TokenCompose: Text-to-Image Diffusion with Token-level Supervision (01/13/2025)

Louis Martinez
Telecom Paris
louis.martinez@telecom-paris.fr

Abstract

This report first proposes an in-depth analysis of the method and results presented in the article. Hence, we reproduce the experiments of the paper, including running the method to generate our own checkpoints, evaluating the latter, as well as the checkpoints provided by the authors on the MultiGen benchmark. Then, provided the properties on the cross-attention maps claimed in the original article and visualized in this report, we evaluate the capabilities of our models on local image editing.

1. Introduction

The article presents two main contributions. On the one hand, it describes a fine-tuning procedure, TokenCompose [9], which enforces token-level supervision during the denoising process. In particular, for multicategory composition, it ensures that all prompted categories are actually generated in the output image. The merthod and the On the other hand, the article introduces MultiGen, a benchmark designed to quantify model performance on multi-category composition tasks.

1.1. TokenCompose

TokenCompose aims to overcome two issues of image generation with Latent Diffusion Models (LDM) [7]:

- Most LDMs are trained on real images or on synthetic images of plausible scenes. For instance, StableDiffusion 1.4 (SD1.4), the architecture on which the authors carried out most of their experiments, is fine-tuned on laion-aesthetics v2 5+ [8]. This dataset is a mixture of real and synthetic images generated with GLIDE [6] and former versions of SD. Then, the main pitfall stated in the article is that SD1.4 is not able to accurately generate scenes combining categories which are not meant to be together in a natural scene.
- Although LDMs generally demonstrate good grounding capabilities between a given prompt and the generated instances, they struggle when dealing with multicategory instance composition, more than three in general.

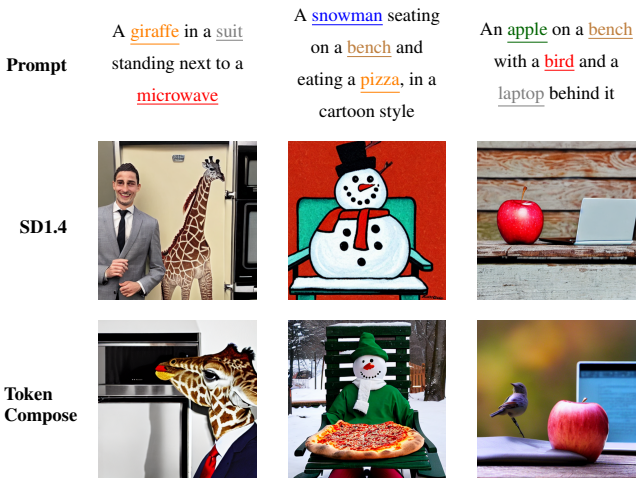

Figure 1. **Images generated with StableDiffusion 1.4 and TokenCompose.** In the column SD1.4 struggles generating a giraffe in a suit and ends up generating the most plausible scene provided the input prompt. The middle and right columns highlight the fact that SD1.4 tends to generate no more than two of the requested categories. While the pizza is missing in the middle column, there is no bird in the right column.

To overcome those aforementioned issues, the article introduces two additional loss terms. These terms explicitly improve grounding between a prompt and the output image by constraining the activations of some cross-attention maps within the denoising U-Net.

1.1.1. Token Loss

The first loss term, denoted $\mathcal{L}_{\text{token}}$ constrains the activations of each cross-attention map associated to a noun to be concentrated in the area where the corresponding instance is within the reference image. To do so, every noun of a given prompt is extracted with a part-of-speech tagger. They are then used as input for a Grounded-SAM model. The lat-

SD1.4 **TokenCompose**

Figure 2. **Samples from SD1.4 and TokenCompose.** See Figures 4 and 3 for additional details.

ter automatically generates the corresponding segmentation mask within the reference image. The expression of $\mathcal{L}_{\text{token}}$ is as follows:

$$\mathcal{L}_{\text{token}} = \frac{1}{N}\sum_{i=1}^{N}\left(1 - \frac{\sum_{u\in\mathcal{B}_i}^{\mathrm{L}_{z_t}} A(i,u)}{\sum_{u}^{\mathrm{L}_{z_t}} A(i,u)}\right)^2 \tag{1}$$

where N is the number of tokens corresponding to nouns, $\mathcal{B}_i$ is the "active" area of the i-th binary segmentation mask (where it's non zero), L_{z_t} represents all the activations of the i-th cross-attention map. Concretely, it pushes activations to be non-zero only within the active area of a segmentation mask. Although this spatial constraint is applied within the latent space, we know that they keep the same spatial coherence as their decoded version.

