Supplementary Material for MAGE: MAsked Generative Encoder to Unify Representation Learning and Image Synthesis

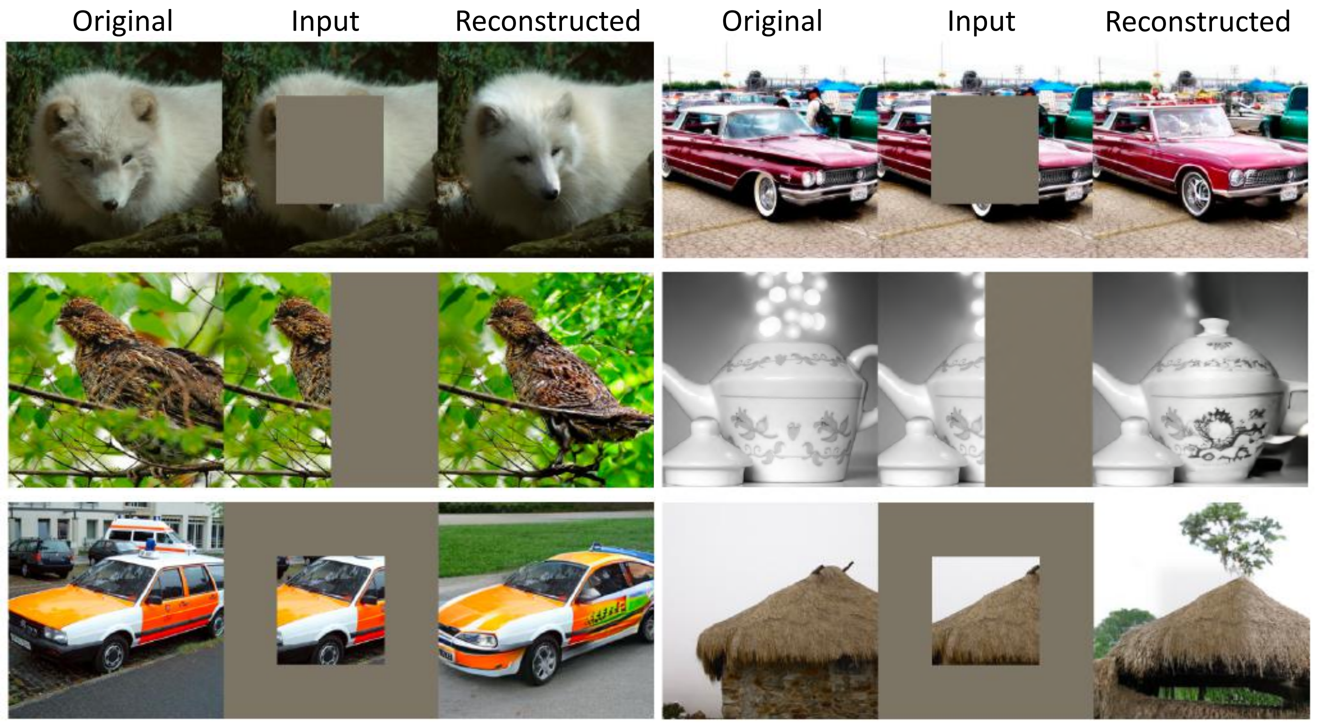

Figure 1. Examples of image inpainting (first row), outpainting (second row), and uncropping (outpainting on a large mask, third row) using MAGE (ViT-L).

1. Additional Results

1.1. Qualitative Results

Image Inpainting and Outpainting. With the superior class-unconditional reconstruction and generation power shown in the main paper, MAGE naturally enables many common image synthesis applications. As shown in Figure 1, MAGE can reconstruct realistic and high-quality images for different image editing tasks such as inpainting (first row), outpainting (second row), and uncropping (outpainting on large masks, third row). We also include more qualitative results in Figure 5, Figure 6, and Figure 7, demonstrating MAGE's excellent ability in such image synthesis tasks. All results are generated using MAGE based on ViT-L trained with default augmentations, and the original images are all from the ImageNet eval set.

Class Unconditional Generation. We include more class unconditional generation results using default strong augmentation (random crop and resize (0.2 to 1) and random flipping) and weak augmentation (random crop and resize (0.8 to 1) and random flipping) in Figure 3 and Figure 4.

1.2. Quantitative Results

Class-Conditional Image Generation. Our model can also be used for class-conditional image generation as downstream task. To enable class-conditional generation, we take the ViT encoder from pre-training, and replace the original ViT decoder with a class-conditional decoder (12-layer ViT with embedding dimension 768, 113M parameters) which takes the class label as another input (concatenated to the padded features). During training, we freeze the encoder parameters to better evaluate the quality of the

Table 1. Quantitative comparison with state-of-the-art generative models on ImageNet 256x256 for class-conditional generation. Our method uses a MAGE pre-trained ViT-B as encoder and only trains a class-conditional decoder with 113M parameters.

Methods	FID↓	IS↑	#params
DCTransformer [15]	36.51	-	738M
VQVAE-2 [17]	31.11	∼45	13.5B
VQGAN [3]	18.65	80.4	227M
VQGAN [9]	15.78	78.3	1.4B
Improved DDPM [16]	12.26	-	280M
ADM [6]	10.94	101.0	554M
LDM [19]	10.56	103.5	400M
BigGAN-deep [2]	6.95	198.2	160M
MaskGIT [3]	6.18	182.1	227M
MAGE (ViT-B)	6.93	195.8	117M+113M

learned representations. Similar to pre-training, the model will take masked tokenized images as input and try to reconstruct the masked tokens. The only difference is that the decoder will not only see representations from the encoder, but also know the class label of the input image. Then during inference, the class label will be used to guide the model to generate images of the same class.

