JeDi: Joint-Image Diffusion Models for Finetuning-Free Personalized Text-to-Image Generation

Supplementary Material

1. Implementation Details

In this section, we describe the key modifications based on StableDiffusion v1.4[1] to implement the proposed method. We point to the original location in StableDiffusion code and highlight the modified lines in each code snippet.

Data loading. We store all images and captions belonging to an image set in a single image file and text file. The images are concatenated vertically, and the captions corresponding to different images are separated by a special token `<|split|>`, as illustrated in Fig. 1. This enables us to easily reuse existing single image dataloaders in PyTorch. We only use square images in training and obtain the image set size by dividing the image height with the image weight. Given a batch of data, we extract individual images and text, and obtain the size of the image set `ng` as follows,

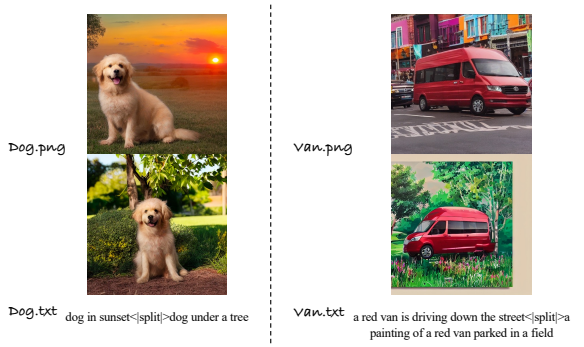

Figure 1. **Data format.** An image file is the concatenation of all images in the set. The text file contains corresponding captions separated by a special token.

```python
# ldm/models/diffusion/ddpm.py#L865-L868
def get_group_data(self,img,txt):
    b,c,h,w = img.shape
    ng = h//w
    assert h==w*ng
    img = img.view(b,c,ng,w,w)
    img = img.transpose(1,2).reshape(b*ng,c,w,w)
    txt = reduce(lambda
    ↪  a,b:a+b,[t.split("<|split|>") for t in
    ↪  txt])
    return img,txt,ng
def shared_step(self, batch, **kwargs):
    k1,kc=self.first_stage_key,self.cond_stage_key
    assert kc=='txt'
    batch[k1],batch[kc],ng =
    ↪  self.get_group_data(batch[k1])
    x, c = self.get_input(batch,
    ↪  self.first_stage_key)
    loss = self(x, c)
    return loss
```

[1] https://github.com/CompVis/stable-diffusion

Joint-image diffusion models. The proposed joint-image diffusion model can be easily implemented based on a single-image diffusion model with a few simple modifications as follows,

```python
# ldm/modules/attention.py#L211C15-L215
# ng: size of the image set
def _forward(self, x, context=None,ng=None):
    b,l,c = x.shape
    if ng is not None:
        x = x.view(-1,ng*l,c)
    x = self.attn1(self.norm1(x)) + x
    if ng is not None:
        x = x.view(b,l,c)
    x = self.attn2(self.norm2(x), context=context)
    ↪  + x
    x = self.ff(self.norm3(x)) + x
    return x
```

Personalization as inpainting. As described in Sec. 3.3 of the paper, we cast the personalized generation problem into an inpainting task. In training, the inpainting masks are generated randomly. First a binary random vector `ng_mask` is sampled with every bit set to zero or one with equal probability. Then a `mask` of the same spatial size as the input image is constructed by replicating `ng_mask` alone the height and width dimensions. The actual input fed into the U-Net is the concatenation of the mask `mask`, noisy image `x`, and masked clean image `x0*(1-mask)`. The key implementation can be found in the following code snippet.

```python
# ldm/modules/diffusionmodules/openaimodel.py
#L730-L730
bs=x.size(0)
if ng_mask is None:
    ng_mask =
    ↪  th.empty(bs,device=x.device).bernoulli_()
mask = ng_mask[:,None,None,None]
x0 = x0*(1-mask)
mask = mask.expand(-1,-1,*x0.shape[-2:])
x = th.cat((x,x0,mask),1)
h = x.type(self.dtype)
```

At test time, `x0` is the concatenation of all input images and an all-zero image. The corresponding elements in `ng_mask` are set to 0 for the input images and 1 otherwise.

