CLIPAG: Towards Generator-Free Text-to-Image Generation Supplementary Material

Roy Ganz
Department of ECE
Technion, Haifa, Israel
ganz@campus.technion.ac.il

Michael Elad
Department of Computer Science
Technion, Haifa, Israel
elad@cs.technion.ac.il

1. Implementation Details

1.1. Training

We implement our code based on the publicly-available CLIP repository of OpenCLIP [6][1]. In all the experiments considered in the paper, we use a pretrained ViT-B/32 architecture, initialized using OpenCLIP's weights[2]. In addition, we consider a threat model of L_2 with a maximum perturbation of 1.5 and 5 PGD steps by extending the standard implementation to the multimodal case. We set the learning rate and weight decay to $2e-5$ and $1e-4$, respectively, and perform 10 gradient accumulation steps to enable larger batch sizes, resulting in an effective batch size of $40,960$. As for the training data, we concatenate SBU, CC-3M, and CC-12M without resampling and a downsampled version of LAION 400M (by $\times 0.04$). We freeze the textual encoder and finetune the vision encoder (88M parameters) on eight A40 GPUs. We study the effects of different design choices in Sec. 3. We will make our code and pretrained model publicly available.

1.2. Text-to-Image Generative Frameworks

In the experiments presented in the main paper, we seamlessly replace the existing CLIP ViT-B/32 in such frameworks with CLIPAG, using the same architecture and hyperparameters as in the baseline. In this way, we ensure a fair comparison that enables us to study the benefits of CLIPAG. We explore our approach in three main frameworks and describe implementation details and relevant information below.

CLIPDraw We utilize the official implementation of CLIPDraw[3] and replace the used "vanilla" CLIP ViT-B/32 by CLIPAG with the same architecture. As stated in the main paper, we experiment with two settings – with and without augmentations. Besides the qualitative demonstrations, we propose a quantitative evaluation procedure to evaluate the performance. To this end, we utilize some prompts suggested in CLIPDraw's paper and request from ChatGPT [3] to provide us additional 100 similar prompts. We synthesize the generated prompts using CLIPDraw with CLIP and CLIPAG, with and without augmentations. Next, we calculate the aesthetic score using a CLIP trained on human aesthetic predictions using a publicly available code[4]. Given an image, such a model outputs a continuous value describing the aesthetic score (higher is better). Using this model, we calculate an aesthetic score for every generated drawing and report two metrics – Average aesthetics score and a pairwise aesthetic preference. Since the proposed aesthetic metrics do not depend on the caption, we also measure caption similarity using CLIP similarity with two CLIP models to validate the results better. Besides CLIP similarity, we utilize the R-Prec metric [7], focusing on image-based text retrieval. Specifically, given a generated image, we use CLIP to pick the most probable prompt across all the 100 textual descriptions. We average the accuracy of such a task for the generated drawings, resulting in the R-Prec metric, similar to [7]. The combination of these metrics captures both the generated drawings' quality and consistency with the caption, enabling a proper evaluation of the generated drawings.

VQGAN+CLIP Similar to CLIPDraw, we use the official code repository [5] and replace CLIP with CLIPAG. We randomly sample 100 captions from the validation set of the MS-COCO dataset and generate two sets of 100 images. Next, we calculate the CLIP similarity to measure the alignment of the generated images with the desired prompts. To better demonstrate the effects of replacing CLIP with CLI-

[1] https://github.com/mlfoundations/open_clip

[2] https://huggingface.co/laion/CLIP-ViT-B-32-laion2B-s34B-b79K

[3] https://colab.research.google.com/github/kvfrans/clipdraw/blob/main/clipdraw.ipynb

[4] https://github.com/christophschuhmann/improved-aesthetic-predictor

[5] https://github.com/nerdyrodent/VQGAN-CLIP

PAG, we provide additional results in Fig. 1.

CLIPStyler We experiment with CLIPStyler official code[6] and replace CLIP with CLIPAG, with and without the style network. To better demonstrate the effectiveness of CLIPAG in the text-guided style transfer context, we provide additional results in Fig. 2.

1.3. Generator-Free Text-to-Image Generation

Initialization mechanism In practice, we propose the following simple-yet-effective initialization process – we randomly sample image candidates from a simple distribution and pick the one that best matches the target text using CLIP cosine similarity. Specifically, we leverage a downsampled version (16×16) of the Tiny-ImageNet dataset [12] and train a Gaussian Mixture Model where each Gaussian represents a class. Next, we sample M candidates from each of the 200 classes, resulting in $M \times 200$ images. Such images are unrealistic and mainly contain colorful blobs (a visualization of the chosen initial images is shown in Figure 3). Next, we upsample these images to 224×224 and pick the one with the highest alignment with the target text as the input to our process, resulting in a generated image. To study the effect of the initialization mechanism, we generate several prompts using the above-described initialization, compared to random Gaussian noise one in Figure 4. As can be seen, CLIPAG is capable of producing meaningful results with both initializations, attesting to its guiding capabilities.

The generation process After the initialization step, we perform an iterative process of K steps (set empirically to 1000) in which we modify the input image to better align with the given textual description. Specifically, we duplicate the image and augment every instance using random augmentations, leading to a batch of different image views. Next, we input the batch to the image encoder to obtain feature representations. Finally, we calculate the cosine similarity loss, calculate the input gradients and use them to update the image. Unlike other works that harness CLIP [2, 5, 8, 10], we do not use additional losses such as direction-loss and total variation regularization to guide the process but rather focus solely on the basic CLIP-loss. Repeating these steps K times results in pleasing generated images corresponding to the target captions.

