ELITE: Encoding Visual Concepts into Textual Embeddings for Customized Text-to-Image Generation – Supplementary Materials –

Yuxiang Wei[1,2] Yabo Zhang[1] Zhilong Ji[3] Jinfeng Bai[3] Lei Zhang[2] Wangmeng Zuo[1,4(✉)]
[1]Harbin Institute of Technology [2] The Hong Kong Polytechnic University [3]Tomorrow Advancing Life [4] Peng Cheng Lab

The following materials are provided in this supplementary file:

- Sec. A: more ablation studies of our ELITE (cf. Sec. 4.2 in the main paper), including the value of λ, layer indexes, local attention map reweighting, and global mapping.
- Sec. B: more training and testing details (cf. Sec. 4.1 in the main paper).

A. More Ablation Studies

A.1. Effect of the value of λ

In Eqn. (6) of the main paper, λ is introduced to control the fusion of information from the global mapping network and the local mapping network. To evaluate its effect, we vary its value from 0 to 1.2 during customized generation. As shown in Fig. 7 of the main paper, with the increase of λ, the consistency between the synthesized image and concept image is improved. Meanwhile, from Fig. A in this supplementary file, the image alignment (*i.e.*, CLIP-I and DINO-I) improves as λ increases. However, when the value of λ is too large, it may lead to degenerated editing results, leading to decreased text alignment (*i.e.*, CLIP-T). Therefore, for a trade-off between inversion and editability, we set $\lambda = 0.6$ for editing prompts and $\lambda = 0.8$ for generating prompts. We find these parameters work well for most cases.

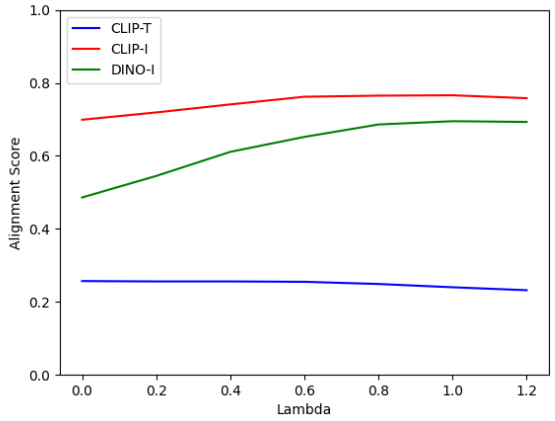

Figure A. Ablation on the value of λ. As λ increases, the CLIP-I and DINO-I improve, yet the text alignment (CLIP-T) slightly deteriorates.

A.2. Effect of the layer indexes

In our experiments, we select the features of five layers of the CLIP image encoder to learn multiple word embeddings, whose indexes are $\{24, 4, 8, 12, 16\}$ in order. We have further conducted ablation studies by putting the deepest layer (*i.e.*, layer 24) in different orders. Specifically, we compare four variants. i) **Single-layer Multi-words**: learning multiple word embeddings from the deepest feature separately. ii) **Multi-layers Multi-words First**: our setting, learning multiple word embeddings from the multiple layer features separately, and the layer indexes are $\{24, 4, 8, 12, 16\}$ in order. iii) **Multi-layers Multi-words Middle**: learning multiple word embeddings from the multiple layer features separately, and the layer indexes are $\{4, 8, 24, 12, 16\}$ in order. iv) **Multi-layers Multi-words Last**: learning multiple word embeddings from the multiple layer features separately, and the layer indexes are $\{4, 8, 12, 16, 24\}$ in order. Fig. B illustrates the visualization of words learned by each variant. As shown in the figure, each variant contains one primary word that describes the subject concept. When learning *multiple word embeddings from multi-layer features*, we observe that the primary word is naturally linked to the features from the deepest layer, regardless of the position indices of layers. Besides, as illustrated in Fig. C, in contrast to the primary word obtained by the single layer feature, the primary word learned by multi-layer features is well-editable. Among them, our setting achieves better editability, which is shown in Table A.