As shown in Table 1, MAGE achieves comparable performance as SOTA image generation methods on the task of class-conditional image generation on ImageNet-1K. Note that our encoder is inherited from the pre-training and *is not* fine-tuned during the downstream class-conditional training. Only the decoder is trained and has information about the class label. This shows that MAGE's encoder can learn high-quality representations that can achieve similar generative performance as models trained end-to-end on class conditional generative tasks.

Few-shot Transfer Learning. In the main paper, we provide transfer learning results of MAGE on 8 different datasets with 25 samples per class. Here we further show our method's performance with 1, 5, and 10 samples per class. As shown in Figure 2, our method is consistently better than MAE and SimCLR on most datasets with different numbers of samples per class, demonstrating the effectiveness of our method on few-shot transfer learning.

2. Ablation Studies

In this section, we conduct extensive ablation studies on our method. Without further notice, we use ViT-B trained with 800 epochs for all ablation studies.

MAE with GAN loss. One trivial solution to force the previous MIM method to generate realistic images is to add a GAN loss on top of the reconstructed image. However, we show that introducing GAN loss during previous MIM pre-training could largely decrease the performance of linear probing. As shown in Table 2, we evaluate the linear probing performance of a ViT-L MAE model pre-trained with an extra GAN loss released in MAE's official GitHub repo [10]. Although this model can reconstruct much more realistic images than the original MAE, the linear probing performance decreases by 11% compared with the ViT-L MAE model pre-trained without the GAN loss. On the other hand, our MAGE framework enables generative modeling and representation learning to help each other, achieving SOTA performances on both tasks using one single model.

Table 2. Top-1 accuracy of linear probing on ImageNet-1k with different method. MAE with GAN loss significantly reduces its performance on linear probing.

Methods	Linear Probing (%)
MAE (ViT-L) [11]	75.1
MAE (ViT-L) + norm pixel loss [11]	75.8
MAE + GAN loss (ViT-L) [10]	64.1
MAGE	**78.9**

Table 3. FID and top-1 accuracy of linear probing on ImageNet-1k by padding with `[C]` or a universal `[MASK]` token.

Padding Token	FID	Linear Probing (%)
`[MASK]`	12.4	72.5
`[C]`	11.6	73.3

Pad with [CLS] token. To pad the output of the encoder, unlike MAE which uses a learnable mask token that is shared for different inputs, we use the class token feature which is specific to each image. This design allows the decoder to take the global features extracted by the encoder as input. As shown in Table 3, this design can improve both class-unconditional generation performance and linear-probing results.

Table 4. FID and top-1 accuracy of linear probing of ViT-B trained 1600 epochs on ImageNet-1k using strong augmentations (s.a.) and weak augmentations (w.a.).

Augmentations	FID	Linear Probing (%)
MAGE + w.a.	**8.67**	70.5
MAGE + s.a.	11.1	**74.7**

Augmentations. As shown in many previous works on generative modeling and representation learning [4, 7, 11, 18], the augmentation used to train the model is important for both generation and representation learning performance. In our paper, we use two different sets of augmentations: default augmentations, or strong augmentations (s.a.), which consist of random crop and resize (0.2 to

1) and random flipping; weak augmentations (w.a.), which consist of random crop and resize (0.8 to 1) and random flipping. The only difference between s.a. and w.a. is the zoom-in scale of random crop and resize. As shown in Table 4, strong augmentations favor representation learning and weak augmentation favor generation quality, which is consistent with findings in prior works [7, 11].

Table 5. FID and top-1 accuracy of linear probing of ViT-B trained 400, 800, and 1600 epochs on ImageNet-1k.

#Pre-training Epochs	FID	Linear Probing (%)
400	12.2	72.2
800	11.6	73.3
1600	11.1	74.7

Pre-training Epochs. One important factor in self-supervised learning methods is the number of pre-training epochs. Prior works have shown that longer pre-training epochs can largely improve the performance of self-supervised methods [4, 11]. We compare MAGE's performance on ViT-B using 400, 800 and 1600 epochs of pre-training in Table 5. We observe that MAGE achieves good performances in both generation and representation learning with 400 epochs of pre-training, and can consistently benefit from longer training epochs.

Table 6. FID and top-1 accuracy of linear probing on ImageNet-1k using different decoder architecture. d denotes decoder depth (number of transformer blocs in the decoder), and w denotes decoder width (feature dimension in the decoder).

Decoder Arch.	FID	Linear Probing (%)
$d = 8, w = 512$	12.4	72.1
$d = 8, w = 768$	11.6	73.3
$d = 8, w = 1024$	11.4	73.5
$d = 6, w = 768$	12.4	71.8
$d = 8, w = 768$	11.6	73.3
$d = 10, w = 768$	11.4	73.2
$d = 12, w = 768$	11.3	73.4

Decoder Design. MAE [11] shows that a small ViT decoder is enough to achieve good performance. We also try different decoder architectures and summarize the results in Table 6. As shown in the table, the decoder with 8 blocks and 768 feature dimension reaches the best balance between computation cost and performance for ViT-B. Therefore, we choose the decoder architecture to be 8 blocks with 768 feature dimensions for ViT-B and 1024 feature dimension for ViT-L in the paper.