Synthetic same-subject (S^3) dataset. Algorithm 1 describes the process of training data generation (Fig. 3 in the paper) in details. Please note that we use the term *instance segmentation* for simplicity, however, in our implementation, we combine an object detection model [4] and a segmentation model [2] to separate object instances rather than using an actual instance segmentation model. The *object-centric* prompts (GPT($l, object$) in Algorithm 1) are generated by instructing ChatGPT to generate details of an object

Table 1. Quantitative comparisons on the unique subject test set.

Method	CLIP-T (↑)	CLIP-I (↑)	MCLIP-I (↑)	DINO (↑)	MDINO (↑)
BLIP Diffusion [3]	0.2851	0.8107	0.8234	0.6091	0.6018
ELITE [5]	0.2193	0.6082	0.6430	0.1862	0.2156
JeDi	**0.2856**	**0.8697**	**0.8838**	**0.7934**	**0.7926**

l, and the *scene-centric* prompts (GPT($l, scene$)) in Algorithm 1) are generated by instructing it to describe a scene involving the object l. The list of object names used in our implementation can be found in the attached text file.

Algorithm 1: Generating the S^3 dataset.

Input: A list of object names L;
A text-to-image model G;
A text-based inpainting model G_I;
ChatGPT GPT;
An instance segmentation model S;
CLIP image model CLIP;
Output: a database $\mathcal{X}$ of image sets $\mathcal{X} = \{X_1, X_2, ..., X_P\}$ where $X_p = \{x_1, x_2, ..., x_{N_p}\}$ is a set of images share a common subject;
$\mathcal{X} \leftarrow \phi$;
for l **in** L **do**
  Generate an object-centric prompt $t_o \leftarrow$ GPT($l, object$);
  Generate an image $x_0 \leftarrow G(t_o)$;
  Extract object instances from x_0: $\{\hat{x}_1, \hat{x}_2, ..., \hat{x}_K\} \leftarrow S(x_0)$;
  Construct an affinity matrix $A, A_{ij} \leftarrow$ CLIP($\hat{x}_i, \hat{x}_j$);
  Construct an adjacent matrix $M, M_{ij} \leftarrow 1$ if $A_{ij} > 0.95$ else 0;
  Find connected components $\mathcal{I} \leftarrow \{I_1, I_2, ..., I_P\}$ of the graph represented by M;
  for I_p **in** $\mathcal{I}$ **do**
    $X \leftarrow \phi$;
    for i **in** I_p **do**
      Generate a scene-centric prompt involving object name l: $t_s \leftarrow$ GPT($l, scene$);
      Paste the object instance image $\hat{x}_i$ at a random location in an empty image x;
      Inpaint the unknown area in x: $x \leftarrow G_I(x, t_s)$;
      $X \leftarrow X \cup \{x\}$;
    end for
    $\mathcal{X} \leftarrow \mathcal{X} \cup \{X\}$
  end for
end for

Evaluation metric. We use the CLIP ViT-B/32 to compute CLIP-T, CLIP-I and MCLIP-I. We use DINO VIT-S/16 to compute DINO and MDINO. We use one input image per subject for comparison with finetuning-free methods, and average the pair-wise scores to all real images of the same subject and over all possible choices of the input image. For comparison with finetuning-based methods, we randomly select three input images. In ablation studies we use one randomly selected input image for each subject and only compute the scores using input/output pairs by default unless stated otherwise.

2. Additional Experiments

2.1. Additional Results

Fig. 4 shows additional personalized generation results on real-world human and object images. Our method can generate high-quality images with diverse content while preserving the key visual features of the subjects in input images. Although the model is not trained on human-specific data, it can still generate reasonable results for human subjects, as shown in the second and third row of Fig. 4.

2.2. Comparison with State-of-the-Art Methods

Fig. 5 provides additional visual comparisons with finetuning-based methods DreamBooth (DB) and CustomDiffusion (CD). Finetuning-based methods suffer from the overfitting issue and might fail to preserve the subject identity. For common subjects, they tend to extensively copy from the reference images, adding only minor adjustments to match the given text, *e.g.* in the first example, for the prompt *a backpack in the snow*, DreamBooth nearly replicate a reference image with slight snow patterns added in the bottom. For unique subjects, the finetuning-based methods often fail to preserve the distinctive features, *e.g.* the cartoon character in the fourth row. This is because these methods use the loss on retrieved or generated images of similar subjects as regularization during finetuning. For unique and rare objects, these images can be visually distinct from the reference images and interfere the model from memorizing the custom concept.

