End-to-End Visual Editing with a Generatively Pre-trained Artist

Andrew Brown[1,2(✉)], Cheng-Yang Fu[2], Omkar Parkhi[2], Tamara L. Berg[2], and Andrea Vedaldi[1,2]

[1] Visual Geometry Group, University of Oxford, Oxford, UK
abrown@robots.ox.ac.uk
[2] Meta AI Research, New York, US
{chengyangfu,omkar,tlberg,vedaldi}@fb.com
https://www.robots.ox.ac.uk/abrown/E2EVE/

Abstract. We consider the targeted image editing problem, namely blending a region in a source image with a driveg that specifiesthe desired change. Differently from prior works, we solve this problem by learning a conditional probability distribution of the edits, *end-to-end* in code space. Training such a model requires addressing the lack of example edits for training. To this end, we propose a self-supervised approach that simulates edits by augmenting off-the-shelf images in a target domain. The benefits are remarkable: implemented as a state-of-the-art auto-regressive transformer, our approach is simple, sidesteps difficulties with previous methods based on GAN-like priors, obtains significantly better edits, and is efficient. Furthermore, we show that different blending effects can be learned by an intuitive control of the augmentation process, with no other changes required to the model architecture. We demonstrate the superiority of this approach across several datasets in extensive quantitative and qualitative experiments, including human studies, significantly outperforming prior work.

1 Introduction

A key part of the creative process is the ability to combine known factors in novel ways. For instance, we can imagine how a dress would look like with a different v-neck, or our bedroom would look like with the large windows we have seen in a magazine. In this paper, we thus consider the problem of generating new variants of a source image, guided by another image containing a feature, such as a component of a dress or window style, that we wish to change in the source. For additional control, we wish the edit operation to focus on a particular target region of the source, leaving the context as unchanged as possible while maintaining the realism of the edit (see Fig. 1).

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-19784-0_2.

S. Avidan et al. (Eds.): ECCV 2022, LNCS 13675, pp. 18–35, 2022.
https://doi.org/10.1007/978-3-031-19784-0_2

Fig. 1. E2EVE combines a driver and source image (resp. to the left and right of the $\oplus$ symbol), generating a new version of the source that resembles the driver in the edit region (marked in blue). The generated output looks realistic while faithfully resembling the driver. Our method can be trained to work well on different types of images, including bedrooms, dresses, and faces, and can use regions of arbitrary shape, from rough rectangles to pixel-accurate segmentations (bottom-right). (Color figure online)

Prior works consider image editing tasks, but often guided by a *textual* description of the desired change [21,43,51]. We argue that specifying edits visually rather than textually offers a far more fine grained and explicit level of control, ultimately resulting in a more useful editor[1]. Formally, we can describe the editing process as drawing a sample from a conditional image distribution $P(\hat{x}|x, y, R)$, where x is the source image, y is the driver image, R the edit region and $\hat{x}$ is an edited version of the source x. The goal of the edit $\hat{x}$ is to look natural while being close to the source x everywhere except for the region R, where it should resemble the driver image y.

A main challenge in learning the model $P(\hat{x}|x, y, R)$ is the lack of suitable training data, namely quadruplets $(\hat{x}, x, y, R)$, that exemplify the desired mapping. Most authors have thus proposed to focus on learning an unconditional prior distribution $P(x)$ of images, for which abundant training data is usually available, and then seeking an edit $\hat{x}$ that is both likely according to the prior and close in some sense to the source x and the driver y. This can be achieved in a pre-processing [8,76] or post-processing stage [4], and often uses a Generative Adversarial Network (GAN) to model the prior $P(x)$. Although demonstrating some impressive results, such approaches offer limited control on the edit $\hat{x}$, which either shows only a weak dependency on the driver image y, or does not stay on the image manifold $P(x)$, resulting in undesirable artifacts.

In this work, we overcome these challenges by considering a different approach where we learn the conditional distribution $P(\hat{x}|x, y, R)$ directly, *end-to-end* in code space. For this, we propose new ways of *synthesising* suitable training quadruplets $(\hat{x}, x, y, R)$ on a large scale and without requiring manual intervention. We do this in a self-supervised manner: given an image $\hat{x}$, we select an edit region R at random and use it to decompose the image into source x and driver y images, so that the edit can be written as a (random) function $(x, y, R) = f(\hat{x})$

[1] After all, a picture is worth a thousand words!.

of $\hat{x}$. A shortcoming is that such x and y are statistically correlated, whereas in a "creative" edit process a user must be able to choose y independently of x. A key contribution is to show that, if the process f is carefully designed, then the resulting images x and y are independent *enough*, meaning that they can be used to learn a high-quality conditional generator $P(\hat{x}|x, y, R)$ which works even when x and y are sampled independently.

We pair this intuition with the adoption of state-of-the-art auto-regressive image modelling using transformers for learning and sampling the distribution $P(\hat{x}|x, y, R)$. Overall, our End-to-End Visual Editor (E2EVE) approach has significant advantages over prior image editing works: (1) E2EVE learns to *directly* map the input to the output representations; (2) hence, based on extensive qualitative, quantitative and human-analysis experiments on several datasets, it results in higher quality edits that are simultaneously more dependent on the driver image and more natural looking than prior works based on GANs and attention which train task-agnostic priors not necessarily optimal for editing; (3) it is generally easier to implement and tune than GAN-based alternatives; and (4) it is efficient because it allows the direct sampling of edits without involving the expensive pre- or post-processing steps required by some prior methods.

2 Related Work

Targeted Generative Image Editing. Approaches for editing images in targeted locations include spatial manipulation of objects [77], adding or removing a closed-set of objects [5,6], or text-driven manipulation using CLIP [4,58]. These GAN-based approaches have some downsides: First, they require inverting GANs to represent the input image—a difficult problem [1,2,7,25,38,77] which can limit the *editability* of the images [2,61]. Second, they are not trained end-to-end. Third, text-driven approaches offer limited fine-grained control of shape and texture [4]. We address such shortcomings by training a sequence-to-sequence model end-to-end for the task of targeted image-based visual editing.

Note that the targeted image manipulation task which we address, differs from spatially-conditioned generation [31,37,45,64,68,75], or global image manipulation, where an entire image is generated/manipulated [35,49]. In global manipulation works, interpolations in the latent space are located which correspond to edits over the entire image, either via visual attribute classifiers [10,33,57,71], unsupervised disentanglement [26,47,56,63,67], or via image-text similarity [3,11,22,40,46,55,68]. Different to in-painting [29,39], in our task, generation depends on the driver image as well as image context.

