VISUALIZING AND UNDERSTANDING GENERATIVE ADVERSARIAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative Adversarial Networks (GANs) have recently achieved impressive results for many real-world applications. As an active research topic, many GAN variants have emerged with immprovements in sample quality and training stability. However, visualization and understanding of GANs is largely missing. How does a GAN represent our visual world internally? What causes the artifacts in GAN results? How do architectural choices affect GAN learning? Answering such questions could enable us to develop new insights and better models.

In this work, we present an analytic framework to visualize and understand GANs at the unit-, object-, and scene-level. We first identify a group of interpretable units that are closely related to object concepts with a segmentation-based network dissection method. Then, we quantify the causal effect of interpretable units by measuring the ability of interventions to control objects in the output. Finally, we examine the contextual relationship between these units and their surrounding by inserting the discovered object concepts into new images. We show several practical applications enabled by our framework, from comparing internal representations across different layers, models, and datasets, to improving GANs by locating and removing "artifacts" units, to interactively manipulating objects in the scene. We will open source our interactive online tools to help researchers and practitioners better understand their models.

1 INTRODUCTION

Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) have been able to produce photorealistic images, often indistinguishable from real images (Figure 1a). This remarkable ability has powered many real-world applications ranging from visual recognition (Wang et al., 2017), to image manipulation (Isola et al., 2017; Zhu et al., 2017), to video prediction (Mathieu et al., 2016). Since its invention in 2014, researchers have proposed many GAN variants (Radford et al., 2016; Zhang et al., 2018), often producing more realistic and diverse samples with better training stability.

Despite this tremendous success, many questions remain to be answered. For example, to produce a church image (Figure 1a), what knowledge does a GAN need to learn? Alternatively, why does a GAN sometimes produce terribly unrealistic images (Figure 1f)? What causes the mistakes? Why does one GAN variant work better than another? What differences are encoded in their weights?

In this work, we investigate the internal representations of GANs. To a human observer, a well-trained GAN appears to have learned facts about the objects in the image: for example a door can appear on a building but not on a tree. We wish to understand how a GAN represents such structure. Do the objects emerge as pure pixel patterns without any explicit representation of objects such as doors and trees, or does the GAN contain internal variables that correspond to the objects that humans perceive? If the GAN does contain variables for doors and trees, do those variables cause the generation of those objects, or do they merely correlate? How are relationships between objects represented?

We present a general method for visualizing and understanding GANs at different levels of abstraction, from each neuron, to each object, to the contextual relationship between different objects. We first identify a group of interpretable units that are related to object concepts (Figure 1b). These units' featuremap closely matches the semantic segmentation of a particular object class (e.g., trees). Second, we directly intervene within the network to identify sets of units that cause a type of object

Figure 1: Overview: (a) A number of realistic outdoor church images generated by Progressive GANs (Karras et al., 2018). (b) Given a pre-trained GAN model (e.g., Progressive GANs), we first identify a set of interpretable units, whose featuremap is highly correlated to the region of an object class across different images. For example, unit #157 can localize tree regions with diverse visual appearance. (c) We ablate the units by forcing the activation to be zero and quantify the average casual effect of the ablation. Here we successfully remove these trees from churches. (d) We can insert these tree causal units to other locations in the generated images. The same set of units can synthesize different trees visually compatible with their surrounding context. In addition, our method can diagnose and improve GANs by identifying artifact-causing units (e). We can remove the artifacts that appear in (f) and significantly improve the results by ablating the "artifacts" units (g). Please see our video for more results.

to disappear (Figure 1c) or appear (Figure 1d). We quantify the causal effect of these units using a standard causality metric. Finally, we examine the contextual relationship between these causal object units and the background. We study where we can insert the object concepts in new images and how this intervention interacts with other objects in the image (Figure 1d). To our knowledge, our work provides the first systematic analysis for understanding the internal representations of GANs.

Finally, we show several practical applications enabled by this new analytic framework, from comparing internal representations across different layers, GAN variants, and datasets; to debugging and improving GANs by locating and ablating "artifact" units (Figure 1e); to understanding contextual relationships between objects in natural scenes; to manipulating images with interactive object-level control. Our interactive tools are demonstrated in a video. We will release our code and data as well as our web-based interactive interface upon publication to help researchers and practitioners better understand and develop their own models.

2 Related Work

Generative Adversarial Networks. GANs (Goodfellow et al., 2014) have kept improving the quality and diversity of results, from generating simple digits and faces (Goodfellow et al., 2014), to synthesizing natural scene images (Radford et al., 2016; Denton et al., 2015), to generating 1k photorealistic portraits (Karras et al., 2018), to producing one thousand object classes (Miyato et al., 2018; Zhang et al., 2018). In addition to image generation, GANs have also enabled many applications such as visual recognition (Wang et al., 2017; Hoffman et al., 2018), image manipulation (Isola et al., 2017; Zhu et al., 2017), and video generation (Mathieu et al., 2016; Wang et al., 2018). Despite the huge success, little work has been done to visualize what GANs have learned. Prior

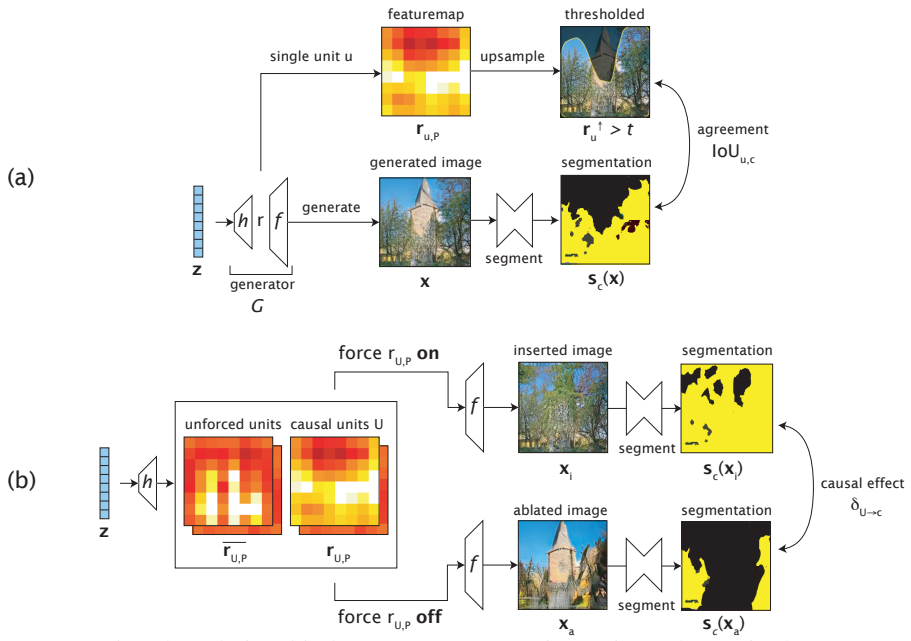

Figure 2: Measuring the relationship between representation units and trees in the output using (a) dissection and (b) intervention. Dissection measures agreement between a unit u and a concept c by comparing a semantic segmentation of the generated image $s_c(x)$ with the thresholded upsampled unit. Intervention measures the causal effect of a set of units U on a concept c by comparing the effect of forcing the unit on (unit insertion) and off (unit ablation). In this case, the segmentation s_c reveals that trees in the generated image increase after insertion and decrease after ablation. The average difference in the tree pixels measures the average causal effect.

work (Radford et al., 2016; Zhu et al., 2016) manipulates latent vectors and analyzes how the results change accordingly. However, none of the methods investigate the internal representations.

