, \ldots, p_K\}$. For each p_k, we replace $\ddot{c}$ with p_k and keep all remaining characters in $\ddot{t}_b$ the same to generate a new adversarial prompt $\ddot{t}_k$. We evaluate $\ddot{t}_k$ by Φ_{init} and generate the corresponding Att^k_{score}. Then we create an examination score list $\mathcal{A}_{\ddot{c}} = \{a_1, \ldots, a_K\}$ where $a_k = \text{Att}^k_{score} - \text{Att}^o_{score}$ and Att^o_{score} is the Att_{score} of t. Therefore, $\ddot{t}_k$ is a successful adversarial prompt if its a_k is positive. Otherwise, $\ddot{t}_k$ is a failure.

Intuitively, the estimated perturbation direction for $\ddot{t}_b$ should be more aligned with the successful adversarial prompts. Figure 1 shows examples of replacing $\ddot{c}$ in $\ddot{t}_b$ (i.e., \U(0175)) with different p_k (e.g., \U(1E85)). The attacking results are in the dashed box of Figure 1 where the solid green arrows denote positive a_k and the red arrow denotes negative a_k. Therefore, the optimal perturbation direction is expected to be closer with more solid green arrows and less with the red arrow. However, the main challenge here is how to quantitatively represent the perturbation direction between $\ddot{c}$ and p_k, given the fact that the characters are non-semantic discrete symbols.

Since character embeddings do not contain explicit semantic meanings, we compute the proxy representation of a character perturbation by comparing the representation of prompt and the attacking term before and after the perturbation.

Definition 10. Proxy Perturbation Representation: Given a prompt t, the attacking target w and a character perturbation $c \rightarrow \ddot{c}$ where $c \in w$, the proxy perturbation representation of $c \rightarrow \ddot{c}$ is defined as $\vec{\delta}_{c|\ddot{c}} = \text{TF}(\ddot{t}) + \text{TF}(\ddot{w}) - \text{TF}(t) - \text{TF}(w)$ where TF is a transformer encoding model (e.g., BERT).

This definition indirectly assigns the character with prompt-level semantics and projects the embedding of different characters into a prompt specific uniform hidden space. For example, if there are two character perturbations $c_1 \rightarrow \ddot{c}_1$ and $c_2 \rightarrow \ddot{c}_2$ that successfully pass Φ_{init} and a failure perturbation $c_3 \rightarrow \ddot{c}_3$, $\vec{\delta}_{c_1|\ddot{c}_1}$ and $\vec{\delta}_{c_2|\ddot{c}_2}$ are more likely to be closer than $\vec{\delta}_{c_3|\ddot{c}_3}$. In this way, the perturbation directions between $\ddot{c}$ and $p_k, k = 1...K$ become comparable, as each green or red arrow in Figure 1 can be represented as $\vec{\delta}_{c|\ddot{c}} - \vec{\delta}_{c|p_k}$.

The *Estimator* module then estimates the perturbation di-

rection for replacing c based on the all p_k from $P_{\ddot{c}}$,

$$\Delta_c = \sum_{k=1}^{K} \text{Norm}(a_k / D(\ddot{c}, p_k)) \times (\vec{\delta}_{c|\ddot{c}} - \vec{\delta}_{c|p_k}) \quad (4)$$

where $D(\ddot{c}, p_k)$ calculates the pixel difference between $\ddot{c}$ and p_k, Norm$(\cdot)$ normalizes the weights of all characters from $P_{\ddot{c}}$.

Then, by comparing $\vec{\delta}_{c|\cdot}$ of each character from the character pool with Δ_c based on cosine similarity, the character with the highest similarity is retrieved to replace the original character c. The replacement process runs iteratively until the current adversarial prompt $\ddot{t}_f$ passes Φ_{init}. After that, we return $\ddot{t}_f$ to the *Initializer* module and obtain a new decision boundary prompt again. Then, the *Estimator* module runs to get another $\ddot{t}_f$. We repeat the *Initializer* and *Estimator* module in the alternative way until the perturbed characters fully converge or the iteration times reach the predefined threshold.