Image Composition. Image composition combines possibly inconsistent images into a single cohesive output. Previous approaches include collaging nearest neighbors [30], composing foregrounds and backgrounds [62], composing a closed set of visual attributes [42,48], or using semantic pyramids [59]. Others fuse images by projecting their composition on the manifold generated by a GAN via inversion [2,8,23,32,70,76]. These methods work well if the driver image is sufficiently aligned to the source (which usually requires manual intervention), but worse than our end-to-end model when this is not the case.

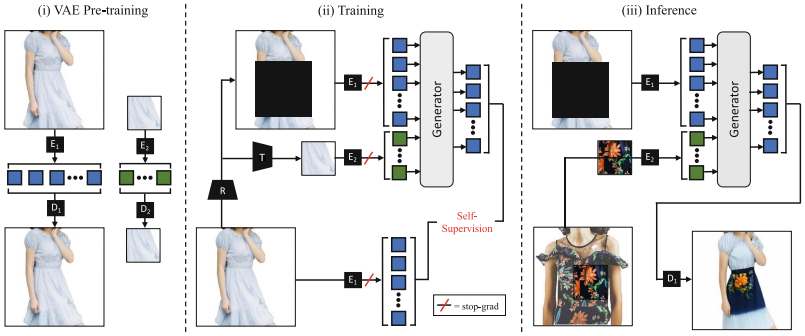

Fig. 2. The E2EVE approach: (i) **VAE Pre-Training**: we train two quantized VAEs (one for whole images, and one for image patches), each consisting of an Encoder, E, and decoder, D. (ii) **Training**: Each data sample produces a masked source image (via the operation R), and driver image (via random transformation T on the masked region). Given these conditioning inputs, the model is self-supervised to predict the data sample. Following prior work, the VAEs are kept frozen while training the generator. (iii) **Inference**: E2EVE generates edited images when the source and driver are sampled independently from different images.

Two Stage Image Synthesis. We adopt state-of-the-art two stage auto-regressive models [9,12,15,18,19,52,69] for the image generator. These models scale better than sequence-to-sequence [14,50] models applied directly to pixels by reducing first the dimensionality of images via a discrete autoencoder [19,20,44,53]. Others have recently built on this work for text- or class-driven image manipulation [11,17,66], whereas we consider image-driven editing [74]. Shows a (single) qualitative result for image-based out-painting using BERT [14], where a small set of tokens in the output sequence is fixed (termed *preservation controls*), and paired training data is sourced from the same image. We do not solve the out-painting problem, but the targeted editing problem, with the added challenge of preserving context while mixing images with very different statistics.

3 Method

We wish to learn a model that can "naturally" blend a given source image x with a user-provided driver image y. Formally, we denote with $x, \hat{x} \in \mathbb{R}^{3\times H\times W}$ the source and output (RGB) images and with $y \in \mathbb{R}^{3\times H'\times W'}$ the driver image (where, usually, $H > H'$ and $W > W'$). Furthermore, we target the edit operation on a region $R \in \{0,1\}^{H\times W}$, expressed as a binary mask. We cast the problem as one of learning a conditional probability distribution $P(\hat{x}|x, y, R)$ and then sample the output image $\hat{x}$ conditioned on the source image x, the driver image y, and the edit region R (Fig. 1).

Next, we discuss the advantages and requirements of this approach (Sect. 3.1), propose a self-supervised learning formulation that does not require any manually-provided labels to train the model (Sect. 3.2), and give the technical details of the neural network that learns the conditional distribution (Sect. 3.3). An overview of our training and inference settings is shown in Fig. 2.

3.1 End-to-End Conditional Generation

Our approach is to learn a conditional distribution $P(\hat{x}|x, y, R)$ *end-to-end*, which we do by means of an auto-regressive transformer network discussed in Sect. 3.3. In order to train such a model, we require training quadruplets $(\hat{x}, x, y, R)$ sampled from the joint distribution $P(\hat{x}, x, y, R)$. Each of these quadruplets represents the outcome of a "creative" process, where a human artist combines images x and y to generate a new image $\hat{x}$. Because obtaining such training data would require the intervention of human artists, it would be very difficult to obtain a sufficiently large dataset to learn the required conditional distribution. Hence, much of the research in image editing focuses on how to avoid this bottleneck and use instead data which is readily available.

A popular approach is to consider an *indirect* formulation and learn instead an unconditional image distribution $P(x)$, for instance expressed as a GAN generator $x = G(z)$. Then, the output image $\hat{x} = G(z^*)$ is "sampled" via an optimization process such as $z^* = \operatorname{argmin}_z d(G(z)|_R, y)$ where $d(\hat{x}|_R, y)$ measures compatibility between the region R of the generated image $\hat{x}$ and the driver image y. The advantage is that the model G can be learned from a collection $\mathcal{X}$ of unedited images $x \sim P(x)$, which is often easy to obtain at scale. The disadvantage is that this model is not optimized for the final task of image editing.

By contrast, in our approach we learn directly the model $P_\theta(\hat{x}|x, y, R)$ minimizing the standard negative log-likelihood loss:

$$\theta^* = \operatorname*{argmin}_\theta \left(-\frac{1}{|\mathcal{T}|} \sum_{(\hat{x},x,y,R)\in\mathcal{T}} \log P_\theta(\hat{x}|x, y, R) \right) \tag{1}$$

where $\mathcal{T}$ is a large collection of training quadruplets. Once learned, we can directly draw samples $\hat{x} \sim P_{\theta^*}(\hat{x}|x, y, R)$. The main challenge is how to obtain the training set $\mathcal{T}$. The key to our method is a way of constructing $\mathcal{T}$ from $\mathcal{X}$ in an automated fashion, at no extra cost. This is explained in the next section.

3.2 Synthesizing a Dataset of Meaningful Edits

Given a training set $\mathcal{X}$ of unedited images x, the goal is to create a dataset $\mathcal{T}$ of "edits" $(\hat{x}, x, y, R)$ consisting of the generated image $\hat{x}$, the source image x, the diver image y, and the edit region R. The difficulty is that these quadruplets should be representative of a "creative" process where the generated image $\hat{x}$ is a meaningful blend of the source and driver images, x and y, according to a human artist. Specifically, $\hat{x}$ should resemble x as much as possible except in the region R, where it should take the character of y, but without introducing unnatural artifacts (e.g., simply pasting y on top of x would not do).