Visualizing Deep Neural Networks. To understand the the representation learned by the networks, various methods have been developed to visualize their internal weights, such as the visualization for CNNs (Zeiler & Fergus, 2014) and RNNs (Karpathy et al., 2016; Strobelt et al., 2018). We can visualize a CNN by locating and reconstructing salient image features (Simonyan et al., 2014; Mahendran & Vedaldi, 2015) or by mining patches that maximize hidden layers' activations (Zeiler & Fergus, 2014), or we can synthsize the input images to invert a feature layer (Dosovitskiy & Brox, 2016). Alternately, we can identify the semantics of each unit (Zhou et al., 2015; Bau et al., 2017) by measuring agreement between unit activation and object segmentation masks. Visualization of an RNN has also revealed interpretable units, which keep track of long-range dependencies (Karpathy et al., 2016). Most previous work on network visualization has focused on networks trained for classification; our work explores deep generative models trained for image generation.

Explaining the Decisions of Deep Neural Networks. We can explain individual network decisions using informative heatmaps (Zhou et al., 2016; Selvaraju et al., 2017), or through modified back-propagation (Simonyan et al., 2014; Bach et al., 2015; Sundararajan et al., 2017). The heatmaps highlight which regions contribute most to the categorical prediction given by the networks. Morcos et al. (2018) examined the effect of individual units by ablating them. Recent work has also studied the contribution of feature vectors (Kim et al., 2017; Zhou et al., 2018) or individual channels (Olah et al., 2018) to the final prediction. Those methods only work for explaining a classification network. Our method aims at explaining how a photo-realistic image can be generated by the network, which is much less explored.

3 Method

Our goal is to analyze how objects such as trees are encoded by the internal representations of a GAN generator $G\colon \mathbf{z} \to \mathbf{x}$. Here $\mathbf{z} \in \mathbb{R}^{|z|}$ denotes a latent vector sampled from a low-dimensional distribution and $\mathbf{x} \in \mathbb{R}^{H \times W \times 3}$ denotes an $H \times W$ generated image. We use *representation* to describe the tensor $\mathbf{r}$ output from a particular layer of the generator G, where the generator creates an image $\mathbf{x}$ from random $\mathbf{z}$ through a composition of layers: $\mathbf{r} = h(\mathbf{z})$ and $\mathbf{x} = f(\mathbf{r}) = f(h(\mathbf{z})) = G(\mathbf{z})$.

Thresholding unit #65 layer 3 of a dining room generator matches 'table' segmentations with IoU=0.34.

Thresholding unit #37 layer 4 of a living room generator matches 'sofa' segmentations with IoU=0.29.

Figure 3: Visualizing the activations of individual units in two GANs. 10 top activating images are shown, and IoU is measured over a sample of 1000 images. In each image, the unit feature is upsampled and thresholded as described in Eqn. 2.

Since $\mathbf{r}$ has all the data necessary to produce the image $\mathbf{x} = f(\mathbf{r})$, $\mathbf{r}$ certainly contains the information to deduce the presence of any visible class c in the image. Therefore the question we ask is not whether information about c is present in $\mathbf{r}$ — it is — but *how* such information is encoded in $\mathbf{r}$. In particular, we seek to understand whether $\mathbf{r}$ explicitly represents the concept c in some way where it is possible to factor $\mathbf{r}$ at locations P into components

$$\mathbf{r}_{\mathrm{U,P}} = (\mathbf{r}_{\mathrm{U,P}}, \mathbf{r}_{\overline{\mathrm{U}},\mathrm{P}}), \tag{1}$$

where the generation of the object c at locations P depends mainly on the units $\mathbf{r}_{\mathrm{U,P}}$, and is insensitive to the other units $\mathbf{r}_{\overline{\mathrm{U}},\mathrm{P}}$. Here we refer to each channel of the featuremap as a unit; U denotes the set of unit indices of interest and $\overline{\mathrm{U}}$ denotes its complement; we will write $\mathbb{U}$ and $\mathbb{P}$ to refer to the entire set of units and featuremap pixels in r. Our method for studying the structure of $\mathbf{r}$ consists of two phases.

- Dissection: starting with a large dictionary of object classes, we identify the classes that have an explicit representation in r by measuring the agreement between individual units of $\mathbf{r}$ and every class c (Figure 1b).
- Intervention: for the represented classes identified through dissection, we measure causal effects between units and classes by forcing sets of units on and off $\mathbf{r}$ (Figure 1c and Figure 1d).

3.1 Characterizing units by dissection

We first focus on individual units of the representation. Recall that $\mathbf{r}_{u,\mathbb{P}}$ is the one-channel $h \times w$ featuremap of unit u in a convolutional generator, where $h \times w$ is typically smaller than the image size. We want to know if a specific unit $\mathbf{r}_{u,\mathbb{P}}$ encodes a semantic class such as a "tree". For image classification networks, Bau et al. (2017) has observed that many units can approximately locate emergent object classes when the units are upsampled and thresholded. In that spirit, we quantify the spatial agreement between the unit U's thresholded featuremap and a concept c' segmentation with the following intersection-over-union (IoU) measure:

$$\mathrm{IoU}_{u,c} \equiv \frac{\mathbb{E}_{\mathbf{z}} \left| (\mathbf{r}^{\uparrow}_{u,\mathbb{P}} > t_{u,c}) \wedge \mathbf{s}_c(\mathbf{x}) \right|}{\mathbb{E}_{\mathbf{z}} \left| (\mathbf{r}^{\uparrow}_{u,\mathbb{P}} > t_{u,c}) \vee \mathbf{s}_c(\mathbf{x}) \right|}, \text{where } t_{u,c} = \arg\max_t \frac{\mathrm{I}(\mathbf{r}^{\uparrow}_{u,\mathbb{P}} > t; \mathbf{s}_c(\mathbf{x}))}{\mathrm{H}(\mathbf{r}^{\uparrow}_{u,\mathbb{P}} > t, \mathbf{s}_c(\mathbf{x}))}, \tag{2}$$

where $\wedge$ and $\vee$ denotes the intersection and union operations, $\mathbf{x} = G(\mathbf{z})$ denotes the image generated from $\mathbf{z}$. The one-channel feature map $\mathbf{r}_{u,\mathbb{P}}$ slices the entire featuremap $\mathbf{r} = h(\mathbf{z})$ at unit u. As shown in Figure 2a, we upsample $\mathbf{r}_{u,\mathbb{P}}$ to the output image resolution as $\mathbf{r}^{\uparrow}_{u,\mathbb{P}}$. $(\mathbf{r}^{\uparrow}_{u,\mathbb{P}} > t_{u,c})$ produces a binary mask by thresholding the $\mathbf{r}^{\uparrow}_{u,\mathbb{P}}$ at a fixed level $t_{u,c}$. The binary mask $\mathbf{s}_c(\mathbf{x})$ denotes a semantic segmentation, whose pixel indicates the class c's presence in the generated image $\mathbf{x}$. We automatically choose a threshold $t_{u,c}$ (using a separate validation set) that maximizes the information quality ratio, that is, the portion of the joint entropy H which is mutual information I (Wijaya et al., 2017).

