ASSET: Autoregressive Semantic Scene Editing with Transformers at High Resolutions

DIFAN LIU, UMass Amherst and Adobe Research, USA
SANDESH SHETTY, UMass Amherst, USA
TOBIAS HINZ, Adobe Research, USA
MATTHEW FISHER, Adobe Research, USA
RICHARD ZHANG, Adobe Research, USA
TAESUNG PARK, Adobe Research, USA
EVANGELOS KALOGERAKIS, UMass Amherst, USA

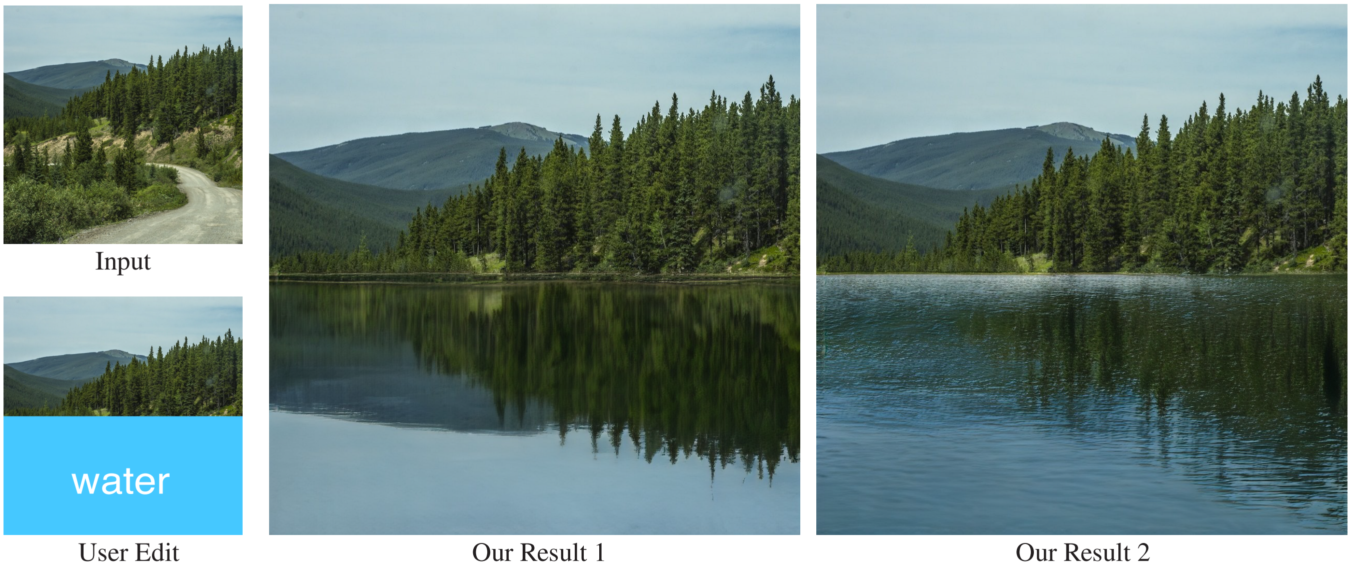


Fig. 1. ASSET allows users to create diverse editing results by specifying a region and a new label on the input image. Our efficient transformer captures long-range dependencies in the image, such as the detailed reflection of the trees on the water, even at high resolutions (1024 × 1024 pixels in this example).

We present ASSET, a neural architecture for automatically modifying an input high-resolution image according to a user's edits on its semantic segmentation map. Our architecture is based on a transformer with a novel attention mechanism. Our key idea is to sparsify the transformer's attention matrix at high resolutions, guided by dense attention extracted at lower image resolutions. While previous attention mechanisms are computationally too expensive for handling high-resolution images or are overly constrained within specific image regions hampering long-range interactions, our novel attention mechanism is both computationally efficient and effective. Our sparsified attention mechanism is able to capture long-range interactions and context, leading to synthesizing interesting phenomena in scenes, such as reflections of landscapes onto water or flora consistent with the rest of the landscape, that were not possible to generate reliably with previous convnets and transformer approaches. We present qualitative and quantitative results, along with user studies, demonstrating the effectiveness of our method. Our code and dataset are available at our project page: https://github.com/DifanLiu/ASSET

CCS Concepts: • **Computing methodologies** → **Image processing**; *Neural networks*; Image representations.

Additional Key Words and Phrases: image editing, neural networks, transformers, sparse attention

ACM Reference Format:
Difan Liu, Sandesh Shetty, Tobias Hinz, Matthew Fisher, Richard Zhang, Taesung Park, and Evangelos Kalogerakis. 2022. ASSET: Autoregressive Semantic Scene Editing with Transformers at High Resolutions. *ACM Trans. Graph.* 41, 4, Article 74 (July 2022), 14 pages. https://doi.org/10.1145/3528223.3530172

Authors' addresses: Difan Liu, dliu@cs.umass.edu, UMass Amherst and Adobe Research, USA; Sandesh Shetty, sandeshshett@umass.edu, UMass Amherst, USA; Tobias Hinz, thinz@adobe.com, Adobe Research, USA; Matthew Fisher, techmatt@gmail.com, Adobe Research, USA; Richard Zhang, rizhang@adobe.com, Adobe Research, USA; Taesung Park, taesung_park@berkeley.edu, Adobe Research, USA; Evangelos Kalogerakis, kalo@cs.umass.edu, UMass Amherst, USA.

1 INTRODUCTION

Semantic image editing allows users to easily edit a given image by modifying a corresponding segmentation map. Ideally, one could change the outline of existing regions, or even freely add or remove regions. However, to obtain realistic and consistent results, an effective system needs to consider global context, from across the full image. For example, consider the example in Figure 1. To properly hallucinate a reflection in the water on the bottom of the image, the model should consider the content from the very top. Traditional CNN based approaches [Chen and Koltun 2017; Isola et al. 2017; Ntavelis et al. 2020; Park et al. 2019; Zhu et al. 2020] rely entirely on convolutional layers which have difficulty modeling such long-range dependencies [Wang et al. 2018].

Transformers are well equipped to handle these long-range dependencies through their attention mechanism allowing them to focus on distant image areas at each sampling step. However, the heavy computational cost for using attention, which usually increases quadratically with the input size, makes it infeasible to use standard transformers for high-resolution image editing. One way to address this is to use a sliding-window approach [Esser et al. 2021a], in which the transformer only attends to a small area around the currently sampled token, thereby reducing the computational cost to a fixed budget. While this approach enables the synthesis of high-resolution images, it forgoes the benefit of modeling long-range dependencies. This leads to inconsistencies when edits are dependent on image regions that are far away in pixel space. Is it possible to retain the ability to model long-range dependencies, while not paying prohibitive computational costs at high resolution?

We introduce a novel attention mechanism, called *Sparsified Guided Attention* (SGA), to facilitate long-range image consistency at high resolutions. While the sliding window approach is limited to local contexts, SGA can attend to far contexts that are relevant for the current sampling location. The core idea is to efficiently determine a small list of relevant locations that are worth attending to, and compute the attention map only over these locations. To achieve this, we use a guiding transformer that operates at the downsampled version of the input image and performs the same edit, but enjoys the full self-attention map thanks to the reduced input size. Based on the guiding transformer's attention map, we rank the importance of different areas of the image, and have the high resolution transformer attend only to the top-K most important image regions. In practice, our SGA leads to a large reduction in computational cost due to the obtained sparse attention matrix. Compared to other approaches, we obtain more realistic and consistent edits while still achieving high diversity in our outputs.

Our model takes as input a quantized representation of the image and its edited segmentation map, both obtained through a modified VQGAN encoder [Esser et al. 2021a]. We then mask out all image tokens in the image representation corresponding to the edited area and replace those tokens with a specific `[MASK]` token. Our transformer then samples new image tokens at the edited areas, conditioned on the original (masked) image and the edited segmentation map. Finally, a VQGAN decoder is used to decode the image tokens into the final RGB image. As the edited tokens are sampled autoregressively based on a likelihood-based model, we can sample a diverse set of image outputs, all of which are consistent with the overall image characteristics.

Contributions. We propose a transformer-based model that outputs realistic and diverse edits specified through modified segmentation maps. We introduce *Sparsified Guided Attention* (SGA), which allows the transformer to only attend to the most important image locations, leading to sparse attention matrices and reduced computational cost. Our model achieves diverse, realistic, and consistent image edits even at 1024 × 1024 resolution.

2 RELATED WORK

CNN-based image editing. CNN-based methods have achieved impressive results by enabling users to move bounding boxes containing objects [Hinz et al. 2019; Hong et al. 2018], modifying scene representations [Dhamo et al. 2020; Su et al. 2021], or following textual instructions [Cheng et al. 2020; Nam et al. 2018; Patashnik et al. 2021]. Other approaches enable user guides with edges or color information [Jo and Park 2019; Liu et al. 2021c] or perform simple inpainting, typically without user guidance [Liu et al. 2018, 2021b; Suvorov et al. 2022; Yang et al. 2017; Yu et al. 2018, 2019]. Exemplar-based image translation methods [Zhang et al. 2020; Zheng et al. 2021; Zhou et al. 2021] can synthesize images from semantic maps, but they cannot hallucinate new content that does not exist in the exemplar image. Other approaches fine-tune or train a generator for a specific image to perform editing on that single image [Bau et al. 2019; Hinz et al. 2021; Shaham et al. 2019; Vinker et al. 2021]. However, these models need to be adapted for each new image. More similarly to us, other methods allow for direct editing via segmentation maps [Gu et al. 2019; Lee et al. 2020; Ling et al. 2021; Ntavelis et al. 2020]. However, these approaches can only generate a single output for a given edit. In addition, previous CNN-based approaches prioritize local interactions between image pixels for image synthesis due to their inductive bias. They also fail to effectively capture long-range interactions between image regions necessary for realistic image synthesis. Our approach is based on a transformer that is able to effectively capture such interactions and also allows the synthesis of diverse results for each edit.