We propose to build such quadruplets as follows (see also Fig. 4). We sample an output image $\hat{x}$ from the unedited collection $\mathcal{X}$, thus pretending that the latter is, in fact, the result of an edit operation. Then, we define the source and driver images for this virtual edit as follows:

$$x = (1 - R) \odot \hat{x}, \qquad y = T(R \odot \hat{x}), \tag{2}$$

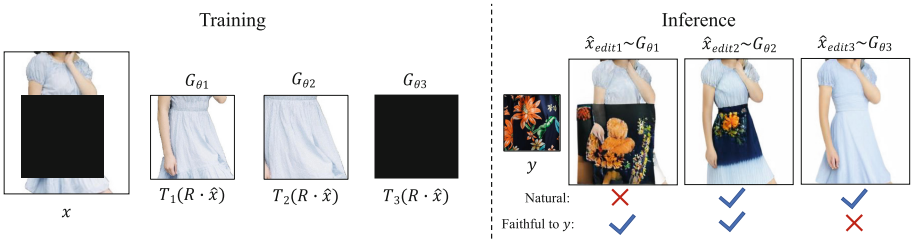

Fig. 3. Intuition for augmenting training inputs. During training, the driver image y is computed by applying a random transformation T to the masked region of the source image x. Left: Three options for T that are used to train three different generators, $G_{\theta n}$. Right: Samples from the three trained models when conditioned on x and an independently sampled y. An optimal choice of T removes just enough information that generated images are both natural-looking, and are faithful to the driver image y.

where R is the mask of a random image region, $\odot$ is the element-wise product (where broadcasting is used as required), and $T : \mathbb{R}^{3\times H\times W} \rightarrow \mathbb{R}^{3\times H'\times W'}$ is a *random image transformation*, also known as an "augmentation".

By optimizing the log-likelihood loss in eq. (1), the model $P_\theta(\hat{x}|x, y, R)$ learns to predict the full image $\hat{x}$ from x (which misses the region R), and y, which preserves some information about the missing region. Because $\hat{x}$ is originally an unedited image, the model learns to predict a natural-looking output.

The key design choice here is the random transformation T. For example, if we set $T = 1$ to be the identity function, then the output image can be reconstructed exactly as $\hat{x} = x + y$; in this case, the model $P_\theta(\hat{x}|x, y, R)$ learns to paste y onto x, which is uninteresting (see model G_{θ_1} in Fig. 3). Inversely, if we set $T = 0$ to be the null function, then y does not provide any information; in this case, the model $P_\theta(\hat{x}|x, y, R)$ learns to inpaint $\hat{x}$, filling in the missing region in a non-trivial manner, but ignoring the driver image y altogether (see model G_{θ_3} in Fig. 3). The augmentations T should find a *sweet spot* and remove just the right amount of information from y (see model G_{θ_2} in Fig. 3). While finding the optimal choice for the random transformations T is ultimately an empirical process, we describe next some important design criteria that were crucial for our results.

Decorrelating Source and Driver Images. A difficulty with our approach is that, because both source image x and driver image y are derived from the same image $\hat{x}$, they are *not* independent but *paired*. This is a problem because the user should be free to choose almost *any* driver image y for editing, so the generator must work well for *unpaired* inputs x and y too. We can approximate this condition by making x and y as uncorrelated as possible during training.

While we cannot achieve this result exactly, we propose two methods to approximate it, *block* and *free-form*, both shown in Fig. 4. **Block edits** are simple: we let the transformation $T(R \odot \hat{x})$ take a further sub-crop R_T of the crop R it is given as input, thus removing the most direct source of correlation between

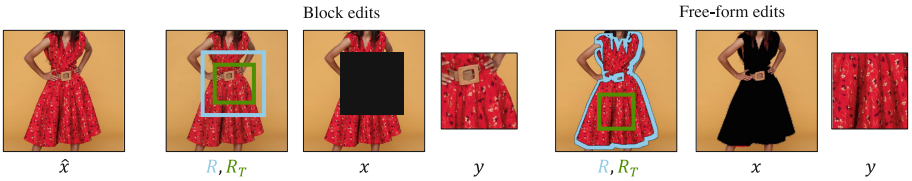

Fig. 4. *Left: block edits.* We sample an output image $\hat{x}$ and generate the corresponding edit input (x, y, R) by sampling a square region R, a transformation T cutting a sub-region R_T from R, and extracting the source image x and a driver y from these two. *Right: free-form edits.* The same approach, but this time R is a pixel-accurate segmentation mask extracted manually or automatically (R_T remains a square sub-region).

x and y: spatial continuity. Furthermore, we found empirically that if the sub-crop is always centered in R and of a fixed relative size w.r.t. R, then the model learns, as one would expect, to paste this crop in the middle. Instead, we let T further randomize the position (`pos_augment`) and size (`size_augment`) of the sub-crop relative to the edit region during training. We parameterize the sub-cropping operation via α, which defines the ratio of the sub-crop width, to the edit region width. The `size_augment` operation allows α to vary during training. The model hence learns to find a meaningful placement for the patch y in the context of x, without assuming spatial continuity or a specific geometric arrangement. Optionally, we further decorrelate source and driver images via **free-form edits**. The difference is that we let R be the output of a semantic segmentation network, while T still takes a square sub-crop R_T from region R. Because y is fully contained in the edit region R and the latter separates a foreground object from the background, this significantly reduces the correlation between x and y. While this approach requires additional machinery (the segmentation network), empirically it can obtain impressive results (Fig. 5).

Controlling the Learned Editor. In order to favour generalization, the augmentations above should remove as much information as possible from the crop y *except* for the information that the editor should transfer from the driver image y to the generated image $\hat{x}$. For example, it would be possible to consider further augmentations such as color jitter, but this would cause the editor to learn to ignore the color, which we usually wish to transfer. In general, by choosing different augmentations we can *control* what information the editor learns to transfer from the driver image to the generated one (e.g., style and colour), and what to ignore (e.g., the specific spatial arrangement).

3.3 Two-Stage Conditional Auto-regressive Image Generation

In order to implement the conditional distribution $P_\theta(\hat{x}|x, y, R)$, we use an *auto-regressive (AR) model.* AR models have been shown to be highly expressive for image generation [19,52], they can be conditioned on multiple signals elegantly and without architectural changes, and, unlike GANs [54], are mode-covering.

In practice, this leads to more varied generation results and the ability to model datasets with more variation. We summarise next how this model is applied to our case and point the reader to the supp. mat. for additional details.

The goal is to model a conditional distribution $P(\hat{x}|c)$, where c lumps together all conditioning information. An AR model further decomposes $\hat{x} = (\hat{x}_1, \ldots, \hat{x}_M)$ into M components and factorizes the distribution as the product $P(\hat{x}|c) = \prod_{m=1}^{M} P(\hat{x}_m|\hat{x}_1, \ldots, \hat{x}_{m-1}, c)$. The model is trained by minimizing the negative log likelihood (1) (avoiding unstable adversarial techniques used in GANs). For modelling images x, a challenge lies in finding a suitable decomposition, such that the individual factors $P(\hat{x}_m|\hat{x}_1, \ldots, \hat{x}_{m-1}, c)$ can be implemented effectively. To this end, we build upon the two stage process of Esser *et al.* [19] and use a transformer on top of a discrete autoencoder. Note that the focus of this work is on end-to-end targeted image editing and the training formulation; we describe the two-stage auto-regressive method for completeness and reproducibility.