Given a dictionary of concepts $c \in \mathbb{C}$ for which we have semantic segmentations $\mathbf{s}_c$, we can use $\mathrm{IoU}_{u,c}$ to rank the concepts related to each unit and label each unit with a candidate concept that matches it best. Figure 3 shows examples of interpretable units with high $\mathrm{IoU}_{u,c}$. They are not the

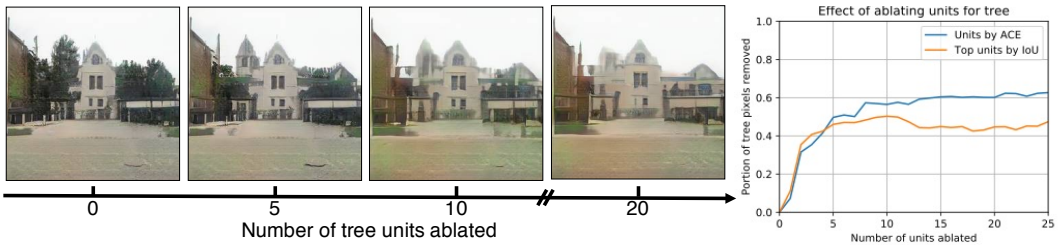

Figure 4: Ablating successively larger sets of tree-causal units from a GAN trained on LSUN outdoor church images, showing that the more units are removed, the more trees are reduced, while buildings remain. The choice of units to ablate is specific to the tree class and does not depend on the image. At right, the causal effect of removing successively more tree units is plotted, comparing units chosen to optimize ACE and units chosen with highest IoU for tree.

only units in their layers to match tables and trees: `layer3` of the dining room generator has 30 other units (of 512) that match tables and table parts; and `layer4` of the church generator has 24 (of 512) tree units.

Once we have identified an object class that a set of units match closely, we next ask: which of those units are responsible for triggering the rendering of that object? A unit that correlates highly with an output object might not actually cause that output. Furthermore, any output will jointly depend on several parts of the representation. Therefore, we need way to identify combinations of units that cause an object.

3.2 Measuring causal relationships using intervention

To answer the above question about causality, we probe the network using interventions: we test whether a set of units U in $\mathbf{r}$ cause the generation of c by forcing the units of U on and off.

Recall that $\mathbf{r}_{\mathrm{U,P}}$ denotes the featuremap $\mathbf{r}$ at units U and locations P. We *ablate* those units by forcing $\mathbf{r}_{\mathrm{U,P}} = \mathbf{0}$. Similarly, we *insert* those units by forcing $\mathbf{r}_{\mathrm{U,P}} = \mathbf{c}$, where $\mathbf{c}$ is a big constant. We decompose the featuremap $\mathbf{r}$ into two parts $(\mathbf{r}_{\mathrm{U,P}}, \mathbf{r}_{\overline{\mathrm{U,P}}})$, where $\mathbf{r}_{\overline{\mathrm{U,P}}}$ are unforced components of $\mathbf{r}$:

$$\begin{aligned} &\text{Original image :} && \mathbf{x} = G(\mathbf{z}) \equiv f(\mathbf{r}) \equiv f(\mathbf{r}_{\mathrm{U,P}}, \mathbf{r}_{\overline{\mathrm{U,P}}}) \\ &\text{Image with U ablated at pixels P :} && \mathbf{x}_a = f(\mathbf{0}, \mathbf{r}_{\overline{\mathrm{U,P}}}) \\ &\text{Image with U inserted at pixels P :} && \mathbf{x}_i = f(\mathbf{c}, \mathbf{r}_{\overline{\mathrm{U,P}}}) \end{aligned} \tag{3}$$

An object is caused by U if the object appears in $\mathbf{x}_i$ and disappears from $\mathbf{x}_a$. Figure 1c demonstrates the ablation of units that remove trees, and Figure 1d demonstrates insertion of units at specific locations to make trees appear. This causality can be quantified by comparing the presence of trees in $\mathbf{x}_i$ and $\mathbf{x}_a$ and averaging effects over all locations and images. Following (Holland, 1988; Pearl, 2009), we define the average causal effect of units U on the generation of on class c as:

$$\delta_{\mathrm{U}\to c} \equiv \mathbb{E}_{\mathbf{z},\mathrm{P}}[\mathbf{s}_c(\mathbf{x}_i)] - \mathbb{E}_{\mathbf{z},\mathrm{P}}[\mathbf{s}_c(\mathbf{x}_a)], \tag{4}$$

where $\mathbf{s}_c(x)$ denotes a segmentation indicating the presence of class c in the image $\mathbf{x}$ at P. To permit comparisons of $\delta_{\mathrm{U}\to c}$ between classes c which are rare, we normalize our segmentation $\mathbf{s}_c$ by $\mathbb{E}_{\mathbf{z},\mathrm{P}}[\mathbf{s}_c(x)]$. While these measures can be applied to a single unit, we have found that objects tend to depend on more than one unit. Thus we need to identify a set of units U that maximize the average causal effect $\delta_{\mathrm{U}\to c}$ for a class c.

Finding sets of units with high ACE. Given a representation $\mathbf{r}$ with d units, exhaustively searching for a fixed-size set U with high $\delta_{\mathrm{U}\to c}$ is prohibitive as it has $\binom{d}{|\mathrm{U}|}$ subsets. Instead, we optimize a continuous intervention $\boldsymbol{\alpha} \in [0,1]^d$, where each dimension $\boldsymbol{\alpha}_u$ indicates the degree of intervention for a unit u. We maximize the following average causal effect formulation $\delta_{\boldsymbol{\alpha}\to c}$:

$$\begin{aligned} &\text{Image with partial ablation at pixels P :} && \mathbf{x}'_a = f((\mathbf{1}-\boldsymbol{\alpha}) \odot \mathbf{r}_{\mathbb{U},\mathrm{P}},\ \mathbf{r}_{\mathbb{U},\overline{\mathrm{P}}}) \\ &\text{Image with partial insertion at pixels P :} && \mathbf{x}'_i = f(\boldsymbol{\alpha} \odot \mathbf{c} + (\mathbf{1}-\boldsymbol{\alpha}) \odot \mathbf{r}_{\mathbb{U},\mathrm{P}},\ \mathbf{r}_{\mathbb{U},\overline{\mathrm{P}}}) \\ &\text{Objective :} && \delta_{\boldsymbol{\alpha}\to c} = \mathbb{E}_{\mathbf{z},\mathrm{P}}\left[\mathbf{s}_c(\mathbf{x}'_i)\right] - \mathbb{E}_{\mathbf{z},\mathrm{P}}\left[\mathbf{s}_c(\mathbf{x}'_a)\right], \end{aligned} \tag{5}$$