2. Explainability

With the introduction of learning-based machines into "real-world" applications, the interest in interpreting the decisions of such models has become a central concern. Thus, the explainability of deep learning-based models is a crucial objective for improving the trust and transparency of such models. Moreover, it enables users to understand model predictions better and detect shortcuts and biases. We hypothesize that due to its more aligned gradients, CLIPAG possesses improved explainability capabilities than the regular CLIP model. To verify if this is indeed the case, we utilize The GradCAM [11] (Gradient-weighted Class Activation Mapping) algorithm, which utilizes the model's gradients to generate visual heatmaps, highlighting the important regions in an input image for a given target. We follow the implementation of [4][7]. Specifically, GradCAM combines the features and the gradients of a network's layer by multiplying it. As this method relies on the gradients of the deepest layers, upsampling its results to the input resolution often leads to coarse results. In addition, GradCAM is designed for convolutional neural networks and is significantly less effective in vision transformers. In Figure 5, we present the results of applying GradCAM on CLIP (using ViT-B/32) with both the original and CLIPAG, using ImageNet images in a zero-shot setting. As can be seen, while GradCAM performs unsatisfactorily on the regular CLIP, applying it on CLIPAG leads to more aligned heatmaps with the target objects. We hypothesize that this improvement stems from the Perceptually Aligned Gradients property of CLIPAG, leading to an improved explainability with GradCAM.

Furthermore, we study the interpretability of CLIPAG under adversarial attacks. To this end, given an input image $\mathbf{x}$ and a textual description of an object t (*e.g.*, "`a cat`"), we perform adversarial attacks to minimize the cosine similarity between $\mathbf{x}$ and y in the feature space, and maximize the one between the image and the negation of the textual description t, denoted as $\tilde{t}$ (*e.g.*, "`not a cat`"). Formally, we solve the following optimization problem:

$$\max_{\delta\in\Delta} \mathcal{L}_{SIM}(f^I_{\theta_I}(\mathbf{x}+\delta), f^T_{\theta_T}(t)) - \mathcal{L}_{SIM}(f^I_{\theta_I}(\mathbf{x}+\delta), f^T_{\theta_T}(\tilde{t})) \tag{1}$$

where $\mathcal{L}_{SIM}$ is the cosine similarity loss, *i.e.*, maximizing it minimizes the cosine similarity. In particular, we perform a Projected Gradient Descent (PGD) attack where $\Delta = \{\ \delta : \|\delta\|_\infty \leq \frac{8}{255}\}$ using 20 steps and a step size of $\frac{1}{255}$. We provide visualizations of the outputs of GradCAM using both CLIPAG and the "vanilla" CLIP on adversarial attacks in Figure 5. As can be seen, the adversarial attacks change the outputs of the baseline significantly; however, CLIPG's outputs are much more robust.

3. Ablation Study

In this section, we explore the effects of different design choices of Vision-Language adversarial finetuning CLIP on its Perceptually Aligned Gradients. To this end, we conduct

[6] `https://github.com/cyclomon/CLIPstyler`

[7] `https://github.com/anguyen8/gScoreCAM`

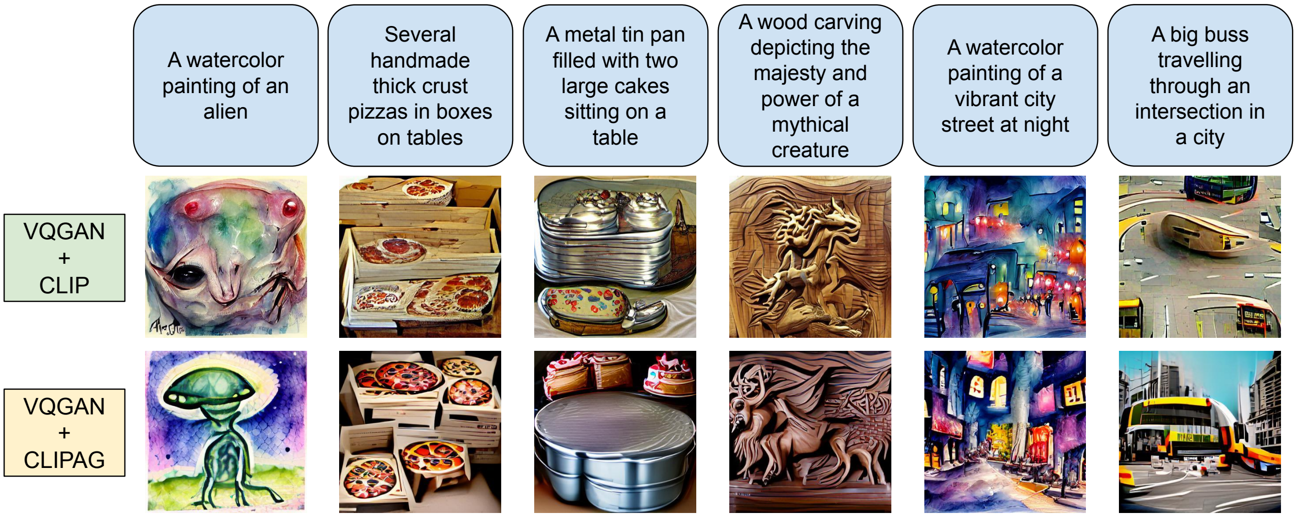

A watercolor painting of an alien
Several handmade thick crust pizzas in boxes on tables
A metal tin pan filled with two large cakes sitting on a table
A wood carving depicting the majesty and power of a mythical creature
A watercolor painting of a vibrant city street at night
A big buss travelling through an intersection in a city
VQGAN + CLIP
VQGAN + CLIPAG