To further demonstrate the advantage of our methods for challenging cases, we collect a new test set containing unique subjects with only single input image for each subject. Most the input images are from Reddit AI Art channel[2]. Fig. 6 visualize the input images. The first five rows in Fig. 7 compare the results of our method to state-of-the-art finetuning-free methods BLIP-Diffusion (BLIPD) and ELITE on the unique subject test set. We also include the results on common subjects from DreamBooth test set in the last two rows for comparison. It can be seen that BLIPD and ELITE can produce reasonable results for common subjects

[2]https://www.reddit.com/r/aiArt/

such as the dog of a typical breed and a common stuffed animal (row 6-7). However, for unique subjects, their results hardly resemble the subject from the reference image (row 1-5). In contrast, our method can faithfully capture the key visual features of the subject. The advantage of our method is also clearly reflected in the quantitative results in Table 1, where our method outperforms ELITE and BLIP-Diffusion by a large margin.

2.3. Additional Analysis

Image guidance. As we have discussed in the paper, the use of image guidance can significantly improve the faithfulness to the input images. This is also supported by the visual comparison in Fig. 3 (column 4-5).

In our main experiments in the paper, we use a simple strategy for image guidance where both the image and text input are set to null for unconditional inference. Here we discuss a more flexible guidance strategy to model trade-off between image alignment and text alignment. The score function with flexible image guidance is as follows,

$$\tilde{\boldsymbol{\epsilon}}(\boldsymbol{x}_t, \hat{\boldsymbol{x}}, \boldsymbol{M}) = \boldsymbol{\epsilon}^0 + \lambda_1(\boldsymbol{\epsilon}^1 - \boldsymbol{\epsilon}^0) + \lambda_2[\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, \hat{\boldsymbol{x}}, \boldsymbol{M}) - \boldsymbol{\epsilon}^1], \tag{1}$$

where $\boldsymbol{\epsilon}^0 = \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, \mathbf{0}, \boldsymbol{M})$ represents the unconditional score when the text prompt and all reference images are set to null; $\boldsymbol{\epsilon}^1$ represents the partially conditional score when either the text prompt or reference images are kept. We can compute the partial conditional score $\boldsymbol{\epsilon}^1$ using image conditioning to emphasize text alignment, or text conditioning to emphasize image alignment. We call these two options *text first* and *image first* strategies, respectively. Table 2 reports the quantitative results based on different strategies averaged over a varying guidance scale in $[1.5, 10]$. It can be seen that the *text first* strategy yields higher DINO and MDINO scores, indicating better image alignment. *Image first* strategy yields a higher CLIP-T score, which indicates better text alignment.

Table 2. Quantitative results with different guidance strategies averaged over a varying guidance scale in $[1.5, 10]$.

Strategy	Text only	Joint	Image first	Text first
DINO	0.4652	0.7268	0.6558	0.7508
MDINO	0.5922	0.8384	0.7863	0.8527
CLIP-T	0.3259	0.3013	0.3156	0.2853

We can also adjust the ratio between the guidance scale of image condition and text condition for more flexible personalized generation. Fig. 2 visualize the change of DINO and CLIP-T scores with the varying ratio. We can see that the use of image guidance is important. Using only text guidance (Text only in Fig. 2) yields low DINO score, indicating low resemblance to the custom subject. By varying the ratio of text and image guidance scales we can balance

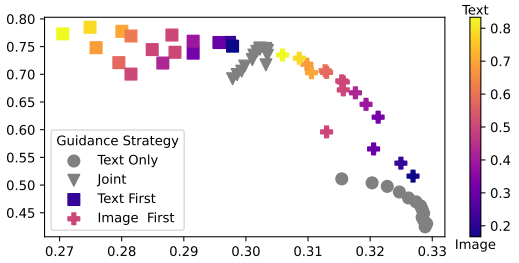

Figure 2. Effect of varying the ratio between the guidance scales for image and text guidance. X-axis: CLIP-T; Y-axis: DINO.

between subject identity preservation and text alignment. We found that the simple joint guidance strategy (Joint in Fig. 2) usually gives good balanced results.

Table 3. Quantitative comparison with three baseline models.