where $\mathbf{r}_{\mathbb{U},\mathrm{P}}$ denotes the all-channel featuremap at locations P, $\mathbf{r}_{\mathbb{U},\overline{\mathrm{P}}}$ denotes the all-channel featuremap at other locations $\overline{\mathrm{P}}$, and $\odot$ applies a per-channel scaling vector $\boldsymbol{\alpha}$ to the featuremap $\mathbf{r}_{\mathbb{U},\mathrm{P}}$. We optimize $\boldsymbol{\alpha}$ over the following loss with an L2 regularization:

$$\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha}}(-\delta_{\boldsymbol{\alpha}\to c} + \lambda||\boldsymbol{\alpha}||_2), \tag{6}$$

where $\lambda = 100$ controls the relative importance of each term. We add the L2 loss as we seek for a minimal set of casual units. We optimize using stochastic gradient descent, sampling over both $\mathbf{z}$ and featuremap locations P and clamping $\boldsymbol{\alpha}$ within the range $[0, 1]^d$ at each step. Finally, we can rank units by $\boldsymbol{\alpha}^*_u$ and achieve a larger causal effect (i.e., removing trees) when ablating successively larger sets of tree-causing units as shown in Figure 4.

4 Results

We study three variants of Progressive GANs (Karras et al., 2018) trained on LSUN scene datasets (Yu et al., 2015). To segment the generated images, we use a recent model (Xiao et al., 2018) trained on ADE20K scene dataset (Zhou et al., 2017). The model can segment the input image into 366 object classes, 29 parts of large objects, and 25 materials. To further identify units that specialize in object parts, we expand each object class c into additional object part classes *c-t*, *c-b*, *c-l*, and *c-r*, which denotes the top, bottom, left, or right half of the bounding box of a connected component.

Below, we use dissection for analyzing units (Section 4.1), comparing units across datasets, layers, and models (Section 4.2), and locating artifact units (Section 4.3). Then, we start with a set of dominant object classes and use intervention to locate causal units that can remove and insert objects in different images (Section 4.4 and 4.5). In addition, our video demonstrates our interactive tool.

4.1 Emergence of individual unit object detectors

We are particularly interested in any units that are correlated to instances of an object class with diverse visual appearances; these would suggest that GANs generate those objects using similar abstractions as humans. Figure 3 illustrates two such units. For example in the dining room dataset, a unit emerges to match the dining table regions. Note that the matched tables have different colors, materials, geometry, viewpoints, and level of clutter: the only obvious commonality among these regions is the concept of table. Evaluating this unit against a recent segmentation network (Xiao et al., 2018) shows that the unit's featuremap correlates to the fully supervised segmentation model with a high IoU of 0.34.

4.2 Comparing units across datasets, layers, and models

Interpretable units for different scene categories The set of all object classes matched by the units of a GAN provides a map of what a GAN has learned about the data. Figure 5 examines units from generators train on four LSUN (Yu et al., 2015) scene categories. The units that emerge are object classes appropriate to the scene type: for example, when we examine a GAN trained on kitchen scenes, we find units that match stoves, cabinets, and the legs of tall kitchen stools. Another striking phenomenon is that many units represent parts of objects: for example, the conference room GAN contains separate units for the body and head of a person.

Interpretable units for different network layers. In classifier networks, the type of information explicitly represented changes from layer to layer (Zeiler & Fergus, 2014). We find a similar phenomenon in a GAN. Figure 6 compares early, middle, and late layers of a progressive GAN with 14 internal convolutional layers. The output of the first convolutional layer, one step away from the input z, remains entangled: individual units do not correlate well with any semantic object classes except for two units that are biased towards the ceiling of the room. Mid-level layers 4 to 7 have a large number of units that match semantic objects and object parts. Units in layers 10 and beyond match local pixel patterns such as materials and shapes.

Interpretable units for different GAN models. Interpretable units can provide insight about how GAN architecture choices affect the structures learned inside a GAN. Figure 7 compares three models (Karras et al., 2018) that introduce two innovations on baseline Progressive GANs. By examining unit semantics, we confirm that providing minibatch stddev statistics to the discriminator

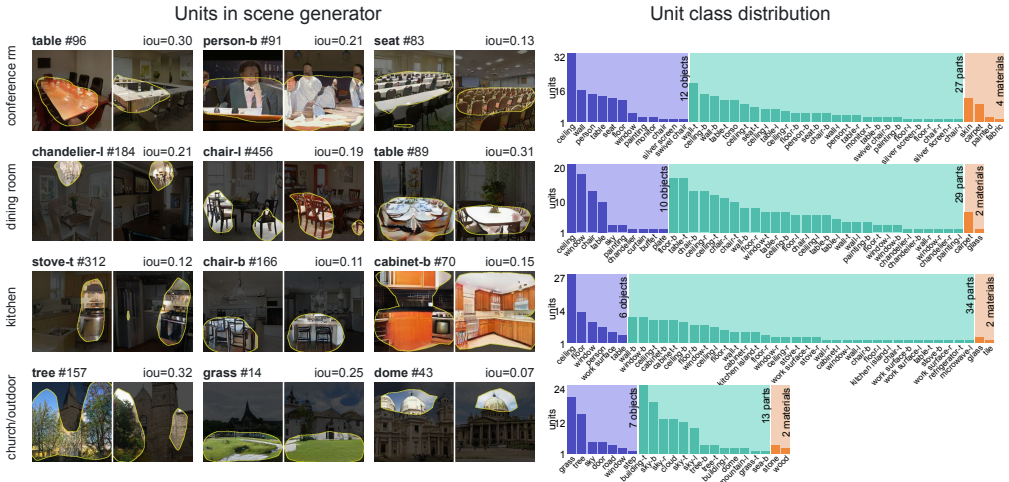

Figure 5: Comparing representations learned by progressive GANs trained on different scene types. The units that emerge match objects that commonly appear in the scene type: seats in conference rooms and stoves in kitchens. Units from layer 4 are shown. A unit is counted as a class predictor if it matches a supervised segmentation class with pixel accuracy > 0.75 and IoU > 0.05 when upsampled and thresholded. The distribution of units over classes is shown in the right column.