1.1.2. Pixel Loss

The authors observed that only using $\mathcal{L}_{\text{token}}$ led to some unwanted effects. In particular, although the activations tend to concentrate in the area where the corresponding instance is being generated, they are only concentrated on sub-regions of that area. The behavior makes token grounding less accurate and can be observed through the attention maps. To control it, the article uses a pixel-wise binary cross-entropy loss:

$$\mathcal{L}_{\text{pixel}} = -\frac{1}{L_{\tau_\theta(y)} L_{z_t}}\sum_i\sum_u\Big(\mathcal{M}(i,u)\log(A(i,u)) + \big(1-\mathcal{M}(i,u)\big)\log\big(1-A(i,u)\big)\Big) \tag{2}$$

where $\mathcal{M}(i)$ denotes the i-th segmentation mask. However, the authors do not justify why this would negatively impact the output image. The only claim is that, given an image generated with different checkpoints of SD1.4 using Null-Text Inversion [5], grounding seems more convincing when this additional loss term is added. The activations of the cross-attention map of a given category cover more the area ultimately occupied by the generated instance. Although finding a metric to quantify the effect of the pixel-level loss term may be tedious, there is no qualitative comparison of it's potential advantage on an image.

1.2. MultiGen benchmark

This benchmark plays a key role in evaluating the efficiency of the proposed method. The benchmark is made up of prompts with five randomly chosen classes among either the ADE20K dataset [10] or the COCO dataset [4]. Then, a vocabulary detector is used to detect whether the requested categories were generated or not.

The rest of the report is organized as follows: we first assess the reproducibility of TokenCompose and validate the quantitative and qualitative results claimed by the authors. Then we use the properties of TokenCompose to improve the results of Blended Latent Diffusion [1], an inference-based method to locally edit images with pre-trained LDMs.

2. Reproducibility

2.1. Usability of the code

The code provided by the authors of TokenCompose was maintained and straightforward enough to be used as a code base for all the experiments we carried out. In particular, finetuning an SD1.4 using TokenCompose and running the MultiGen benchmark could be performed without any error or unexpected behavior. To extract cross-attention maps during inference, we adapted the code of the training loop. Results are illustrated on Figures 4 and 3 Finally, as stated in the paper, using our own checkpoints for inference could be done in a plug-and-play fashion, making comparisons between different checkpoints fast and easy.

2.2. Results on the MultiGen benchmark

TokenCompose was evaluated by running the MultiGen benchmark with different checkpoints on a fine-tuned SD1.4. We first ran the benchmark on the checkpoints provided by the authors. They are indicated as A and B in Table 1. We also evaluated our own checkpoint, obtained after fine-tuning an SD1.4 with TokenCompose. Despite the fact that we did not retrieve the same results as in the paper, the authors claimed their method to be state-of-the-art on their benchmark with no additional remark. This is concerning given that the second best model in terms of score on the MultiGen benchmark often lies within the error margin of the score obtained by the best model. As a consequence, our retrained version of TokenCompose (last row of Table 1) turns out not to be the state-of-the-art model in at least one subcategory of the benchmark.

Model	COCO Instances				ADE20K Instances			
	MG2	MG3	MG4	MG5	MG2	MG3	MG4	MG5
TokenCompose (Original)	**98.08**$_{0.40}$	**76.16**$_{1.04}$	28.81$_{0.95}$	3.28$_{0.48}$	**97.75**$_{0.34}$	**76.93**$_{1.09}$	**33.92**$_{1.47}$	**6.21**$_{0.62}$
TokenCompose (Checkpoint A)	97.15$_{0.47}$	76.15$_{1.01}$	**29.43**$_{0.68}$	**3.66**$_{0.42}$	97.24$_{0.37}$	76.57$_{1.12}$	33.56$_{1.33}$	5.91$_{0.44}$
TokenCompose (Checkpoint B)	96.53$_{0.55}$	75.54$_{1.22}$	27.27$_{1.76}$	2.82$_{0.40}$	97.26$_{0.20}$	76.52$_{1.18}$	34.17$_{1.37}$	5.92$_{0.62}$
Attn-Exct [2]	93.64$_{0.76}$	65.10$_{1.04}$	28.01$_{0.95}$	6.01$_{0.61}$	91.74$_{0.49}$	62.51$_{0.94}$	26.12$_{0.78}$	5.89$_{0.40}$
TokenCompose (Retrained)	96.83$_{0.67}$	73.47$_{1.40}$	27.34$_{0.74}$	3.30$_{0.56}$	96.50$_{0.43}$	75.99$_{1.02}$	33.45$_{1.30}$	5.87$_{0.57}$

Table 1. **Results of the MultiGen benchmark.** We compare the performance claimed in the paper and the results of the benchmark on the checkpoints provided by the authors, as well as a retrained version. Except for the hardware, all the experiments, training and benchmark included, were conducted with the same hyper-parameters as the one used for the paper. Hence, it is relevant to directly compare the results claimed by the paper and those we obtained. Overall, the scores for the reference range within the same order of magnitude as those for the reference. However, there is an obvious gap, up to more than 2.5% (MG3) between our retrained version and the reference. This leads TokenCompose no longer being the state-of-the-art method for multicategory composition. For example, with a score of 27.34 on MG4 with COCO instances, Attend-and-excite by Chefer *et al*. [2] performs better than TokenCompose. Given the variance of the scores and some variability induced during training, these poorer results may just be due to this randomness.