Complement MIM with Contrastive Loss. We show in the main paper that MAGE can be further combined with a simple contrastive loss (MAGE-C) to achieve better representation learning performance. In Table 7 we show more ablations regarding this contrastive loss. The performance of simply applying the contrastive loss without the reconstructive loss is worse than the SimCLR baseline. This is likely because we do not use augmentations such as color jittering and random grey scale, so applying only the contrastive loss could result in learning shortcut semantics such as color distribution [4, 18]. However, the reconstructive loss can prevent the network from falling into such shortcut solutions and help the network learn richer semantics. To choose λ which balances contrastive and reconstructive loss, we ablate different values of λ as shown Table 8.

Table 7. Top-1 accuracy of linear probing on ImageNet-1k using different methods. C denotes our contrastive loss, R denotes our reconstructive loss, † denotes our re-implementation.

Methods	Linear Probing (%)
MAE [11]	68.0
R only	73.3
SimCLR † [4]	74.2
C only	72.9
C+R (MAGE-C)	**77.1**

Table 8. Top-1 accuracy of linear probing with different λ.

λ	0	0.01	0.05	0.1	0.2	0.5
Linear Probing	73.3	75.5	76.7	**77.1**	76.9	76.6

Table 9. FID and top-1 accuracy of linear probing of MAGE-C on ImageNet-1k using different maximum masking ratio $\max(m_r)$.

	FID	Linear Probing (%)
$\max(m_r)$=1.0	14.1	75.0
$\max(m_r)$=0.7	23.5	76.3
$\max(m_r)$=0.6	27.0	77.1

We also notice one problem of applying contrastive training to MAGE: MAGE can see very high masking ratios during training, but applying positive loss to two augmented views of the same image both with a very high masking ratio is problematic. This is because such two views can share very little common information, leading to a performance drop as shown in [21]. Therefore, we only apply contrastive loss when the masking ratio is relatively low ($m_r < 0.6$). In Table 9, we show the performance of generation and representation learning w.r.t. the maximum masking ratio of our variable masking ratio distribution. Smaller $\max(m_r)$ leads to better linear probing but worse FID. We believe it is because, with smaller $\max(m_r)$, the contrastive loss can operate on more samples in the batch whose masking ratio

$m_r < 0.6$, which is important for contrastive learning as shown in [4]. On the other hand, small $\max(m_r)$ harms the generation performance because the network should see a relatively high masking ratio to enable generation from blank image (100% masking ratio). We leave a further investigation of this phenomenon and a better combination strategy for future work.

3. Implementation Details

Tokenizer and Detokenizer. We use a CNN-based VQGAN encoder and quantizer to tokenize the 256x256 input images to 16x16 discrete tokens. The detokenizer operates on the 16x16 discrete tokens and reconstructs the 256x256 image. The encoder consists of 5 blocks and each block consists of 2 residual blocks. After each block in the encoder, the feature is down-sampled by 2 using average pooling. The quantizer then quantizes each pixel of the encoder's output feature map using a codebook with 1024 entries, each entry with dimension 256. The detokenizer consists of another 5 blocks where each block consists of 2 residual blocks. After each block in the decoder, the feature map is up-sampled by 2. The tokenizer consists of 23.8M parameters and the detokenizer consists of 30.5M parameters. Our VQGAN tokenizer and detokenizer are trained on ImageNet-1K with 256 batch size. Please refer to our code and the original VQGAN paper for more details [9].

ViT architecture. After the tokenizer, the latent sequence length becomes 256 (plus one 'fake' class token). We then follow a similar encoder-decoder Transformer architecture similar to MAE [11]. More specifically, we use standard ViT architecture [8], which consists of a stack of Transformer blocks [22], where each block consists of a multi-head self-attention block and an MLP block. We use two learnable positional embeddings, one added to the input of the encoder and another added to the input of the decoder.

We use features from the encoder output for classification tasks, such as linear probing, few-shot transfer learning, and fine-tuning. We average pool the encoder output without the class token to get the input of the linear classifier.

Pre-training. Please refer to Table 10 for our default pre-training setting. We use only random crop and resize (0.2 to 1) and random horizontal flip as our default augmentations.

Generation. We use iterative decoding as in MaskGIT [3] to iteratively fill in masked tokens and generate images. To generate an image at inference time, we start from a blank canvas with all the tokens masked out, i.e., $Y_M^{(0)}$. For iteration $t = 1, \cdots, T$, the algorithm runs as follows:

1. **Predict.** Given $Y_M^{(t)}$ which is the unmasked tokens at the beginning of iteration t, we first predict the probability of the remaining masked tokens using our model, denoted as $p^{(t)} \in \mathbb{R}^{N_t \times K}$, where N_t is the number of remaining masked tokens and K is the number of entries in the code-

Table 10. **Pre-training Setting.**

config	value
optimizer	AdamW [14]
base learning rate	1.5e-4
weight decay	0.05
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.95$
batch size	4096 (B), 2048 (L)
learning rate schedule	cosine decay [13]
warmup epochs	40
training epochs	1600
gradient clip	3.0
label smoothing [20]	0.1
dropout	0.1
masking ratio min	0.5
masking ratio max	1.0 (MAGE) 0.6 (MAGE-C)
masking ratio mode	0.55
masking ratio std	0.25
MAGE-C only	
contrastive loss weight	0.1
temperature	0.2

Table 11. **Linear Probing Setting.**

config	value
optimizer	LARS [23]
base learning rate	0.1 (B) 0.05 (L)
weight decay	0
optimizer momentum	0.9
batch size	4096
learning rate schedule	cosine decay [13]
warmup epochs	10
training epochs	90
augmentation	RandomResizedCrop