Figure 1. **VQGAN+CLIP additional results.**

a relatively short training and study the effect of different architectures (ViT-B/32, ViT-B/16, and ConvNext). Moreover, we also compare an L_∞ ($\epsilon = \frac{2}{255}$) to an L_2-based one ($\epsilon = 1.5$). To assess the impact of such design choices, we use them for generator-free text-to-image generation using our proposed framework, described in **??**, and visualize the results in Fig. 6. As can be seen, all the different design choices lead to satisfactory outputs, attesting to their PAG. However, there are some differences:

- **Different ViTs** – We consider both ViT-B/16 and ViT-B/16, which differ in the patch size, using L_∞-based threat model. The ViT-B/16, which utilizes a smaller path size, leads to some visual artifacts. We hypothesize that maximizing the consistency with the caption with a small patch architecture leads to fine-grained modifications that result in undesired artifacts.

- **CNN vs. ViT** – While both are trained on the same threat model, the ConvNext guides the generation process towards images with significantly more saturated colors than the ViTs.

- **Threat model** – We compare the L_2, $\epsilon = 1.5$ to L_∞, $\epsilon = \frac{2}{255}$ using ViT-B/32. Interestingly, despite these threat models being substantially different, they both lead to generated images with similar characteristics. Nevertheless, we find the results of the L_2 case more pleasing.

Thus, we mainly focus on the ViT-B/32 architecture and the L_2 threat model.

4. Generator-Free Text-To-Image Analysis

In this section, we provide additional information and study different aspects of our text-to-image generation results in the generator-free setting, presented in Figure **??**. According to our procedure, we perform an iterative process of updating K steps (empirically set to 1000). During such updates, the generated image is modified to be more aligned with the textual description, according to CLIPAG. To better study the effect of the iterative process, we depict samples in three timesteps in Figure 3– (i) the initialization image, (ii) an intermediate image, and (iii) the final result. In particular, `Initialization` depicts the starting point of our iterative process, *i.e.*, a sample from our GMM that best aligns with the given text. As can be seen in the Figure, the starting point is not a real image, as modeling images via GMMs is limited due to their high dimensionality. Interestingly, CLIPAG is capable of transforming such inputs into perceptually meaningful content that corresponds with the text. One can see that in the `intermediate` point, the resulting images contain most of the high-level features. The process from the intermediate point towards the `output` adds mainly low-level details and refines the visual content.

To better understand the generative capabilities and explore different trends in the synthesis process, we provide additional qualitative results in Figure 7. As can be seen, in the top two rows, the outputs of our proposed algorithm are relatively natural and realistic. However, CLIPAG often prefers “cartoonish” outputs over realistic ones, as can be seen in the third row. We hypothesize that this might be affected by certain words in the target text prompt that guides the model towards such outputs (*e.g.,* “`magical`”). We suspect such a tendency leads to lower FID scores when measured w.r.t. natural images dataset, such as MS-COCO.

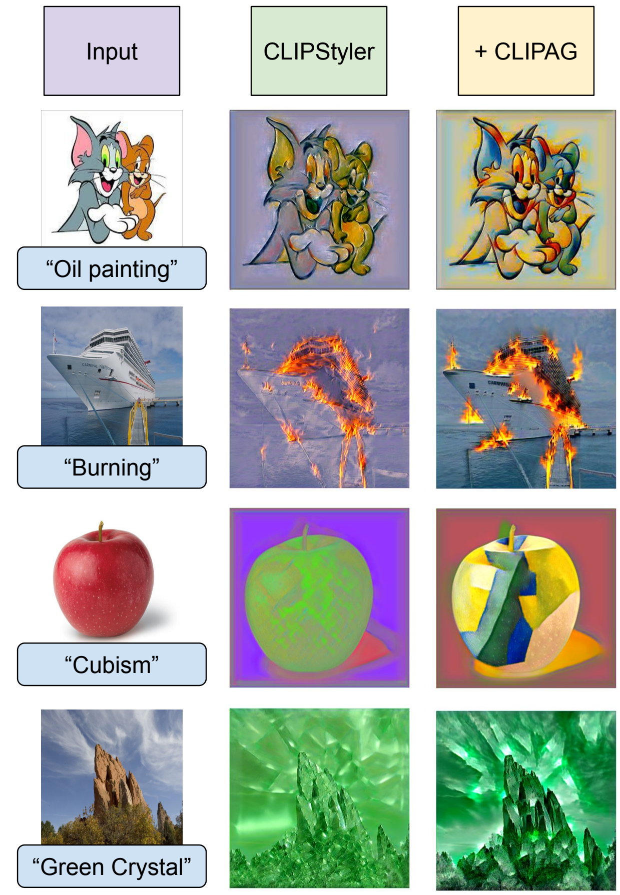


Figure 2. **CLIPStyler additional results.** Style transfer results of multiple images using different textual prompts with CLIP and CLIPAG while using the style network.