3. Local image editing

3.1. Motivation

As demonstrated in both quantitative and qualitative results, TokenCompose achieves to strongly improve grounding between text tokens associated to nouns and specific regions of the output image. We can therefore take advantage of the local effect of activations encouraged by TokenCompose for local image editing. There are several ways to locally edit images with LDMs. They all try to tackle the two same main issues:

- We first want the unedited region of the source image not to be modified at all. This is quite challenging, as the source image is either noised or used to condition the denoising process in most methods. For this particular case, the behavior observed within the cross-attention maps of an LDM fine-tuned with TokenCompose is what would naturally be expected when locally editing an image. Indeed, we expect the model to focus only on the object or area to be edited and not on the whole image.
- Then, we want the raw output image (with no post-processing) to have a seamless transition between the edited area and the background.

3.2. Blended Latent Diffusion

(Note: In this section, the terms *edited region* and *target region* are used interchangeably)

Among all the previous work on image editing, Blended Latent Diffusion (Blended Diffusion for short) introduced by Avrahami *et al*. [1] is a natural choice to get the best out of TokenCompose in terms of editing. The key motivation for using this method is that it involves using a segmentation mask to ensure that the editing is applied only in the region of interest. Blended Diffusion therefore minimizes any alteration of the background as much as possible. This approach and its motivations are close to the ones of TokenCompose. So in this particular framework, TokenCompose would act as a prior conditioning of the SD1.4 model to better suit the editing task. Hence, we intuitively expect the preservation of the background in the resulting image to be even more accurate with an LDM fine-tuned with TokenCompose. A minor but still meaningful motivation for using Blended Diffusion is that it is inference-based. Thus, experiments could be carried out easier and faster.

3.3. Experiments

To evaluate the editing capabilities of TokenCompose within the framework of Blended Diffusion, we reused the code base provided by the authors of the method. The latter allows to use different checkpoints of SD1.4 in the same plug-and-play fashion as the code from TokenCompose. By visualizing the cross-attention maps during the editing process, we noticed several relevant details:

- As observed before, the activations of the cross-attention maps are concentrated on the area of the image associated with the category at stake. In contrast, cross-attention maps obtained when using a frozen SD1.4 backbone for editing keep the same aspect. It highlights that introducing a segmentation during editing is not a strong enough mechanism to concentrate activations within the area of interest.
- Surprisingly, regardless of the token associated with a given cross-attention map, activations tend to have a higher value in the area being edited. This kind of semantic leakage could have several reasons that we did not investigate in the report. From a purely qualitative point of view, this can be encouraged by some small overlapping between the area to edit and the one corresponding to another category present in the image. This however would not explain everything, because we observed this