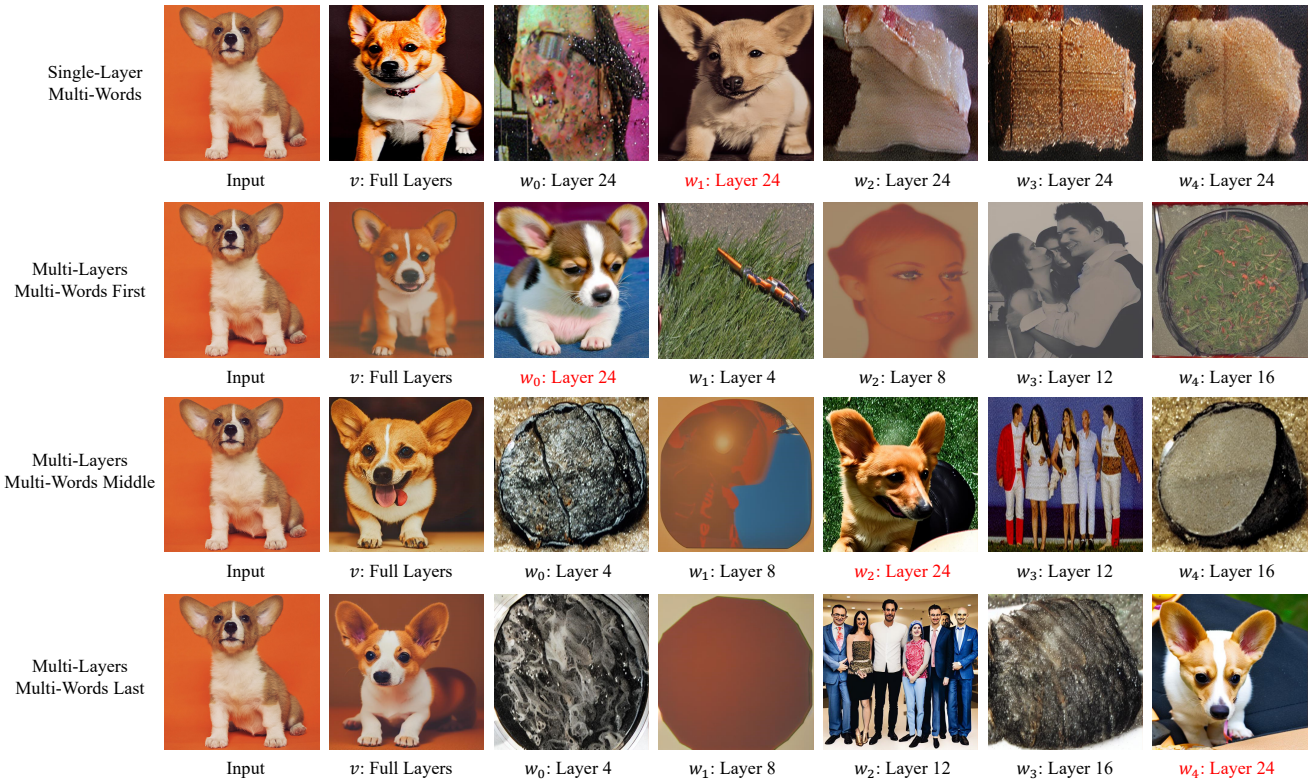

Figure B. **Visualization of learned word embeddings for different variants.** First, Middle, and Last denote the order of deepest feature in learned word embeddings. The learned primary word is highlighted by red color.

Figure C. **Visual comparisons of different variants.** [v] denotes the generation results with full word embeddings v, while [w] denotes the generation results with the primary word embedding w (see Fig. B). The primary word learned by multi-layer features is well-editable.