Table 12. **End-to-end fine-tuning Setting.**

config	value
optimizer	AdamW [14]
base learning rate	2.5e-4
weight decay	0.05
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$
layer-wise lr decay [1]	0.65 (B) 0.75 (L)
batch size	1024
learning rate schedule	cosine decay [13]
warmup epochs	5
training epochs	100 (B) 50 (L)
label smoothing [20]	0.1
augmentation	RandAug (9, 0.5) [5]
mixup [25]	0.8
cutmix [24]	1.0
random erase	0
drop path [12]	0.1 (B) 0.2 (L)

Table 13. **Supervised training from scratch setting with ViT on semantic tokens.**

config	value
optimizer	AdamW [14]
base learning rate	1e-4
weight decay	0.3
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.95$
batch size	4096
learning rate schedule	cosine decay [13]
warmup epochs	20
training epochs	300 (B) 200 (L)
label smoothing [20]	0.1
augmentation	RandAug (9, 0.5) [5]
mixup [25]	0.8
cutmix [24]	1.0
drop path [12]	0.1 (B) 0.2 (L)
exp. moving average (EMA)	0.9999

book.

2. **Sample.** At each masked location i, we sample a token $y_i^{(t)}$ based on the prediction probability $p_i^{(t)} \in \mathbb{R}^K$ over all possible tokens in the codebook, and form the unmasked prediction $Y^{(t)}$. After $y_i^{(t)}$ is sampled, its corresponding prediction score plus a noise sampled from a random Gumbel distribution multiplied by temperature τ is used as the ”confidence” score indicating the model’s belief of the prediction at location i. The confidence scores at the unmasked locations are set to $+\infty$.

3. **Mask.** We then determine the number of tokens N_{t+1} for the next iteration $t+1$ based on a cosine masking schedule $N_{t+1} = N_0 \cdot \cos\left(\frac{\pi t}{2T}\right)$. We then sample N_{t+1} locations with the lowest confidence scores and mask those locations from $Y^{(t)}$ to generate $Y_M^{(t+1)}$.

For class unconditional generation, we choose $\tau = 6.0$ and $T = 20$ to generate images in our experiment.

Linear Probing & Fine-tuning. Our linear probing and fine-tuning setup follow MAE [11]. Please see Table 11 and Table 12 for detailed configurations.

Training from scratch. We also follows MAE [11] for our training-from-scratch baseline. Table 13 summarizes our configurations.

Code. For more implementation details, please refer to our code included in the supplementary material. We will also release our code and pre-trained model to the public.

References

[1] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

[2] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[3] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11315–11325, 2022.

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *icml*, pages 1597–1607. PMLR, 2020.

[5] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.

[6] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

[7] Jeff Donahue and Karen Simonyan. Large scale adversarial representation learning. *Advances in neural information processing systems*, 32, 2019.

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *Int. Conf. on Learning Representations (ICLR)*, 2021.

[9] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021.

[10] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. `https://github.com/facebookresearch/mae`, 2021.

[11] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16000–16009, June 2022.

[12] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer, 2016.

[13] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[14] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[15] Charlie Nash, Jacob Menick, Sander Dieleman, and Peter W Battaglia. Generating images with sparse representations. *arXiv preprint arXiv:2103.03841*, 2021.

[16] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[17] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

[18] Joshua Robinson, Li Sun, Ke Yu, Kayhan Batmanghelich, Stefanie Jegelka, and Suvrit Sra. Can contrastive learning avoid shortcut solutions? *Advances in neural information processing systems*, 34:4974–4986, 2021.

[19] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[20] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[21] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019.

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[23] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

[24] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.