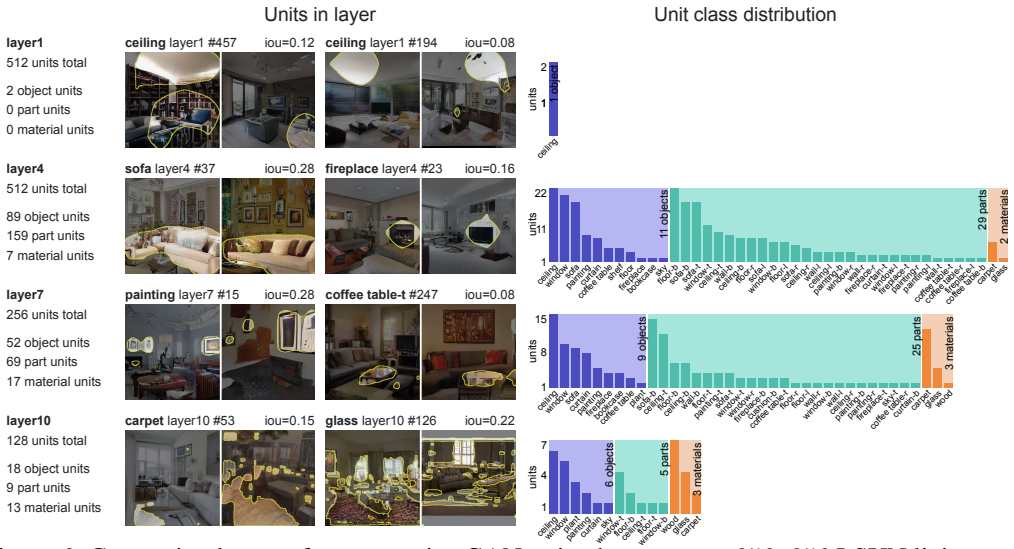

Figure 6: Comparing layers of a progressive GAN trained to generate $256x256$ LSUN living room images. The output of the first convolutional layer has almost no units that match semantic objects, but many objects emerge at layers 4-7. Later layers are dominated by low-level materials and shapes.

increases not only the visible GAN output, but also the diversity of concepts represented by units of a GAN: the number of types of objects, parts, and materials matching units increases by more than 70%. The second architecture applies pixelwise normalization to achieve better training stability. As applied to Progressive GANs, pixelwise normalization increases the number of units that match semantic classes by 19%.

4.3 Diagnosing and Improving GANs

While our framework can reveal how GANs succeed in producing realistic images, it can also analyze the causes of failures in their results. Figure 14a shows several annotated units that are responsible for typical artifacts consistently appearing across different images. Human annotation is efficient and it typically takes 10 minutes to locate 20 artifact-causing units out of 512 units in `layer4`.

More importantly, we can fix these errors by ablating the above 20 artifact-causing units. Figure 14b shows that artifacts are successfully removed and the artifact-free pixels stay the same, improving the

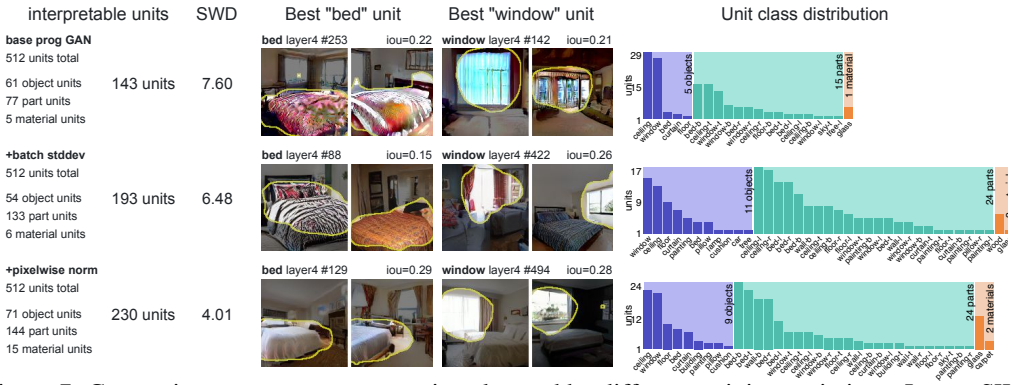

Figure 7: Comparing `layer4` representations learned by different training variations. Lower SWD indicates a higher-quality model: as the quality of the model improves, the number of interpretable units also rises. Progressive GANs apply several innovations including making the discriminator aware of minibatch statistics, and pixelwise normalization at each layer. We can see batch awareness increases the number of object classes matched by units, and pixel norm increases the number of units matching objects.

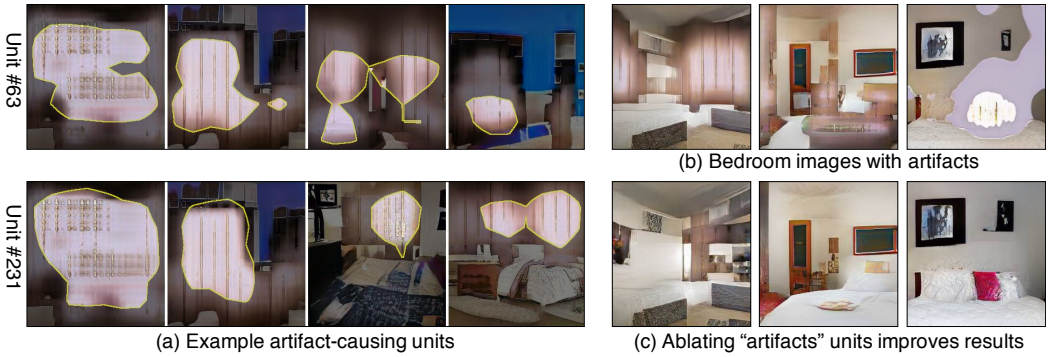

Figure 8: (a) We show two example "artifacts" units that are responsible for visual artifacts in GAN results. There are 20 units in total. By ablating these units, we can fix the artifacts in (b) and largely improve the visual quality as shown in (c).

generated results. To further quantify the improvement, we report two standard metrics. First, we compute the popular Fréchet Inception Distance (Heusel et al., 2017) between the generated images and real images. We use 50 000 real images and generate 10 000 images with high activations on these units. Second, we ask human participants on Amazon MTurk to determine which image looks more realistic, given two images produced by different methods. In total, we collected 20 000 annotations for 1 000 images per method. In Table 1, we compare our improved images to both the original artifacts images and a simple baseline that ablates 20 randomly chosen units. As demonstrated, our framework significantly improves GAN results based on these two metrics.

4.4 Locating Causal Units with Ablation

Errors are not the only type of output that can be affected by directly intervening in a GAN. A variety of specific object types can also be removed from GAN output by ablating a set of units in a GAN. In Figure 9 we apply the method in Section 3.2 to identify sets of 20 units that have causal effects on common object classes in conference rooms scenes. We find that, by turning off these small sets of units, most of the output of people, curtains, and windows can be removed from the generated scenes. However, not every object has a simple causal encoding: tables and chairs cannot be removed. Ablating those units will reduce the size and density of these objects, but will rarely eliminate them.

The ease of object removal depends on the scene type. Figure 10 shows that, while windows can be removed well from conference rooms, they are more difficult to remove from other scenes. In particular, windows are just as difficult to remove from a bedroom as tables and chairs from a conference room. We hypothesize that the difficulty of removal reflects the level of choice that a

Table 1: We compare generated images before and after ablating 20 "artifacts" units. We also report a simple baseline that ablates 20 randomly chosen units.