A.3. Effect of local attention map reweighting

The local mapping network aims to inject the fine-grained details of the given subject during generation. To further emphasize its effect on the subject region rather than irrelevant areas (*e.g.*, background), we reweight the obtained local attention map by multiplying it with the attention map of primary word (refer to Sec. 3.3 in the main paper):

$$A^l = A^l * \frac{A^g_{w_0}}{\max(A^g_{w_0})}, \tag{1}$$

where $A^l = \text{Softmax}\left(\frac{QK^{l\,T}}{\sqrt{d'}}\right)$ denotes the attention map of local mapping network and $A^g = \text{Softmax}\left(\frac{QK^{g\,T}}{\sqrt{d'}}\right)$ denotes the attention map of global mapping network. d' is the output dimension of key and query features. $A^g_{w_0}$ is the attention map of primary word w_0. To verify its effectiveness,

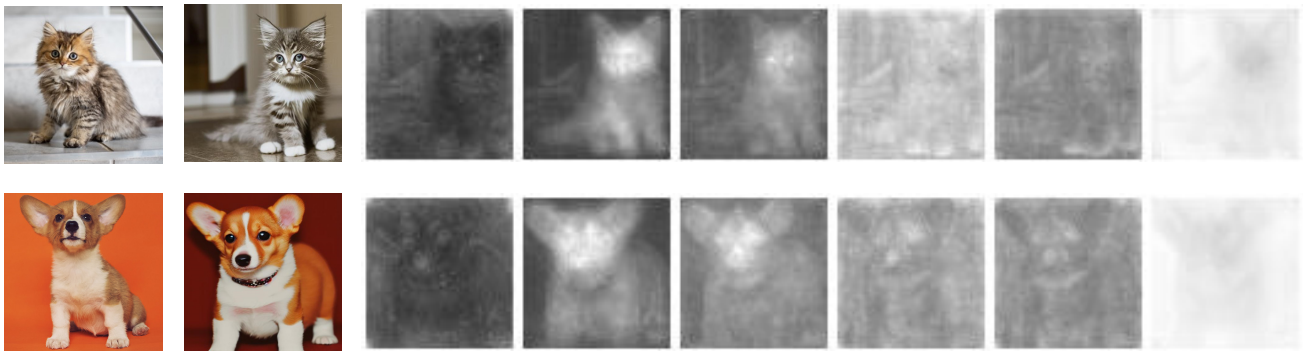

Figure D. Cross-attention map visualization. We show the average attention across timestep and layers for each word embedding. The attention map corresponding to the learned primary word w_0 delineates the subject region.

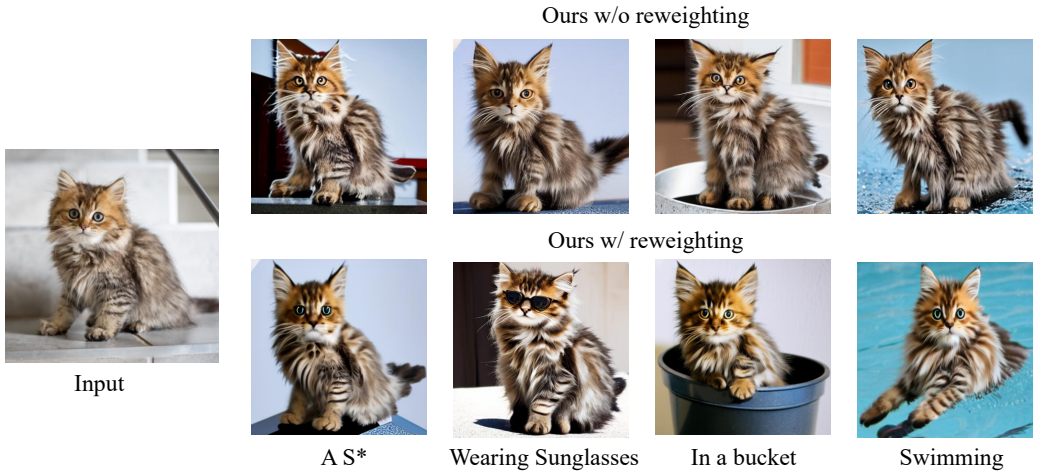

Figure E. Ablation of local attention map reweighting. Without the local attention reweighting, the learned model tends to be less editable.

