A. Details for AugVAE

A.1. Architecture

The AugVAE encoder and decoder are ResNet [14] with bottleneck-style Resblocks. Our AugVAE is specifically based on the encoder-decoder from official VQGAN [12] implementation available at https://github.com/CompVis/taming-transformers. From VQGAN implementation, we removed the attention block and applied the modification we describe in 3.3. The high-level architecture of our AugVAE is depicted in Figure 9. Before we start AugVAE-SL fine-tuning, we change the model architecture by removing 16 × 16 and 8 × 8 latent map from AugVAE-ML and replacing concatenation with 1 × 1 convolution for channel upsampling. Precise details for the architecture are given in files latent-verse/models/vqvae.py and latent-verse/modules/vqvae/vae.py of our source code available at https://github.com/tgisaturday/L-Verse.

A.2. Training

Our AugVAE is trained on ImageNet1K [8]. We resize each image into 256 × 256 × 3 and apply random crop with 0.75 crop ratio for training. We train both AugVAE-ML and AugVAE-SL using AdamW [26] optimizer with \( \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8} \), weight decay multiplier \( 1 \epsilon - 5 \), and the learning rate 4.5e − 6 multiplied by the batch size. We half the learning rate each time the training loss appeared to plateau. For the loss term, we use a combination of mean-squared-error (MSE) and LPIPS [52] losses between the input and the reconstructed image. For stable training, we multiply the LPIPS loss by 0.1.

B. Details for BiART

B.1. Architecture

Our BiART is similar to the GPT architecture [3]. We utilize the minGPT implementation of GPT architectures available at https://github.com/karpathy/minGPT. We only add segment embedding with dimension size 256 for [REF] and [GEN]. Each segment embedding is added to the positional encoding of an input token. We use a 32-layer decoder-only transformer with 1024 dimensional states and 16 masked self-attention heads. While BiART uses an integrated embedding matrix for image and text tokens, each token groups appear to plateau. For the loss term, we use a combination of mean-squared-error (MSE) and LPIPS [52] losses between the input and the reconstructed image. For stable training, we multiply the LPIPS loss by 0.1.

B.2. Training

BiART is trained on MS-COCO Captions [24] and Conceptual Captions [39]. We resize each image into 256 × 256 × 3 and apply random crop with 0.75 crop ratio for training. We apply BPE dropout [30] with a rate of 0.1 to our byte-pair encoder. We also apply residual, embedding, and attention dropouts [42] with a rate of 0.1. We train BiART using AdamW [26] optimizer with \( \beta_1 = 0.9, \beta_2 = 0.95, \epsilon = 10^{-8} \), weight decay multiplier \( 1 \epsilon - 2 \), and the learning rate 4.5e − 7 multiplied by the batch size. We don’t apply weight decay to embedding parameters. We half the learning rate each time the training loss appeared to plateau.

C. Examples for Image Reconstruction

We provide more examples of in-domain image reconstruction in Figure 11 and out-of-domain in Figure 12. We also provide the reconstruction FID of AugVAE-SL on various datasets in Table 4 as a reference for future works. AugVAE-SL trained on ImageNet1K shows “≤ 8” FID for all data domain without extra finetuning. The resolution of each reconstructed image is 256 × 256.

D. Examples for Image-to-Text Generation

We provide an example task interface of our human evaluation mentioned in Section 4.2 in Figure 10. We also provide more examples of image-to-text generation on MS-COCO Captions in Figure 13. All examples in Figure 13 received “Both captions well describe the image” in our human evaluation. The resolution of each input image is 256 × 256.