4.3 The *Redistributor* Module

Given the final adversarial prompts $\ddot{t}$ from the *Initializer* and *Estimator* module, we develop a *Redistributor* module to further optimize the prompts, as we observe that some adversarial prompts may not optimally attack the target Text2Image model even if they already passed Φ_{init}. For example, in Figure 3, replacing the "a" with "o" for the attacking term "man" directly removes the human head but also has influence on the "sitting" behavior. In contrast, replacing the "n" with "p" only corrupts the male face by hiding the male features (e.g., Beards) and keeps the remaining visual subjects matched with the prompt. However, both adversarial prompts are considered as successful because the *Estimator* module always pushes an adversarial prompt to approach the decision boundary and Φ_{init} cannot examine the issues reported above. Therefore, the goal of the *Redistributor* module is to calculate the pixel-level differences between $\ddot{t}$ and t, and redistribute the same amount of pixel perturbation to different characters, such that a new adversarial prompt can be generated to achieve a higher Att_{score}.

The *Redistributor* would not make $\ddot{t}$ worse, as an existing successful adversarial prompt $\ddot{t}$ has the Att_{score} that is the lower bound for the equal or even less pixel-level differences as $\ddot{t}$. In other words, a higher Att_{score} can be achieved if the equal or less pixel-level differences are distributed to perturb different characters of the attacking term.

To perform the pixel re-distribution strategy, we calculate the pixel-level differences between $\ddot{t}$ and t as X. Given all available characters from the attacking term (e.g., "man" in Figure 3), we randomly select a character (e.g., "m") and replace the character with another character that shares less than X pixel-level differences. After that, we update the remaining pixel-level differences and select another unperturbed character for replacement until all characters are perturbed or X is not sufficient for a replacement. We then test the new adversarial prompt $\ddot{t}_d$ and obtain the corresponding Att_{score}. If the Att_{score} is higher than the current lower bound Att_{score}, we update Φ_{init} by replacing $\frac{\mathcal{S}[t \setminus w, \mathcal{M}(t)]}{\mathcal{S}[w, \mathcal{M}(t)]}$ with Att_{score} and start over with the *Initializer* module. Otherwise, we drop half number of the new perturbed characters from $\ddot{t}_d$ whose $\vec{\delta}_{c|\cdot}$ share the least similarity with the original perturbed characters from $\ddot{t}$, so as to maximize the possibility of removing non-important perturbed characters and keep the important perturbations. We update X with the returned pixel-level differences and repeat the re-distribution for unperturbed characters until X runs out again. We stop the *Redistributor* if a higher Att_{score} does not exist after a fixed number of re-distributions or the updating iterations reach the pre-defined threshold. We show the overall workflow of CharGrad in the Appendix.

Figure 3: Different attack examples that pass Φ_{init}.

5 Experimental Evaluation

In this section, we conduct extensive experiments on two image captioning datasets to answer the following questions: **Q1)** Can CharGrad achieve a better attacking performance than the state-of-the-art baselines? **Q2)** Is CharGrad more efficient than the state-of-the-art baselines to successfully attack different prompts? **Q3)** How sensitive is the CharGrad with different hyper-parameters? **Q4)** How does each component of CharGrad contribute to its overall performance?

5.1 Dataset and Experiment Setup

Dataset. We use the annotations from two public captioning datasets in our experiments: i) COCO2017 [Lin *et al.*, 2014] is an image object detection dataset that contains captioning annotations for more than 600,000 images, and ii) Flickr30k [Young *et al.*, 2014] is an image captioning dataset that contains 31,000 images with image descriptions. We follow previous research [Ye *et al.*, 2022a; Jin *et al.*, 2020; Maheshwary *et al.*, 2021] by taking 1,000 annotations as prompts from each dataset above to conduct the adversarial prompt attacking. Given a prompt, we tokenize the prompt and randomly select a token as the attacking target. We follow the previous research [Ye *et al.*, 2022a] to generate the Part-of-speech (POS) taggings for all tokens and filter the tokens with specific taggings ("ADJ", "ADV", "VERB", "NOUN") to improve the quality of the selected tokens.

Experiment Setup. In our experiments, we select Stable Diffusion [Rombach *et al.*, 2022] as the target Text2Image model that is widely used by previous research [Ye *et al.*, 2022a; Rando *et al.*, 2022]. We follow the standard parameter settings as shown in *Diffusers* library and remove the safety checker module to achieve the best attacking performance. We set the number of sampled characters from the *Estimator* module as K=5 and the number of times applying the *Redistributor* module as N_{re}=3. We further evaluate the sensitivity

of CharGrad against these two hyper-parameters in Section 5.3 below.

5.2 Baselines

We compare the performance of CharGrad with state-of-the-art prompt or text adversarial attacking models as follows.

HumanTrial [Struppek *et al.*, 2022a]: we simulate the human trials on generating adversarial prompts by randomly replacing the characters from the attacking term with different characters from the character pool.

GreedyPert [Boucher *et al.*, 2022]: we simulate the human imperceptible perturbation approach by starting the perturbation of characters from attacking the term with other characters that share the least pixel-level differences until passing the attack examiner.