Table A. **Ablation study**. [v] denotes the generated testing results with full word embeddings v, while [w] denotes thee generated testing results with the primary word embedding w. First, Middle, and Last denote the position of word embedding with respect to the deepest feature.

Method	CLIP-T (↑)	CLIP-I (↑)	DINO-I (↑)
Single-Layer Multi-Words [v]	0.198	0.683	0.431
Single-Layer Multi-Words [w]	0.212	0.692	0.443
Multi-Layers Multi-Words Middle [v]	0.211	0.726	0.592
Multi-Layer Multi-Words Middle [w]	0.249	0.673	0.444
Multi-Layers Multi-Words Last [v]	0.217	0.722	0.585
Multi-Layers Multi-Words Last [w]	0.247	0.694	0.474
Multi-Layers Multi-Words First [v]	0.204	**0.771**	**0.658**
Multi-Layer Multi-Words First [w]	**0.257**	0.699	0.486

we firstly visualize the cross-attention map of each word in input text prompt in Fig. D. As one can see, the learned primary word w_0 is associated with the subject concept and its attention map $A^g_{w_0}$ accurately delineates the subject region, so we can leverage it to reweight the local attention map A^l. Furthermore, as illustrated in Fig. E, without the local attention reweighting, the features of local mapping network may affect the subject-irrelevant areas, resulting in degraded editability. In contrast, our ELITE with reweighting strategy reduces the disturbances on subject-irrelevant areas, and achieves better editability.

A.4. Effect of global mapping

We have further conducted the ablation study to evaluate the effect of our global mapping network. For comparison, we remove the global mapping network, while replace the pseudo word `S*` with a ground-truth category word to learn the local mapping network (*e.g.*, `S*` $\rightarrow$ `dog` in Fig. F). As shown in Fig. F, without the global mapping, using the local mapping network only provides a few fine-grained details (*e.g.*, fur color), yet fails to keep the structure of the concept (*e.g.*, ear). In contrast, by adding global mapping network to encode a suitable primary word embedding, our ELITE faithfully recovers the target concept with higher visual fidelity while enabling robust editing.

Figure F. Ablation of global mapping network. With the global mapping network, our ELITE faithfully recovers the target concept with higher visual fidelity while enabling robust editing.

B. More Experimental Details

B.1. Training Details

Textual Inversion [1]. We use the official stable diffusion version of Textual Inversion[1]. For each subject, experiment is conducted with the batch size of 1 and a learning rate of 0.005 for 5,000 steps. The new token is initialized with the category word, *e.g.*, "cat".

Custom Diffusion [2]. We use the official implementation of Custom Diffusion[2]. We train it with a batch size of 1 for 300 training steps. The learning rate is set as 1e-5. The regularization images are generated with 50 steps of the DDIM sampler with the text prompt "`A photo of a [category]`".

DreamBooth [3]. We use the third-party implementation of DreamBooth[3]. Training is done with finetuning both the U-net diffusion model and the text transformer. The training batch size is 1 and the learning rate is set as 1e-6. The regularization images are generated with 50 steps of the DDIM sampler with the text prompt "`A photo of a [category]`". For each subject, we train it for 800 steps.

B.2. Testing Datasets

For customized generation, we adopt concept images from existing works [1–3] with 20 subjects, including dog, cat, and toy, *etc*. Fig. G illustrates the full image samples.

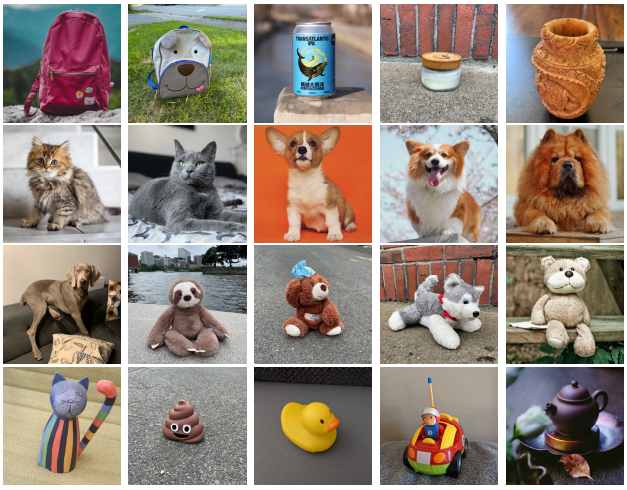

Figure G. Testing image samples.