Model	CLIP baseline	Concate. baseline	Learned baseline	JeDi
DINO	0.3411	0.3379	0.7065	0.7501
MDINO	0.4394	0.4292	0.7740	0.8639
CLIP-T	0.3325	0.3247	0.3015	0.3020

Comparison with baseline models. We have discussed a CLIP-encoder baseline in the paper (CLIP baseline). Here we include the comparison to another two baseline models. Concate baseline: the input images are concatenated to the noisy images to be fed into the UNet. Learned baseline: similar to SuTI [1] where an learnable encoder is used to extract a feature vector from the input images. All models are trained on the same training data described in Sec. 4 of the paper and are based on the same StableDiffusion v1.4 backbone for fair comparison. Table 3 reports the quantitative comparison results and the visual comparison can be found in Fig. 3 (column 1,2,3,5). The results indicate that our method significantly outperforms all baseline models in terms of image alignment, as evidenced by considerably higher DINO and MDINO scores.

Training data identity similarity. Table 4 reports the average CLIP and DINO scores on training samples over the 1,000 ImageNet categories, which indicates a high overall identity similarity for a wide array of categories. For context, we also show the scores on real images from the DreamBooth test set, which has a slightly higher identity similarity but covers much less categories than our dataset.

	Subjects	Categories	CLIP-I (↑)	DINO (↑)
S^3 dataset	$1.6M$	$\sim 2K$	0.849	0.751
Real images	30	15	0.885	0.774

Table 4. Training data statistics.

Quantitative results with more input images. Although our method does not have an inherent constraint on the number of inputs, for simplicity, we only use 1-3 input images

in the current implementation. We find that our method still outperforms DB and CD, even when they are finetuned with the maximum available reference images in the test set (4-6 images), as shown in Table 5. We will add the experiments with more reference images, *e.g*. 10, in the revised version.

Table 5. Comparison to DB and CD with the maximum available reference images.

	CLIP-T	CLIP-I (↑)	MCLIP-I (↑)	DINO (↑)	MDINO (↑)
DB	0.2971	0.8025	0.8736	0.6226	0.7175
CD	0.3071	0.7864	0.8586	0.6198	0.7011
Ours (1 input)	0.3040	0.7818	0.8764	0.6190	0.7510
Ours (3 inputs)	0.2932	0.8139	0.9011	0.6791	0.8037

Inference cost. The inference cost is comparable to other methods when N is small (reported in the table below). When N is substantially larger, *e.g*. a database, we can reduce the inference cost by first finetuning the model on the database, and then retrieving the few images closest to the text prompt to be the actual test time input (please refer to the future work section).

Table 6. Inference time for one diffusion step on one A100 GPU.

Method	BLIPD	ELITE	Ours
Time (second) ↓	0.0492	0.0719	0.0564

3. Limitations and Future Work

A limitation of `JeDi` is that it needs to process all reference images at inference time. This enables finetuning-free personalization but leads to efficiency drop when the number of reference images increases. Therefore, `JeDi` is more suitable for subject image generation given a few reference images, and are less efficient in adapting to a new domain given a large database of reference images. A potential solution is to combine `JeDi` with finetuning-based methods. When a large database of reference images are available, we can first finetune `JeDi` on the database. Then at inference time, given a text prompt, we retrieve the most relavent images from the database to use as the test-time inputs to `JeDi`. Another limitation is that the current implementation cannot be directly applied for multi-subject image generation. There are two possible ways to extend `JeDi` for multi-subject generation: (1) generate multiple subjects sequentially through inpainting, and (2) construct a multi-subject S^3 dataset by combining multiple sets of subjects. We will explore these directions in future work.

References

[1] Wenhu Chen, Hexiang Hu, Yandong Li, Nataniel Ruiz, Xuhui Jia, Ming-Wei Chang, and William W Cohen. Subject-driven text-to-image generation via apprenticeship learning. *Conference on Neural Information Processing Systems*, 36, 2024. 3

[2] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023. 1

[3] Dongxu Li, Junnan Li, and Steven CH Hoi. Blip-diffusion: Pre-trained subject representation for controllable text-to-image generation and editing. *arXiv preprint arXiv:2305.14720*, 2023. 2

[4] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023. 1

[5] Yuxiang Wei, Yabo Zhang, Zhilong Ji, Jinfeng Bai, Lei Zhang, and Wangmeng Zuo. Elite: Encoding visual concepts into textual embeddings for customized text-to-image generation. *arXiv preprint arXiv:2302.13848*, 2023. 2