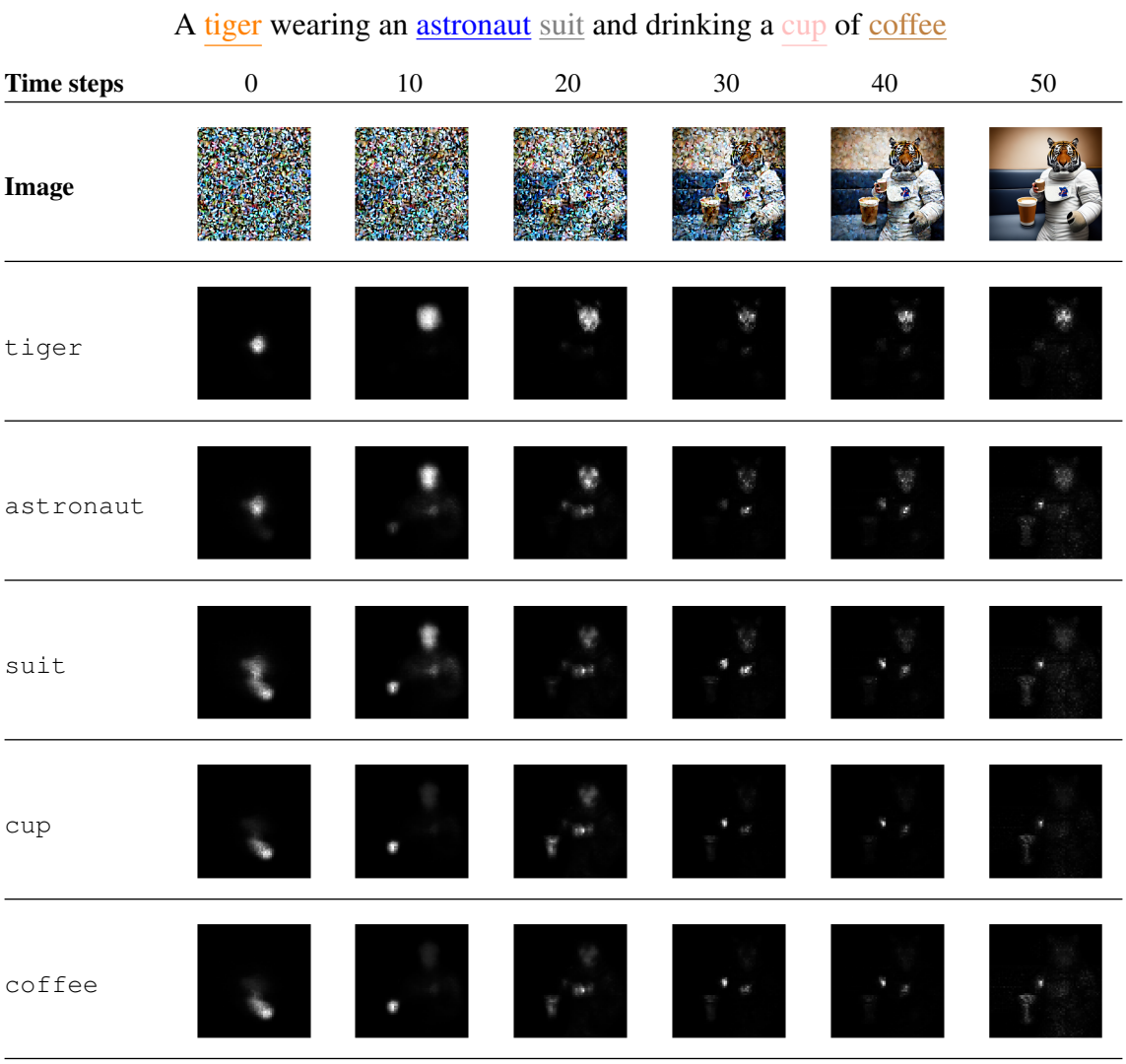

Figure 3. **TokenCompose generation.** For each word token, we show here the evolution of the 64×64 cross-attention maps of the decoder block of the denoising U-Net. See Figure 4 for the result obtained with a default SD1.4. The fully generated image can also be seen on Figure 2

phenomenon even when there is explicitly no overlap between both regions.

3.4. Results

3.4.1. Qualitative results

Generally speaking, qualitative results vary a lot from one image to another. Both SD1.4 and TokenCompsoe editing may provide satisfactory results or fail at at least one of the two aforementioned challenges of image editing. However, after several trials on different images, the following trends emerged.

- Images edited with an SD1.4 backbone look more appealing at first glance, mostly because the transition between the edited region and the background is almost seamless. In contrast, editing with TokenCompose as a backbone outputted more often neatly, thus undesired transitions between both regions. This especially occurs when the newly generated category does not cover a major part of the target region. The strong grounding capability of TokenCompose is the most plausible reason for this observation. Filling the target region with something else that the requested category boils down to an inpainting task. However, this involves generating another category in that same region. As a consequence of its grounding property, TokenCompose is less able than a frozen SD1.4 to inpaint an area. In other terms, it struggles generating categories than what was prompted.
- In terms of scene coherence, TokenCompose, as claimed