Transformers for image synthesis. To apply transformers for image synthesis, they are trained on discrete sequences of image elements. Some models first learn a discrete image region representation [Esser et al. 2021a; Ramesh et al. 2021], whereas other approaches work directly on pixels [Chen et al. 2020; Child et al. 2019; Jiang et al. 2021; Parmar et al. 2018]. However, most of them model images in a row-major format, and thus cannot capture bidirectional context, leading to inconsistent editing results. PixelTransformer [Tulsiani and Gupta 2021], iLAT [Cao et al. 2021], and ImageBART [Esser et al. 2021b] add bidirectional context to their transformer models but do not support editing via segmentation maps. More importantly, due to their quadratic complexity in the number of image tokens, these methods are trained on small image resolutions of 256 × 256 pixels. Alternatively, some approaches model the image directly at a low resolution (e.g., 32 × 32) and then use a deterministic upsampling network [Wan et al. 2021; Yu et al. 2021]. In this case, fine-grained edits are difficult to achieve due to the small resolution at which the images are modeled. For high-resolution image synthesis, [Esser et al. 2021b,a] proposed transformers with attention constrained on sliding windows. However, this hampers long-range interactions and, as a result, they often generate artifacts and inconsistent image edits. In contrast, our work incorporates a novel sparsified attention mechanism that can capture such interactions without compromising the synthesized image plausibility.

Efficient transformers. Much work has been invested in reducing the computational cost of the transformer's attention mechanism [Tay et al. 2020]. Broadly speaking, there are two ways to achieve this. One way is to reduce the computational cost of the attention mechanism directly, e.g., by approximating full attention through mechanisms where the computation cost grows linearly with the input length [Kitaev et al. 2020; Wang et al. 2020]. Alternatively, several works explore reducing the cost by replacing full attention with a sparse variant [Beltagy et al. 2020; Zaheer et al. 2020]. A few recent vision transformers reduce the computational complexity by spatially reducing attention [Liu et al. 2021a; Wang et al. 2021; Yang et al. 2021; Zhang et al. 2021]. However, these methods use encoder-only transformers for feature extraction and do not support autoregressive image generation. Our method is inspired by BigBird's sparse attention mechanism for long-document NLP tasks [Zaheer et al. 2020]. BigBird achieves high efficiency by using a random sparse attention map over blocks of tokens. However, when applying the random attention mechanism of BigBird to our task it fails to capture correct context for a given edit. Instead of randomly choosing tokens, our approach picks the most relevant tokens for attention at each spatial location.

Image synthesis with a guidance image. Synthesizing high resolution outputs is challenging in terms of both quality and computational cost. The idea of utilizing a high-resolution guide to upsample a low-resolution output has been explored in computer graphics [Chen et al. 2016; Kopf et al. 2007]. In particular, constructing a high-resolution depth map from coarse sensor data, guided by an RGB image has been extensively investigated [Ferstl et al. 2013; Liu et al. 2013; Park et al. 2011; Yang et al. 2014, 2007]. More recently, learning-based approaches were developed for similar tasks, by posing it as an image-to-image translation problem [Lutio et al. 2019], fusing the guidance and low-res information at multiple scales [Hui et al. 2016], or transferring the mapping learned at low resolution to high resolution [Shocher et al. 2018]. While these works primarily aim at leveraging high-resolution information in the input as a guide, our application must synthesize information from a flat input. In fact, our guide is a low-resolution version of the same image. Relatedly, [Shaham et al. 2021] use a low-resolution network to predict parameters of a lightweight high-resolution network, for the purpose of fast image translation, using a convolutional network architecture.

3 METHOD

Overview. Our method synthesizes images guided by user input in the form of an edited label map ("semantic map") of an input image. More specifically, given an RGB image and its corresponding label map, the user paints some desired changes on the label map, e.g., replace mountain regions with water (Figure 2). Since there exist several possible output images reflecting the input edits, our method generates a diverse set of outputs allowing the user to select the most preferable one. Moreover, our method generates high-resolution images of up to 1024 × 1024 resolution.

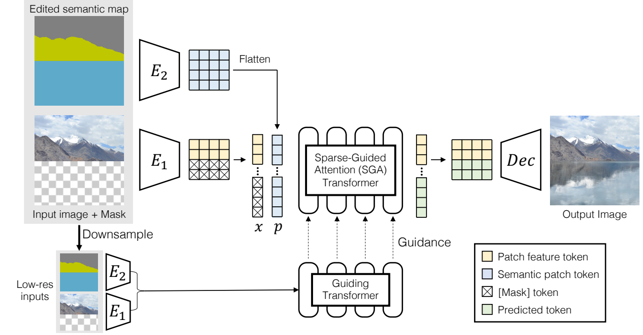


Fig. 2. Overview of our semantic image editing model. Our transformer model operates in the codebook space using the encoder E_1. To incorporate user edits the tokens inside the edited region are masked and are augmented with a semantic encoder E_2. The mask tokens are filled in by our SGA-Transformer Encoder-Decoder network, whose sparse attention mechanism is guided by the Guiding Transformer that computes the full attention on downsampled inputs. Finally, the generated tokens are decoded into the output image via Dec.

Our architecture is shown in Figure 2. Inspired by recent approaches [Esser et al. 2021a] we represent images and label maps as a spatial collection of quantized codebook entries (Section 3.1). These codebook entries are processed by a transformer model which aims to update the codebook entries of the edited areas in an autoregressive manner (Section 3.2). All codebook entries are subsequently decoded to the output set of images (Section 3.3). A crucial component of the transformer is its attention mechanism, which enables long-range interaction between different parts of the image such that the synthesized output is coherent as a whole. E.g., if a lake is generated by the semantic edits it must also capture any reflections of landscape (Figure 1). One complication is that the quadratic complexity of the traditional attention mechanism leads to a large time and memory cost for high-resolution images. The key idea of our method is to compute the full attention at lower resolution first, and then use that as guidance for sparsifying the attention at full resolution (Section 3.2). This approach allows us to model long-range dependencies even at high resolutions, resulting in more coherent and plausible output images compared to existing approaches that constrain attention within sliding windows [Esser et al. 2021a] or alternative attention models [Zaheer et al. 2020].

3.1 Image encoder

The input RGB image $\mathbf{X}$ of size $H_{\text{im}} \times W_{\text{im}} \times 3$ is processed by a convolutional encoder resulting in a feature map $\mathbf{F}$ of size $\frac{H_{\text{im}}}{16} \times \frac{W_{\text{im}}}{16} \times d$. We also create a $H_{\text{im}} \times W_{\text{im}}$ binary mask indicating image regions that must be replaced according to the semantic map edits. Masked image regions should not affect features produced for unmasked regions, e.g., information about the edited area of Figure 2 should

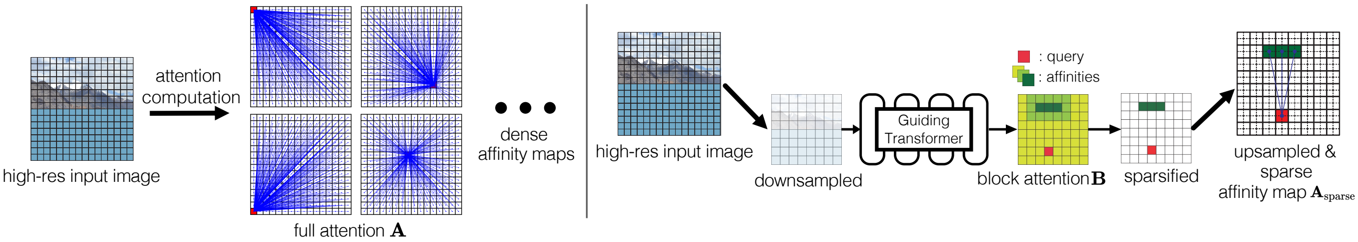


Fig. 3. Details of our Sparsified Guided Attention (SGA, right) compared to full attention (left). We downsample the input and the user-edited semantic layout and use a Guiding Transformer to obtain the full attention map, which identifies the most important attention locations for each sampling step. By keeping only the locations with high attention weight, we construct the high-resolution, sparse attention map for the SGA transformer.

not "leak" into the feature map of the unmasked area. To avoid information leakage, we employ partial convolutions [Liu et al. 2018] and region normalization [Yu et al. 2020] in our encoder while processing the unmasked regions. The feature map $\mathbf{F}$ is subsequently quantized following VQGAN [Esser et al. 2021a] with the help of a learned codebook $\mathcal{Z}$, i.e., each feature map entry $\mathbf{f}_{i,j}$ at position (i, j) in $\mathbf{F}$ is mapped to the closest codebook entry $\hat{\mathbf{f}}_{i,j} = \arg\min_{\mathbf{z}_\kappa \in \mathcal{Z}} ||\mathbf{f}_{i,j} - \mathbf{z}_\kappa||$, where $\{\mathbf{z}_\kappa\}_{\kappa=1}^{|\mathcal{Z}|}$ are codebook entries with dimensionality d. The codebook indices of the edited regions, as indicated by the binary mask, are replaced with a special [MASK] token (Figure 2). We use a second encoder with regular convolutions to obtain a feature representation of the edited semantic map $\mathbf{P}$ which is subsequently quantized in the same way as the RGB image, resulting in codebook entries $\hat{\mathbf{g}}_{i,j}$.

3.2 Autoregressive transformer

Our transformer follows a sequence-to-sequence architecture inspired by [Vaswani et al. 2017] which consists of a bidirectional encoder and an autoregressive decoder, both of which are equipped with our novel sparsified attention mechanism. The transformer encoder captures bi-directional context of the image, which is used by the transformer decoder to generate new codebook indices autoregressively.

Traditional Dense Attention. Traditional attention transforms each embedding linearly into a learned query, key, and value representation $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ of size $L \times d$, where $L = H_{\text{feat}} W_{\text{feat}} = \frac{H_{\text{im}}}{16} \frac{W_{\text{im}}}{16}$ is the length of the flattened codebook indices in our case [Vaswani et al. 2017]. The output embedding is then computed as $\text{softmax}(\mathbf{A}/\sqrt{d})\mathbf{V}$, where attention $\mathbf{A} = \mathbf{Q}\mathbf{K}^T \in \mathbb{R}^{L \times L}$. The advantage of attention is that it allows for interactions across all positions in the sequence, i.e., in our case the whole encoded image, as illustrated in Figure 3 *(left)*. The disadvantage is computation of the attention matrix $\mathbf{A}$ has quadratic time and memory complexity in terms of sequence length: $O(L^2) = O(H_{\text{feat}}^2 W_{\text{feat}}^2)$. For an input image with resolution 1024 × 1024, the sequence has length $L = 4096$, and practically the cost of performing the above matrix multiplication becomes prohibitively high, as discussed by several other works [Tay et al. 2020]. One simple way to reduce the computational cost is to use a sliding window approach [Esser et al. 2021a], i.e., crop a fixed sized patch around the currently sampled token and feed it into the transformer. However, this introduces a bias towards preferring interactions only within local regions of the image, missing other useful longer-range interactions.