Fréchet Inception Distance (FID)	
original images	52.87
"artifacts" units ablated (ours)	**32.11**
random units ablated	52.27

Human preference score	original images
"artifacts" units ablated (ours)	**79.0**%
random units ablated	50.8%

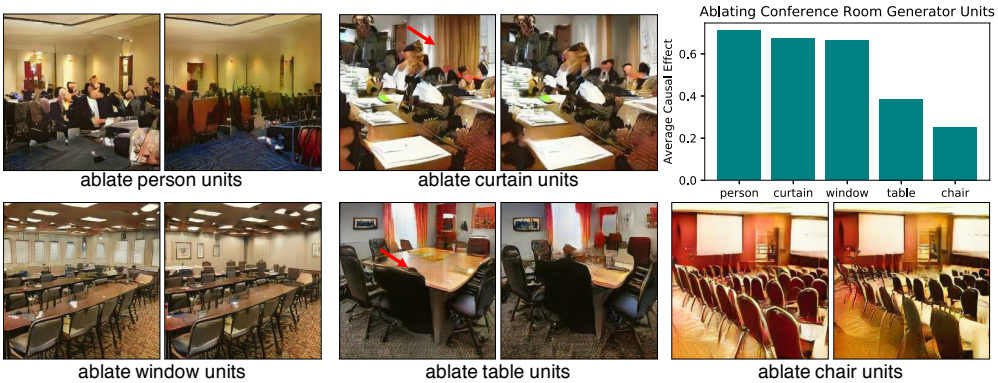

Figure 9: Measuring the effect of ablating units in a GAN trained on conference room images. Five different sets of units have been ablated related to a specific object class. In each case, 20 (out of 512) units are ablated from the same GAN model. The 20 units are specific to the object class and independent of the image. The average causal effect is reported as the portion of pixels that are removed in $1,000$ randomly generated images. We observe that some object classes are easier to remove cleanly than others: a small ablation can erase most pixels for people, curtains, and windows, whereas a similar ablation for tables and chairs only reduces object sizes without erasing them.

GAN has learned for a concept: a conference room is defined by the presence of chairs, so they cannot be removed. And modern building codes mandate that all bedrooms must have windows; the GAN seems to have caught on to that pattern.

4.5 Characterizing contextual relationships via insertion

We can also learn about the operation of a GAN by forcing units on and inserting these features into specific locations in scenes. Figure 11 shows the effect of inserting 20 `layer4` causal door units in church scenes. In this experiment, we insert units by setting their activation to a fixed 99% percentile level at a single feature pixel. Although this intervention is the same in each case, the effects vary widely depending on the context. For example, the doors added to the five buildings in Figure 11 appear with a diversity of visual attributes, each with an orientation, size, material, and style that matches the building.

We also observe that doors cannot be added in most locations. The locations where a door can be added are highighted by a yellow box. The bar chart in Figure 11 shows average causal effects of insertions of door units, conditioned on the object class at the location of the intervention. We find that, on average, the easiest way to increase doors in the output is to enlarge an existing door. This is shown in example (d). New doors can also be created, but only in appropriate locations. In general it is not possible to trigger a door in the sky or on trees. Interventions provide insight on how a GAN enforces relationships between objects. Even if we try to add a door in `layer4`, that choice can be vetoed later if the object is not appropriate for the context.

5 Discussion

By carefully examining representation units, we have found that many parts of GAN representations can be interpreted, not only as signals that correlate with object concepts but as variables that have a causal effect on the synthesis of semantic objects in the output. These interpretable effects can be used to compare, debug, modify, and reason about a GAN model. Our method can be potentially applied to other encoder-based generative models such as VAEs (Kingma & Welling, 2014) and RealNVP (Dinh et al., 2017).

Figure 10: Comparing the effect of ablating 20 window-causal units in GANs trained on five scene categories. In each case, the 20 ablated units are specific to the class and the generator and independent of the image. In some scenes, windows are reduced in size or number rather than eliminated completely, or replaced by visually similar objects such as paintings.

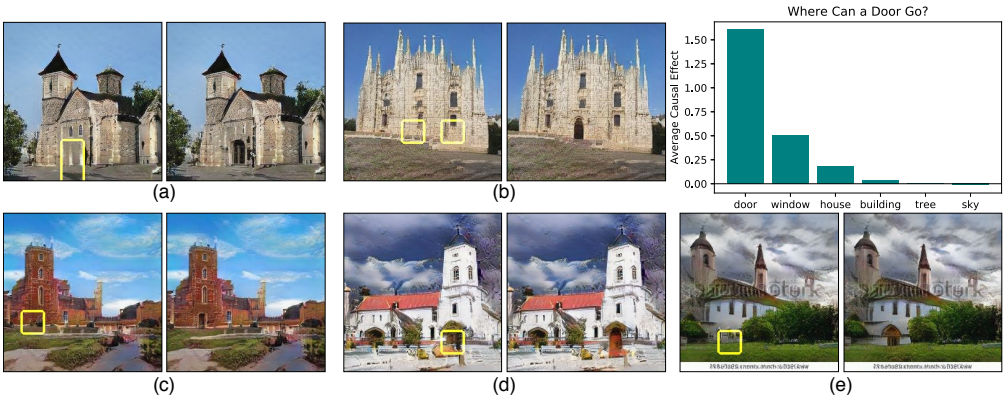

Figure 11: Inserting door units by setting 20 causal units to a fixed high value at one pixel in the representation. Whether the door units can cause the generation of doors is dependent on local context: we highlight every location that is responsive to insertions of door units on top of the original image, including two separate locations in (b) (we intervene at left). The same units are inserted in every case, but the door that appears has a size, alignment, and color appropriate to the location. One way to add door pixels is to emphasize a door that is already present: the result is a larger door (d). The chart summarizes the causal effect of inserting door units at one pixel with different context.

We have focused on the generator rather than the discriminator (as did in Radford et al. (2016)) because the generator must represent all the information necessary to approximate the target distribution, while the discriminator only learns to capture the difference between real and fake images. Alternatively, we can train an encoder to invert the generator (Donahue et al., 2017; Dumoulin et al., 2017). However, this incurs additional complexity and errors. Many GANs also do not have an encoder.

Our method is not designed to compare the quality of GANs with one other, and it is not intended as a replacement for well-studied GAN metrics such as SWD and FID. Our goal has been to identify interpretable structure and provide a window into the internal mechanisms of a GAN.

Prior visualization methods (Zeiler & Fergus, 2014; Bau et al., 2017; Karpathy et al., 2016) have brought many new insights to CNN and RNNs research. Motivated by that, in this work we have taken a small step towards understanding the internal representations of a GAN, and we have uncovered many questions that we cannot yet answer with the current method. For example: why can't a door be inserted in the sky? How does the GAN suppress the signal in the later layers? Further work will be needed to understand the relationships between layers of a GAN. Nevertheless, we hope that our work can help researchers and practitioners better analyze and develop their own GANs.