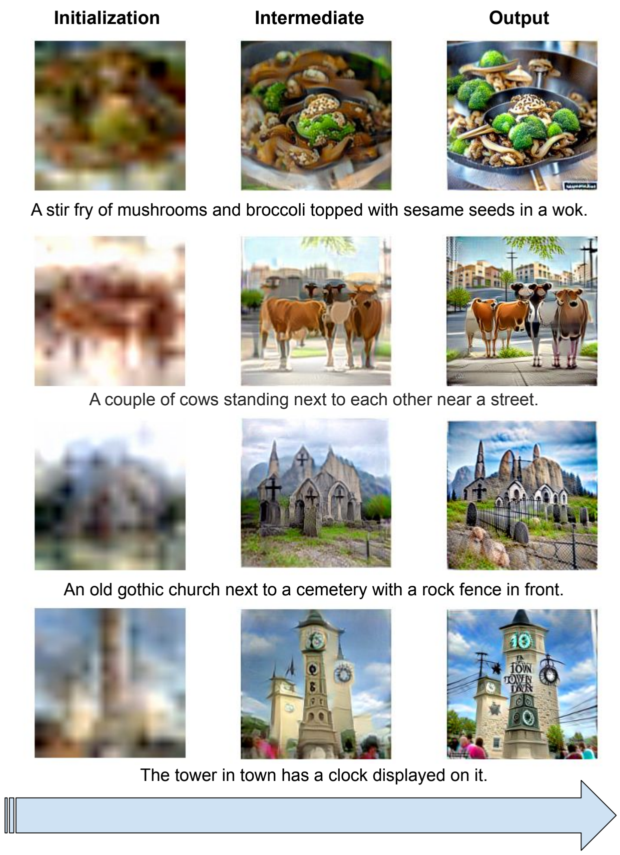

Initialization
Intermediate
Output
A stir fry of mushrooms and broccoli topped with sesame seeds in a wok.
A couple of cows standing next to each other near a street.
An old gothic church next to a cemetery with a rock fence in front.
The tower in town has a clock displayed on it.

Figure 3. **Generation Trajectory**. Visualization of three different time steps in the generation process.

In addition, our model sometimes maximizes the alignment with the given text by producing OCR [1, 9] information that corresponds with the caption. For example, in the third row in Figure 7, the model attempts to spell the word “`fiesta`”, which appears in the caption. Similarly, in the bottom row of Figure 3, the model spells “`town`” and “`tower`” that are included in the caption.

Moreover, we aim to explore the level of stochasticity of our framework. Our scheme includes two random steps that introduce randomness to the synthesis process – the initialization and the random augmentations. To better understand the variability that these mechanisms introduce, we generate the same caption several times and visualize such results in Figure 9.

Lastly, we investigate the effect of the chosen prompt on the generated image. Until now, we do not prepend to the target caption any guiding prefix. Now, we study the impact of adding such prefixes that describe the style of the desired image. In particular, we consider the following prompts – “`oil painting of`”, “`a pencil drawing of`”, “`a graffiti of`”, and “`a childish cartoon of`”. We depict the results in Figure 8. As can be seen, the prefix strongly determines the style of the generated images, strongly attesting to CLIPAG’s capability in guiding towards different stlyes, although mainly trained on natural images.

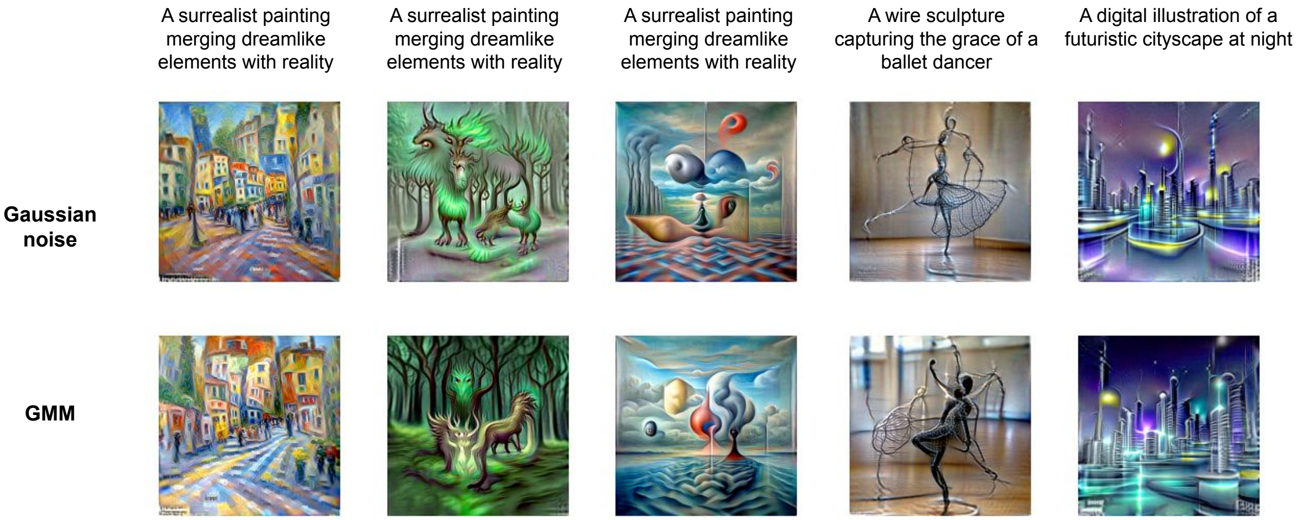

A surrealist painting merging dreamlike elements with reality
A surrealist painting merging dreamlike elements with reality
A surrealist painting merging dreamlike elements with reality
A wire sculpture capturing the grace of a ballet dancer
A digital illustration of a futuristic cityscape at night
Gaussian noise
GMM

Figure 4. **Initialization ablation.** Comparison of our GMM initialization with random Gaussian noise in text-to-image generation.