Sparsified Guided Attention (SGA). We propose an efficient sparsification strategy, without sliding windows. The key idea is to first compute coarse full attention with downsampled images, determine which locations are worth attending to, and then use it to avoid computing most entries of the attention matrix $\mathbf{A}$.

To do this, we first proceed by downsampling the original input image and semantic map to 256 × 256 resolution. We further encode them to obtain a feature map of size 16 × 16. At this resolution, computing full attention is fast, because the sequence length of codebook indices is only $16^2 = 256$. Then we employ a *guiding transformer*, which has the same architecture as the main transformer but is trained at the low resolution, to calculate the dense attention matrix $\mathbf{A}_{\text{low}} \in \mathbb{R}^{256 \times 256}$.

Then we leverage $\mathbf{A}_{\text{low}}$ to construct a block attention matrix $\mathbf{B} \in \mathbb{R}^{L \times L}$ at the original feature resolution that will guide the sparsification. To do this, we divide the original feature map into non-overlapping blocks, as illustrated in Figure 3 *(right)* for an 8 × 8 grid of blocks. For each block, we find the corresponding locations in $\mathbf{A}_{\text{low}}$ and average their affinity values. Note that the matrix $\mathbf{B}$ essentially consists of blocks, each of which is populated with a single affinity value.

Then we construct the sparse attention matrix $\mathbf{A}_{\text{sparse}}$ by considering only the attention weights that are likely important in approximating the true attention. To this end, we keep the attention if the corresponding affinity is high in the block attention matrix $\mathbf{B}$. In other words, we argue that the selection of sparse attention pairs can be reliably guided by the dense attention evaluated at lower resolution. In addition, following the proposition in the context of NLP models [Zaheer et al. 2020], we always compute attention within the current and adjacent blocks, no matter their affinity values.

$$\mathbf{A}_{\text{sparse}}(r,t) = \begin{cases} \mathbf{A}(r,t), & \text{if } t \in \mathcal{N}(r) \text{ or } t \in \mathcal{K}(r). \\ -\infty, & \text{otherwise,} \end{cases} \quad (1)$$

where $\mathcal{N}(r)$ contains the entries of the neighborhood blocks of r, and $\mathcal{K}(r)$ contains the entries of the blocks with the top-K highest affinities outside the neighborhood.

In our experiments, we set $K = 3$, resulting in the sparsity ratio <10%, and significantly reduce the computational cost of attention.

Transformer encoder. The input to our transformer encoder is a sequence of embeddings jointly representing the masked image codebook indices $\mathbf{x} = \{x_l\}_{l=1}^{L}$ and semantic codebook indices $\mathbf{p} = \{p_l\}_{l=1}^{L}$ produced by the image encoders (flattened using row-major format), and position information of each index in the corresponding sequence. Note that here, the transformers are operating both at full-resolution and low-resolution. Specifically, for each position in the sequence, three d-dimensional learned embeddings are produced: (i) an image embedding $E_{\text{im}}(x_l)$ representing the token x_l at position l in our sequence and in turn the corresponding RGB image region, (ii) an embedding $E_{\text{map}}(p_l)$ of the semantic token p_l at the same position, and finally (iii) a positional embedding $E_{\text{pos}}(l)$ for that position l. The summation of token embeddings and positional embeddings follows other popular transformers [Dosovitskiy et al. 2021; Vaswani et al. 2017]:

$$\mathbf{e}_l = E_{\text{im}}(x_l) + E_{\text{map}}(p_l) + E_{\text{pos}}(l) \tag{2}$$

The sequence of embeddings $\{\mathbf{e}_l\}_{l=1}^{L}$ is fed to the first transformer encoder layer and is transformed to a continuous representation by the stack of transformer encoder layers. To preserve the position information of each index at subsequent layers, the position encoding generator (PEG) is placed before each encoder layer [Chu et al. 2021a,b] (more details in the appendix).

Transformer decoder. The decoder predicts codebook indices for the edited region with the help of the global context obtained through the transformer encoder. Similar to BART [Lewis et al. 2020], the autoregressive generation starts by pre-pending a special index (token) `[START]` to the decoder input. At each step, the decoder predicts a distribution over codebook indices from our dictionary $\mathcal{Z}$ learned in our image encoder (Section 3.1). Specifically, the decoder predicts $p(\mathcal{X}_l|\{\chi_{<l}\})$, where $\mathcal{X}_l$ is a categorical random variable representing a codebook index to be generated at position l in the sequence and $\{\chi_{<l}\}$ are all indices of the previous steps. We note that the tokens corresponding to unmasked image regions (i.e., image regions to be preserved) are set to the original image codebook indices. We predict the distributions only for positions corresponding to the edited image regions.

To predict the output distribution at each step, the decoder first takes as input a learned embedding $D_{\text{im}}(x_l)$ representing the input token x_l, and a learned positional embedding $D_{\text{pos}}(l)$ for that position l. It sums the two embeddings $\mathbf{d}_l = D_{\text{im}}(x_l) + D_{\text{pos}}(l)$, then passes $\mathbf{d}_l$ into a self-attention layer (attention between generated tokens) and a cross-attention layer (attention between generated tokens and encoder output features). For both self-attention and cross-attention we make use of the sparsified guided attention mechanism. We also note that the self-attention in the decoder layer is modified to prevent tokens from attending to subsequent positions. For more details about the architecture of our decoder, we refer to the appendix. Based on the predicted distribution of codebook indices, we use top-k sampling [Esser et al. 2021a; Holtzman et al. 2019] ($k = 100$ in our experiments) to create multiple candidate output sequences, each of which can be mapped to a new image by the image decoder. The generated images are ordered by the joint probability of the distributions predicted by the decoder.

3.3 Image decoder

The image decoder takes as input the quantized feature map and decodes an RGB image of size $H_{\text{im}} \times W_{\text{im}} \times 3$ following VQGAN [Esser et al. 2021a] (see appendix for architecture details). Due to the quantization process, the reconstruction of the encoder-decoder pair is not perfect and leads to minor changes in the areas that are not edited. To avoid this, we follow the same strategy as SESAME [Ntavelis et al. 2020] and retain only the generated pixels in the masked regions while the rest of the image is retrieved from the original image. To further decrease any small artifacts around the borders of the masked regions we apply Laplacian pyramid image blending as a final post-processing step.

3.4 Training

We randomly sample free-form masks following [Ntavelis et al. 2020] and use the semantic information in the masked area as user edits. The image encoder, decoder, and transformer are trained in a supervised manner on training images which contain ground-truth for masked regions. We first train our image encoders and decoders following VQGAN [Esser et al. 2021a]. We then train our transformer architecture on images with 256 × 256 resolution using the original attention mechanism (full attention), which will be used as the *guiding transformer*. Following that, we switch to train our SGA-transformer with the sparsified guided attention on high resolution, specifically, we initialize its weights from the previously trained guiding transformer and fine-tune it at 512 × 512 resolution again at 1024 × 1024 resolution. In all cases, we use the same losses proposed in VQGAN [Esser et al. 2021b]. For more details about our model and training procedures please see the appendix.

4 RESULTS

In this section, we present qualitative and quantitative evaluation for ASSET.

Dataset. We evaluate our ability to perform high-resolution semantic editing on the Flickr-Landscape dataset consisting of images crawled from the Landscape group on Flickr. It contains 440K high-quality landscape images. We reserve 2000 images for our testing set, while the rest are used for training. Following [Ntavelis et al. 2020], we use 17 semantic classes including mountain, clouds, water, and so on. To avoid expensive manual annotation, we use a pre-trained DeepLabV2 [Chen et al. 2017] to compute segmentation maps for all images. We also use the ADE20K [Zhou et al. 2017] and COCO-Stuff [Caesar et al. 2018] datasets for additional evaluation.

Evaluation metrics. To automatically generate test cases, we simulate user edits by masking out the pixels belonging to a random semantic class for each test image. As explained in Section 3.2, we can sample several output images and also rank them according to their probability. For our experiments, we sample 50 images, and keep the top 10 of them, as also done in other works [Liu et al. 2021b; Zheng et al. 2019]. This results in 20K generated images for our test set. To evaluate the perceptual quality of our edits we compute the FID [Heusel et al. 2017], LPIPS [Zhang et al. 2018], and SSIM metrics [Wang et al. 2004]. For each ground-truth image of the test split, we evaluate these metrics against the synthesized image that achieves

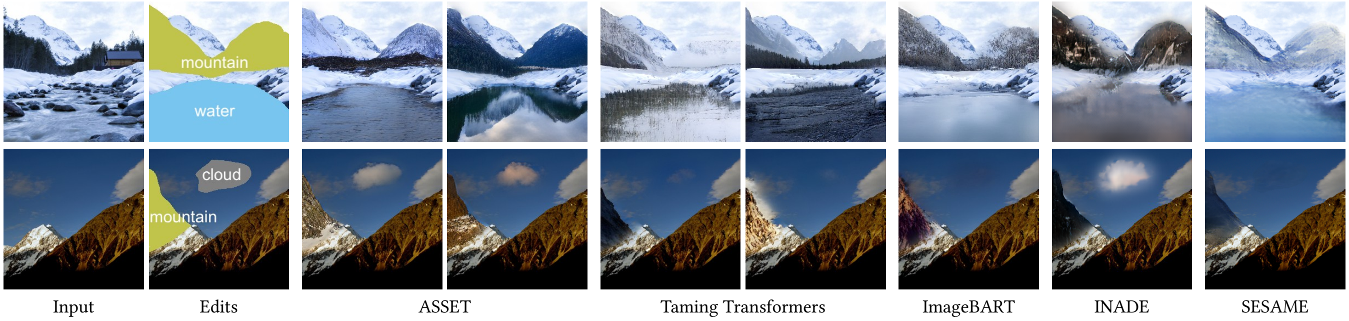


Fig. 4. Comparison of our approach with Taming Transformers [Esser et al. 2021a], ImageBART [Esser et al. 2021b], INADE [Tan et al. 2021], and SESAME [Ntavelis et al. 2020] on 256 × 256 images.

Table 1. Quantitative evaluation on the Flickr-Landscape dataset at various resolutions.

