supplemental material for Fast Imagic: Solving Overfitting in Text-guided Image Editing via Disentangled UNet with Forgetting Mechanism and Unified Vision-Language Optimization

Shiwen Zhang*
Bytedance Inc
shiwen.zhang@bytedance.com

Editors: Marco Fumero, Clementine Domine, Zorah Lähner, Donato Crisostomi, Luca Moschella, Kimberly Stachenfeld

1 Appendix / supplemental material

1.1 Visual Storytelling

Our Fast Imagic could precisely preserve the characteristics of multiple actors and is capable of conducting complex non-rigid editing, which makes our Fast Imagic an ideal tool for visual storytelling and long video generation with strong consistency and very arbitrary scene and action. In Figure **??**, we input a random image generated by SDXL (2) and then use Fast Imagic with Realistic Vision V6.0 B1 noVAE, a variant of Stable Diffusion to generate various samples for different target prompts. With image to video models, for example Stable Video Diffusion (1), we could generate movies with high consistency of several minutes.

1.2 decomposition in text embedding space

We compare two different reasoning methods to merge e_{src} and e_{tgt} to get the final text embedding e, shown in Figure 1 . These two methods are complementary to each other, with vector projection better at preserving the identity, and vector subtraction showing stronger editing capability. Thus in the workflow of Figure **??**, we use vector subtraction to be the default option. When the characteristics could not be preserved by vector subtraction, we switch to vector projection.

For the dog and the cat examples, vector projection can preserve more details of the appearance of the dog and the cat than vector subtraction. However, for a glass of milk and cookie example, vector subtraction performs better than vector projection which struggles to change the milk to juice and also introduces wave-like blurs in the image. We observe such phenomenons in many other cases for vector projection, which demonstrates that it is more suitable for edits where the identity of object should be kept instead of changed.

1.3 Disentangled UNet

The default workflow is in the highlighted flow in Figure 6, where the default forgetting mechanisms are 'encoderattn' and 'decoderattn'. However, we still demonstrate in Figure 3 and Figure 2 how changing which part of parameters to merge the models influence the editing result. We first inference

*Or contact me via my personal email witcherofresearch@gmail.com. Codes are available at https://github.com/witcherofresearch/Forgedit.

Proceedings of the II edition of the Workshop on Unifying Representations in Neural Models (UniReps 2024).

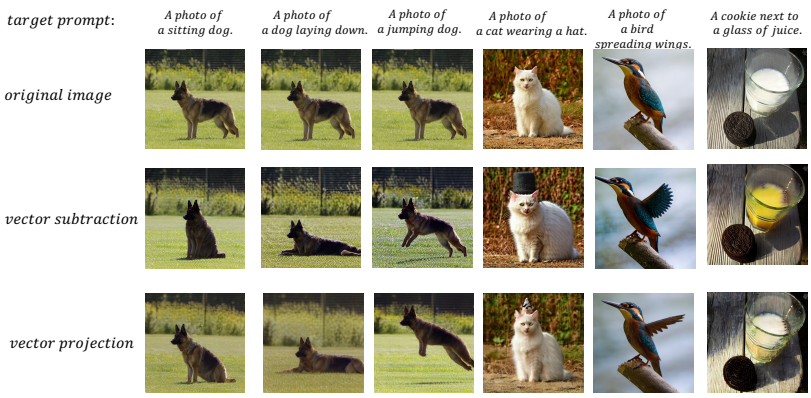

Figure 1: Comparisons of vector subtraction and vector projection, which are complementary.