Specifically, in the first stage we learn a compressed and discretized representation $z = \Phi(\hat{x}) \in \{1, \ldots, K\}^M$ of the images, where here K is the size of the discrete encoding space and M the resulting number of discrete tokens. For this, we use the VQ-GAN [19], achieving 16-fold image compression (we use separate encoders for x and y). Naturally, the encoder comes with a paired decoder $\hat{x} = \Psi(z)$ that allows to reconstruct the image from the code. This achieves two important goals: it allows (1) to scale the generator model to higher resolution images, and (2) to predict discrete distributions for the second stage.

The second stage uses a transformer to model the factors $P(\hat{x}_m|\hat{x}_1, \ldots, \hat{x}_{m-1}, c)$. Specifically, the conditioning information $c = (x, y, R)$ consists of source image x, driver image y and region R. The sequence of tokens $S_m = (z_1, \ldots, z_m) \oplus \Phi(x) \oplus \Phi(y)$ (where $\oplus$ denotes concatenation), comprising the partially-predicted output tokens along with the conditioning tokens, is fed to the transformer to output a K-dimensional histogram $P(z_m = \cdot|S_m)$. Spatial encodings are added to the tokens, but there is no need to explicitly encode the region R as it can be inferred from x because $x = (1 - R) \odot \hat{x}$ has a R-shaped 'hole'. While this way of encoding for R may seem naïve, it is in fact simple and powerful: prior work such as EdiBERT [32] use "occlusion tokens", and hence lose the ability to express pixel-accurate edit regions, which we can do effortlessly.

In practice, we train a GPT-2 [50] style transformer. Because the factors $P(\hat{x}_m|\hat{x}_1, \ldots, \hat{x}_{m-1}, c)$ require each predicted token to depend only on those prior to it in the sequence, GPT-2 uses causal masking allowing only unidirectional attention towards earlier tokens in the sequence. All factors are trained efficiently in parallel using *teacher forcing* [24,65]. During inference, only the conditioning information is provided so the target sequence is predicted iteratively, sampling one symbol z_m at a time from the corresponding histogram. In practice, inference is faster than for some GAN alternatives, as shown in the sup. mat.

4 Experiments

We compare our method to others that, given a source image x and a driver image y, produce one or more edits $\hat{x}$. Good edits have three properties: (1) *naturalness*

(the edit $\hat{x}$ looks like a sample from the prior $P(x)$); (2) *locality* ($\hat{x}$ is close to x outside the edit region R—although a certain amount of slack is necessary to allow the edit to blend in naturally); and (3) *faithfulness* ($\hat{x}$ resembles y within the edit region R). Achieving only one of the three objectives is trivial (for example, setting $\hat{x} = x$ ignoring y is natural and local but unfaithful whereas copying y on top of x is local and faithful but unnatural) so a good model must seek for a trade-off between these properties.

Measuring these properties is not entirely trivial; for the quantitative analysis, we take the standard FID measure for naturalness [27], the L^1 distance $\|(1 - R) \odot (\hat{x} - x)\|_1$ to measure locality, and a retrieval approach to measure faithfulness. For the latter, we consider a set $\mathcal{Y}_{\text{dstr}}$ of 100 distractor images of the same size as the driver image y, use the edited region $\hat{x}|_R$ as a query, and find its nearest neighbour $y^* = \text{argmin}_{\hat{y} \in \{y\} \cup \mathcal{Y}_{\text{dstr}}} d(\hat{x}|_R, \hat{y})$, incurring the loss $\delta_{y^* \neq y}$. In this expression, $d(\cdot, \cdot)$ is the Inception v3 [60] feature distance (pre-trained on ImageNet [13]), which is the same encoder used for FID calculation.

Evaluation Data. Recall that our goal is to evaluate the quality of automated editing algorithms. To feed such algorithms, we need triplets (x, y, R) consisting of a source image x, a driver image y and an edit region R. In Sect. 3.2 we explained how to build such a dataset for the purpose of *training* our model—with the added complexity that, for training, we *also* need to know the result $\hat{x}$ of the edit process. We could use the same dataset for evaluation, but this would unfairly advantage our model. Instead, since knowing the output $\hat{x}$ is *not* required to measure naturalness, locality and faithfulness, we are free to choose *new* and less constrained triplets (x, y, R) for evaluation, resulting in more challenging edits and a fairer evaluation. However, we wish to avoid too many cases in which cohesive blending is impossible (*e.g.*, where y is a patch of sky and $x|_R$ is a face). Hence, we build evaluation triplets as follows: given a sample image x and an edit region R, we define $y = x'|_R$ to be a crop taken at the same spatial location from a *different* image x' in the dataset. The effect is to (very) weakly constrain $x|_R$ and y to be compatible (e.g., both sky regions, or face regions) by exploiting the photographer bias in the datasets we consider.

We conduct experiments on three datasets: (1) the private *Dresses-7M* dataset containing 7 million images mainly depicting a woman wearing a dress; (2) LSUN bedrooms [72] containing 3M images of bedrooms; and (3) FFHQ [35], containing 70K aligned faces. We sample x by considering 1024 images from the validation sets of *Dresses-7m* and FFHQ, and 256 for LSUN bedrooms (due to its small size). For each (x, y, R), we consider 10 edit samples and obtain naturalness, locality and faithfulness by averaging over all images thus generated (totalling 10,240 and 2,560 samples, respectively). All images in this paper are from UnSplash[2] (dresses and bedrooms), or DFDC [16] (faces).

Implementation Details. We use a transformer architecture with 24 layers, 16 head multi-head attention, embedding size 1024, and we train it using standard

[2] www.unsplash.com.

cross-entropy loss. The masked source image $(1-R)\odot x$ and the driver image y are encoded using VQ-GANs with 16× compression and 1024 codebook size. Source x and output $\hat{x}$ images have resolution 256×256 and y resolution 64×64. The token sequences S_m (comprising the coded x, y and partial $\hat{x}$) has maximum length 516. Our final model uses `pos_augment` and `size_augment`—the latter forms y by taking a sub-crop in the region R, with α varying from 0.4 to 0.7. We use a batch size of 512, and the AdamW [41] optimizer with learning rate 4.5e-6. During inference, for each input (x, y, R), we generate 20 samples $\hat{x}$ and keep the 10 with highest similarity to y (a method termed *Filter*, in Table 2). In order to focus the sampling on more realistic/likely outputs we use *nucleus sampling* [28] with a p-value of 0.9. Following prior work [19,52], the VQ-GANs are kept frozen when training the transformer.

