Detail Me More: Improving GAN's photo-realism of complex scenes

Raghudeep Gadde
Amazon

Qianli Feng
Amazon

Aleix M Martinez
Amazon

Figure 1: 1MP synthetic images of living rooms. Images generated using the proposed approach with 5 fine-grained discriminators for couches, chairs, coffee tables, end tables, and lamps. Zoom in to see the improved photo-realism of the details of the objects in the scene as compared to previous generative models.

Abstract

Generative models can synthesize photo-realistic images of a single object. For example, for human faces, algorithms learn to model the local shape and shading of the face components, i.e.*, changes in the brows, eyes, nose, mouth, jaw line, etc. This is possible because all faces have two brows, two eyes, a nose and a mouth, approximately in the same location. The modeling of complex scenes is however much more challenging because the scene components and their location vary from image to image. For example, living rooms contain a varying number of products belonging to many possible categories and locations,* e.g.*, a lamp may or may not be present in an endless number of possible locations. In the present work, we propose to add a "broker" module in Generative Adversarial Networks (GAN) to solve this problem. The broker is tasked to mediate the use of multiple discriminators in the appropriate image locales. For example, if a lamp is detected or wanted in a specific area of the scene, the broker assigns a fine-grained lamp discriminator to that image patch. This allows the generator to learn the shape and shading models of the lamp. The resulting multi-fine-grained optimization problem is able to synthesize complex scenes with almost the same level of photo-realism as single object images. We demonstrate the generability of the proposed approach on several GAN algorithms (BigGAN, ProGAN, StyleGAN, StyleGAN2), image resolutions (256^2 to 1024^2), and datasets. Our approach yields significant improvements over state-of-the-art GAN algorithms.*

1. Introduction

In recent years, the improvement in the photo-realism of the images synthesized by Generative Adversarial Networks (GANs) has been extraordinary. We now have algorithms than can synthesize high-resolution images of human faces, bodies, cats, dogs, cars and other object categories, with results basically indistinguishable from real photos to the untrained eye [20, 21, 22, 19, 36, 2, 3, 10, 38].

However, the high-fidelity of these synthetic images does not translate to the generation of complex scenes with multiple and varying object categories [6, 40]. For example, GAN-generated images of living rooms and city streets, although good, are easily distinguishable from real photos even by the untrained eye.

The classical (vanilla) single discriminator GAN is sufficient to help the generator estimate the underlying distribution of a single object class such as a face. In this case, the underlying distribution models the shape and shading changes of the face components located approximately at the same spatial location. But, a single discriminator GAN has a much harder time aiding the generator to identify the underlying distribution that models the shape and shading variations of all possible objects in all possible image locations.

The present paper derives a solution to this problem. Our key contribution is to include a "broker" module in GANs to mediate the use of multiple fine-grained discriminators to areas of the image that need them. For example, when generating an image of a living room, the broker may decide to use a couch discriminator, a coffee table discriminator,

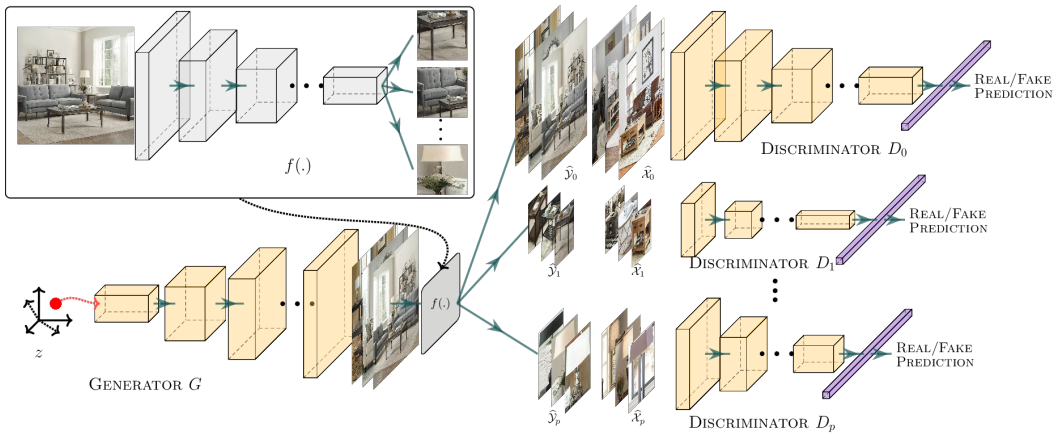

Figure 2: **DMM-GAN overview.** The broker assigns multiple fine-grained discriminators to generated images.

and a floor lamp discriminator, Figure 1. The broker will also decide to which image pixels each discriminator applies. The couch discriminator may be assigned to several pixels around the middle of the image, the coffee table discriminator to pixels slightly below that, and the floor lamp discriminator to pixels left of these two.