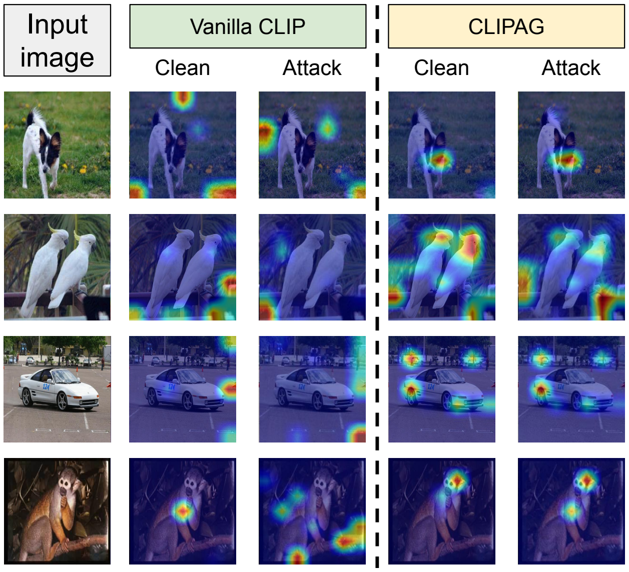


Figure 5. **Explainability visualizations.** GradCAM [11] heatmaps for both the baseline CLIP ViT-B-32 and CLIPAG on ImageNet images. The targets for the GradCAM are `dog`, `parrot`, `cars` and `monkey`, respectively. As can be seen, both in the clean and adversarial cases, CLIPAG heatmaps are more aligned with the objects, providing better interpretability.