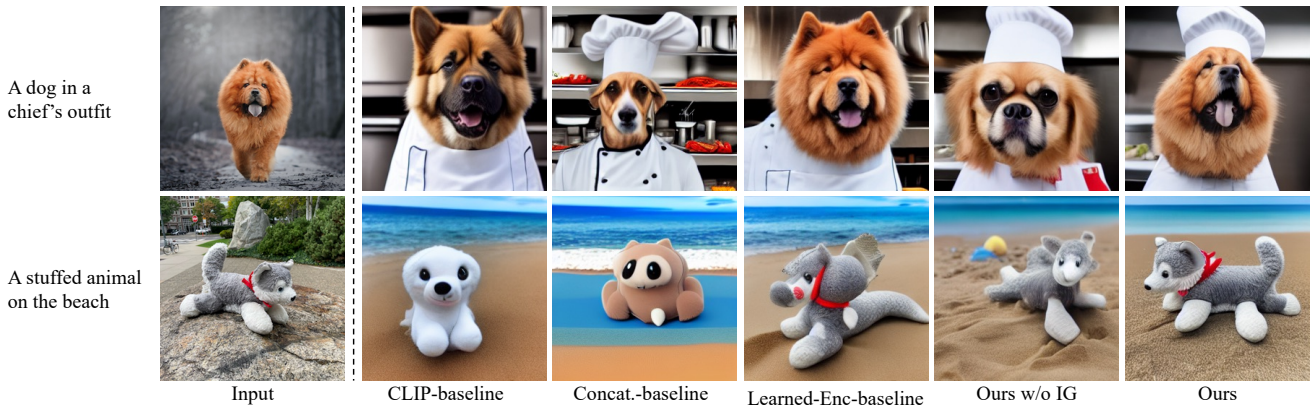

Figure 3. Visual comparisons to baseline models.