Baselines. We compare our method against the following image composition baselines: (1) *Copy-paste* generates $\hat{x}$ by pasting y onto x at the specified location R; (2) *Inpaint* ablates our method by removing the tokens y from the input, thus generating $\hat{x}$ by inpainting the region R unconditionally while disregarding y; (3) *GAN inv*, inspired by [32,70,76], takes the copy-paste output and uses the StyleGANv2 [36] or StyleGANv2-ADA [34] networks to re-encode and thus denoise the resulting image via GAN inversion [2], "blending" the edit; (4) *EdiBERT* [32] is a related transformer-based approach, which iteratively refines the output of copy-paste output using BERT [14] (for fairness, we use the same VQ-GAN and sample filtering by similarity to driver as for our method); (5) *In-Domain GAN* [76] uses a regularised form of GAN inversion to blend source and driver images. Pre-trained models are available for all test datasets except *Dresses-7M*; unfortunately, we were unable to successfully train the GAN-based models on the latter (possibly due to the significant diversity of this data), so in this case we limit the other baselines. Some off-the-shelf models are trained on the validation sets that we use for testing, which disadvantages our approach in the comparison. For more details on experimental settings, please see supp. mat.

4.1 Quantitative Evaluation

Block Edits. Table 1 reports the evaluation metrics for all baselines and datasets. Our approach significantly outperforms others in **naturalness**: because our method is trained explicitly with the goal of blending source and driver images, it works even for cases where the images are poorly aligned, where prior works based on fitting priors on unedited images fail (see Fig. 5).

The copy-paste baseline outperforms other methods on the **faithfulness** metric but has very poor naturalness—this is expected as the edited image contains a 1-to-1 copy of the driver image. The opposite is true for the inpainting baselines, which attain good naturalness but very poor faithfulness as they ignore the driver image altogether. Our method is second only to copy-paste in faithfulness while also scoring best in naturalness. Other baselines sit somewhere in between, but generally do not fair very well in faithfulness because, by projecting the composite image to the prior manifold, they distort the cue too much.

Table 1. Results for *block-* and *free-form edits*. Naturalness is computed over the whole image (Image), and just the edit-region (Edit-R) using FID. Faithfulness is computed via retrieval, where R@K measures whether or not the sample is retrieved in the top-k instances. Locality is measured using the L^1 distance outside of the edit-region

		Naturalness (↓)		Faithfulness (↑)			Locality (↓)
		Image	Edit-R	R@1	R@5	R@20	(L1)
Dresses-7m (*block-edits*)	Baseline: Copy-Paste	21.457	35.924	1.000	1.000	1.000	0.000
	Baseline: Inpaint	15.797	25.769	0.071	0.214	0.515	0.095
	EdiBERT [32]	17.193	32.621	0.554	0.837	0.963	**0.052**
	(ours) E2EVE	**14.411**	**24.743**	**0.797**	**0.937**	**0.978**	0.056
FFHQ (*block-edits*)	Baseline: Copy-Paste	33.330	25.811	1.000	1.000	1.000	0.000
	Baseline: Inpaint	18.328	12.665	0.421	0.704	0.895	0.139
	GAN inv [2]: StyleGANv2	26.583	16.223	0.590	0.823	0.948	0.198
	GAN inv [2]: StyleGANv2-Ada	26.657	16.290	0.593	0.821	0.949	0.199
	In-domain [76]	19.880	14.270	0.539	0.800	0.938	0.178
	EdiBERT [32]	13.192	12.230	0.718	0.925	0.983	**0.093**
	(ours) E2EVE	**12.770**	**10.574**	**0.853**	**0.970**	**0.994**	0.106
LSUN Bedrooms (*block-edits*)	Baseline: Copy-Paste	24.402	28.828	1.000	1.000	1.000	0.000
	Baseline: Inpaint	15.080	21.493	0.113	0.297	0.596	0.161
	GAN inv [2]: StyleGANv2	23.735	33.530	0.405	0.689	0.866	0.259
	In-domain [76]	32.333	43.544	0.171	0.363	0.608	0.208
	EdiBERT [32]	16.518	27.528	0.537	0.816	0.946	**0.111**
	(ours) E2EVE	**14.107**	**22.187**	**0.789**	**0.923**	**0.981**	0.119
Dresses-7m (*free-form edits*)	Baseline: Copy-Paste	23.107	58.259	0.581	0.700	0.817	0.000
	Baseline: Inpaint	13.718	24.516	0.193	0.385	0.659	0.103
	EdiBERT [32]	15.277	27.359	0.650	0.843	0.937	0.079
	(ours) E2EVE	**14.000**	**25.973**	**0.814**	**0.920**	**0.951**	**0.072**

As for **locality**, copy-paste is also optimal, as it does not change the context region at all. Compared to non-trivial baselines, our method is first or second best in this metric, affecting the context region much less than GAN methods. However, EdiBERT is also very competitive as it is designed to leave the context nearly exactly unchanged (via *periodic collage* of the output with the context). However, relaxing locality is often necessary to obtain a more reasonable blending effect—an intuitive fact that we show qualitatively in Fig. 5f.

Finally, we also conduct a **human-study** on the *Dresses-7M* dataset using Amazon Mechanical Turk. We showed 256 edit samples using our method and EdiBERT to 3 human assessors each, asking two questions: which of the two outputs is more realistic, and which is more faithful to the driver image. The results show that by majority vote, human annotators think that our samples are more natural 83.2% of the time, and more faithful 80.5% of the time. The key to the performance improvements are that E2EVE is trained *end-to-end* for targeted visual editing, whereas prior work are not. Furthermore, we outperform EdiBERT, which uses the same VQ-GAN with a 50% larger transformer.

Free-Form Edits. Here, we use a semantic segmentation network to extract the edit region R from image x, while we define y as a crop taken from within a semantic region of a different image x' (see Sect. 3.2) We conduct experiments on *Dresses-7M* and compare to EdiBERT, with results in Table 1. Our approach again outperforms the previous methods in terms of **naturalness** and **faithfulness** and, this time, **locality** too, because we can better capture irregular