LeapAttack [Ye *et al.*, 2022a]: a gradient based text adversarial attacking framework that gradually replaces the words of a given text with their synonyms by estimating the gradient of the perturbation direction for the text based on the disparity of semantic word embeddings.

TextHoaxer [Ye *et al.*, 2022b]: a gradient based text adversarial attacking framework that samples virtual perturbation directions from Gaussian distributions and utilizes the gradients of the directions to retrieve the word candidates with matched word embeddings.

GA-Attack [Maheshwary *et al.*, 2021]: a Genetic Algorithm (GA) based attacking approach that exhaustively queries the candidates words with the target model and generates new candidate set based on the best performed candidates.

ExAttack [Liu *et al.*, 2022]: a text adversarial attacking framework that exchanges the words based on their contributions to a prompt being correctly and mistakenly classified.

For adapting LeapAttack and TextHoaxer from word-level attack to character-level attack, we use CANINE [Clark *et al.*, 2022], a character-level transformer network, to generate character-level embeddings. Then LeapAttack and TextHoaxer can be used for the character-level attacks. We strictly follow the parameters and configurations of all schemes as documented in their papers.

5.3 Evaluation Results

Prompt Adversarial Attacking Performance (Q1)

To answer Q1, we evaluate the generated adversarial prompts by CharGrad and all compared baselines on both datasets. In particular, we average the pixel-level differences (PLD) between all adversarial prompts and the original prompts to evaluate how imperceptible the adversarial prompts are. To make a fair comparison, we follow [Ye *et al.*, 2022a] to adopt perturbation rate (Pert) as another evaluation metric that averages the rate of perturbed characters to all characters within the attacking terms. For the generated images, we calculate the per-pixel attacking score (PPAS) across all images to evaluate the attacking effect of different schemes. The PPAS for each adversarial prompt is calculated based on the corresponding Att_{score} divided by the pixel-level differences. The evaluation results are shown in Table 1. We observe that CharGrad significantly outperforms all compared baselines in terms of Pert and PPAS on both datasets. The results demonstrate the effectiveness of CharGrad on attacking different prompts in the character level by perturbing limited number of characters, which is important for approaching the human imperceptible perturbations. While the PLD of GreedyPert is slightly lower than CharGrad because GreedyPert focuses on human imperceptible perturbations. However, the PPAS and Pert of GreedyPert are much worse than those of CharGrad, which demonstrates the necessity of learning the optimal adversarial characters from the character pool rather than only relying on the human imperceptible characters.

Dataset	COCO2017			Flickr30k		
Metric	PLD	Pert	PPAS (1e-3)	PLD	Pert	PPAS (1e-3)
HumanTrial	523.35	0.788	0.182	455.41	0.824	0.239
LeapAttack	104.99	0.247	0.879	98.92	0.232	1.148
TextHoaxer	107.32	0.339	0.615	92.13	0.218	0.890
GreedyPert	**53.29**	0.617	0.372	**44.78**	0.552	0.399
GA-Attack	148.84	0.249	0.491	153.38	0.290	0.582
ExAttack	202.34	0.498	0.711	237.40	0.535	0.798
CharGrad	71.22	**0.171**	**4.049**	79.24	**0.115**	**5.971**

Table 1: Prompt Adversarial Attacking Performance

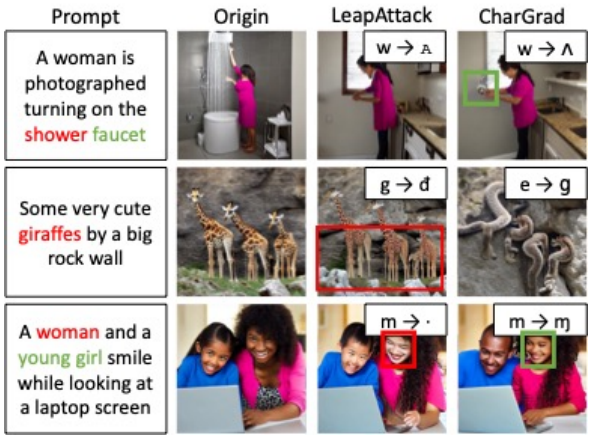

Figure 4: Visualization of Generated Images.