Res	Method	LPIPS ↓	FID ↓	SSIM ↑	mIoU ↑	accu ↑	div ↑
256	*INADE*	0.233	11.2	0.826	48.6	59.1	0.145
	SESAME	0.213	10.2	0.830	50.3	61.7	0.000
	TT	0.201	10.4	0.839	46.1	58.3	**0.187**
	ImageBART	0.196	10.0	0.841	47.3	58.5	0.163
	ASSET	**0.187**	**9.2**	**0.846**	**51.5**	**63.0**	0.151
512	*TT*	0.203	10.6	0.850	52.2	63.7	**0.186**
	ImageBART	0.199	10.4	0.851	52.4	63.3	0.168
	ASSET	**0.186**	**8.4**	**0.856**	**53.5**	**64.7**	0.145
1024	*TT*	0.210	10.9	0.881	50.4	61.7	**0.160**
	ImageBART	0.201	10.4	0.880	50.8	62.1	0.139
	ASSET	**0.160**	**7.7**	**0.887**	**54.1**	**65.2**	0.124

Table 2. Quantitative evaluation on the COCO-Stuff and ADE20K datasets at 512 × 512 resolution.

Dataset	Method	LPIPS ↓	FID ↓	SSIM ↑	mIoU ↑	accu ↑	div ↑
COCO-Stuff	*TT*	0.237	20.2	0.820	36.9	51.7	**0.192**
	ASSET	**0.194**	**14.7**	**0.845**	**43.4**	**58.7**	0.156
ADE20K	*TT*	0.197	15.7	0.860	51.8	68.1	**0.155**
	ASSET	**0.191**	**14.0**	**0.862**	**52.6**	**68.8**	0.140

the best balance of them, as done in [Liu et al. 2021b; Zheng et al. 2019]. To evaluate how well the models adhere to the specified label map at the edited areas we also compare the mean Intersection over Union (mIoU) and the pixel-accuracy between the ground truth semantic map and the inferred one using the pretrained DeepLabV2 model [Chen et al. 2017]. Finally, to evaluate diversity, we utilize the LPIPS metric following [Liu et al. 2021b; Zheng et al. 2019]. The diversity score is calculated as the average LPIPS distance between 5K pairs of images randomly sampled from all generated images, as also done in [Liu et al. 2021b; Zheng et al. 2019]. We also perform a user study to evaluate the perceptual quality of several models.

Baselines. We compare our method with several semantic image editing baselines: SESAME [Ntavelis et al. 2020], INADE [Tan et al. 2021], Taming Transformers (TT) [Esser et al. 2021a], and ImageBART [Esser et al. 2021b]. SESAME and INADE are based on convolutional networks and only support image resolutions of up to 256 × 256 pixels. TT and ImageBART are based on Transformers, but use a sliding window approach at high resolution, in which the attention is only calculated on a local neighborhood for each sampling location. While SESAME can only produce a single output, the other three methods can generate diverse outputs for a given edit. For a fair comparison, when generating image samples using TT or ImageBART, we also selected top-10 images out of 50 sampled ones based on their probability. We selected top-10 images for INADE based on discriminator scores as done in [Liu et al. 2021b; Zheng et al. 2019]. For all baselines, we use the authors' implementation and train them on the same datasets as our method.

Quantitative evaluation. For multimodal editing tasks we aim to obtain outputs that are both diverse and consistent. There is an inherent trade-off between diversity and consistency, as higher diversity can be achieved by sacrificing image consistency. As such, we aim to achieve maximal diversity without generating inconsistent results. For all models that can generate more than one solution for a given edit, we choose the sample with the best balance of quantitative measures (out of the top 10 samples), as done in [Liu et al. 2021b] and [Zheng et al. 2019]. Table 1 shows the comparison with competing models on the Flickr-Landscape dataset at different resolutions.

Except for the diversity metric our model outperforms all competing methods on all resolutions. While TT and ImageBART achieve a higher diversity than our model, we observe that this higher diversity comes at the cost of inconsistent images (see Figure 5 and Figure 6). In contrast, our approach also achieves high diversity but shows much more consistent image outputs, both at lower and higher resolutions. At low resolution (256 × 256), our method differs from TT by using a bidirectional transformer encoder to capture global context and partial convolutions to prevent information leakage from masked regions. As we increase the resolution (512 × 512 and higher), our approach continues to obtain consistent results, thanks to the Sparsified Guided Attention (SGA) that captures long-range dependencies. In contrast, the perceptual performance of TT

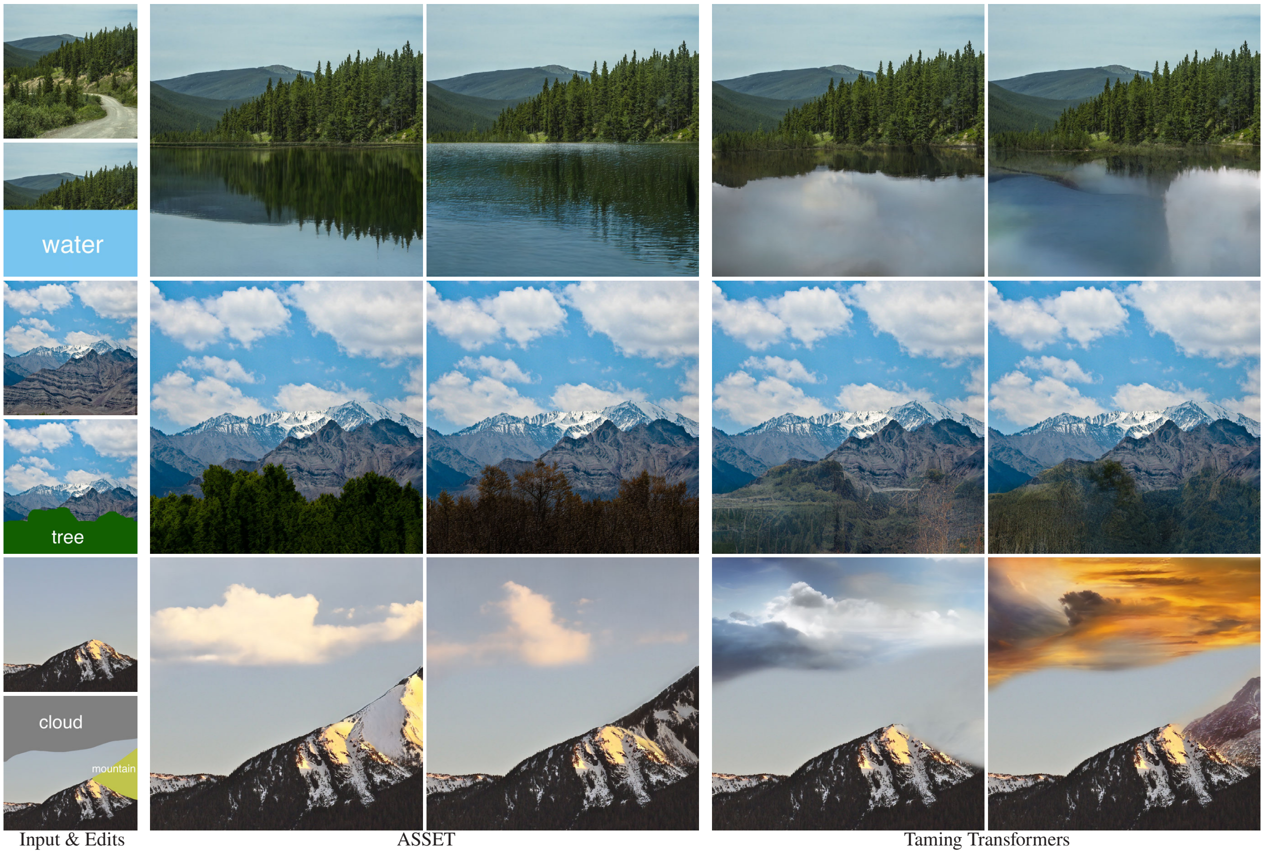


Fig. 5. Comparison of ASSET with Taming Transformers [Esser et al. 2021a] at 1024 × 1024 resolution.

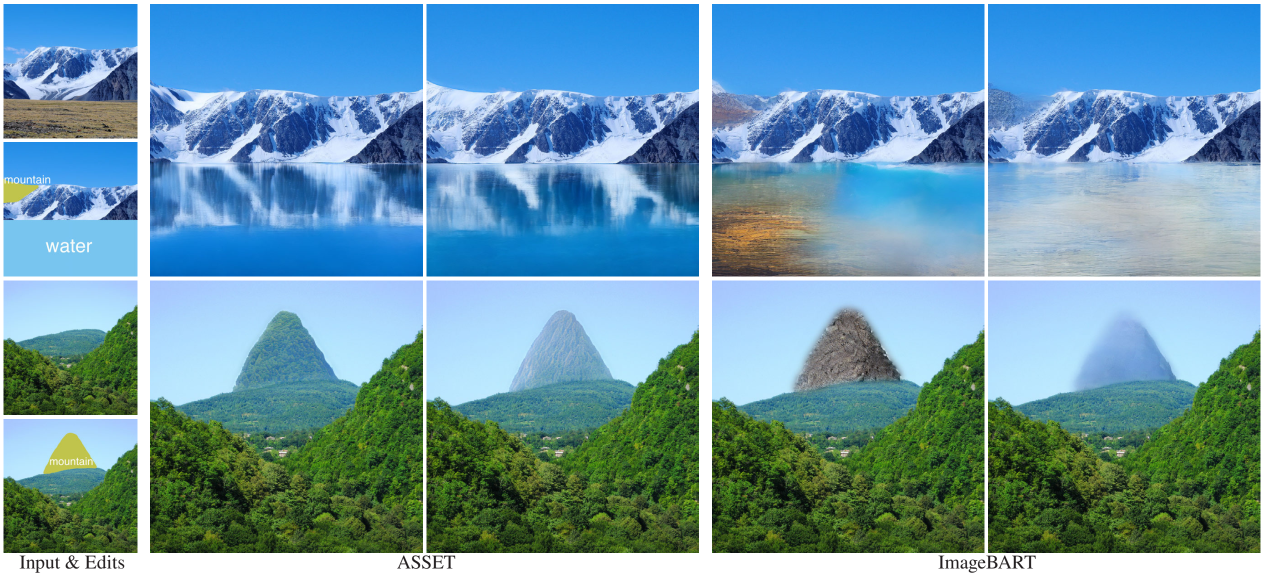


Fig. 6. Comparison of ASSET with ImageBART [Esser et al. 2021b] at 1024 × 1024 resolution.