To make the above determination the broker mediates the result of a full-image discriminator and those of the fine-grained discriminators. Figure 2 provides an overview. For example, the full-image discriminator may specialize in the design of entire rooms, while the fine-grained discriminators focus on the photo-realism of the different elements of the room, like couches, coffee tables, and lamps.

Note, however, that our approach is general, allowing for other alternative configurations. For example, the full-image discriminator may be a graph representing the interrelationship between different objects in that type of scene [18, 14, 39]. Similarly, the fine-grained discriminators may be guided by an embedding space representing styles or object similarities, among others [30].

We show the general use of the proposed approach on BigGAN, ProGAN, StyleGAN, and StyleGAN2. In each case, these GAN algorithms are extended to include our broker module with up to five fine-grained discriminators. In all instances, the addition of the broker module yields statistically significant improvements over the FID score. On BigGAN, the addition of our module yields an improvement of up to 73.9%. On ProGAN it is 16.8%. On StyleGAN it is 35.1%. And, on StyleGAN2 an improvement of 25.6%.

Crucially, we show that FID scores improve significantly on the areas of the image with specific objects. For example, in a living room synthetic image, the FID score for the pixels in image patches containing a couch or a lamp see a comparable improvement to the ones detailed in the preceding paragraph. This allows us to generate complex scenes with a diverse number of objects in varying poses, scales and locations while maintaining their realism, Figure 1. As seen in the sample images in this figure, the overall scenes look very realistic and do not contain the typical large compositional errors seen in the results of previous algorithms.

We show these improvements over multiple image resolutions, 256×256 to 1024×1024, and on three datasets datasets – LSUN [43], Cityscapes [9], and a newly collected dataset of 1MP living room images. Our approach allows for a successful training with qualitatively and quantitatively good photo-realistic results.

2. Methods

2.1. Single discriminator GANs

Classical GANs are given by a generator network $G(.)$ with associated loss function $\mathcal{L}_G$ and a discriminator network $D(.)$ with associated loss $\mathcal{L}_D$. The original proposed losses are $\mathcal{L}_D = \mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}}[\log(1 - D(G(\mathbf{z})))]$ and $\mathcal{L}_G = -\mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}}[\log(D(G(\mathbf{z})))]$, $p_{\mathbf{z}}$ the prior distribution on latent space and $p_{\mathbf{x}}$ the distribution on image space [13]. Since then, many alternative loss functions [4, 25, 26, 27] and network topologies for $G(.)$ and $D(.)$ have been defined [20, 21, 22, 5, 19, 36].

Other successful extensions of GANs include adding: *a.* an attention module [32, 44] to specify which area of the image need to be edited, e.g., to locally edit the expression on a face image, *b.* an embedding space to control the normalization layers [17, 22], e.g., by aligning the mean and variance of the features with those of the style feature, and *c.* Lipschitz continuity on $D(.)$ [28, 34, 45], orthogonal regularization to $G(.)$ [12, 5], and others [15, 33, 26, 41, 23, 7].

In the sections to follow, we derive an alternative method that uses multiple fine-grained discriminators. We call our approach DMM-GAN (Detail Me More GAN). This approach is general and can be combined with any of the extensions described above. Although one could theoret-

ically use multiple discriminators to improve the photo-realism of object's parts (e.g., the eyes or mouth of a human face), single-discriminator GANs already excel at this task. Our goal is to use multiple discriminators to improve the photo-realism of complex scenes where the results of single-discriminator GANs are easily distinguishable as synthetic images.

2.2. Multi-discriminator GANs

We start by extending the classical (vanilla) discriminator defined above to fine-grained discriminators.

Let $D_k(.)$, $k = 1, \ldots, p$, denote p discriminators. Each $D_k(.)$ is used to improve the photo-realism of a (fine-grained) component of the image generated by $G(.)$. For example, if we are modeling living rooms, these fine-grained discriminators focus on the photo-realism of couches, chairs, coffee tables, end tables, lamps, etc., respectively. Note that each of these fine-grained discriminators $D_k(.)$ may be used in multiple image locations. For instance, if a living room scene contains two chairs, the chair discriminator will be applied to the pixels where we wish to draw the two chairs.

Let the loss function of the k^{th} fine-grained discriminator be $\mathcal{L}_k$. And, let $\mathcal{X}_k = \{X_{kj}\}_{j=1}^{n_k}$, with X_{kj} one of the n_k real photos of the object category $D_k(.)$ specializes on.

Similarly, let $\mathcal{Y} = \{Y_i\}_{i=1}^{m}$ be the set of synthetic (generated) images, where $Y_i = G(\mathbf{z}_i)$, $\mathbf{z}_i \sim p_{\mathbf{z}}$. Note that X_{kj} are images of a single object, whereas Y_i are images of complex scenes with multiple objects.