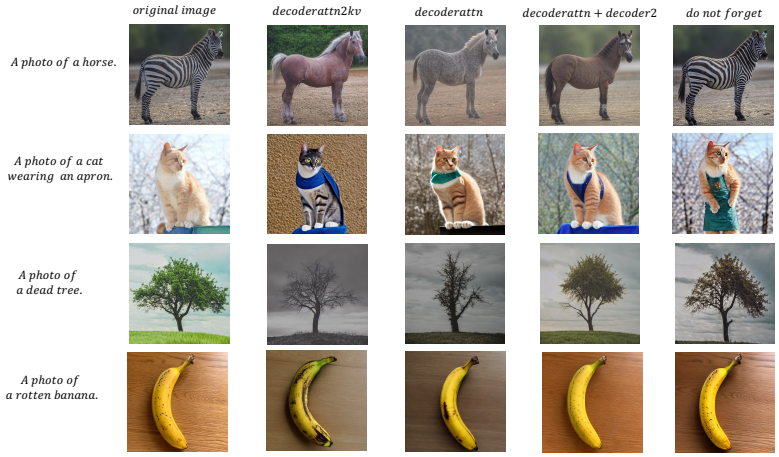

Figure 2: We explore various forgetting strategies for decoder. All learned encoder parameters are preserved. In the 2^{nd} to 4^{th} columns, we preserve decoder cross-attention parameters, decoder self-attention and cross-attention, decoder self-attention, cross-attention and the entire decoder2 block, forgetting all the other parameters of decoder.

without forgetting strategies. If overfitting happens, we choose from the default 'encoderattn' or 'decoderattn' strategy according to the UNet property and target prompt intention. The 'encoderattn' means forgetting all encoder parameters except attention-related parameters. 'decoderattn' means forgetting all decoder parameters except attention-related parameters. The user may choose to forget more or fewer parameters according to the editing results, which we demonstrate and explain in 3 and 2.

1.4 The interpolation hyper-paramters of vector subtraction and vector projection

We explore the effect of hyper-parameters to the editing result in Figure 4.

1.5 Joint optimization of vision and language

The fine-tuning process of Imagic is composed of two stages, text embedding optimization for 500 steps and UNet optimization for 1000 steps. Our Fast Imagic employ unified vision language optimization with a batch-wise traing on 1 a100 GPU, which leads to 40 steps in total, speeding up Imagic for 14 times. This unified optimization of vision and lanugage goes beyond speeding up Imagic. It also eases the overfitting issue to some extent, shown in Figure 5.

Figure 3: We explore different forgetting strategies for encoder. All learned decoder parameters are preserved. For the second to fourth column each, we preserve none of the encoder parameters, encoder self attention and cross attention, encoder self attention and cross attention and the entire encoder1 block, forgetting all the other parameters of encoder.