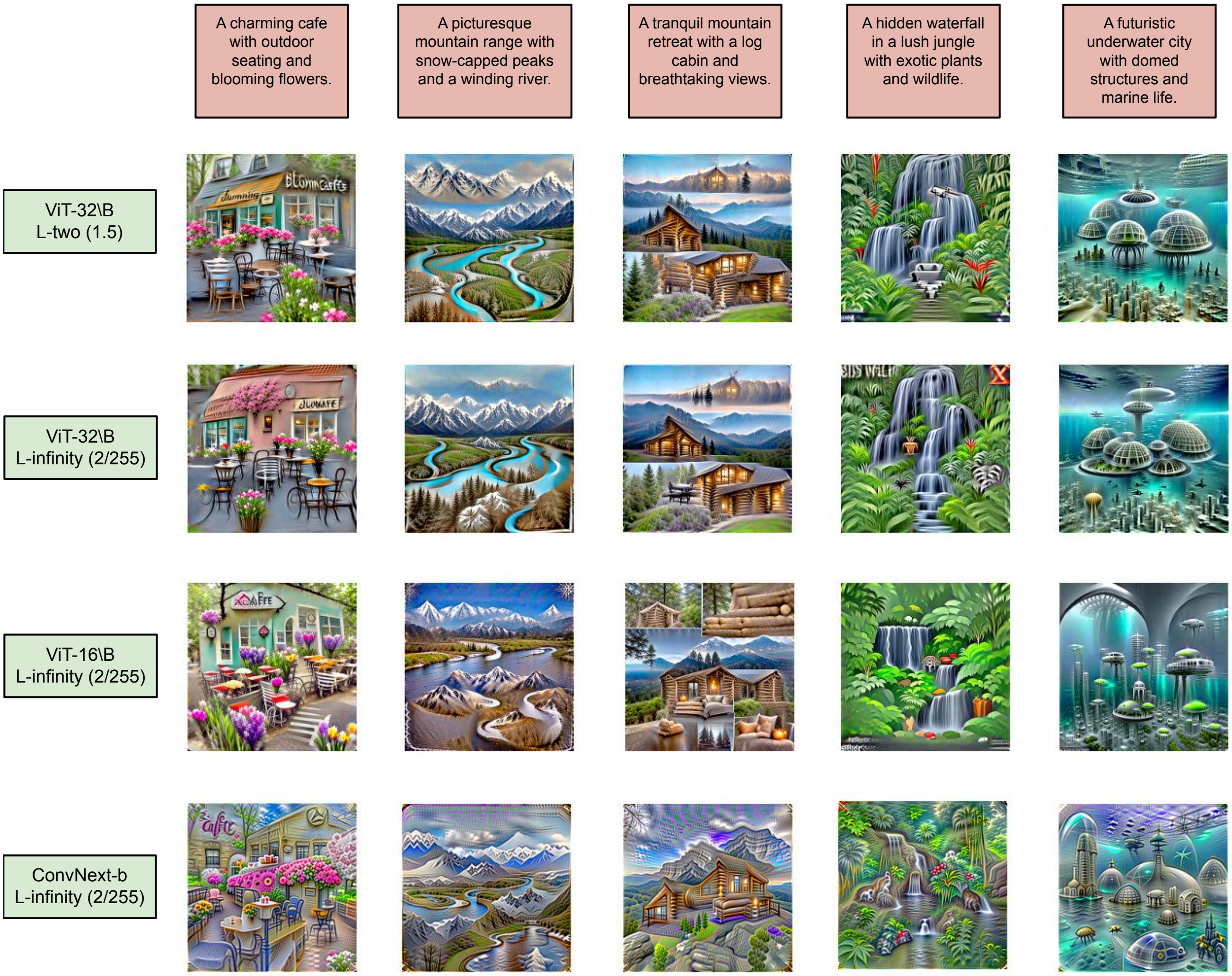
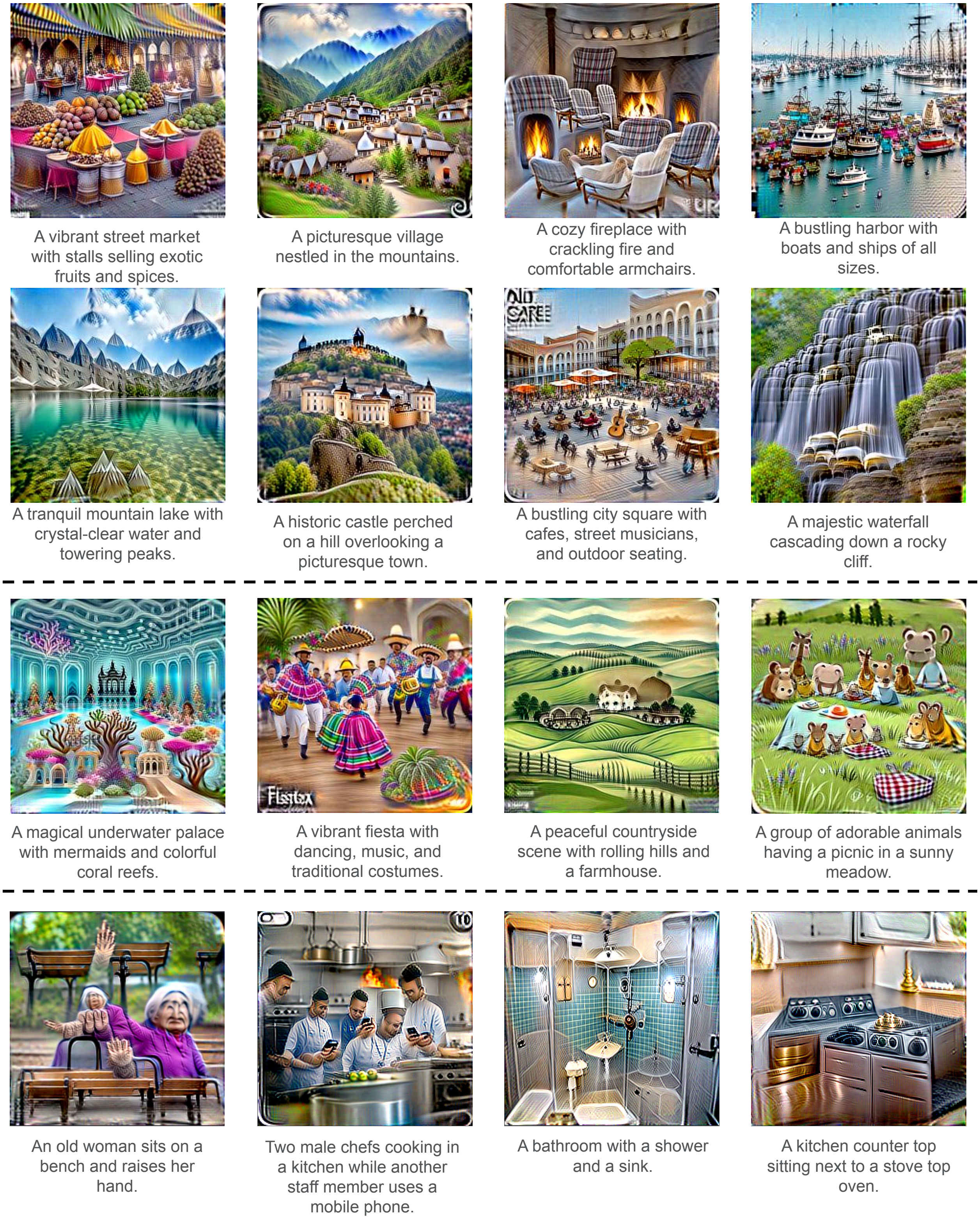

A charming cafe with outdoor seating and blooming flowers.
A picturesque mountain range with snow-capped peaks and a winding river.
A tranquil mountain retreat with a log cabin and breathtaking views.
A hidden waterfall in a lush jungle with exotic plants and wildlife.
A futuristic underwater city with domed structures and marine life.
ViT-32\B L-two (1.5)
ViT-32\B L-infinity (2/255)
ViT-16\B L-infinity (2/255)
ConvNext-b L-infinity (2/255)

Figure 6. **CLIPAG generator-free text-to-image ablation study**. The effect of different design choices in the context of generator-free image synthesis, considering different architectures and threat models.



A vibrant street market with stalls selling exotic fruits and spices.

A picturesque village nestled in the mountains.

A cozy fireplace with crackling fire and comfortable armchairs.

A bustling harbor with boats and ships of all sizes.

A tranquil mountain lake with crystal-clear water and towering peaks.

A historic castle perched on a hill overlooking a picturesque town.

A bustling city square with cafes, street musicians, and outdoor seating.

A majestic waterfall cascading down a rocky cliff.

A magical underwater palace with mermaids and colorful coral reefs.

A vibrant fiesta with dancing, music, and traditional costumes.

A peaceful countryside scene with rolling hills and a farmhouse.