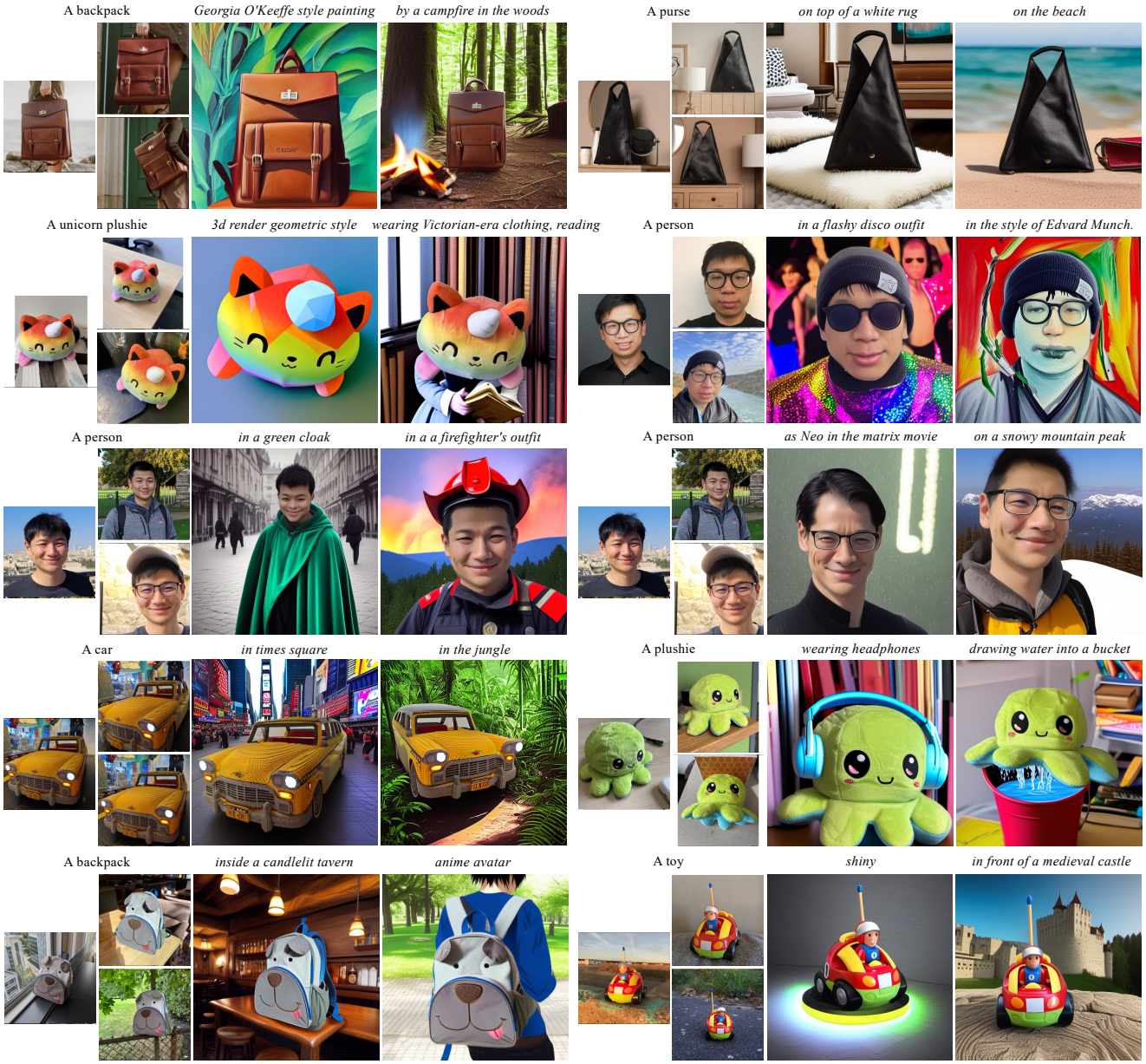

Figure 4. Personalized generation results for human and real-world objects.