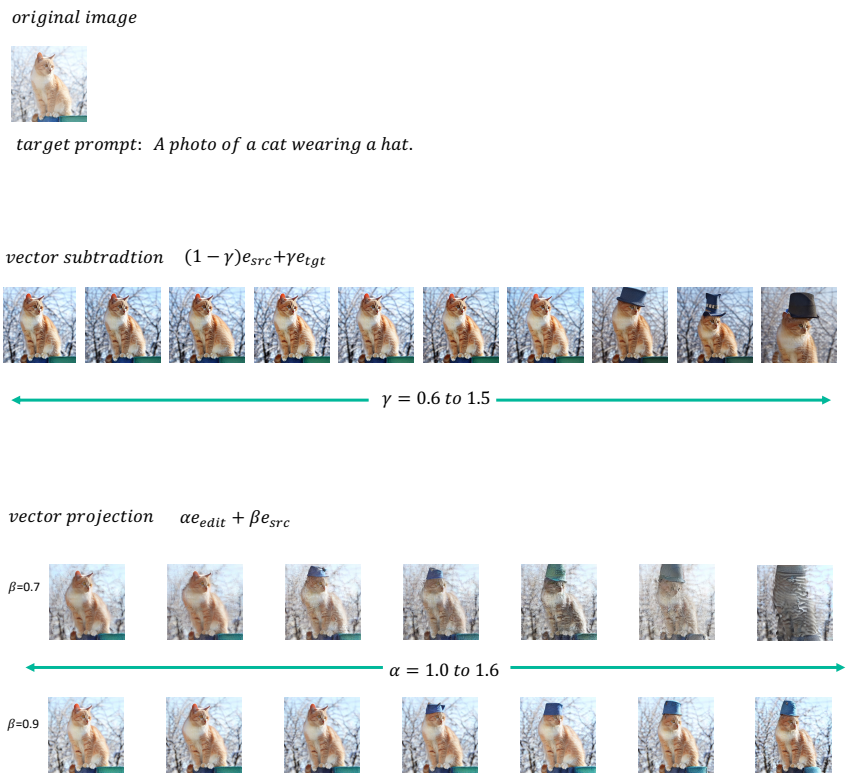

Figure 4: γ for vector subtraction and α, β for vector projection.

Figure 5: What should the source prompt be? Excluding the usage of forgetting strategies for ablation, we could find that Fast Imagic using target prompt leads to severe overfitting, yet Fast Imagic using BLIP generated source prompt eases overfitting.

1.6 Practical Examples of the Workflow

The overall workflow is shown in Figure 6. No matter what the input image is, we use the same set of hyper-paramters for finetuning stage. In the editing stage, the default workflow is to use vector subtraction with γ in the range of 0.8 to 1.6. In general, a proper editing result should already been obtained from one of these 8 images. However, if a perfect editing did not show up, there are two possibilities, overfitting or underfitting. Underfitting leads to the fact that the edited object suffers from identity shift, which means with the editing strengthened, the appearance of target object becomes gradually inconsistent with input image. In this circumstance, one needs to apply vector projection instead, which I will show in another paper with examples from TEdBench. The more often case is overfitting, which means that Fast Imagic could reconstruct the input image well yet cannot conduct the edit successfully. With the disentangled property of UNet, we could utilize the forgetting strategy to tackle the overfitting issue. If the target prompt aims to edit space and structure, one should use the default "encoderattn" forgetting strategy. If the target prompt aims to edit appearance and texture, one should use the default "decoderattn" forgetting strategy. Using the examples from EditEvalv1 benchmark, we demonstrate several cases on how to adjust the hyper-parameters. The base model used in the following examples is Stable Diffusion 1.4.

For the first case where the input image is a polar bear on the ice field, the target prompt is "A polar bear raising its hand". To begin the workflow in Figure, we first run vector projection without forgetting strategy with with γ in the range of 0.8 to 1.6. Shown in Figure , we could find that we are facing the overfitting issue and the polar bear is incapable of raising its hands. Following Figure , we then run the default forgetting strategy on UNet's encoder, i.e. "encoderattn", which means that newly learned parameters of self attention blocks and cross attention blocks are preserved in UNet encoder and all learned parameters of UNet decoder are preserved as well. The hyper-parameter γ still ranges from 0.8 to 1.6. This time we could find successful edits in the results.