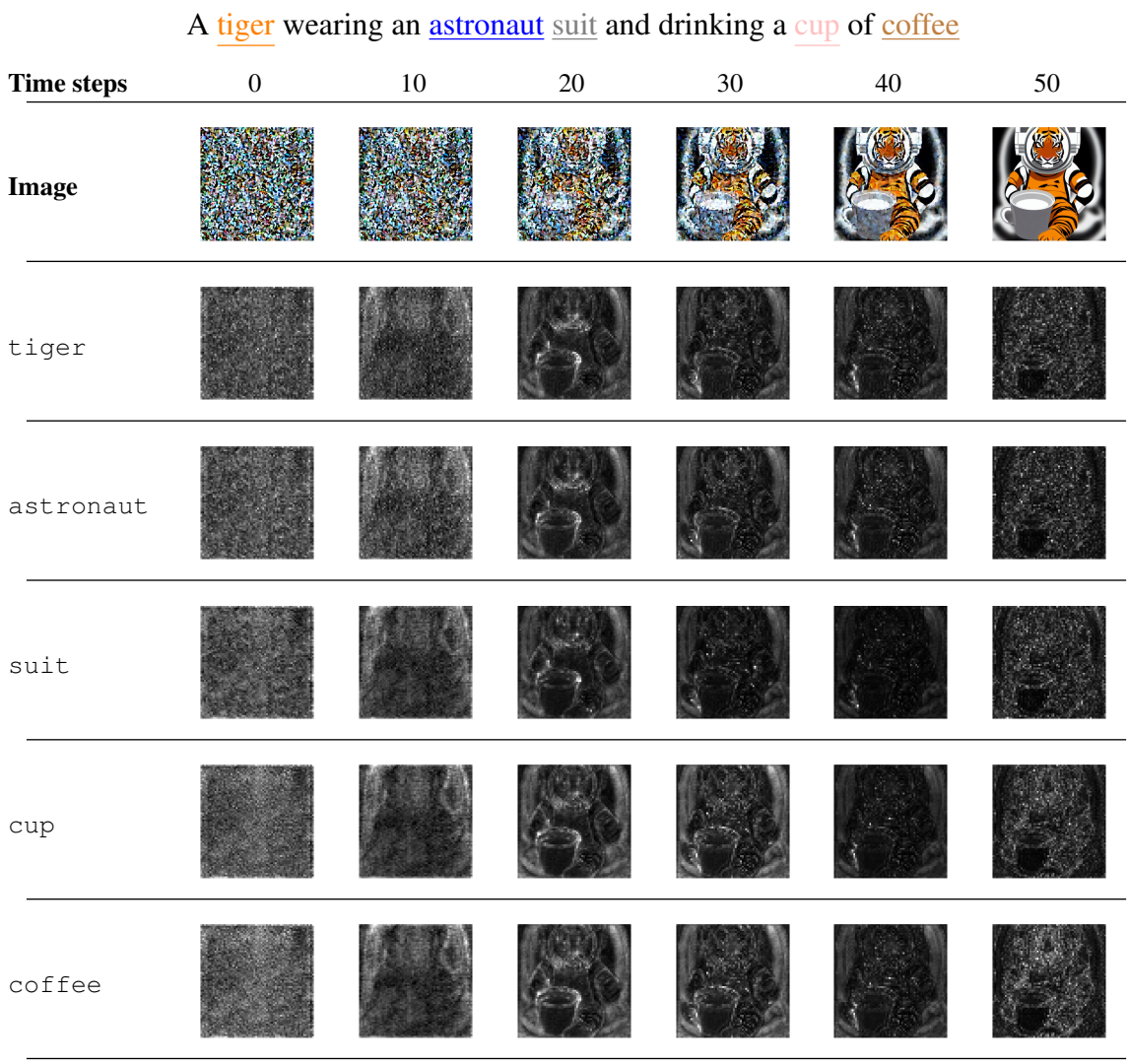

Figure 4. **SD1.4 generation.** As explained in Figure 3, we visualize the cross-attention maps of the decoder block of the denoising U-Net. We notice the weaker grounding between tokens and target regions, as stated by the authors of [9]. The fully generated image can also be seen on Figure 2

Metrics	Structure	Background Preservation				CLIP Similarity	
Model	Distance$_{\times 10^3}$ ↓	PSNR↑	LPIPS$_{\times 10^3}$ ↓	MSE$_{\times 10^4}$ ↓	SSIM$_{\times 10^2}$ ↑	Whole↑	Edit↑
Direct Inversion+P2P [3]	**11.65**	27.22	54.55	32.86	84.76	18.02	22.10
Blended Diffusion (SD1.4)	56.47	28.44	**38.64**	23.85	86.05	18.55	22.80
Blended Diffusion (TokenCompose)	56.61	**28.46**	39.19	**23.63**	**86.08**	**26.02**	**23.23**

Table 2. **Results of the PIE-Bench benchmark**
(Direct Inversion+P2P figures retrieved from the original paper)

in the paper, implicitly makes the scene coherent. Figure 6 shows for instance that the stone was generated right under the cat's paws. This behavior is probably encouraged by the fact that in TokenCompose, SD1.4 is fine-tuned with a subset of the MS-COCO dataset, which only contains real images. The latter therefore intrinsically strengthen the spatial coherence of the generated images.

Figure 5. **Background reconstruction defects.** (Extracted from the original paper [1]) With other local editing methods we observe undesired artifacts in the reconstructed background and rough transitions between the edited region and the background.