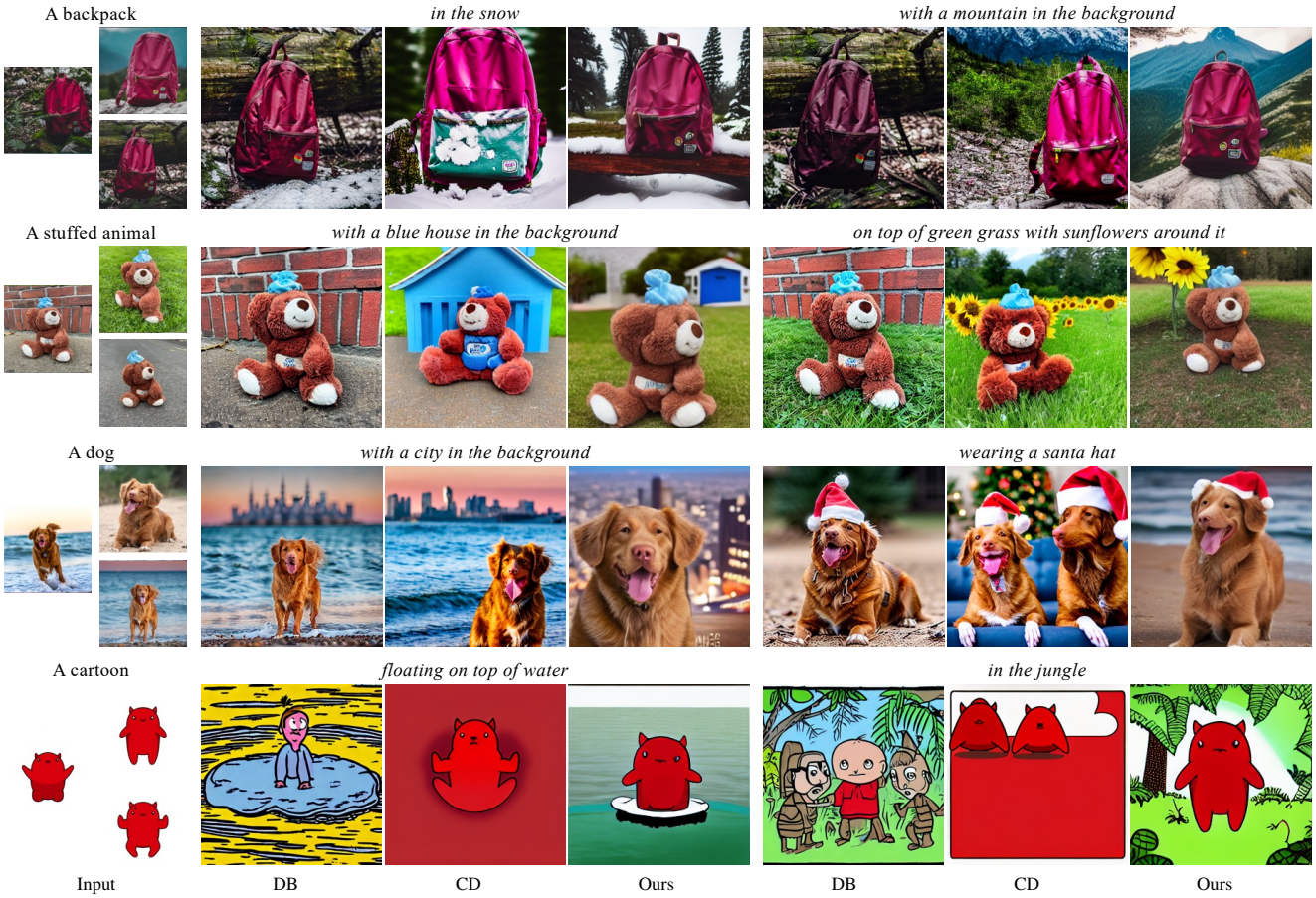

Figure 5. Visual comparison with finetuning-based methods on DreamBooth test set. DreamBooth (DB) and CustomDiffusion (CD) tend to overfit for common subjects and fail to capture the visual features of unique subjects.

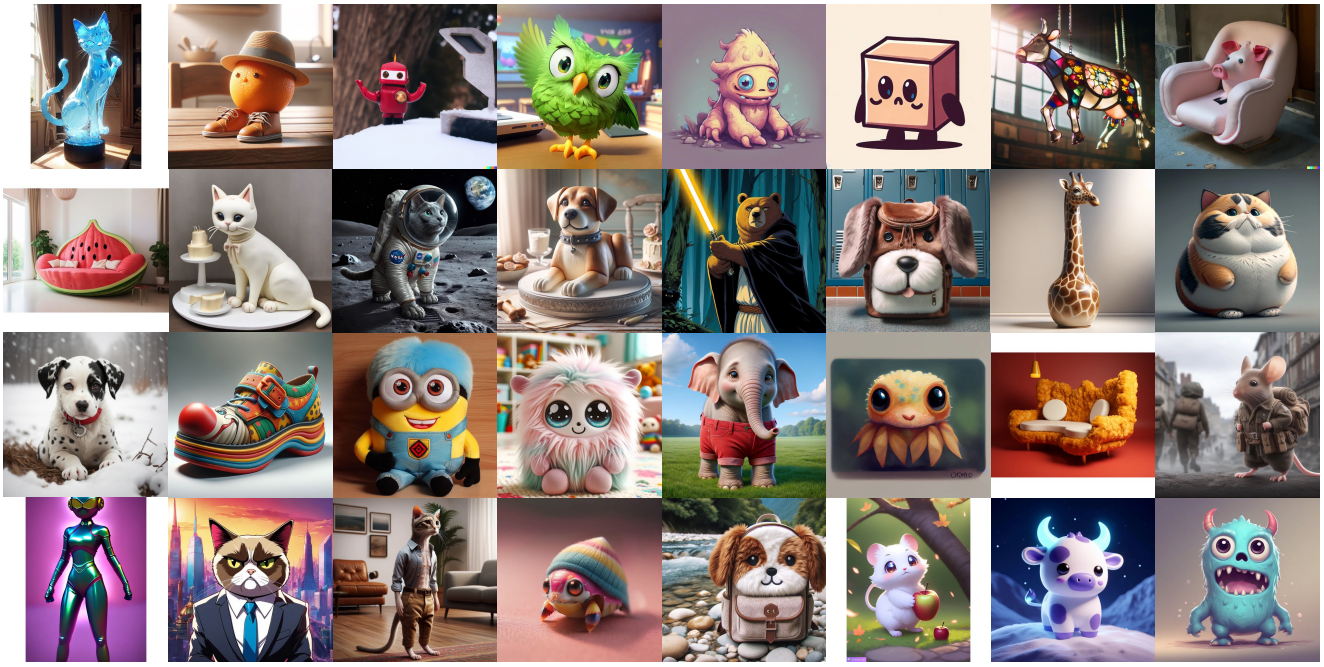

Figure 6. Input images in the unque subject test set.