[25] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

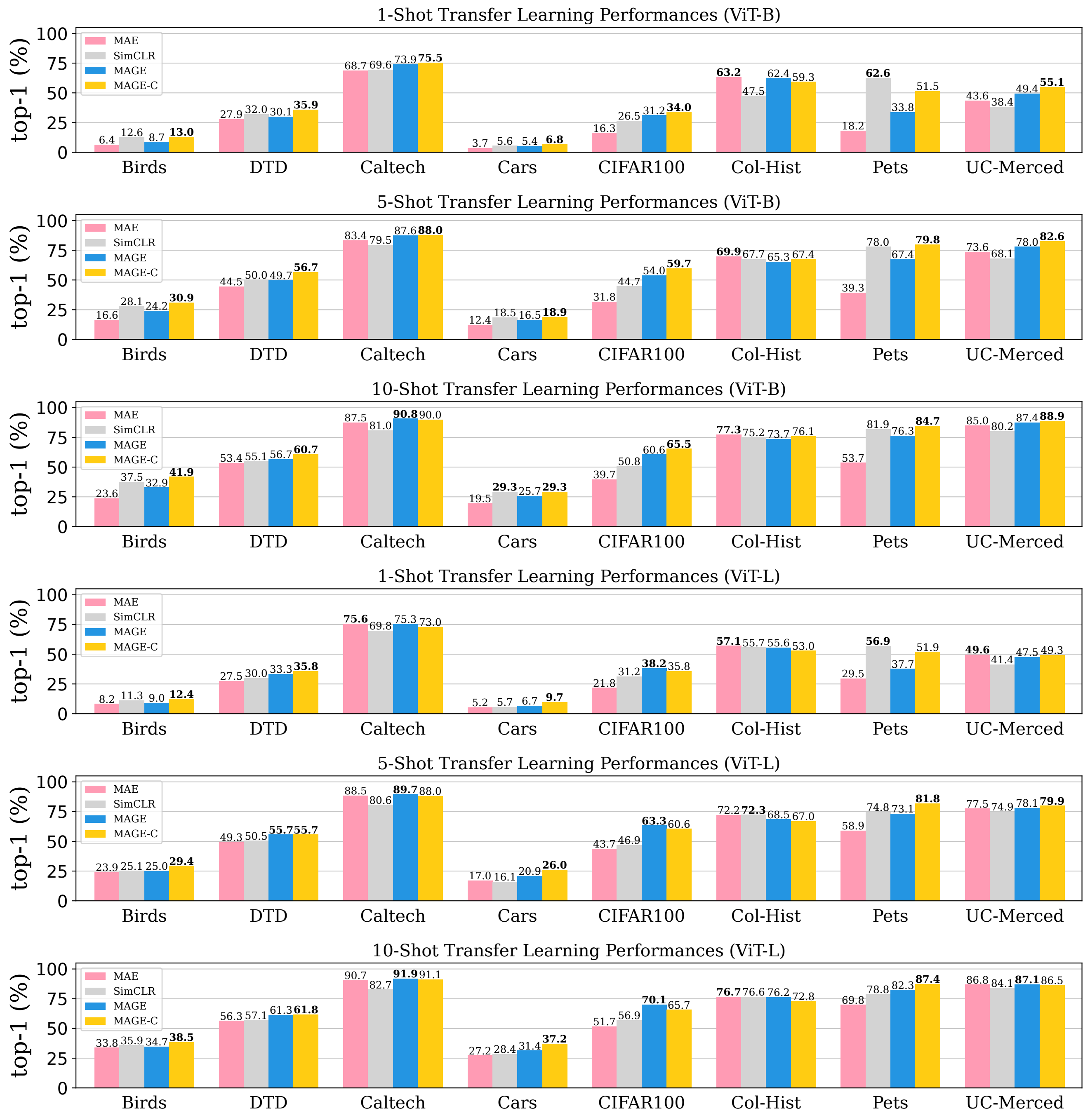

Figure 2. Transfer learning performance of ViT-B and ViT-L pre-trained on ImageNet-1K using different methods. Our methods outperform SimCLR [4] and MAE [11] on most datasets under different few-shot settings.

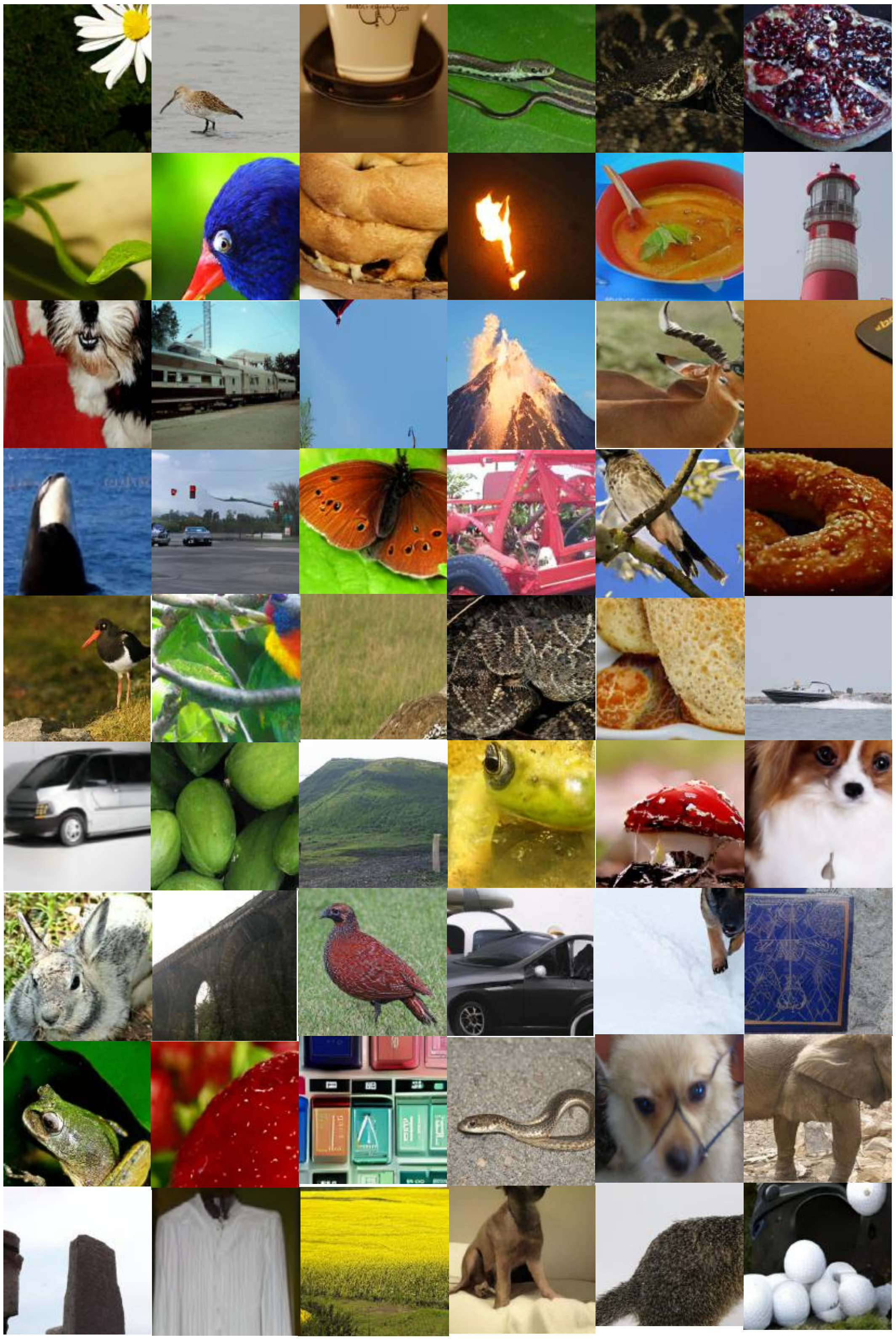

Figure 3. More uncurated examples of Class-unconditional image generation on ImageNet using MAGE trained with default strong augmentation.