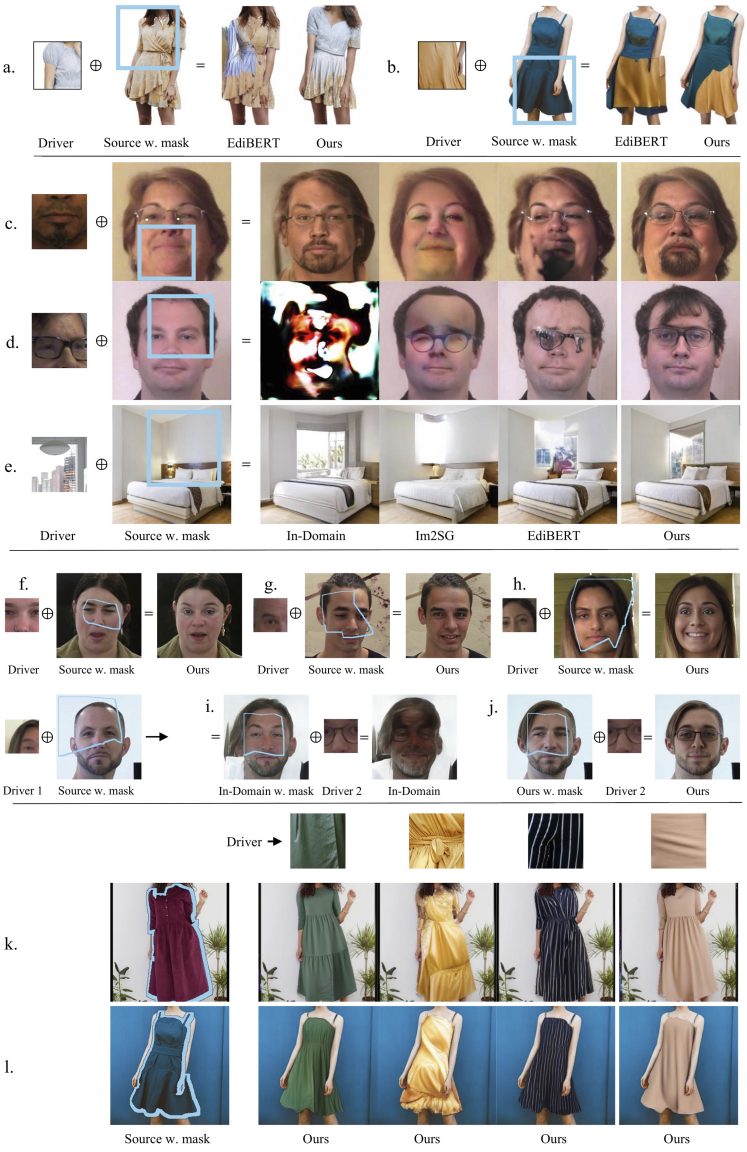

Fig. 5. Qualitative results from E2EVE. Rows a, b: Block edits from E2EVE trained on *Dresses-7M*. Row c, d: Comparisons to prior work trained on FFHQ. Row e: Comparisons to prior work trained on LSUN-Bedrooms. Rows f, g, h: Edits from E2EVE on FFHQ using random masks of increasing size. Rows i, j: one edit after another (sequential) on the same source image using prior work, and E2EVE, respectively. Rows k, l: *free-form* edits from E2EVE trained on *Dresses-7M*. Please zoom in for details. In each case, the masked region in the source image is that contained within the blue line. (Color figure online)

edit regions compared to EdiBERT which results in blocky artifacts. Additional experiments with random free-form masks can be found in the supp. mat.

Fig. 6. E2EVE generalises surprisingly well to out-of-domain driver images (*e.g.* images of weather and nature). Three examples using the same source image from *block-edit Dresses-7M* model. As shown, E2EVE generates cohesive and varied samples.

4.2 Qualitative Evaluation

We show qualitative comparisons against prior work in Fig. 5. Our edits combine naturalness, faithfulness and locality, whereas others fail at achieving all three goals as well as we do. Due to the augmentations in our training edits, our method is better able to cope with uncorrelated driver images y than other approaches that only rely on a pre-learned unconditional prior distribution $P(x)$. For example, in Fig. 5c, d our approach can successfully mix images coming from faces with different gender or pose, showing better naturalness and faithfulness. As for locality, while EdiBERT is highly competitive in Table 1, this comes at a cost: in Fig. 5a, d our method achieves better naturalness by coloring both sleeves in the same way and by completing the glasses even though part of them lie outside of the edit region, whereas EdiBERT cannot. Fig. 5k, l shows free-form edits where the entire clothing item is masked. Although structural details of the dress are hidden by the mask, E2EVE generates natural and varied structure that is different to the source and faithful to the driver image. In Fig. 5e E2EVE generates more natural looking samples than prior work that edit with respect to the spatial geometry of the room. We also see that E2EVE generalises surprisingly well to out-of-domain driver images, as shown in Fig. 6 for the block-edit *Dresses-7M* model. In Fig. 5f, g, h, E2EVE generates impressive samples with small, medium or large random free-form masks. In Fig. 5j E2EVE makes sequential edits with random Rs while maintaining naturalness – empirically this fails with prior works (Fig. 5i). See the supp. mat. for additional results.

4.3 Ablations

In Table 2 we analyse and ablate design choices in E2EVE. We report additional metrics: negative log likelihood (NLL) on the validation set and sample diversity, computed pairwise between samples from the same inputs using LPIPS [73].

Table 2. Model ablations and sweeps for *block edits*. Key: NLL: Negative log likelihood. α, pos-aug, size-aug: the parameters used to define the sub-cropping transformation T. Filter: filtering E2EVE samples by visual similarity to the driver image. 2VQ: using two VQ-GANs rather than one. Datasets: D: *Dresses-7m*, B: Bedrooms, F-FFHQ.

							Naturalness (↓)		Faithfulness (↑)			Locality (↓)	NLL (↓)	Diversity (↑)	
	α	pos-aug	size-aug	Filter	2VQ	Data	Image	Edit-R	R@1	R@5	R@20	(L1)		Image	Edit-R
a.	0.8	✗	✗	✓	✓	D	17.241	30.076	0.882	0.980	0.996	0.056	2.181	0.135	0.309
b.	0.6	✗	✗	✓	✓	D	15.593	29.364	0.920	0.986	0.997	0.056	1.704	0.137	0.315
c.	0.4	✗	✗	✓	✓	D	13.967	26.975	0.811	0.954	0.988	0.056	1.594	0.139	0.327
d.	0.0	✗	✗	✓	✓	D	15.797	25.769	0.071	0.214	0.515	0.095	1.537	0.190	0.419
e.	0.6	✓	✗	✓	✓	D	15.605	26.513	0.887	0.980	0.997	0.056	1.518	0.142	0.338
f.	0.5-0.6	✓	✓	✓	✓	D	14.951	26.186	0.856	0.968	0.992	0.056	1.460	0.143	0.344
g.	0.4-0.7	✗	✓	✓	✓	D	14.589	27.824	0.864	0.970	0.992	0.056	1.494	0.139	0.328
h.	0.4-0.7	✓	✓	✓	✓	D	14.411	24.743	0.797	0.937	0.960	0.056	1.448	0.143	0.344
i.	0.4-0.7	✓	✓	✗	✓	D	13.913	24.583	0.611	0.817	0.929	0.056	1.448	0.145	0.351
j.	0.4-0.7	✓	✓	✗	✓	B	14.347	22.998	0.636	0.831	0.929	0.119	2.942	0.287	0.460
k.	0.4-0.7	✓	✓	✗	✓	F	12.636	10.699	0.723	0.899	0.976	0.106	2.392	0.203	0.321
l.	EdiBERT [32]			✗		B	16.643	29.775	0.356	0.627	0.823	0.111	-	0.291	0.575
m.	EdiBERT [32]			✗		F	13.036	12.891	0.536	0.778	0.925	0.093	-	0.181	0.423
n.	0.4-0.7	✓	✓	✓	✗	D	14.107	23.916	0.720	0.891	0.963	0.056	1.454	0.144	0.347