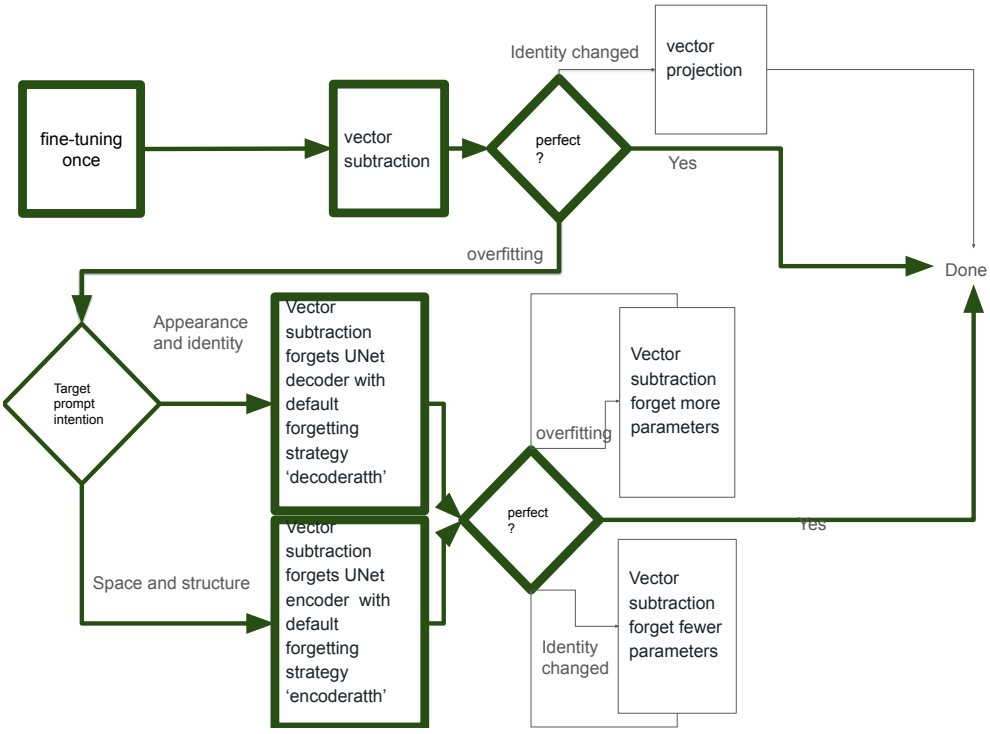

Figure 6: The workflow of Fast Imagic, the most usual flow of editing process is highlighted in the figure, i.e. simple vector subtraction and default forgetting strategies according to our findings of the disentangle rules of UNet.

1.7 Limitations

First the effect of Fast Imagic is influenced by randomness. The fine-tuning process inevitably introduces randomness thus for some particular cases, we cannot guarantee to perfectly reconstruct the details of original image thus we have to run the fine-tuning stage several times for these challenging cases. The sampling procedure is also related to the initial random seed of reverse process, thus for some extremely challenging cases we have to sample tens of images or even hundreds, though rarely the case, before we could get a proper edited one.

Second, the editing capability of Fast Imagic is restricted by the utilized Diffusion Model. If the target prompt cannot even be generated by the Diffusion Model itself, it is almost impossible to accomplish the edit according to the target prompt. For example, the prompt 'a sitting flamingo' cannot be generated by Stable Diffusion at all, thus Fast Imagic cannot successfully edit it either. Such an issue could possibly be solved by switching to better Diffusion Models.

We show some typical bad cases in Figure 8.