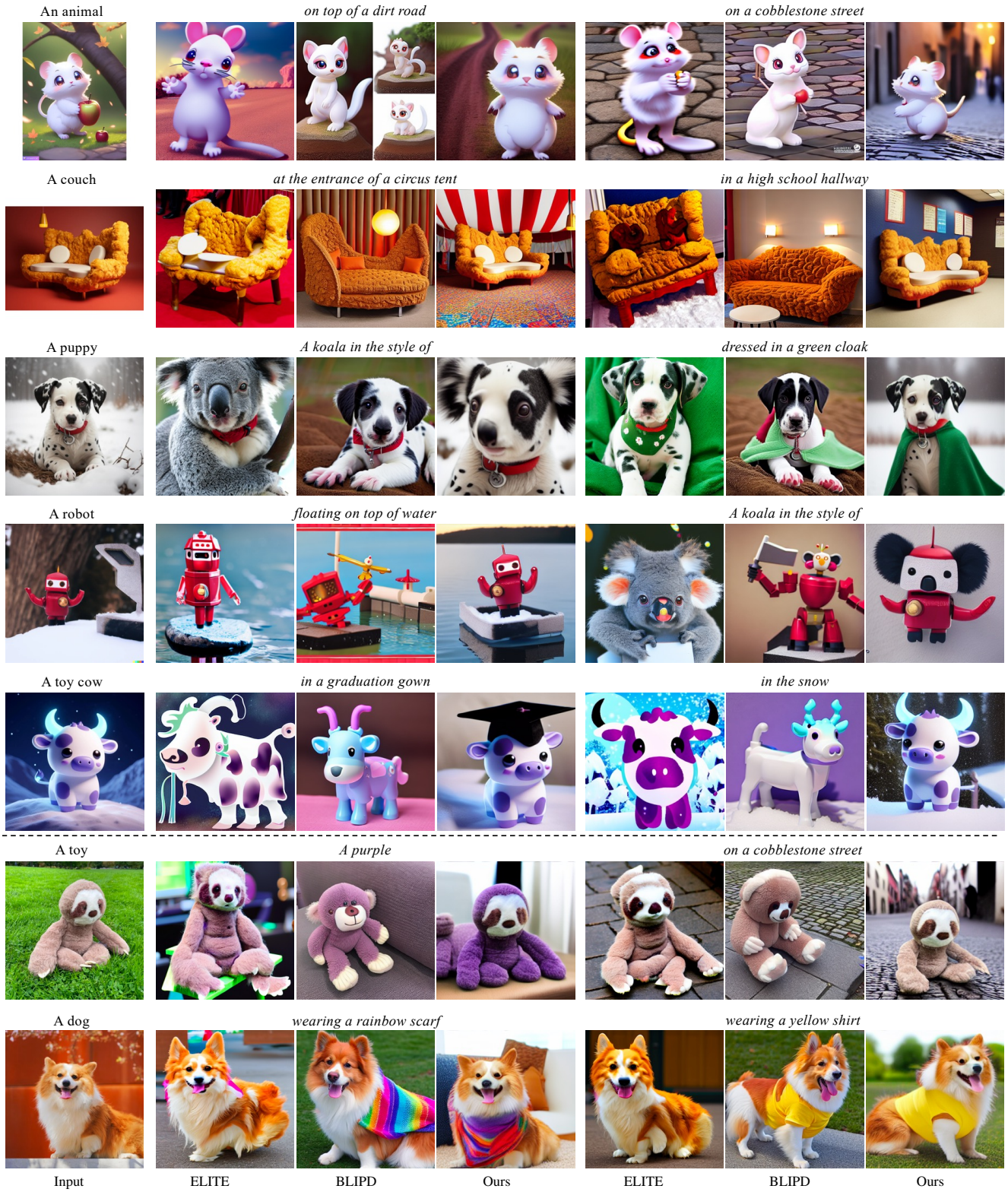

Figure 7. Visual comparison with finetuning-free methods on the unique subject test set (row 1-5) and on Dreambooth test set (row 6-7). BLIP Diffusion and ELITE can generate reasonable results for common subjects but often fail in challenging cases involving unique subjects. In contrast, our method can handle challenging cases and generate personalized images with well-preserved details.