We further demonstrate the attacking performance of CharGrad by visualizing a set of generated images by CharGrad and LeapAttack (the best performed baseline in Table 1). We show the visualization results in Figure 4. In particular, the red words in the prompt column represent the attacking terms. The green words represent one of the remaining subjects that should be kept the same. Similarly, the red bounding boxes in the LeapAttack column represent the unsuccessful attacking on the target subjects, while the green bounding boxes in the CharGrad column denote the successful preservation of non-target visual subjects. We observe that CharGrad can consistently generate adversarial prompts that maximize the attacking capability on the target visual subjects while minimizing the negative impact on non-target visual subjects. We show more visualization examples in the Appendix.

Prompt Adversarial Attacking Efficiency (Q2)

To answer Q2, we evaluate the efficiency of CharGrad by investigating the query number of each scheme to the tar-

Dataset	COCO2017				Flickr30k			
N_{re}	0	1	2	3	0	1	2	3
LeapAttack	112	202	355	579	90	148	392	554
TextHoaxer	139	257	410	614	101	217	401	693
GA-Attack	640	702	759	905	589	742	870	993
CharGrad	**92**	**114**	**157**	**220**	**84**	**121**	**185**	**243**

Table 2: The number of queries to Text2Image when completing attacks by applying the Redistributor N_{re} times

get Text2Image model. In particular, we select LeapAttack, TextHoaxer and GA-Attack for comparison as they require dynamical queries to the model to improve the adversarial prompts. To make fair comparisons, we separately average the query numbers based on the times of applying the *Redistributor* module (N_{re}) and show the results in Table 2. We observe that the query number of CharGrad is consistently lower than other compared schemes on both datasets. We also observe that the gap of the query number between CharGrad and other schemes becomes larger as N_{re} increases. This is because the attack examiner with a higher Att_{score} makes the character perturbation more challenging. The baselines thus require more trials as their perturbation direction is sub-optimal due to the only usage of the pretrained character embeddings. Therefore, the performance gain of CharGrad can be attributed to the proxy representation of perturbation $\vec{\delta}$ from the *Estimator* module that accurately estimates the perturbation directions for the prompts.

Hyper-parameter Sensitivity Study (Q3)

We study the hyper-parameter sensitivity of CharGrad by tuning 1) the number of sampled characters (K) in $P_{\vec{c}}$ from the *Estimator* module, and 2) the number of times for applying the *Redistributor* module (N_{re}). In particular, we vary K from 2 to 6 and N_{re} from 0 to 3. We report the PPAS in Figure 5. We firstly observe that CharGrad consistently outperforms LeapAttack with all ranges of K and N_{re}, which further demonstrate the effectiveness of CharGrad on generating high-quality adversarial prompts. We then observe that the increase of K can improve the PPAS for both schemes and the improvement on CharGrad is more significant. We attribute it to our proxy perturbation representation $\vec{\delta}$ that accurately estimates the gradients of perturbation directions for different prompts. Moreover, we observe that the increase of N_{re} can boost the attacking performance of CharGrad while having little effect on LeapAttack. This is because reaching a high Att_{score} requires the scheme to identify the optimal perturbation direction for prompts with a faster gradient, which is not achievable based on the character-level embeddings of LeapAttack. Therefore, the generated prompts by LeapAttack are positioned close to the decision boundary to satisfy the examination and thus fall into a lower PPAS.

Ablation Study (Q4)

Finally, we perform a comprehensive ablation study to evaluate the contributions of important components in CharGrad. We create variants of CharGrad by changing its key components: 1) *CharGrad-NoOrder*: we remove the perturbed character reordering process from the *Initializer* module and randomly push the original character back; 2) *CharGrad-Binary*: we replace the weighted perturbation direction from the *Estimator* module with the binary mean-pooling [Ye *et al.*, 2022a]; and 3) *CharGrad-Random*: we replace the pixel re-distribution strategy from the *Redistributor* module with a random re-distribution process regardless of their $\vec{\delta}$ with the original perturbed characters. The results are shown in Table 3. We observe CharGrad outperforms other variants in terms of all evaluation metrics. The results demonstrate the importance and necessity of key components of CharGrad.