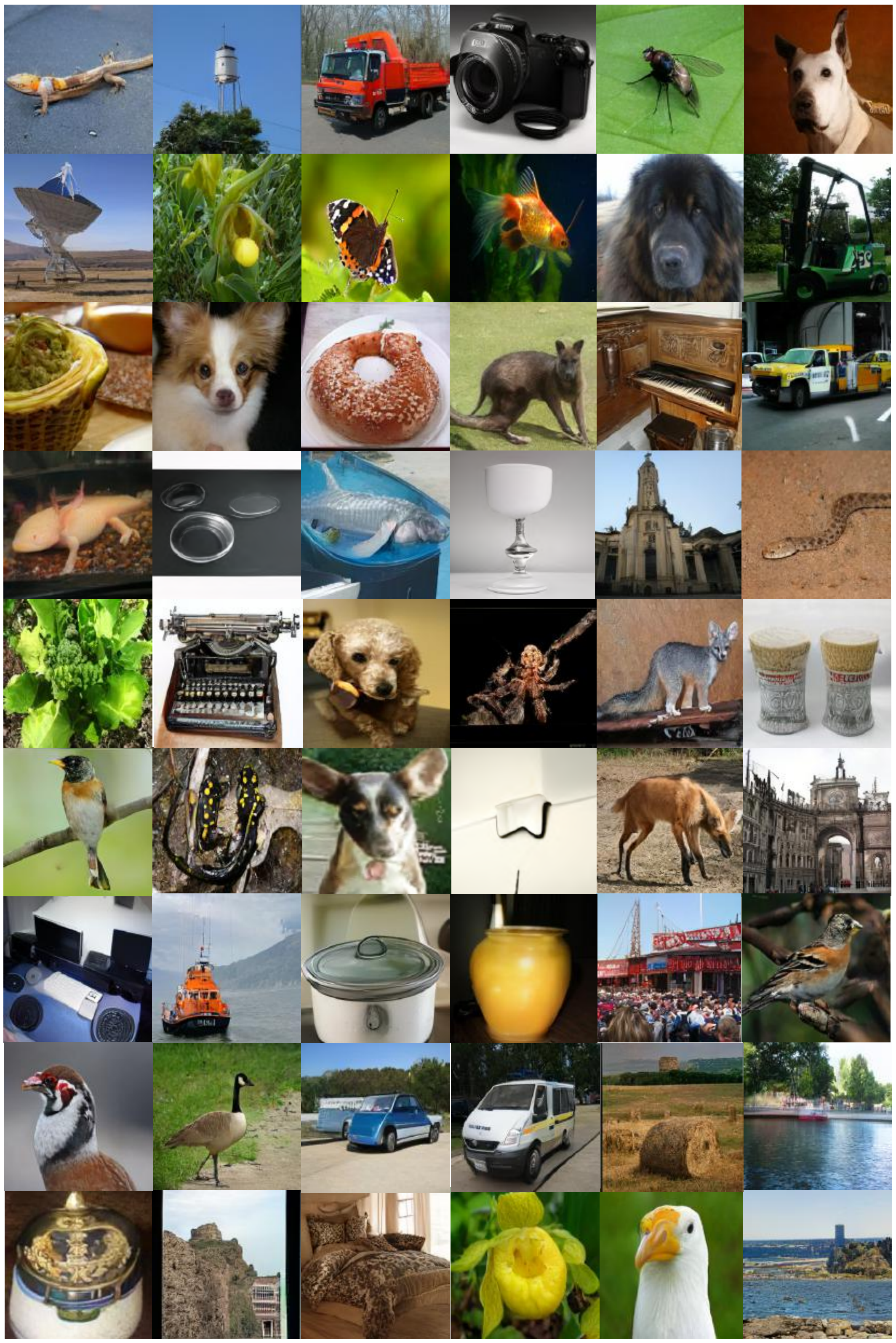

Figure 4. More uncurated examples of Class-unconditional image generation on ImageNet using MAGE trained with weak augmentation.