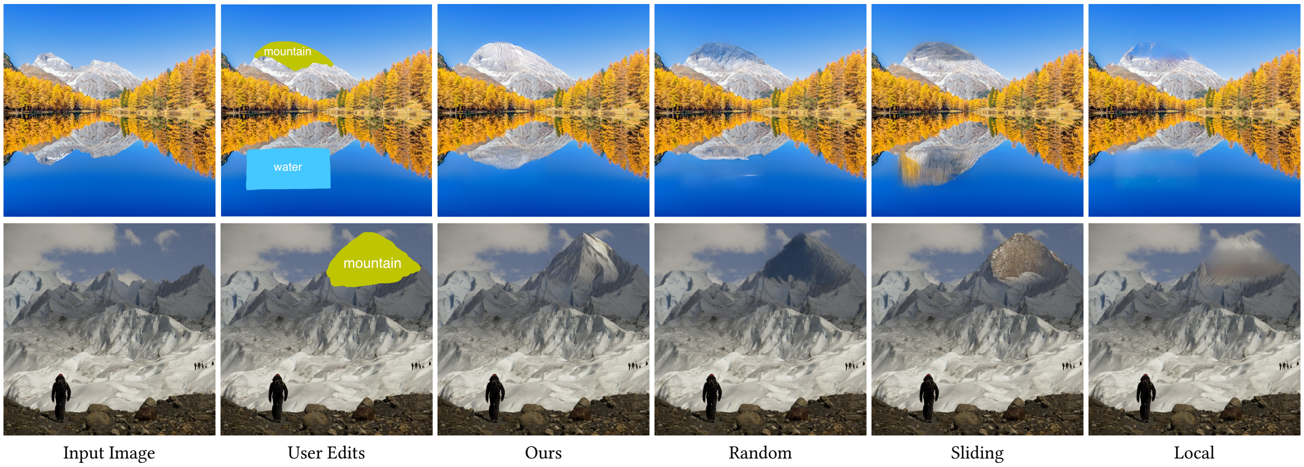


Fig. 7. Comparison showing the effects of using different kinds of attention at 1024 × 1024 resolution.

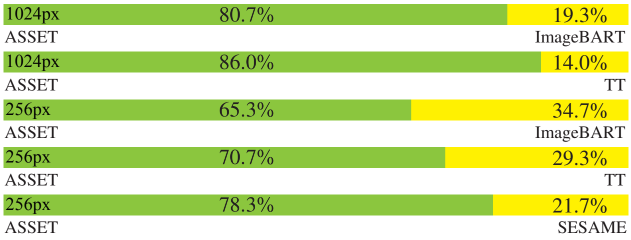


Fig. 8. User study results. At both low and high resolution, our method ASSET is dominantly preferred over the baselines.

and ImageBART decreases with increasing resolution, as the sliding window approach is unable to enforce consistency over long distances.

We also conduct experiments on the COCO-Stuff and ADE20K datasets. Table 2 shows the comparison with TT at 512 × 512 resolution. Our model outperforms TT on both datasets.

Qualitative evaluation. For qualitative evaluation, a brush of a semantic class is used, painting over the image. Figure 4 shows comparison with all competing methods at 256×256 resolution. Since we do not need SGA at small resolutions, we only use our guiding transformer for these examples. Compared to other approaches, our method produces more coherent content with fewer artifacts. In Figure 5 and Figure 6, we show the comparison on 1024 × 1024 images against Taming Transformers and ImageBART respectively. Figure 9 and Figure 10 show qualitative results on COCO-Stuff and ADE20K at 512 × 512 resolution. Even at high resolution, our method can synthesize coherent content across the whole image, while Taming Transformers and ImageBART fail to capture long-range dependency and sometimes ignore the user edits.

Table 3. Effects of different forms of attention on the Flickr-Landscapes dataset at 512 × 512 resolutions.

Method	LPIPS ↓	FID ↓	SSIM ↑	mIoU ↑	accu ↑	div ↑
Sliding	0.214	10.7	0.847	52.6	63.1	**0.180**
Local	0.209	10.1	0.853	51.1	62.4	0.152
Random	0.202	9.6	0.851	50.0	61.6	0.157
ASSET	**0.186**	**8.4**	**0.856**	**53.5**	**64.7**	0.145

User study. To further evaluate perceptual image quality we also conduct an Amazon MTurk study. We showed participants a masked input image, along with a randomly ordered pair of images synthesized by ASSET and one of our baseline algorithms. The participants were then asked which edited image looks more photo-realistic and coherent with the rest of the image. Figure 8 summarizes 1500 user responses for 256 × 256 and 1024 × 1024 resolutions. The study shows that our method receives the most votes for better synthesis compared to other methods in both resolutions, with the largest margin at the highest 1024 × 1024 resolution.

Ablation study. To evaluate the effect of our SGA, we perform several ablations with different attention approaches. The following variants are evaluated at 512 × 512 resolution: **(1)** ***Sliding***: we use our guiding transformer with the sliding window approach as in [Esser et al. 2021a]. **(2)** ***Local***: we remove our top-K attention and only use neighboring window attention $\mathcal{N}(r)$. **(3)** ***Random***: we use random instead of top-K attention similar to [Zaheer et al. 2020]. Table 3 shows the performance of all variants compared to our full model. Our model outperforms all variants in all metrics except for diversity. As before, we observe that higher diversity can be achieved at the cost of poorer image consistency. In Figure 7 we show qualitative comparisons with the proposed variants trained at 1024 × 1024 resolution. As we can see, without the SGA component, the image consistency and perceptual quality decreases as the model either only attends to local areas (*sliding* and *local*) or fails to attend to important image regions at each sampling step (*random*).

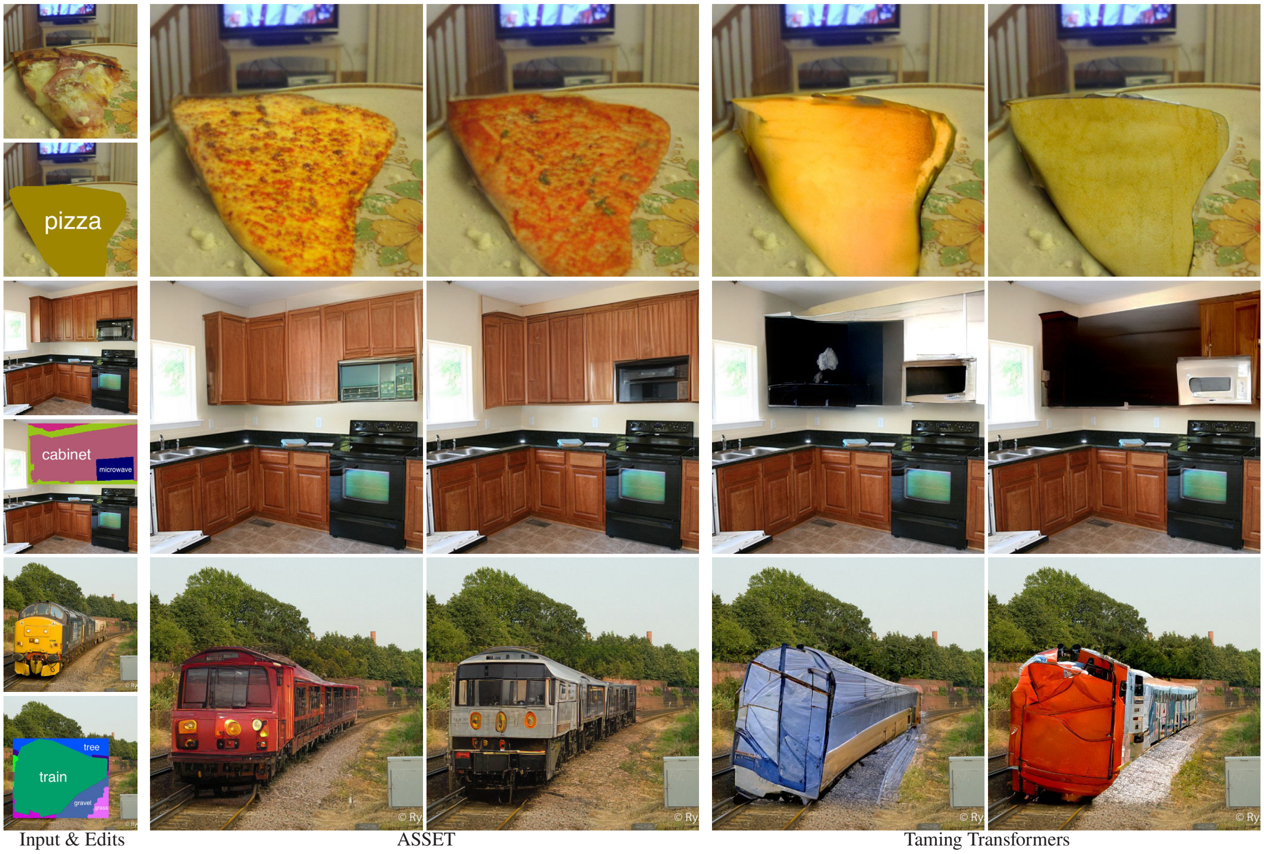


Fig. 9. Qualitative results and comparisons with Taming Transformers [Esser et al. 2021a] on COCO-Stuff at 512 × 512 resolution.

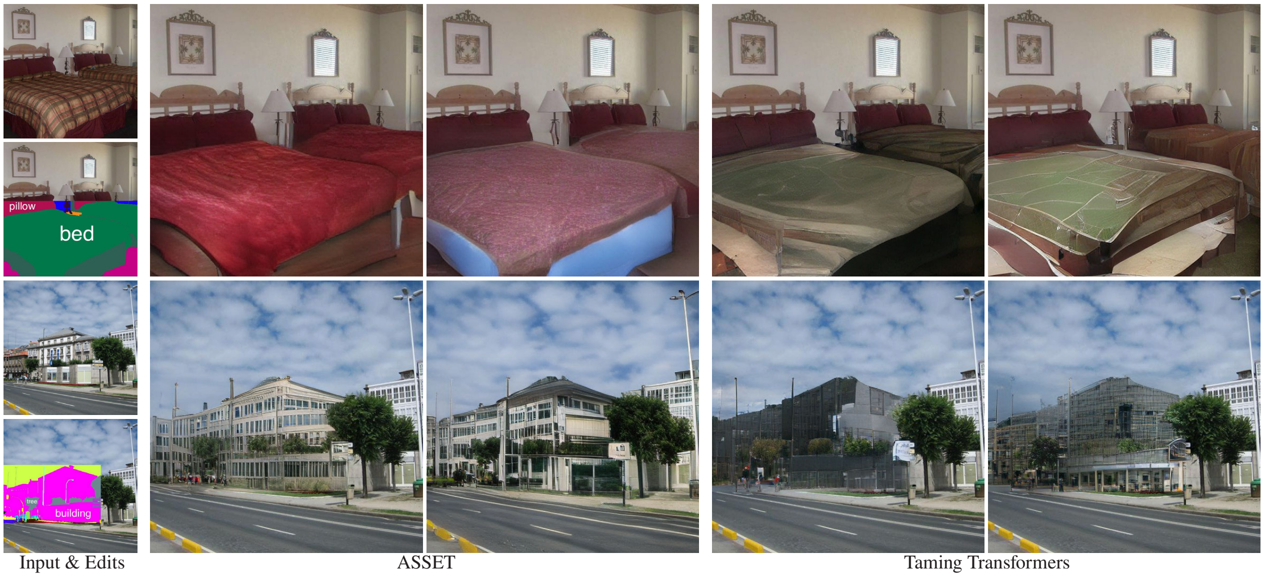


Fig. 10. Qualitative results and comparisons with Taming Transformers [Esser et al. 2021a] on ADE20K at 512 × 512 resolution.