A group of adorable animals having a picnic in a sunny meadow.

An old woman sits on a bench and raises her hand.

Two male chefs cooking in a kitchen while another staff member uses a mobile phone.

A bathroom with a shower and a sink.

A kitchen counter top sitting next to a stove top oven.

Figure 7. **CLIPAG generator-free text-to-image additional results**. The top two rows present additional generator-free synthesis results using CLIPAG. The third row demonstrates a phenomenon in which CLIPAG often opts for cartoonish and artistic content rather than a realistic one. In the last row, we depict some fail cases in which the resulting images are inconsistent.

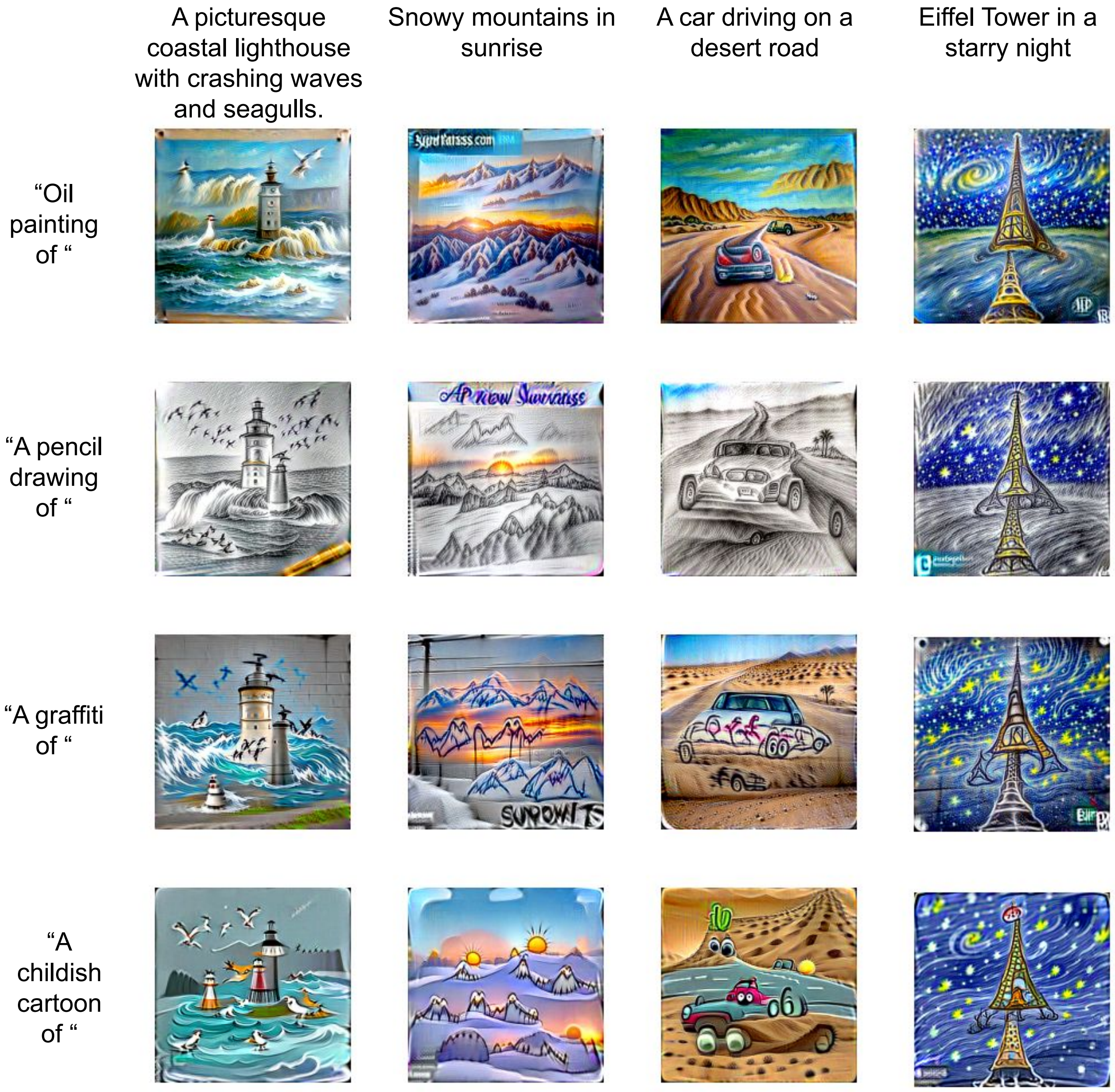
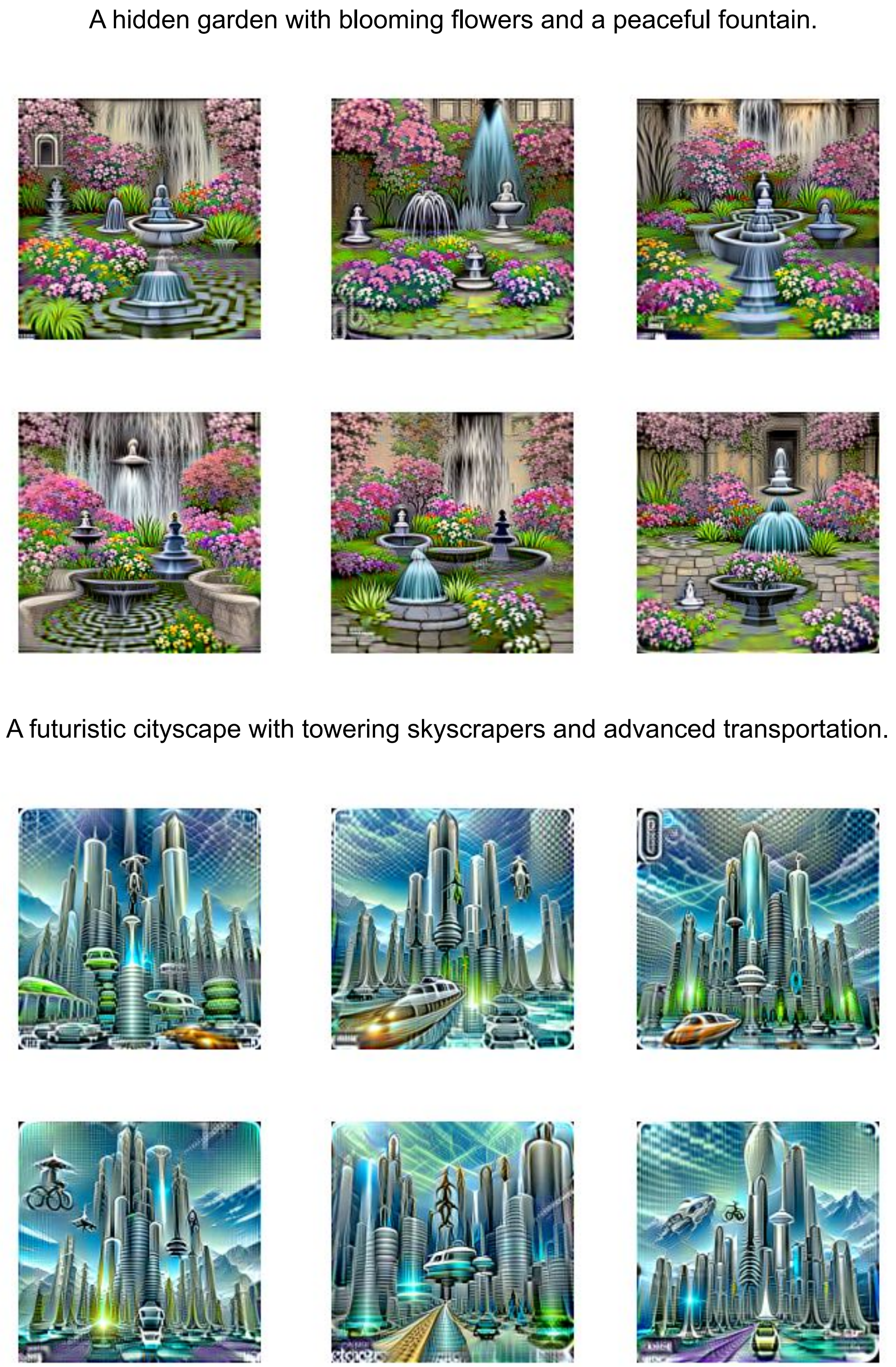