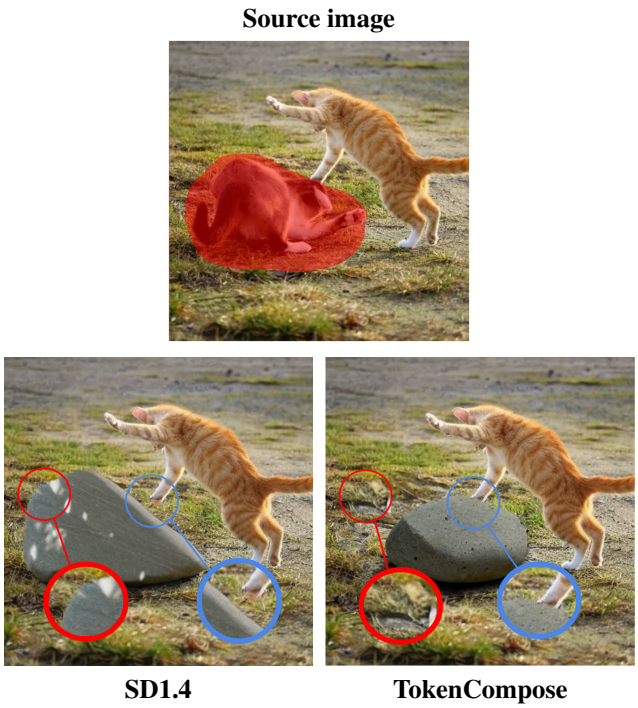

Figure 6. **Natural image edited either with SD1.4 or TokenCompose in Blended Diffusion.** In addition to the source image, we prompted Blended Diffusion to replace the cat with a stone. Key differences between both Stable Diffusion checkpoints are highlighted with the red and blue circles.

3.4.2. Quantitative results

We used the benchmark PIE-Bench introduced by Ju *et al*. [3]. This benchmark computes several metrics for both the target region and the background. In addition to Blended Diffusion, we added the results of Direct Inversion, as PIE-Bench was introduced in the same paper. Although Blended Diffusion with a TokenCompose backbone tends to perform better for every metric but the LPIPS and the Distance, scores for both approaches are extremely close. Therefore, we cannot conclude that one of them performs significantly better than the other for these metrics. However, we notice a clear increase of the CLIP similarity for both the target region and the background with TokenCompose.

Those observations illustrate that, although TokenCompose is still not the perfect choice for image editing, it overcomes some spatial coherence issues. This might be the desired behavior in some real-world use cases. They also show that there is for now a trade-off between visual and semantic quality.

4. Further improvements

The combination of TokenCompose with Blended Diffusion provides promising results in terms of scene coherence and semantic preservation. However, the improved grounding brought by TokenCompose turns out to affect the viasual quality of the edited image. In the particular context of image editing, it should be less strict or more adapted. That is why we could fine-tune SD14 specifically for image editing. Not only should we keep the same loss terms as TokenCompose to benefit from the improved grounding, but also should we add a contrastive loss term to ensure an accurate semantic reconstruction.

References

[1] Omri Avrahami, Ohad Fried, and Dani Lischinski. Blended Latent Diffusion. *ACM Transactions on Graphics*, 42(4):1–11, 2023. arXiv:2206.02779 [cs]. 2, 3, 6, 7

[2] Hila Chefer, Yuval Alaluf, Yael Vinker, Lior Wolf, and Daniel Cohen-Or. Attend-and-Excite: Attention-Based Semantic Guidance for Text-to-Image Diffusion Models, 2023. arXiv:2301.13826 [cs]. 3

[3] Xuan Ju, Ailing Zeng, Yuxuan Bian, Shaoteng Liu, and Qiang Xu. Direct inversion: Boosting diffusion-based editing with 3 lines of code, 2023. 5, 6