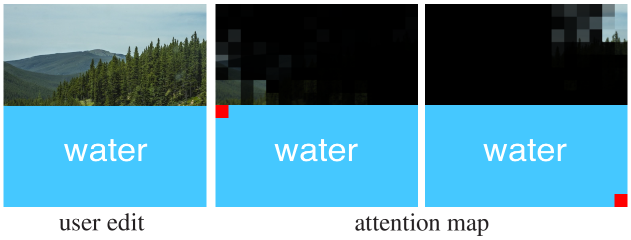


Fig. 11. Encoder self-attention visualized for two different query points (shown as red). The image regions acquiring higher attention are the ones more relevant to generate water reflections at each of these points.

Table 4. Number of transformer parameters for TT and ASSET.

Dataset	TT	ASSET
Landscape	**307M**	343M
COCO-Stuff	651M	**365M**
ADE20K	405M	**269M**

Model capacity comparison with TT. We compare the number of transformer parameters with TT in Table 4. Our transformer's number of parameters is ~12% larger than the one in TT for the Landscape dataset, and much smaller for the COCO-Stuff and ADE20K datasets. We note that ASSET's and TT's CNNs have the same number of parameters.

Attention visualization. We use Attention Rollout [Abnar and Zuidema 2020] to visualize the attention map of our guiding transformer encoder. Specifically, we average attention weights of the guiding transformer encoder across all heads and then recursively multiply the resulting averaged attention matrices of all layers. The attention maps for two different query points are presented in Figure 11. The guiding transformer can attend to informative regions for different query points. For the query points in Figure 11, the regions with high attention correspond to image areas useful to synthesize reflection of scenery at each of these points.

VQGAN leakage visualization. We employ partial convolutions [Liu et al. 2018] and region normalization [Yu et al. 2020] in our image encoder while processing the unmasked image regions. The reason is that the features produced for unmasked regions should not be affected by the masked image regions. Partial convolutions and region normalization avoid any information leakage of the masked area. In Figure 12, we visualize leakage for the original VQGAN and our improved image encoder. With our modification, the latent features produced for unmasked regions are independent of the masked area.

Inference speed comparisons. Following [Cao et al. 2021; Esser et al. 2021b], we record the average inference time on Flickr Landscape and ADE20K as shown in Table 5. The inference speed is influenced by the size of the masked region relevant to the size of the input image (i.e., ratio of the masked region). Following [Cao et al. 2021], we report the average masked ratio in this table. Our method achieves similar inference speed with TT, while producing much higher-quality results than TT.

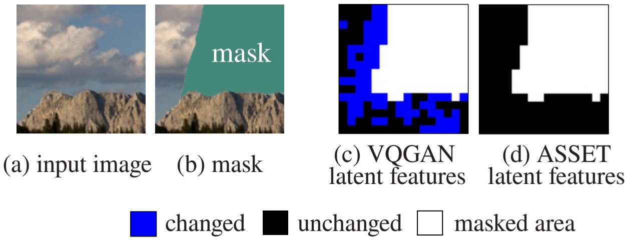


Fig. 12. The masked area of the input image is replaced with random noise. The difference of 16 × 16 latent features between the original image (a) and the masked image (b) are visualized on the right. Changed and unchanged latent features are visualized in blue and black respectively. Image (c) shows how unmasked image latent tokens are affected (blue tokens) by the masked area in the original VQGAN. Image (d) shows that our image encoder successfully prevents leakage from the masked area to the unmasked area.

Table 5. Average inference time in seconds per image.

Resolution	Dataset	Masked Ratio	TT	ASSET
1024	Landscape	0.287	**52.3**	55.8
512	ADE20K	0.296	16.9	**10.6**

Inference time of guiding transformer. Measured on the Landscape dataset, the average inference time of our guiding transformer (256 × 256 resolution) represents only a small fraction (3.4%) of the total inference time of the full ASSET pipeline. The majority of the inference time (96.6%) is taken by our architecture operating at high resolution, which is the crucial part significantly accelerated by our SGA mechanism.

Comparison with full attention. Based on an NVIDIA A100 (40GB VRAM) at 1024 × 1024 resolution with a batch size of 1, the transformer architecture requirements with full attention exceeds the available memory during training. Using our Sparsified Guided Attention mechanism, the transformer architecture utilizes 37GB at train time. In terms of inference time during testing, the cost of the guiding transformer is significantly lower: ASSET is about 20 times faster at test time compared to using full attention at 1024 × 1024 resolution.

5 LIMITATIONS AND CONCLUSION

We introduce a novel transformer-based approach for semantic image editing at high resolutions. Previous approaches have difficulty in modeling long-range dependencies between image areas that are far apart, resulting in unrealistic and inconsistent image edits. To this end, we introduce a novel attention mechanism called *Sparsified Guided Attention* (SGA), which uses the full attention map at the coarse resolution to produce a sparse attention map at full resolution. Our experiments show that SGA outperforms other variants of localized or sparse attention, and allows us to obtain realistic and diverse image edits even at high resolutions of 1024 × 1024 pixels.

While our approach can perform consistent and diverse edits at high resolutions of up to 1024×1024 pixels, there are still avenues for further improvements. A common issue in transformers including ours is that directly applying a trained model to generate content at a higher resolution degrades performance, since the learned positional embedding cannot adapt to the new resolution. In addition, the repeated autoregressive sampling takes several minutes to perform a full edit at 1024 × 1024 resolution. To alleviate this issue, we can sample a diverse set of outputs for a given edit in parallel on multiple GPUs. Finally, the synthesized content may not be perfectly aligned with the provided mask since the masking takes place at a low resolution in the latent space.

ACKNOWLEDGMENTS

This work is funded by Adobe Research.

REFERENCES

Samira Abnar and Willem Zuidema. 2020. Quantifying attention flow in transformers. In *Proc. ACL*.

David Bau, Hendrik Strobelt, William Peebles, Jonas Wulff, Bolei Zhou, Jun-Yan Zhu, and Antonio Torralba. 2019. Semantic Photo Manipulation with a Generative Image Prior. *ACM Trans. Graph.* 38, 4 (2019).

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. arXiv:2004.05150.

Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. 2018. Coco-stuff: Thing and stuff classes in context. In *Proc. CVPR*.

Chenjie Cao, Yuxin Hong, Xiang Li, Chengrong Wang, Chengming Xu, XiangYang Xue, and Yanwei Fu. 2021. The Image Local Autoregressive Transformer. In *Proc. NeurIPS*.

Jiawen Chen, Andrew Adams, Neal Wadhwa, and Samuel W. Hasinoff. 2016. Bilateral Guided Upsampling. *ACM Trans. Graph.* 35, 6 (2016).

Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. 2017. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40, 4 (2017).

Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. 2020. Generative pretraining from pixels. In *Proc. ICML*.

Qifeng Chen and Vladlen Koltun. 2017. Photographic image synthesis with cascaded refinement networks. In *Proc. ICCV*.

Yu Cheng, Zhe Gan, Yitong Li, Jingjing Liu, and Jianfeng Gao. 2020. Sequential attention GAN for interactive image editing. In *Proc. ACM Multimedia*.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. arXiv:1904.10509.

Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. 2021a. Twins: Revisiting the design of spatial attention in vision transformers. In *Proc. NeurIPS*.

Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. 2021b. Conditional positional encodings for vision transformers. arXiv:2102.10882.

Helisa Dhamo, Azade Farshad, Iro Laina, Nassir Navab, Gregory D Hager, Federico Tombari, and Christian Rupprecht. 2020. Semantic image manipulation using scene graphs. In *Proc. CVPR*.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proc. ICLR*.

Patrick Esser, Robin Rombach, Andreas Blattmann, and Björn Ommer. 2021b. ImageBART: Bidirectional Context with Multinomial Diffusion for Autoregressive Image Synthesis. In *Proc. NeurIPS*.

Patrick Esser, Robin Rombach, and Bjorn Ommer. 2021a. Taming transformers for high-resolution image synthesis. In *Proc. CVPR*.

David Ferstl, Christian Reinbacher, Rene Ranftl, Matthias Ruether, and Horst Bischof. 2013. Image Guided Depth Upsampling Using Anisotropic Total Generalized Variation. In *Proc. ICCV*.

Shuyang Gu, Jianmin Bao, Hao Yang, Dong Chen, Fang Wen, and Lu Yuan. 2019. Mask-guided portrait editing with conditional gans. In *Proc. CVPR*.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Proc. NeurIPS*.

Tobias Hinz, Matthew Fisher, Oliver Wang, and Stefan Wermter. 2021. Improved techniques for training single-image gans. In *Proc. WACV*.

Tobias Hinz, Stefan Heinrich, and Stefan Wermter. 2019. Generating multiple objects at spatially distinct locations. In *Proc. ICLR*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *Proc. ICLR*.

Seunghoon Hong, Xinchen Yan, Thomas Huang, and Honglak Lee. 2018. Learning hierarchical semantic image manipulation through structured representations. In *Proc. NeurIPS*.

Tak-Wai Hui, Chen Change Loy, and Xiaoou Tang. 2016. Depth Map Super-Resolution by Deep Multi-Scale Guidance. In *Proc. ECCV*.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-Image Translation with Conditional Adversarial Networks. In *Proc. CVPR*.

Yifan Jiang, Shiyu Chang, and Zhangyang Wang. 2021. TransGAN: Two Transformers Can Make One Strong GAN. In *Proc. NeurIPS*.