Starting from the construction of the training edits $\mathcal{T}$ (Sect. 3.2), reducing α (rows a–d) means removing more of the image $\hat{x}$ from the crop y. As predicted in Sect. 3.2, removing information from y increases naturalness (lower FID) at the expense of weaker faithfulness (lower R). $\alpha = 0.6$ provides a balance. `pos_augment` (row e vs. b) increases naturalness by preventing the model from simply pasting the driver image in the centre of the edit region. `size_augment` (row h) randomizes the choice of α in a range during training, so that the editor learns to automatically resize the driver image as needed. This significantly improves naturalness at the cost of a reduction of faithfulness (row h vs. e). In part, this is likely due to limitations of the retrieval model used to measure faithfulness, which struggles to cope with geometric deformations even when they preserve the style of the driver. α and augmentation have no effect on the locality (rows a–h). A benefit is that our final model generates more diverse samples (row h vs g) by learning to place the driver image at different positions and sizes. Interestingly, NLL is also minimised by the final model despite the fact that y is *less* correlated to $\hat{x}$ than in other cases: this is likely because additional augmentations reduce overfitting to the training data. For further discussion, please see supp. mat.

5 Conclusions, Limitations and Future Work

We present E2EVE, a new approach for targeted visual image editing. The key innovation is an effective method for self-supervising the model end-to-end, based on only an unlabelled collection of natural images. E2EVE learns a conditional image generator that responds well to diverse user inputs, significantly outperforming prior work qualitatively and quantitatively without any manual supervision. Limitations remain: our data generation technique might be difficult to extend to text-based edits and some proposed edits are unreasonable

(see sup. mat.) because the model lacks a full understanding of the semantic content of images. Furthermore, as our model is unsupervised and data-driven, it may contain surprising unwanted biases. Next steps include extending E2EVE beyond images to spatial editing in 3D scenes and spatio-temporal video editing.

Acknowledgements. We are grateful to the advice and support of Yanping Xie, Antoine Toisoul, Thomas Hayes, and the EdiBERT authors.

References

1. Abdal, R., Qin, Y., Wonka, P.: Image2stylegan: How to embed images into the stylegan latent space? In: ICCV (2019)
2. Abdal, R., Qin, Y., Wonka, P.: Image2stylegan++: how to edit the embedded images? In: CVPR (2020)
3. Abdal, R., Zhu, P., Femiani, J., Mitra, N.J., Wonka, P.: Clip2stylegan: unsupervised extraction of stylegan edit directions. arXiv:2112.05219 [cs.CV] (2021)
4. Bau, D., et al.: Paint by word. arXiv:2103.10951 [cs.CV] (2021)
5. Bau, D., et al.: Semantic photo manipulation with a generative image prior. ACM Trans, Graph (2019)
6. Bau, D., Zhu, J.Y., Strobelt, H., Lapedriza, A., Zhou, B., Torralba, A.: Understanding the role of individual units in a deep neural network. In: Proceedings of the National Academy of Sciences (2020)
7. Bau, D., et al.: Inverting layers of a large generator. In: ICLR 2019 Debugging Machine Learning Models Workshop (2019)
8. Chai, L., Wulff, J., Isola, P.: Using latent space regression to analyze and leverage compositionality in gans. In: ICLR (2021)
9. Chen, M., et al.: Generative pretraining from pixels. In: ICML (2020)
10. Choi, Y., Choi, M., Kim, M., Ha, J.W., Kim, S., Choo, J.: Stargan: unified generative adversarial networks for multi-domain image-to-image translation. In: CVPR (2018)
11. Crowson, K.: VQGAN-CLIP (2021). https://github.com/nerdyrodent/VQGAN-CLIP
12. Dai, B., Wipf, D.: Diagnosing and enhancing VAE models. In: ICLR (2019)
13. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: a large-scale hierarchical image database. In: CVPR (2009)
14. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. In: NAACL (2019)
15. Ding, M., et al.: Cogview: mastering text-to-image generation via transformers. In: NeurIPS (2021)
16. Dolhansky, B., et al.: The deepfake detection challenge dataset (2020)
17. Esser, P., Rombach, R., Blattmann, A., Ommer, B.: Imagebart: bidirectional context with multinomial diffusion for autoregressive image synthesis. In: NeurIPS (2021)
18. Esser, P., Rombach, R., Ommer, B.: A disentangling invertible interpretation network for explaining latent representations. In: CVPR (2020)
19. Esser, P., Rombach, R., Ommer, B.: Taming transformers for high-resolution image synthesis. In: CVPR (2021)
20. Fauw, J.D., Dieleman, S., Simonyan, K.: Hierarchical autoregressive image models with auxiliary decoders. arXiv:1903.04933 [cs.CV] (2019)