[4] 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, 2015. 2

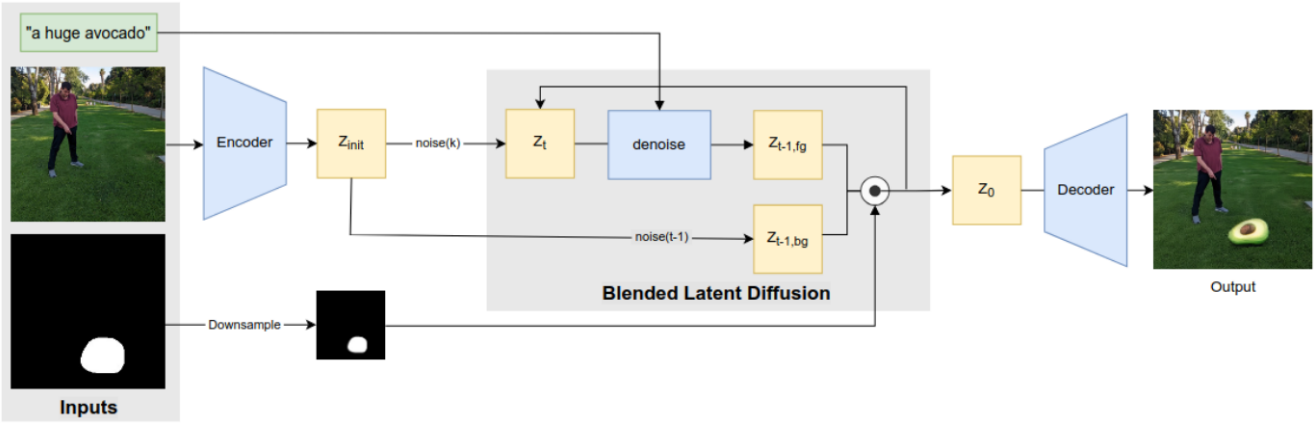

Figure 7. **Diagram of Blended Latent Diffusion.** (Extracted from the original paper [1]) The method works as follows: After encoding the image to edit, a noised is added in a single step to reach a target time step of the noising scheduler. Typically, this time step is $t = 750$, given that the scheduler used with SD1.4 has 1000 noising steps in total. In other words, the input image is 75% degraded. A denoising step conditioned by the edit prompt is then performed. The mask is applied to the output activations with a point-wise multiplication. The obtained non-zero region is denoted as the foreground (fg). In parallel, the background (bg) is unconditionally denoised. After applying the necessary number of iterations (here 750) to completely denoise the image, we get an edited version of the input image.

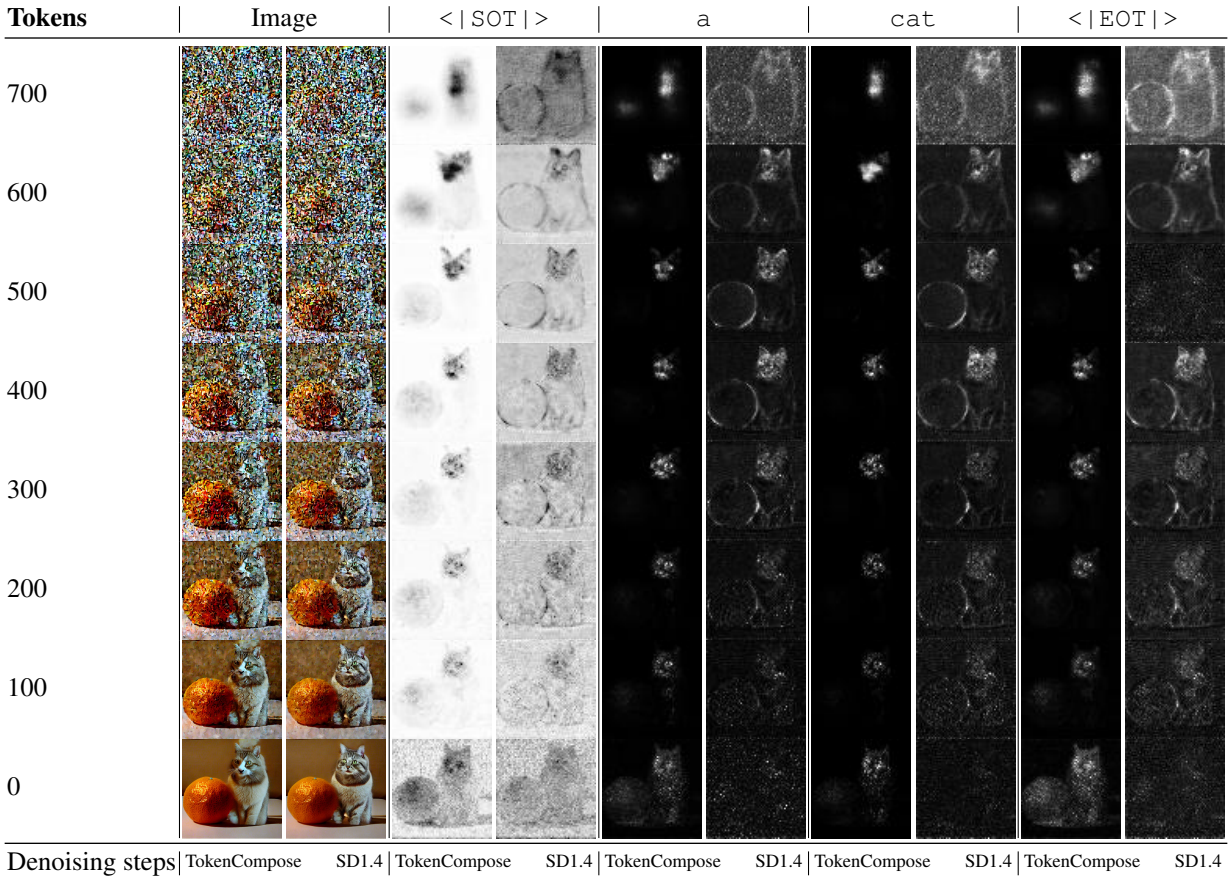

Figure 8. **Image editing process.** Here we visualize the denoising process during editing with Blended Diffusion. Both TokenCompose and default SD1.4 achieve satisfactory results. We clearly observe here the enhanced grounding between tokens and the area being edited brought by TokenCompose. Denoising is performed in 750 steps. See Figure **??** to better observe the differences between edited images.