Youngjoo Jo and Jongyoul Park. 2019. Sc-fegan: Face editing generative adversarial network with user's sketch and color. In *Proc. ICCV*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proc. ICLR*.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. In *Proc. ICLR*.

Johannes Kopf, Michael F Cohen, Dani Lischinski, and Matt Uyttendaele. 2007. Joint bilateral upsampling. *ACM Trans. Graph.* 26, 3 (2007).

Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. 2020. Maskgan: Towards diverse and interactive facial image manipulation. In *Proc. CVPR*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proc. ACL*.

Huan Ling, Karsten Kreis, Daiqing Li, Seung Wook Kim, Antonio Torralba, and Sanja Fidler. 2021. EditGAN: High-Precision Semantic Image Editing. In *Proc. NeurIPS*.

Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. 2018. Image inpainting for irregular holes using partial convolutions. In *Proc. ECCV*.

Hongyu Liu, Ziyu Wan, Wei Huang, Yibing Song, Xintong Han, and Jing Liao. 2021b. PD-GAN: Probabilistic Diverse GAN for Image Inpainting. In *Proc. CVPR*.

Hongyu Liu, Ziyu Wan, Wei Huang, Yibing Song, Xintong Han, Jing Liao, Bin Jiang, and Wei Liu. 2021c. DeFLOCNet: Deep Image Editing via Flexible Low-level Controls. In *Proc. CVPR*.

Ming-Yu Liu, Oncel Tuzel, and Yuichi Taguchi. 2013. Joint geodesic upsampling of depth images. In *Proc. CVPR*.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021a. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proc. ICCV*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proc. ICLR*.

Riccardo de Lutio, Stefano D'aronco, Jan Dirk Wegner, and Konrad Schindler. 2019. Guided super-resolution as pixel-to-pixel transformation. In *Proc. ICCV*.

Seonghyeon Nam, Yunji Kim, and Seon Joo Kim. 2018. Text-adaptive generative adversarial networks: manipulating images with natural language. In *Proc. NeurIPS*.

Evangelos Ntavelis, Andrés Romero, Iason Kastanis, Luc Van Gool, and Radu Timofte. 2020. SESAME: semantic editing of scenes by adding, manipulating or erasing objects. In *Proc. ECCV*.

Jaesik Park, Hyeongwoo Kim, Yu-Wing Tai, Michael S. Brown, and Inso Kweon. 2011. High quality depth map upsampling for 3D-TOF cameras. In *Proc. ICCV*.

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. 2019. Semantic image synthesis with spatially-adaptive normalization. In *Proc. CVPR*.

Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. 2018. Image transformer. In *Proc. ICML*.

Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. 2021. Styleclip: Text-driven manipulation of stylegan imagery. In *Proc. ICCV*.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. In *Proc. ICML*.

Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. 2019. Singan: Learning a generative model from a single natural image. In *Proc. ICCV*.

Tamar Rott Shaham, Michaël Gharbi, Richard Zhang, Eli Shechtman, and Tomer Michaeli. 2021. Spatially-Adaptive Pixelwise Networks for Fast Image Translation. In *Proc. CVPR*.

Assaf Shocher, Nadav Cohen, and Michal Irani. 2018. "zero-shot" super-resolution using deep internal learning. In *Proc. CVPR*.

Sitong Su, Lianli Gao, Junchen Zhu, Jie Shao, and Jingkuan Song. 2021. Fully Functional Image Manipulation Using Scene Graphs in A Bounding-Box Free Way. In *Proc. ACM Multimedia*.

Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor

Lempitsky. 2022. Resolution-robust Large Mask Inpainting with Fourier Convolutions. In *Proc. WACV*.

Zhentao Tan, Menglei Chai, Dongdong Chen, Jing Liao, Qi Chu, Bin Liu, Gang Hua, and Nenghai Yu. 2021. Diverse Semantic Image Synthesis via Probability Distribution Modeling. In *Proc. CVPR*.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020. Efficient transformers: A survey. arXiv:2009.06732.

Shubham Tulsiani and Abhinav Gupta. 2021. PixelTransformer: Sample Conditioned Signal Generation. In *Proc. ICML*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. NeurIPS*.

Yael Vinker, Eliahu Horwitz, Nir Zabari, and Yedid Hoshen. 2021. Image Shape Manipulation from a Single Augmented Training Sample. In *Proc. ICCV*.

Ziyu Wan, Jingbo Zhang, Dongdong Chen, and Jing Liao. 2021. High-Fidelity Pluralistic Image Completion with Transformers. In *Proc. ICCV*.

Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. arXiv:2006.04768.

Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. 2021. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proc. ICCV*.

Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. 2018. Non-local neural networks. In *Proc. CVPR*.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing* 13, 4 (2004).

Chao Yang, Xin Lu, Zhe Lin, Eli Shechtman, Oliver Wang, and Hao Li. 2017. High-resolution image inpainting using multi-scale neural patch synthesis. In *Proc. CVPR*.

Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. 2021. Focal Self-attention for Local-Global Interactions in Vision Transformers. In *Proc. NeurIPS*.

Jingyu Yang, Xinchen Ye, Kun Li, Chunping Hou, and Yao Wang. 2014. Color-Guided Depth Recovery From RGB-D Data Using an Adaptive Autoregressive Model. *IEEE Transactions on Image Processing* 23, 8 (2014).

Qingxiong Yang, Ruigang Yang, James Davis, and David Nister. 2007. Spatial-Depth Super Resolution for Range Images. In *Proc. CVPR*.

Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. 2018. Generative image inpainting with contextual attention. In *Proc. CVPR*.

Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. 2019. Free-form image inpainting with gated convolution. In *Proc. ICCV*.

Tao Yu, Zongyu Guo, Xin Jin, Shilin Wu, Zhibo Chen, Weiping Li, Zhizheng Zhang, and Sen Liu. 2020. Region normalization for image inpainting. In *Proc. AAAI*.

Yingchen Yu, Fangneng Zhan, Rongliang Wu, Jianxiong Pan, Kaiwen Cui, Shijian Lu, Feiying Ma, Xuansong Xie, and Chunyan Miao. 2021. Diverse image inpainting with bidirectional and autoregressive transformers. In *Proc. ACM Multimedia*.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big Bird: Transformers for Longer Sequences.. In *Proc. NeurIPS*.

Pengchuan Zhang, Xiyang Dai, Jianwei Yang, Bin Xiao, Lu Yuan, Lei Zhang, and Jianfeng Gao. 2021. Multi-scale vision longformer: A new vision transformer for high-resolution image encoding. In *Proc. ICCV*.

Pan Zhang, Bo Zhang, Dong Chen, Lu Yuan, and Fang Wen. 2020. Cross-domain correspondence learning for exemplar-based image translation. In *Proc. CVPR*.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. CVPR*.

Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. 2019. Pluralistic image completion. In *Proc. CVPR*.

Haitian Zheng, Zhe Lin, Jingwan Lu, Scott Cohen, Jianming Zhang, Ning Xu, and Jiebo Luo. 2021. Semantic Layout Manipulation with High-Resolution Sparse Attention. arXiv:2012.07288.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. 2017. Scene parsing through ade20k dataset. In *Proc. CVPR*.

Xingran Zhou, Bo Zhang, Ting Zhang, Pan Zhang, Jianmin Bao, Dong Chen, Zhongfei Zhang, and Fang Wen. 2021. CoCosNet v2: Full-Resolution Correspondence Learning for Image Translation. In *Proc. CVPR*.

Peihao Zhu, Rameen Abdal, Yipeng Qin, and Peter Wonka. 2020. Sean: Image synthesis with semantic region-adaptive normalization. In *Proc. CVPR*.

A IMPLEMENTATION DETAILS

Here we provide implementation details of our network architecture and training procedure. Our model is implemented in PyTorch.

Image encoder / decoder. Our image encoder (Section 3.1) and decoder (Section 3.3) use the architecture shown in Table 7. The design of the networks follows the architecture proposed in VQGAN [Esser et al. 2021a]. One difference is that we employ partial convolutions [Liu et al. 2018] and region normalization [Yu et al. 2020] in our image encoder while processing the unmasked image regions. The reason is that the features produced for unmasked regions should not be affected by the masked image regions. Partial convolutions and region normalization avoid any information leakage of the masked area.

Table 6. Ablation for use of global blocks at 512 × 512 resolution.

Method	LPIPS ↓	FID ↓	SSIM ↑	mIoU ↑	accu ↑
R+G	0.207	9.5	0.849	50.3	61.7
Random	0.202	9.6	0.851	50.0	61.6
ASSET	**0.186**	**8.4**	**0.856**	**53.5**	**64.7**

The semantic map encoder follows the same architecture with regular convolutions. We note that during training, VQGAN measures the reconstruction error in terms of both the image and semantic map. Thus, along with the decoder for the image, we also use a decoder to reconstruct the semantic map. The semantic map decoder uses an architecture following the "decoder" column in Table 7. A minor difference is that the number of input/output channels is changed from 3 to the number of categories C in the semantic map encoder and decoder.

Transformer. Our transformer (Section 3.2) follows the architecture presented in BART [Lewis et al. 2020]. All the hyperparameters for the transformer are described in Table 8.

Following [Chu et al. 2021a,b], the position encoding generator (PEG) is placed before each transformer encoder layer. The position encoding generator is a 5 × 5 depth-wise convolution with the padding size of 2, which convolves with each block independently. Similar to the encoder layer, we add one PEG before the first decoder layer which takes the encoder output representation as input to produce positional embeddings for the transformer decoder.

Sparsified Guided Attention. For each transformer layer and attention head, a full attention matrix $\mathbf{A}_{low} \in \mathbb{R}^{256\times256}$ is computed from the downsampled input image. The computation of the block attention matrix $\mathbf{B}$ for high-resolution is guided by $\mathbf{A}_{low}$. Specifically, in our experiments, the number of blocks N is set to 64. Each block consists of $\frac{256}{64} = 4$ tokens at low-resolution. The affinity value of each block corresponds to a 4×4 region in $\mathbf{A}_{low}$. In our implementation, we use a 2D average pooling layer with kernel size 4 and stride 4 to downsample $\mathbf{A}_{low}$ into $\mathbf{B}$.