E. Examples for Text-to-Image Generation

We provide examples of zero-shot text-to-image generation with L-Verse-CC in Figure 14. Captions are randomly sampled from MS-COCO Captions 2017 validation set. The resolution of each generated image is 256 × 256.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>FID</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFHQ [19]</td>
<td>4.92</td>
</tr>
<tr>
<td>AFHQ [5]</td>
<td>4.36</td>
</tr>
<tr>
<td>MS-COCO [24]</td>
<td>4.77</td>
</tr>
<tr>
<td>OpenImages V6 [21]</td>
<td>3.15</td>
</tr>
</tbody>
</table>

Table 4. Reconstruction Fréchet Inception Distance (FID) of AugVAE on various datasets. For all settings, we use ImageNet1K trained AugVAE-SL without any fine-tuning on each dataset. Images are resized to 256 × 256 with LANCZOS [6] filter.
F. Discussion

Bidirectional Learning  L-Verse internally learns a reversible and densely connected mapping between images and texts. From this, L-Verse can generate a text or an image in accordance with the given condition without any fine-tuning or extra object detection framework. Bidirectional learning not only saves time and computational cost for training and application. As we mentioned in Section 3.5, our bidirectional approach also mitigates the heterogeneity of data and enables stable FP16 (O2) mixed-precision training.

Efficiency  The bidirectional training enables L-Verse to efficiently learn the vision-language cross-modal representation with smaller dataset and model size. L-Verse requires 97.6% less data (compared to OSCAR [23]) for image-to-text and 98.8% less data (compared to DALL-E [32]) for text-to-image generation to achieve comparable performances. L-Verse also has 95% less parameters compared to DALL-E, which makes L-Verse more suitable to the environment with limited computing resources.

Vision-Language Pre-Training  Vision-Language (VL) pre-training from OSCAR surely brings positive effects in learning the cross-modal representation. This also follows the current trend of large scale model training: pre-training with a large data set on a general task and fine-tuning with smaller set to solve downstream tasks. Since we mainly focus on the efficiency over the amount of training data and computing resources, VL pre-training is out-of-scope of this work. However, we also believe that combining VL pre-training with bidirectional training will further improve the performance of L-Verse.

Large Scale Training  With limited amount of training data and computational resources, we couldn’t consider training L-Verse in larger scale like OSCAR, DALL-E or CogView [10]. Nevertheless, our bidirectionally trained L-Verse shows competitive results to other large scale models. As 400M well-filtered text-image dataset [37] has been released recently, we are optimistic about training L-Verse in larger scales.

Zero-Shot Image Captioning  L-Verse also has an ability to perform zero-shot image captioning when trained on Conceptual Captions (CC) [39]. Unlike MS-COCO Captions [24] which is carefully annotated by humans, images and their raw descriptions in CC are harvested from the web. While texts in CC represent a wider variety of styles, its diversity also adds noise to the caption that L-Verse generates. For this reason, we mainly use L-Verse trained with MS-COCO for the experiment on image captioning.

Potential Negative Impact  Our findings show excellent performance in both image-to-text and text-to-image generation. L-Verse has a wide range of beneficial applications for society, including image captioning, visual question answering, and text visualization. However, there are still potential malicious or unintended uses of L-Verse including image-to-text or text-to-image generation with social bias. To prevent potential negative impact to our society, we provide open access only to AugVAEs for now.
Figure 9. Proposed AugVAE. Trained with cross-level feature augmentation, AugVAE-ML is finetuned into AugVAE-SL to reduce the length of encoded image sequence. We remove unnecessary encoders and decoders from AugVAE-ML and replace the concatenation operation with a $1 \times 1$ convolution which expands the last dimension of the input tensor by two.

Figure 10. Example interface for human evaluation. Random sampled 30 examples are shown to each participant.
Figure 11. More examples of input images (top) and reconstructions from AugVAE-SL (bottom) on Imagenet1K validation set. The resolution of each image is $256 \times 256$ pixels.
Figure 12. More examples of input images (*top*) and reconstructions from AugVAE-SL (*bottom*) with unseen image domains (256 × 256 pixels).
Figure 13. More examples of image-to-text generation on MS-COCO with corresponding ground-truths.
Figure 14. Examples of text-to-image generation on MS-COCO.