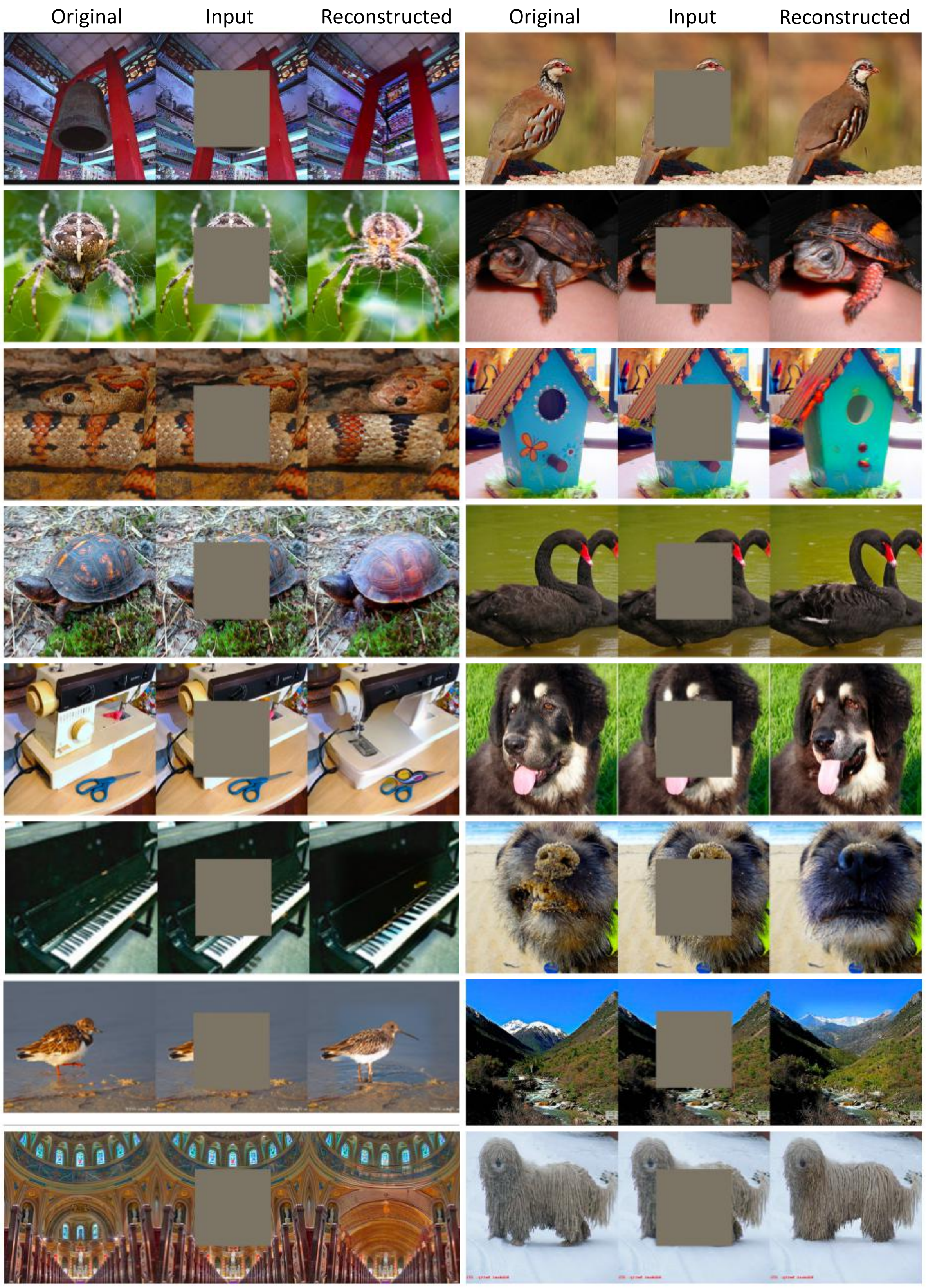

Figure 5. More examples of image inpainting using MAGE (ViT-L).