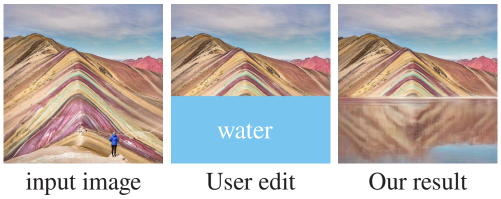
Training details. For the image encoder/decoder and semantic encoder/decoder, we used the Adam optimizer [Kingma and Ba 2015] with learning rate $7.2 \cdot 10^{-6}$ and batch size 16. During the training of the guiding transformer, we used the AdamW optimizer [Loshchilov and Hutter 2019] with learning rate $3.2 \cdot 10^{-5}$ and batch size 224. During the finetuning of the SGA-transformer, we used the AdamW optimizer [Loshchilov and Hutter 2019] with learning rate $1.2 \cdot 10^{-5}$ and batch size 8. All training is done on 8 A100 GPUs.

Table 7. Architecture of the image encoder and decoder. Note that $H_{\text{feat}} = \frac{H_{\text{im}}}{16}$, $W_{\text{feat}} = \frac{W_{\text{im}}}{16}$.

Encoder	Decoder
$\mathbf{X} \in \mathbb{R}^{H_{\text{im}} \times W_{\text{im}} \times 3}$	$\hat{\mathbf{F}} \in \mathbb{R}^{H_{\text{feat}} \times W_{\text{feat}} \times 256}$
Conv2D $\longrightarrow \mathbb{R}^{H_{\text{im}} \times W_{\text{im}} \times 128}$	Conv2D $\longrightarrow \mathbb{R}^{H_{\text{feat}} \times W_{\text{feat}} \times 512}$
4× { Residual Block, Downsample Block} $\longrightarrow \mathbb{R}^{H_{\text{feat}} \times W_{\text{feat}} \times 512}$	Residual Block $\longrightarrow \mathbb{R}^{H_{\text{feat}} \times W_{\text{feat}} \times 512}$
Residual Block $\longrightarrow \mathbb{R}^{H_{\text{feat}} \times W_{\text{feat}} \times 512}$	Non-Local Block $\longrightarrow \mathbb{R}^{H_{\text{feat}} \times W_{\text{feat}} \times 512}$
Non-Local Block $\longrightarrow \mathbb{R}^{H_{\text{feat}} \times W_{\text{feat}} \times 512}$	Residual Block $\longrightarrow \mathbb{R}^{H_{\text{feat}} \times W_{\text{feat}} \times 512}$
Residual Block $\longrightarrow \mathbb{R}^{H_{\text{feat}} \times W_{\text{feat}} \times 512}$	4× { Residual Block, Upsample Block} $\longrightarrow \mathbb{R}^{H_{\text{im}} \times W_{\text{im}} \times 128}$
GroupNorm, Swish, Conv2D $\longrightarrow \mathbb{R}^{H_{\text{feat}} \times W_{\text{feat}} \times 256}$	GroupNorm, Swish, Conv2D $\longrightarrow \mathbb{R}^{H_{\text{im}} \times W_{\text{im}} \times 3}$

Table 8. Transformer hyperparameters. For every experiment, the number of total blocks N, the number of blocks in $\mathcal{N}(r)$, the number of blocks in $\mathcal{K}(r)$ are set to 64, 3, 3 respectively. n_E denotes the number of transformer layers in the bidirectional encoder, n_D is the number of transformer layers in the autoregressive decoder, # params is the number of transformer parameters, n_h is the number of attention heads in the transformer, $|\mathcal{Z}|$ is the number of codebook entries, dropout is the dropout rate used for training the transformer, and n_e is the token embedding dimensionality.

Dataset	n_E	n_D	# params $[M]$	n_h	$\lvert\mathcal{Z}\rvert$	dropout	n_e
Flickr-Landscape	7	15	343	16	1024	0.0	1024
COCO-Stuff	7	15	365	16	8192	0.0	1024
ADE20K	7	10	269	16	4096	0.1	1024



input image User edit Our result

Fig. 13. Example of a less successful result.

B ADDITIONAL RESULTS AND COMPARISONS

Additional comparisons. Please see Figure 14 and Figure 15 for more comparisons at 1024 resolution.

Adding global blocks. In our ablation study, we also experimented with the global block presented in BigBird [Zaheer et al. 2020]. Specifically, we make the first and last blocks "global", which attend over the entire sequence. Similar to [Zaheer et al. 2020], we use the global attention together with the local attention and random attention – this variant is referred to as *R+G*. The results did not improve compared to the *Random* variant in terms of our evaluation metrics (see Table 6).

Failure case. Structured textures such as the mountain in Figure 13 is challenging for reflection synthesis. In this case, our result may not reproduce the texture well.

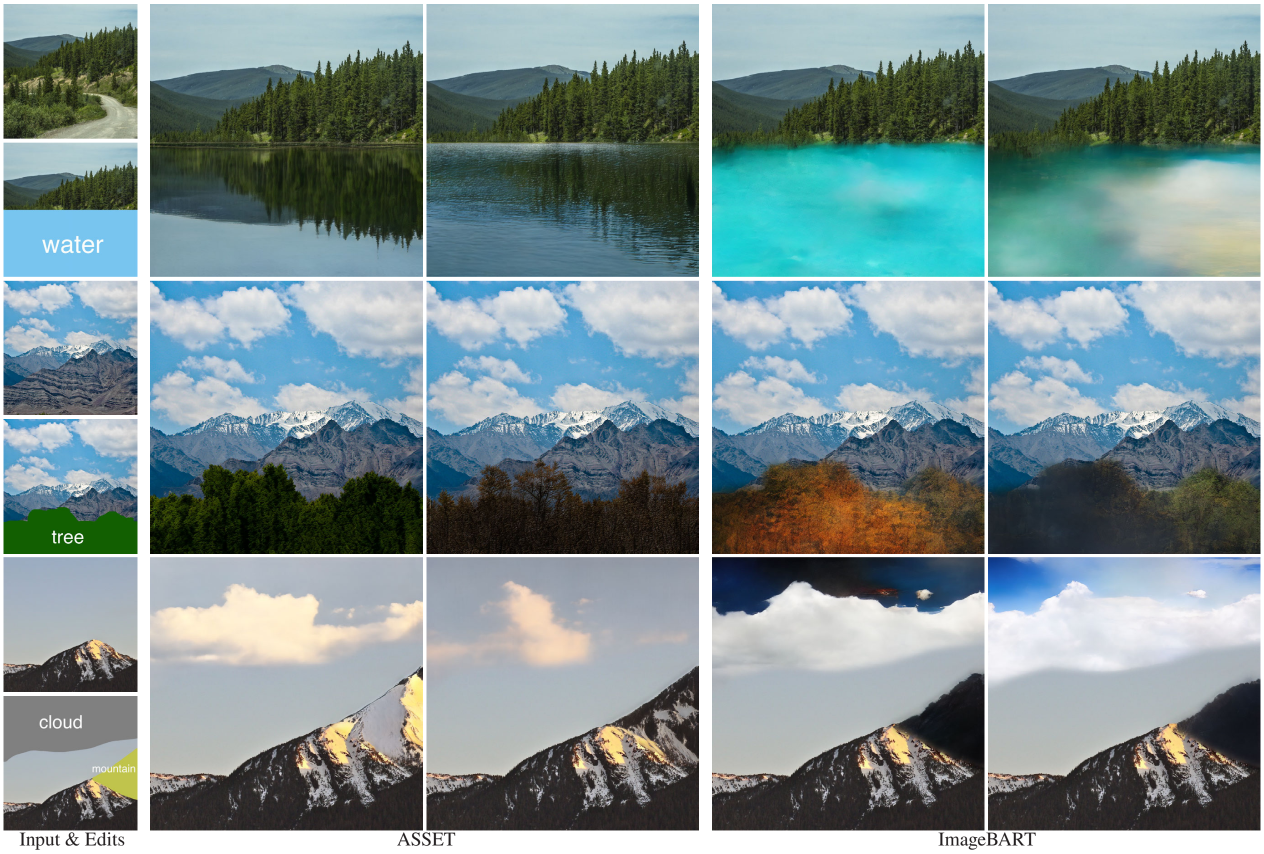


Fig. 14. Comparison of ASSET with ImageBART [Esser et al. 2021b] at 1024 × 1024 resolution.

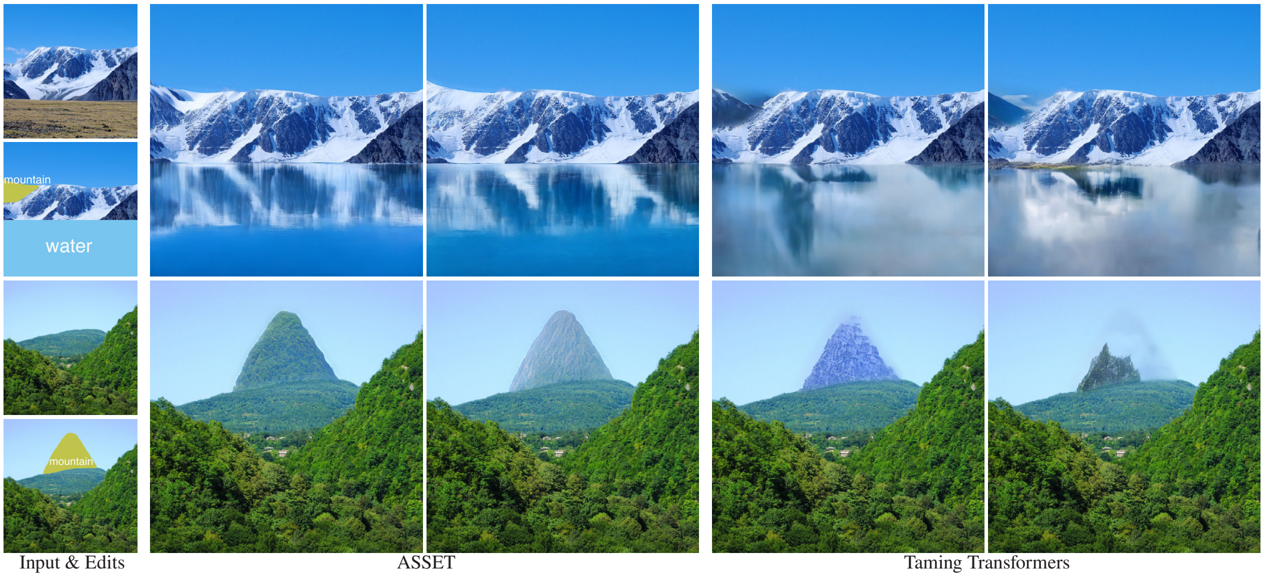


Fig. 15. Comparison of ASSET with Taming Transformers [Esser et al. 2021a] at 1024 × 1024 resolution.