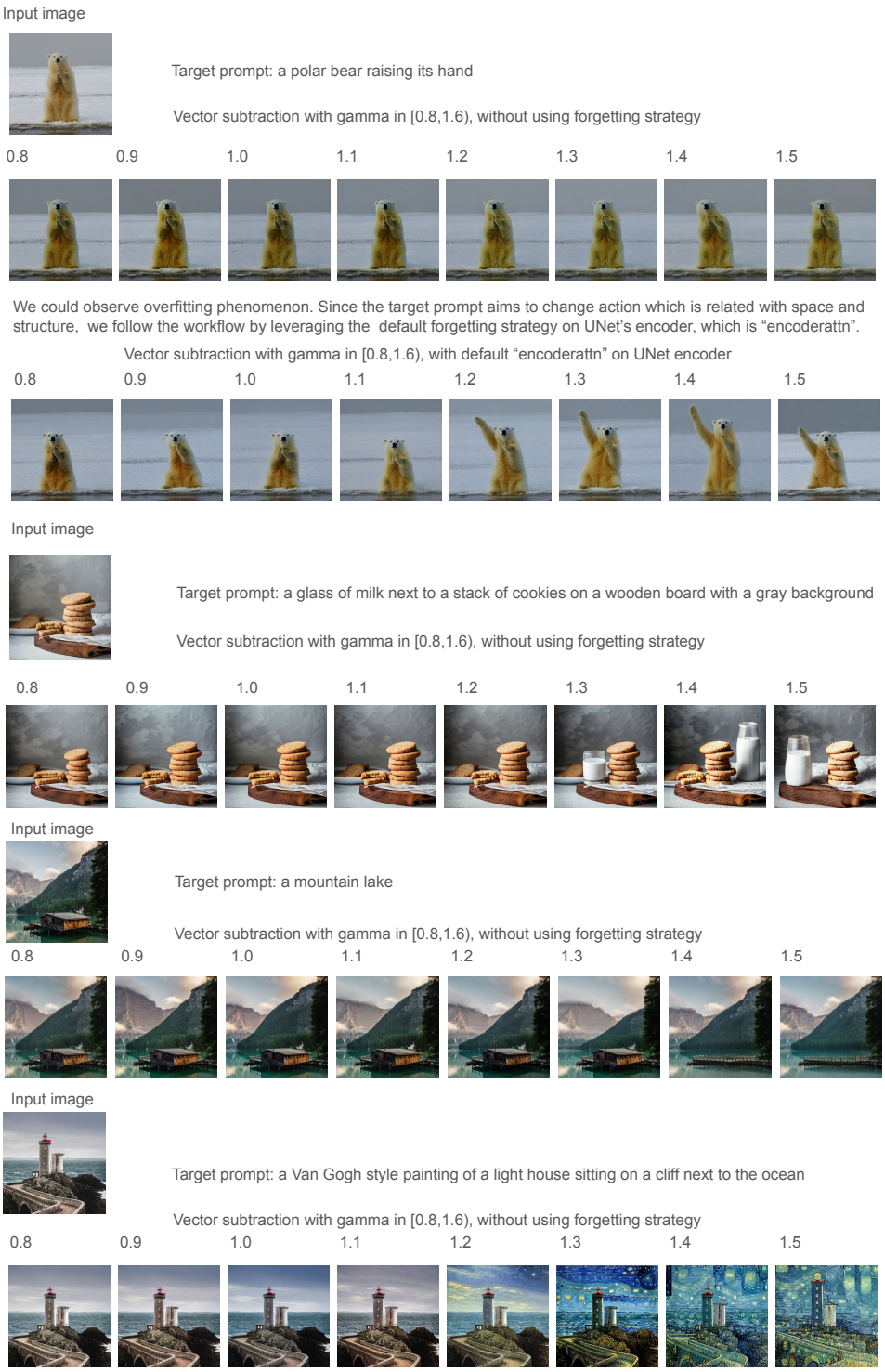

Figure 7: We show the practical workflow of Fast Imagic, with testing images from EditEval. In most cases, simple vector subtraction would finish the job. For other hard cases, the default forgetting strategies, 'encoderattn' or 'decoderattn' according to editing intention on structrue or appearance, could solve the problems.

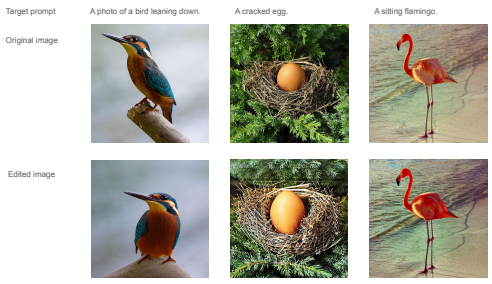

Figure 8: Bad cases from TEdBench.

References

[1] Blattmann, A., Dockhorn, T., Kulal, S., Mendelevitch, D., Kilian, M., Lorenz, D., Levi, Y., English, Z., Voleti, V., Letts, A., et al.: Stable video diffusion: Scaling latent video diffusion models to large datasets. arXiv preprint arXiv:2311.15127 (2023)

[2] Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: SDXL: improving latent diffusion models for high-resolution image synthesis. CoRR **abs/2307.01952** (2023)