A picturesque coastal lighthouse with crashing waves and seagulls.
Snowy mountains in sunrise
A car driving on a desert road
Eiffel Tower in a starry night
“Oil painting of “
“A pencil drawing of “
“A graffiti of “
“A childish cartoon of “

Figure 8. **CLIPAG generator-free text-to-image prefix effect**.



A hidden garden with blooming flowers and a peaceful fountain.

A futuristic cityscape with towering skyscrapers and advanced transportation.

Figure 9. **CLIPAG generator-free text-to-image stochasticity**. We generate each textual description six times, to demonstrate the stochasticity introduces by our framework.

References

[1] Aviad Aberdam, Roy Ganz, Shai Mazor, and Ron Litman. Multimodal semi-supervised learning for text recognition. *CoRR*, abs/2205.03873, 2022. 4

[2] Omer Bar-Tal, Dolev Ofri-Amar, Rafail Fridman, Yoni Kasten, and Tali Dekel. Text2live: Text-driven layered image and video editing. In Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XV*, volume 13675 of *Lecture Notes in Computer Science*, pages 707–723. Springer, 2022. 2

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are unsupervised multitask learners. *arXiv preprint arXiv:2005.14165*, 2020. 1

[4] Peijie Chen, Qi Li, Saad Biaz, Trung Bui, and Anh Nguyen. gscorecam: What objects is clip looking at? In *Proceedings of the Asian Conference on Computer Vision*, pages 1959–1975, 2022. 2

[5] Rinon Gal, Or Patashnik, Haggai Maron, Amit H. Bermano, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. *ACM Trans. Graph.*, 41(4):141:1–141:13, 2022. 2

[6] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, July 2021. If you use this software, please cite it as below. 1

[7] Ajay Jain, Amber Xie, and Pieter Abbeel. Vectorfusion: Text-to-svg by abstracting pixel-based diffusion models, 2022. 1

[8] Gihyun Kwon and Jong Chul Ye. Clipstyler: Image style transfer with a single text condition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18062–18071, June 2022. 2

[9] Ron Litman, Oron Anschel, Shahar Tsiper, Roee Litman, Shai Mazor, and R. Manmatha. Scatter: Selective context attentional scene text recognizer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 4

[10] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 2065–2074. IEEE, 2021. 2

[11] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 618–626. IEEE Computer Society, 2017. 2, 5

[12] Jiayu Wu, Qixiang Zhang, and Guoxi Xu. Tiny imagenet challenge. *Technical report*, 2017. 2