REFERENCES

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015. 3

David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *CVPR*, 2017. 3, 4, 10

Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS*, 2015. 2

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. In *ICLR*, 2017. 9

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. In *ICLR*, 2017. 10

Alexey Dosovitskiy and Thomas Brox. Generating images with perceptual similarity metrics based on deep networks. In *NIPS*, 2016. 3

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martin Arjovsky, Olivier Mastropietro, and Aaron Courville. Adversarially learned inference. In *ICLR*, 2017. 10

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014. 1, 2

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NIPS*, 2017. 8

Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *ICML*, 2018. 2

Paul W Holland. Causal inference, path analysis and recursive structural equations models. *ETS Research Report Series*, 1988(1):i–50, 1988. 5

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 1, 2

Andrej Karpathy, Justin Johnson, and Li Fei-Fei. Visualizing and understanding recurrent networks. In *ICLR*, 2016. 3, 10

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *ICLR*, 2018. 2, 6

Been Kim, Justin Gilmer, Fernanda Viegas, Ulfar Erlingsson, and Martin Wattenberg. Tcav: Relative concept importance testing with linear concept activation vectors. *arXiv preprint arXiv:1711.11279*, 2017. 3

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *ICLR*, 2014. 9

Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *CVPR*, 2015. 3

Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. In *ICLR*, 2016. 1, 2

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018. 2

Ari S Morcos, David GT Barrett, Neil C Rabinowitz, and Matthew Botvinick. On the importance of single directions for generalization. *arXiv preprint arXiv:1803.06959*, 2018. 3

Chris Olah, Arvind Satyanarayan, Ian Johnson, Shan Carter, Ludwig Schubert, Katherine Ye, and Alexander Mordvintsev. The building blocks of interpretability. *Distill*, 3(3):e10, 2018. 3

Judea Pearl. *Causality*. Cambridge university press, 2009. 5

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016. 1, 2, 3, 10

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017. 3

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *ICLR*, 2014. 3

Hendrik Strobelt, Sebastian Gehrmann, Hanspeter Pfister, and Alexander M. Rush. LSTMVis: A tool for visual analysis of hidden state dynamics in recurrent neural networks. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):667–676, Jan 2018. ISSN 1077-2626. doi: 10.1109/TVCG.2017.2744158. 3

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3319–3328, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL http://proceedings.mlr.press/v70/sundararajan17a.html. 3

Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. In *NIPS*, 2018. 2

Xiaolong Wang, Abhinav Shrivastava, and Abhinav Gupta. A-fast-rcnn: Hard positive generation via adversary for object detection. In *CVPR*, 2017. 1, 2

Dedy Rahman Wijaya, Riyanarto Sarno, and Enny Zulaika. Information quality ratio as a novel metric for mother wavelet selection. *Chemometrics and Intelligent Laboratory Systems*, 160:59–71, 2017. 4

Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *ECCV*, 2018. 6

Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 6

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, 2014. 3, 6, 10

Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018. 1, 2

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. In *ICLR*, 2015. 3

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017. 6

Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Interpretable basis decomposition for visual explanation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 119–134, 2018. 3

Tinghui Zhou, Philipp Krahenbuhl, Mathieu Aubry, Qixing Huang, and Alexei A Efros. Learning dense correspondence via 3d-guided cycle consistency. In *CVPR*, 2016. 3

Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. Generative visual manipulation on the natural image manifold. In *ECCV*, 2016. 3

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 1, 2

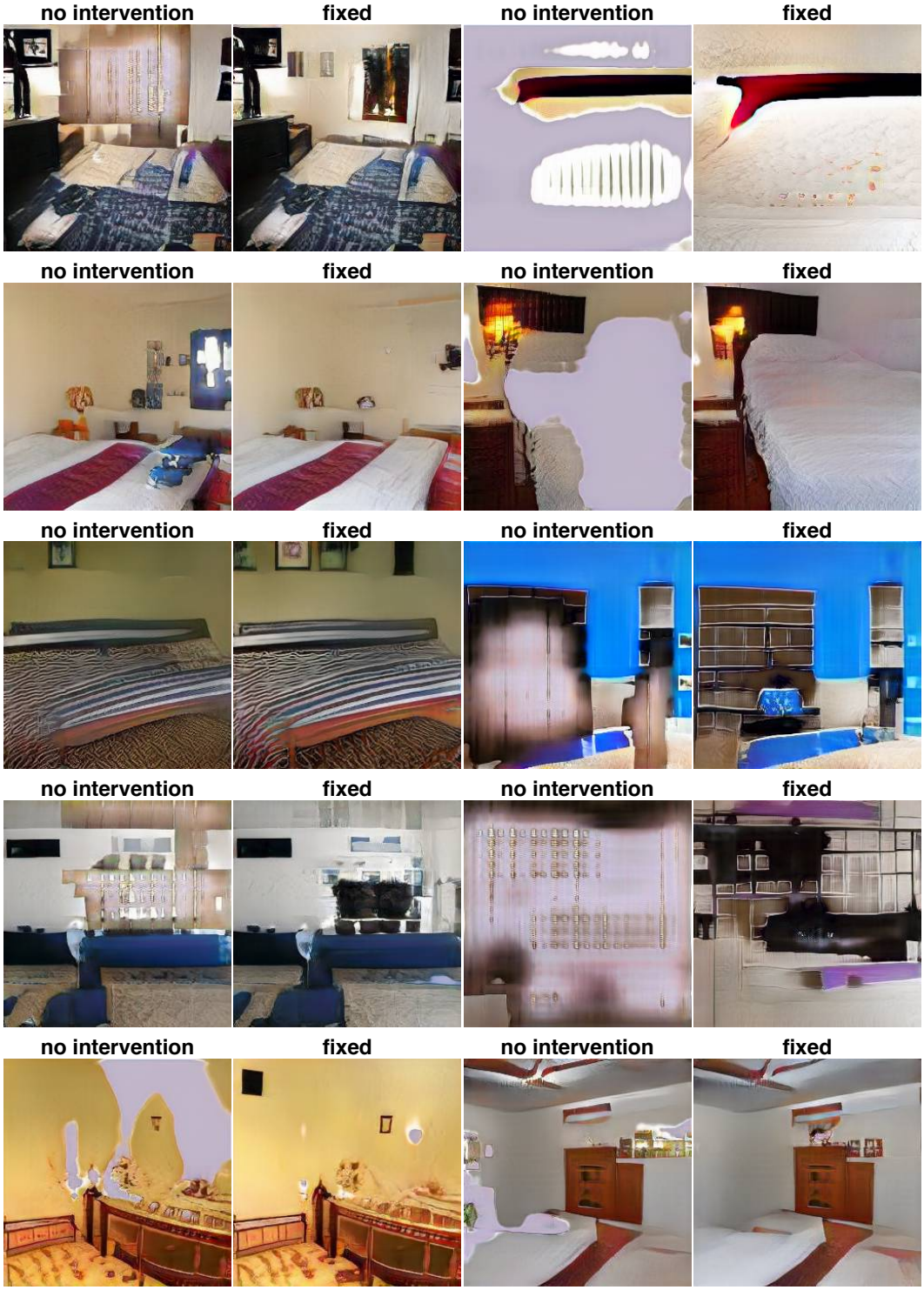

Figure 12: Further examples of images, before and after ablation of 20 artifact units.

6 Supplementary Material

6.1 Ablation of artifact units

Further examples of generated images are shown, before and after ablations of artifact units.