B.3. Text prompts

We adopt the text prompt list used in Textual Inversion [1] for training, which is provided as below:

- "a photo of a S_*",
- "a rendering of a S_*",
- "a cropped photo of the S_*",
- "the photo of a S_*",
- "a photo of a clean S_*",
- "a photo of a dirty S_*",
- "a dark photo of the S_*",
- "a photo of my S_*",
- "a photo of the cool S_*",
- "a close-up photo of a S_*",
- "a bright photo of the S_*",
- "a cropped photo of a S_*",
- "a photo of the S_*",
- "a good photo of the S_*",
- "a photo of one S_*",
- "a close-up photo of the S_*",
- "a rendition of the S_*",

[1] https://github.com/rinongal/textual_inversion
[2] https://github.com/adobe-research/custom-diffusion
[3] https://github.com/XavierXiao/Dreambooth-Stable-Diffusion

Table B. Text prompt list for quantitative evaluation.

Text prompts for non-live objects	Text prompts for live objects
"a S_* in the jungle"	"a S_* in the jungle"
"a S_* in the snow"	"a S_* in the snow"
"a S_* on the beach"	"a S_* on the beach"
"a S_* on a cobblestone street"	"a S_* on a cobblestone street"
"a S_* on top of pink fabric"	"a S_* on top of pink fabric"
"a S_* on top of a wooden floor"	"a S_* on top of a wooden floor"
"a S_* with a city in the background"	"a S_* with a city in the background"
"a S_* with a mountain in the background"	"a S_* with a mountain in the background"
"a S_* with a blue house in the background"	"a S_* with a blue house in the background"
"a S_* on top of a purple rug in a forest"	"a S_* on top of a purple rug in a forest"
"a S_* with a wheat field in the background"	"a S_* wearing a red hat"
"a S_* with a tree and autumn leaves in the background"	"a S_* wearing a santa hat"
"a S_* with the Eiffel Tower in the background"	"a S_* wearing a rainbow scarf"
"a S_* floating on top of water"	"a S_* wearing a black top hat and a monocle"
"a S_* floating in an ocean of milk"	"a S_* in a chef outfit"
"a S_* on top of green grass with sunflowers around it"	"a S_* in a firefighter outfit"
"a S_* on top of a mirror"	"a S_* in a police outfit"
"a S_* on top of the sidewalk in a crowded street"	"a S_* wearing pink glasses"
"a S_* on top of a dirt road"	"a S_* wearing a yellow shirt"
"a S_* on top of a white rug"	"a S_* in a purple wizard outfit"
"a red S_*"	"a red S_*"
"a purple S_*"	"a purple S_*"
"a shiny S_*"	"a shiny S_*"
"a wet S_*"	"a wet S_*"
"a cube shaped S_*"	"a cube shaped S_*"

- "a photo of the clean S_*",
- "a rendition of a S_*",
- "a photo of a nice S_*",
- "a good photo of a S_*",
- "a photo of the nice S_*",
- "a photo of the small S_*",
- "a photo of the weird S_*",
- "a photo of the large S_*",
- "a photo of a cool S_*",
- "a photo of a small S_*",

For qualitative evaluation, we employ the editing templates used in [1–3]. For quantitative evaluation, we employ the editing prompts from DreamBench [3], which contains 25 editing prompts for each subject. The full prompts are listed in Table B.

References

[1] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022. 4, 5

[2] Nupur Kumari, Bingliang Zhang, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Multi-concept customization of text-to-image diffusion. *arXiv preprint arXiv:2212.04488*, 2022. 4, 5

[3] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. *arXiv preprint arXiv:2208.12242*, 2022. 4, 5