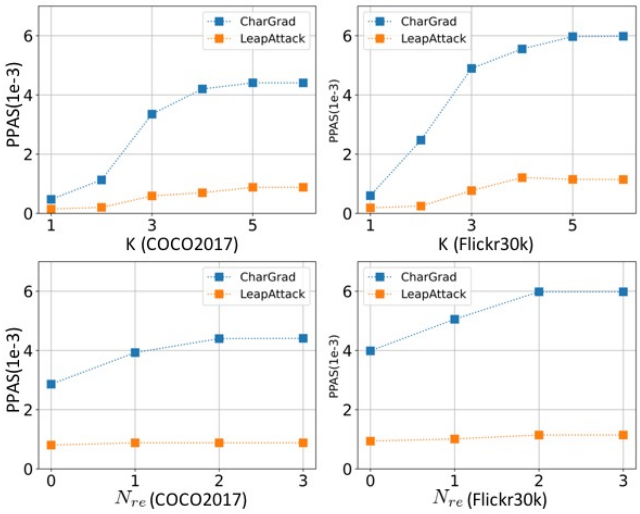

Figure 5: Hyper-parameter Sensitivity Study of CharGrad

Dataset	COCO2017			Flickr30k		
Metric	PLD	Pert	PPAS(1e-3)	PLD	Pert	PPAS(1e-3)
CharGrad-NoOrder	122.58	0.298	2.905	109.23	0.271	4.062
CharGrad-Binary	75.40	0.188	3.769	81.24	0.133	5.290
CharGrad-Random	89.04	0.207	3.485	98.23	0.207	5.144
CharGrad	**71.22**	**0.171**	**4.049**	**79.24**	**0.115**	**5.971**

Table 3: Ablation Study

6 Conclusion

This paper presents a novel CharGrad framework to address the controllable prompt adversarial attacking problem for Text2Image models. CharGrad designs a perturbation representation metric to quantitatively estimate the perturbation directions between Unicode characters and an iterative attack examiner updating strategy to generate adaptive attack examinations for different prompts. Evaluation results on two captioning datasets demonstrate more accurate and efficient adversarial attacking performance, and lower pixel-level perturbations for attacking various prompts than state-of-the-arts.

Acknowledgements

This research is partially supported by the National Science Foundation (NSF) under Grant No. 2202693.

References

[Boucher *et al.*, 2022] Nicholas Boucher, Ilia Shumailov, Ross Anderson, and Nicolas Papernot. Bad characters: Imperceptible nlp attacks. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1987–2004. IEEE, 2022.

[Clark *et al.*, 2022] Jonathan H Clark, Dan Garrette, Iulia Turc, and John Wieting. Canine: Pre-training an efficient tokenization-free encoder for language representation. *Transactions of the Association for Computational Linguistics*, 10:73–91, 2022.

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

[Ho *et al.*, 2020] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[Jin *et al.*, 2020] Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8018–8025, 2020.

[Lin *et al.*, 2014] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2014. cite arxiv:1405.0312.

[Liu *et al.*, 2022] Huijun Liu, Jie Yu, Jun Ma, Shasha Li, Bin Ji, Zibo Yi, Miaomiao Li, Long Peng, and Xiaodong Liu. Textual adversarial attacks by exchanging text-self words. *International Journal of Intelligent Systems*, 2022.

[Maheshwary *et al.*, 2021] Rishabh Maheshwary, Saket Maheshwary, and Vikram Pudi. Generating natural language attacks in a hard label black box setting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13525–13533, 2021.

[Radford *et al.*, 2021] 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.

[Ramesh *et al.*, 2022] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[Rando *et al.*, 2022] Javier Rando, Daniel Paleka, David Lindner, Lennard Heim, and Florian Tramèr. Red-teaming the stable diffusion safety filter. *arXiv preprint arXiv:2210.04610*, 2022.

[Ren *et al.*, 2019] Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 1085–1097, 2019.

[Rombach *et al.*, 2022] 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.

[Struppek *et al.*, 2022a] Lukas Struppek, Dominik Hintersdorf, and Kristian Kersting. The biased artist: Exploiting cultural biases via homoglyphs in text-guided image generation models. *arXiv preprint arXiv:2209.08891*, 2022.

[Struppek *et al.*, 2022b] Lukas Struppek, Dominik Hintersdorf, and Kristian Kersting. Rickrolling the artist: Injecting invisible backdoors into text-guided image generation models. *arXiv preprint arXiv:2211.02408*, 2022.

[Tian *et al.*, 2023] Yijun Tian, Kaiwen Dong, Chunhui Zhang, Chuxu Zhang, and Nitesh V Chawla. Heterogeneous graph masked autoencoders. In *AAAI*, 2023.

[Ye *et al.*, 2022a] Muchao Ye, Jinghui Chen, Chenglin Miao, Ting Wang, and Fenglong Ma. Leapattack: Hard-label adversarial attack on text via gradient-based optimization. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2307–2315, 2022.

[Ye *et al.*, 2022b] Muchao Ye, Chenglin Miao, Ting Wang, and Fenglong Ma. Texthoaxer: Budgeted hard-label adversarial attacks on text. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3877–3884, 2022.

[Young *et al.*, 2014] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.