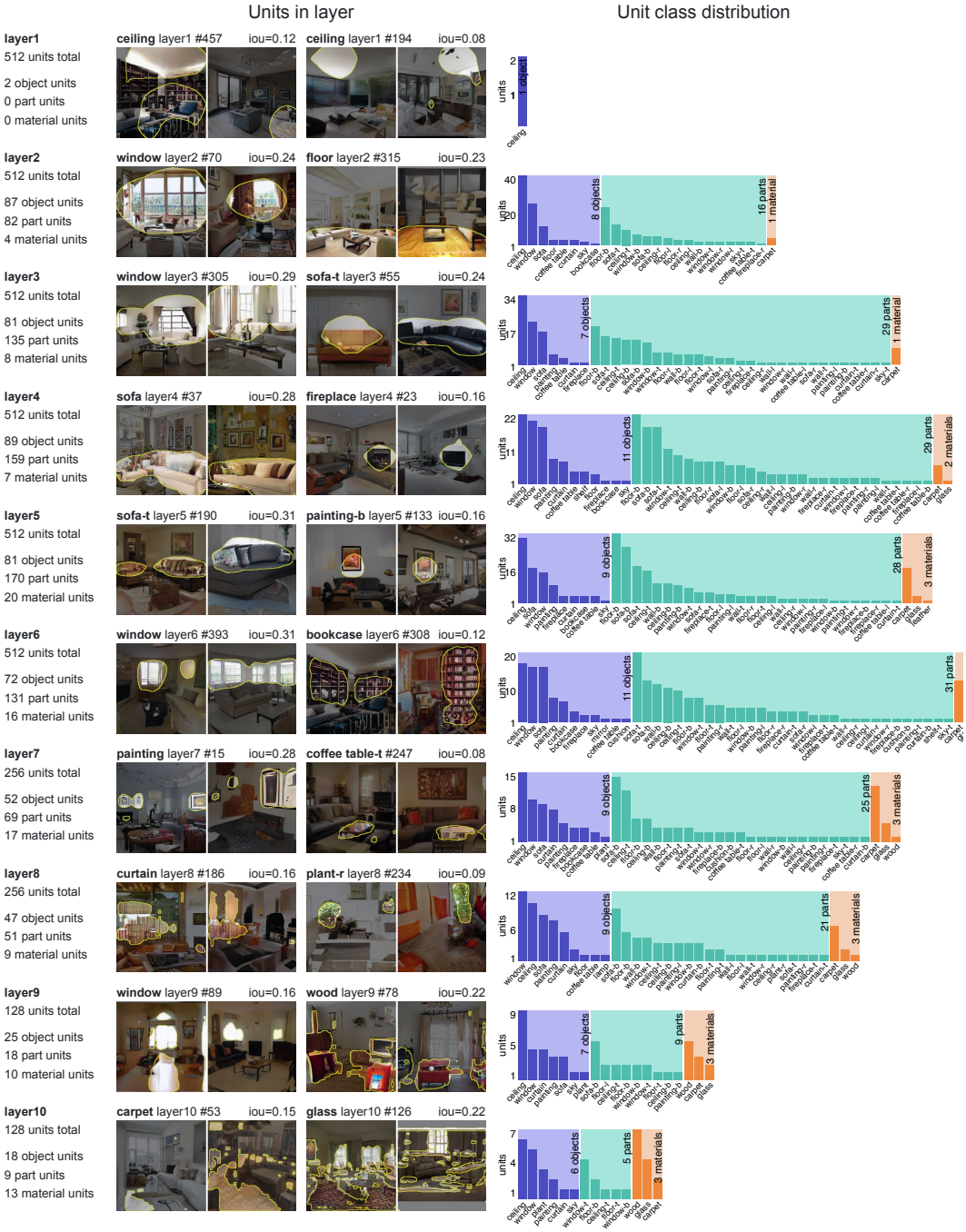

Figure 13: Layers 1-8

6.2 All layers of a GAN

A complete listing of the matching concepts that emerge at the 14 layers of a Progressive GAN trained on LSUN living room scenes.

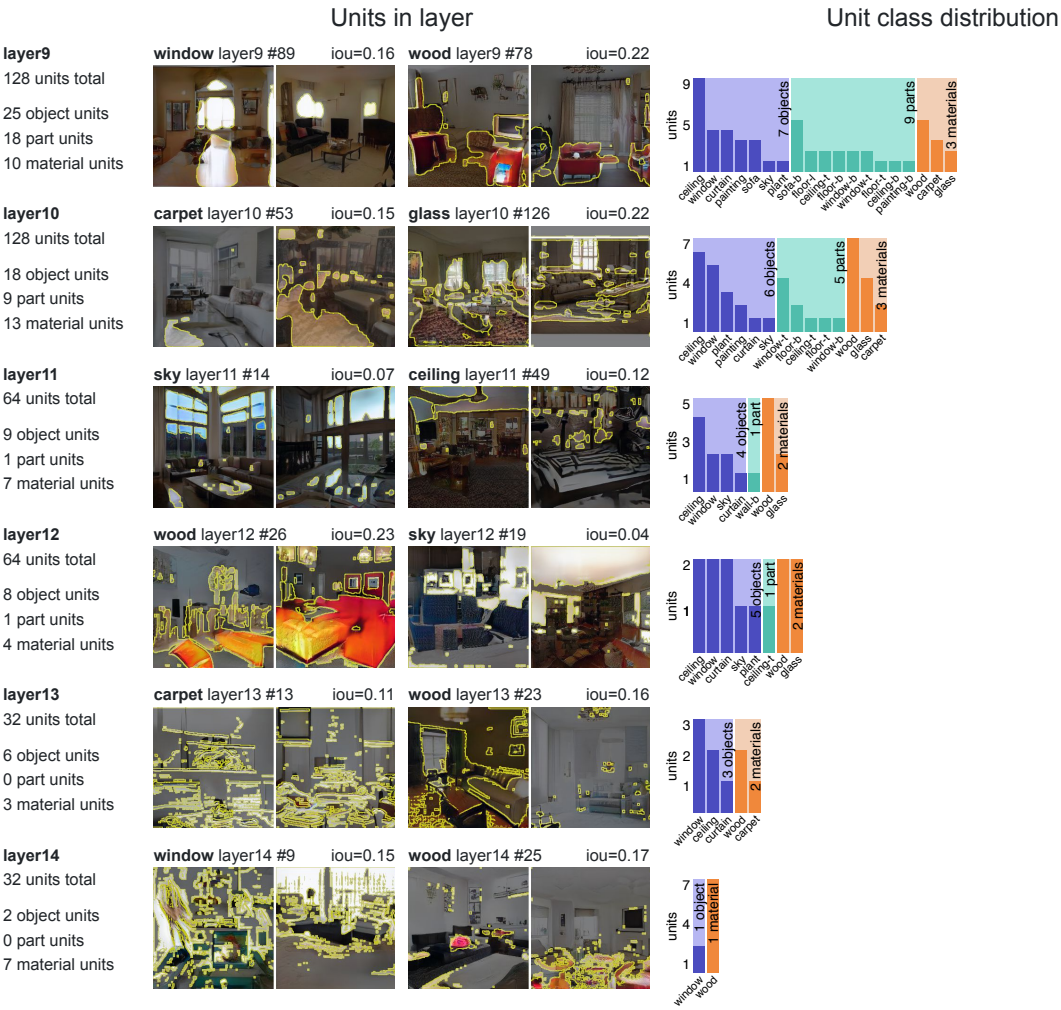

Figure 14: Layers 9-14