21. Gafni, O., Polyak, A., Ashual, O., Sheynin, S., Parikh, D., Taigman, Y.: Make-a-scene: scene-based text-to-image generation with human priors (2022)
22. Galatolo., F., Cimino., M., Vaglini, G.: Generating images from caption and vice versa via clip-guided generative latent space search. In: Proceedings of the International Conference on Image Processing and Vision Engineering (2021)
23. Ghosh, P., Zietlow, D., Black, M.J., Davis, L.S., Hu, X.: Invgan: invertible gans. arXiv:2112.04598 [cs.CV] (2021)
24. Goyal, A., Lamb, A., Zhang, Y., Zhang, S., Courville, A., Bengio, Y.: Professor forcing: a new algorithm for training recurrent networks. In: NeurIPS (2016)
25. Guan, S., Tai, Y., Ni, B., Zhu, F., Huang, F., Yang, X.: Collaborative learning for faster stylegan embedding. arXiv:2007.01758 [cs.CV] (2020)
26. Härkönen, E., Hertzmann, A., Lehtinen, J., Paris, S.: Ganspace: discovering interpretable gan controls. arXiv:2004.02546 [cs.CV] (2020)
27. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: NeurIPS (2017)
28. Holtzman, A., Buys, J., Du, L., Forbes, M., Choi, Y.: The curious case of neural text degeneration. In: ICLR (2020)
29. Iizuka, S., Simo-Serra, E., Ishikawa, H.: Globally and locally consistent image completion. ACM Trans. Graph. **36**(4), 1–14 (2017)
30. Isola, P., Liu, C.: Scene collaging: analysis and synthesis of natural images with semantic layers. In: ICCV (2013)
31. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: CVPR (2017)
32. Issenhuth, T., Tanielian, U., Mary, J., Picard, D.: Edibert, a generative model for image editing. arXiv:2111.15264 [cs.CV] (2021)
33. Jahanian, A., Chai, L., Isola, P.: On the "steerability" of generative adversarial networks. In: ICLR (2020)
34. Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., Aila, T.: Training generative adversarial networks with limited data. In: NeurIPS (2020)
35. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: CVPR (2019)
36. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of StyleGAN. In: CVPR (2020)
37. Kim, H., Choi, Y., Kim, J., Yoo, S., Uh, Y.: Exploiting spatial dimensions of latent in gan for real-time image editing. In: CVPR (2021)
38. Lipton, Z.C., Tripathi, S.: Precise recovery of latent vectors from generative adversarial networks. arXiv:1702.04782 [cs.LG] (2017)
39. Liu, G., Reda, F.A., Shih, K.J., Wang, T.C., Tao, A., Catanzaro, B.: Image inpainting for irregular holes using partial convolutions. In: ECCV (2018)
40. Liu, X., et al.: More control for free! image synthesis with semantic diffusion guidance. arXiv:2112.05744 [cs.CV] (2021)
41. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR (2019)
42. Mokady, R., Benaim, S., Wolf, L., Bermano, A.: Mask based unsupervised content transfer. arXiv:1906.06558 [cs.CV] (2018)
43. Nichol, A., et al.: Glide: towards photorealistic image generation and editing with text-guided diffusion models (2021)
44. van den Oord, A., Vinyals, O., Kavukcuoglu, K.: Neural discrete representation learning. arXiv:1711.00937 [cs.LG] (2017)
45. Park, T., Liu, M.Y., Wang, T.C., Zhu, J.Y.: Semantic image synthesis with spatially-adaptive normalization. In: CVPR (2019)

46. Patashnik, O., Wu, Z., Shechtman, E., Cohen-Or, D., Lischinski, D.: Styleclip: Text-driven manipulation of stylegan imagery. arXiv:2103.17249 [cs.CV] (2021)
47. Peebles, W., Peebles, J., Zhu, J.-Y., Efros, A., Torralba, A.: The hessian penalty: a weak prior for unsupervised disentanglement. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12351, pp. 581–597. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58539-6_35
48. Press, O., Galanti, T., Benaim, S., Wolf, L.: Emerging disentanglement in auto-encoder based unsupervised image content transfer. In: ICLR (2019)
49. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv:1511.06434 [cs.LG] (2016)
50. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
51. Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with clip latents (2022)
52. Ramesh, A., et al.: Zero-shot text-to-image generation. In: ICML (2021)
53. Razavi, A., van den Oord, A., Vinyals, O.: Generating diverse high-fidelity images with vq-vae-2. In: NeurIPS (2019)
54. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. arXiv:1606.03498 [cs.LG] (2016)
55. Schaldenbrand, P., Liu, Z., Oh, J.: Styleclipdraw: Coupling content and style in text-to-drawing synthesis. arXiv:2111.03133 [cs.CV] (2021)
56. Schwettmann, S., Hernandez, E., Bau, D., Klein, S., Andreas, J., Torralba, A.: Toward a visual concept vocabulary for gan latent space. In: ICCV (2021)
57. Shen, Y., Gu, J., Tang, X., Zhou, B.: Interpreting the latent space of gans for semantic face editing. In: CVPR (2020)
58. Shi, J., Xu, N., Zheng, H., Smith, A., Luo, J., Xu, C.: Spaceedit: learning a unified editing space for open-domain image editing. arXiv:2112.00180 [cs.CV] (2021)
59. Shocher, A. et al.: Semantic pyramid for image generation. In: CVPR (2020)
60. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: CVPR (2016)
61. Tov, O., Alaluf, Y., Nitzan, Y., Patashnik, O., Cohen-Or, D.: Designing an encoder for stylegan image manipulation. arXiv:2102.02766 [cs.CV] (2021)
62. Tsai, Y.H., Shen, X., Lin, Z.L., Sunkavalli, K., Lu, X., Yang, M.H.: Deep image harmonization. In: CVPR (2017)
63. Voynov, A., Babenko, A.: Unsupervised discovery of interpretable directions in the gan latent space. In: ICML (2020)
64. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-resolution image synthesis and semantic manipulation with conditional gans. In: CVPR (2018)
65. Williams, R.J., Zipser, D.: A learning algorithm for continually running fully recurrent neural networks. Neural Comput. **1**(2), 270–280 (1989)
66. Wu, C., et al.: N\" uwa: visual synthesis pre-training for neural visual world creation. arXiv:2111.12417 [cs.CV] (2021)
67. Wu, Z., Lischinski, D., Shechtman, E.: Stylespace analysis: Disentangled controls for stylegan image generation. arXiv:2011.12799 [cs.CV] (2020)
68. Xia, W., Yang, Y., Xue, J.H., Wu, B.: Tedigan: text-guided diverse face image generation and manipulation. In: CVPR (2021)
69. Xiao, Z., Yan, Q., Chen, Y.A., Amit, Y.: Generative latent flow. arXiv:1905.10485 [cs.CV] (2019)
70. Xu, Y., Shen, Y., Zhu, J., Yang, C., Zhou, B.: Generative hierarchical features from synthesizing images. In: CVPR (2021)

71. Yang, C., Shen, Y., Zhou, B.: Semantic hierarchy emerges in deep generative representations for scene synthesis. Int. J. Comput. Vis. **129**(5), 1451–1466 (2021). https://doi.org/10.1007/s11263-020-01429-5
72. Yu, F., Zhang, Y., Song, S., Seff, A., Xiao, J.: Lsun: construction of a large-scale image dataset using deep learning with humans in the loop. arXiv:1506.03365 [cs.CV] (2015)
73. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018)
74. Zhang, Z., et al.: UFC-BERT: unifying multi-modal controls for conditional image synthesis. In: NeurIPS (2021)
75. Zhao, S., et al.: Large scale image completion via co-modulated generative adversarial networks. In: ICLR (2021)
76. Zhu, J., Shen, Y., Zhao, D., Zhou, B.: In-domain gan inversion for real image editing. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12362, pp. 592–608. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58520-4_35
77. Zhu, J.-Y., Krähenbühl, P., Shechtman, E., Efros, A.A.: Generative visual manipulation on the natural image manifold. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9909, pp. 597–613. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46454-1_36