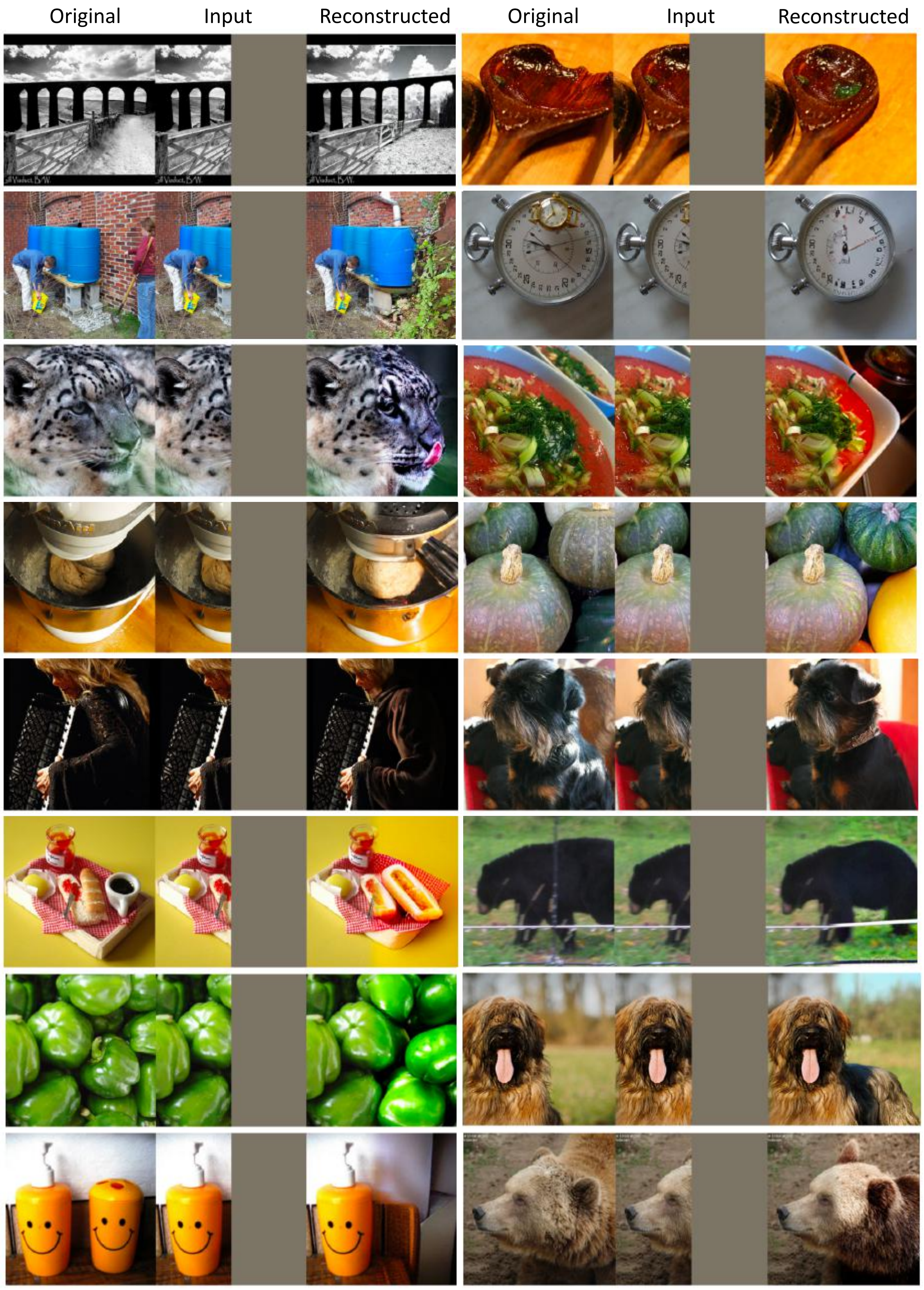

Figure 6. More examples of image outpainting using MAGE (ViT-L).

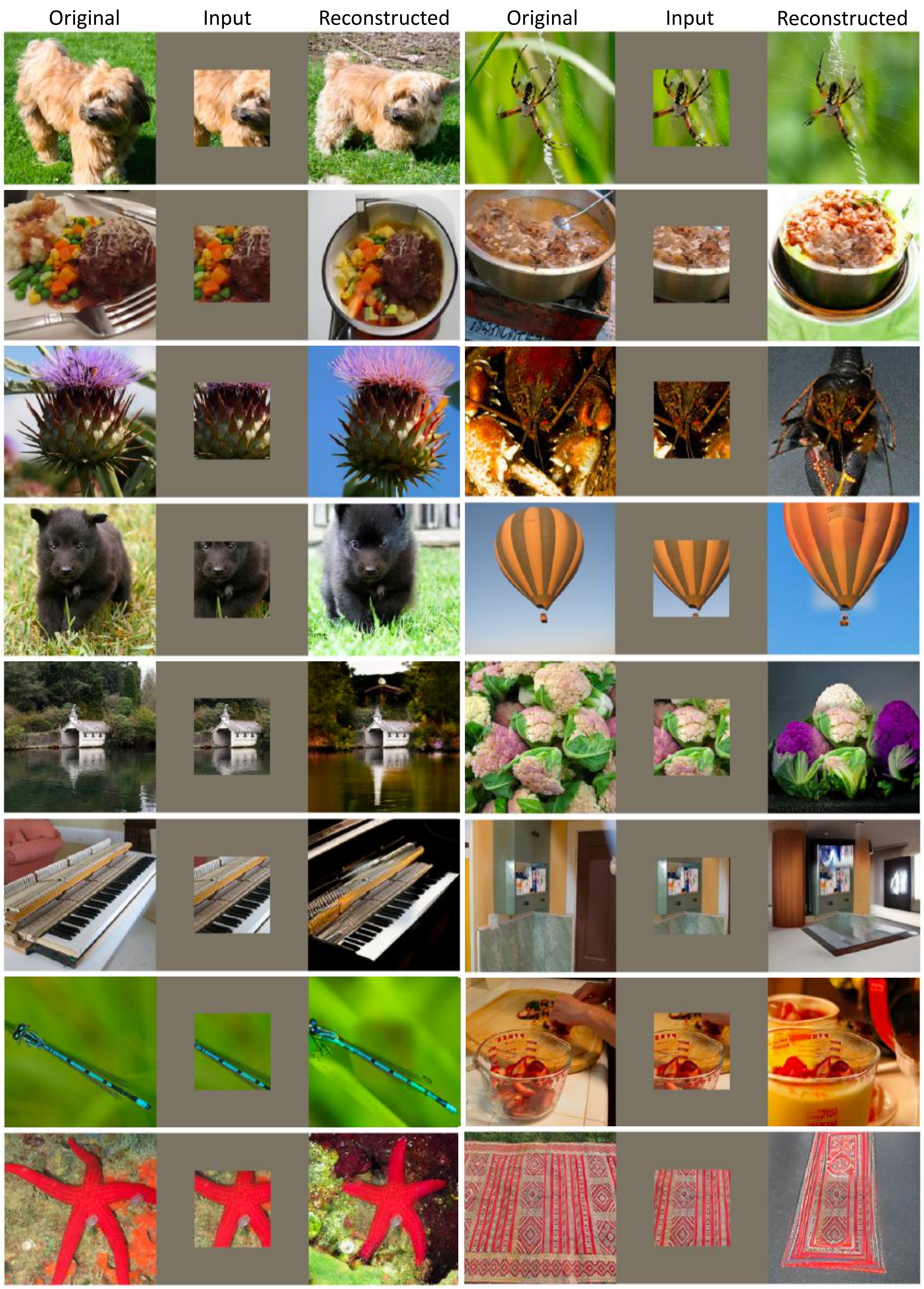

Figure 7. More examples of image outpainting on large outpainting mask (uncropping) using MAGE (ViT-L).