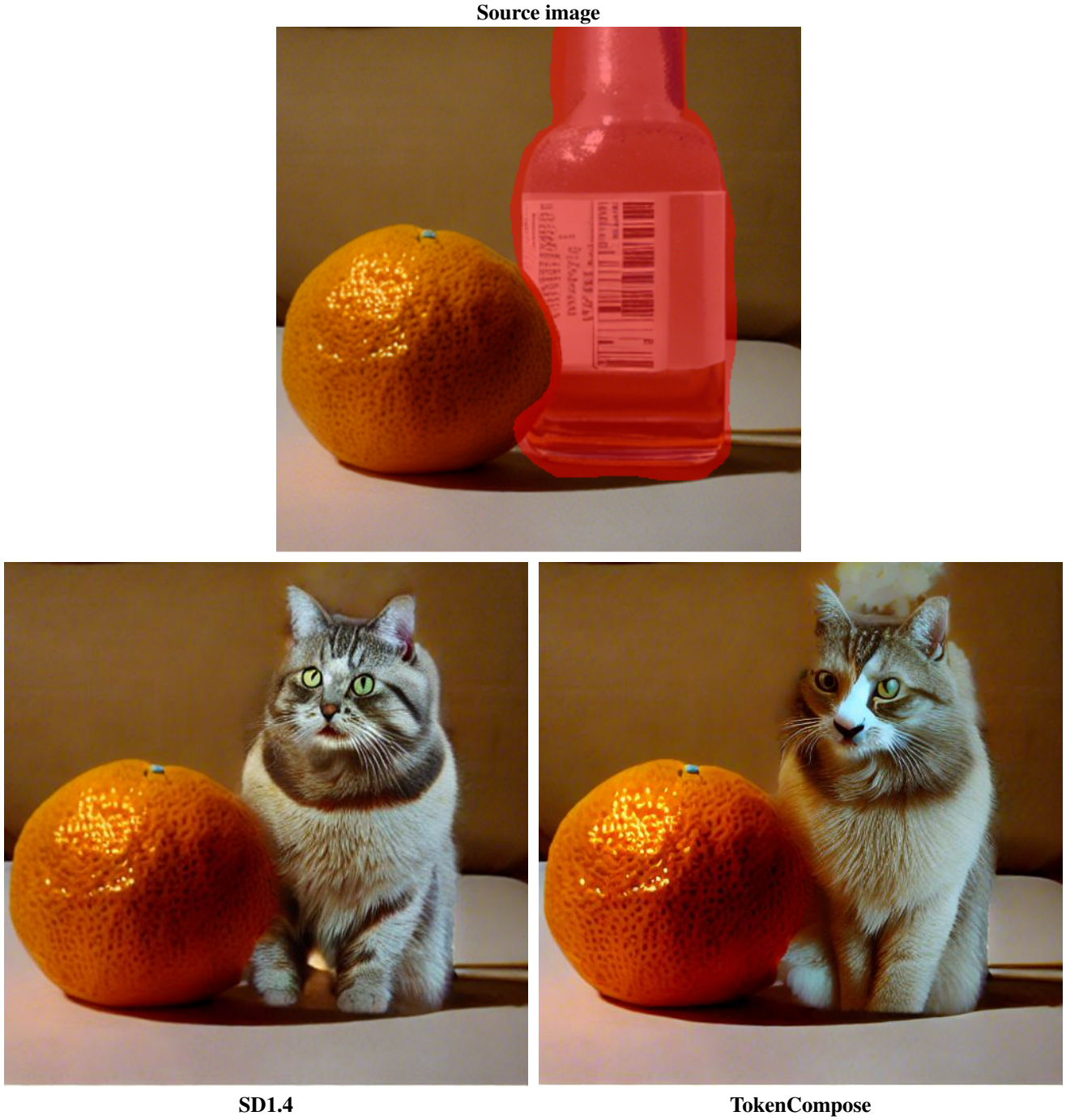

Figure 9. **Synthetic image edited either with SD1.4 or TokenCompose in Blended Diffusion.** The original image wa genrated using TokienCompose with the prompt "A bottle next to an orange". We then used the segmentation mask as with the prompt "a cat" to generate the edited image. This example demonstrates again the poor inpainting capabilities of TokenCompose. The top part of the bottle is replaced by an artifact. In contrast, SD1.4 successfully replaced that same region by the background color.

[5] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text Inversion for Editing Real Images using Guided Diffusion Models, 2022. arXiv:2211.09794 [cs]. 2

[6] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and

Mark Chen. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models, 2022. arXiv:2112.10741 [cs]. 1

[7] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models, 2022. arXiv:2112.10752 [cs]. 1

[8] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5B: An open large-scale dataset for training next generation image-text models, 2022. arXiv:2210.08402 [cs]. 1

[9] Zirui Wang, Zhizhou Sha, Zheng Ding, Yilin Wang, and Zhuowen Tu. TokenCompose: Text-to-Image Diffusion with Token-level Supervision, 2024. arXiv:2312.03626 [cs]. 1, 5

[10] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset, 2018. 2