Hence, next, we need to identify the pixels (or regions) of the synthetic images where each of the fine-grained objects of interest are. To do this, we define the function

$$f\left(Y_i\right) = \left\{\left(\widehat{Y}_{ij}, k_{ij}\right)\right\}_{j=1}^{q_i} \tag{1}$$

which identifies the areas $\widehat{Y}_{ij}$ belonging to fine-grained category k_{ij}. This means that $(\widehat{Y}_{ij}, k_{ij})$ includes a single object who's class is the same as that of $\mathcal{X}_{k_{ij}}$, $k_{ij} \in [1, p]$. For example, Y_i might be the image of a living room and $\widehat{Y}_{i1}$ the pixels of a couch, $\widehat{Y}_{i2}$ the pixels of a coffee table, and $\widehat{Y}_{i3}$ the pixels of a lamp.

In the above, $f(.)$ is typically a differentiable function given by a deep neural network.

We can now define the set $\widehat{\mathcal{Y}}_k$ as the image patches of the synthetic images $\mathcal{Y}$ that correspond to the object class k. Formally,

$$\widehat{\mathcal{Y}}_k = \left\{\widehat{Y}_{ij}\right\}_{\forall(\widehat{Y}_{ij}, k_{ij}=k)} \tag{2}$$

and $k = 1, \ldots, p$.

The task of $D_k(.)$ is to discriminate between the real photos in $\mathcal{X}_k$ and the synthetic images in $\widehat{\mathcal{Y}}_k$. To have the same number of real and synthetic images we select a subset $\widehat{\mathcal{X}}_k$ from $\mathcal{X}_k$ of the same size as $\widehat{\mathcal{Y}}_k$, i.e., $\widehat{\mathcal{X}}_k \sim \mathcal{X}_k$, with $|\widehat{\mathcal{X}}_k| = |\widehat{\mathcal{Y}}_k|$, $|\mathcal{A}|$ the cardinality of the set $\mathcal{A}$.

We are finally in a position to do a forward pass on $D_k(.)$ using the sets $\widehat{\mathcal{X}}_k$ and $\widehat{\mathcal{Y}}_k$, i.e., D_k.forward_pass$(\widehat{\mathcal{X}}_k, \widehat{\mathcal{Y}}_k)$. We follow with the computation of the loss $\mathcal{L}_k$. And, then, we do a backward pass with the gradient of the loss, i.e., D_k.backward_pass$(\nabla\mathcal{L}_k)$.

These fine-grained discriminators allow us to improve the photo-realism of the different elements in a complex scene. This yields much improved, photo-realistic synthetic images of complex scenes. However, we still need to define a way to assign these discriminators to their appropriate image locals. We do this by incorporating a new broker module to GANs.

2.3. Broker module

Let $D_0(.)$ denote the discriminator that applies to the whole image, i.e., the complex scene. This discriminator is tasked to learn the distribution of objects and background elements in the image using a training set of real photos $\mathcal{X}_0 = \{X_{0j}\}_{j=1}^{n_0}$. Its loss function is $\mathcal{L}_0$.

Learning the underlying distribution of complex scenes is however a very difficult task. We use the function $f(.)$ defined above to help solve the problem. We call the combination of these two network components, $D_0(.)$ and $f(.)$, the broker module.

This new GAN module works as follows. First $G(.)$ generates m synthetic images. Formally, $\mathcal{Y} = \{G(\mathbf{z}_i)\}_{i=1}^{m}$, with $\mathbf{z}_i \sim p_{\mathbf{z}}$. Then, $f(.)$ is used to identify any possible regions (or pixels) corresponding to one of the possible p object classes.

It is key to note that lowering the threshold of the probability of detecting these object classes will allow the function $f(.)$ to guide the generator to synthesize images with those objects. This is because once $f(.)$ assigns an image region to class k, the fine-grained discriminator $D_k(.)$ is assigned on it. This forces the generator to synthesize more photo-realistic versions of the object.

Thus, if we want synthetic images that mostly resemble the scenes in the training set $\mathcal{X}_0$, we can set a large threshold for $f(.)$ to assign fine-grained discriminators to each region of the image. But if we want $f(.)$ to have a larger influence, we will lower that threshold value. This threshold *mediates* the influence of $D_0(.)$ and $f(.)$. Hence, the name broker.

2.4. DMM-GAN

We are finally in a position to define the training algorithm. Our approach is general and can be applied to any GAN architecture/topology, loss, etc. In this paper we will show examples of adding the proposed broker module to BigGAN, ProGAN, StyleGAN and StyleGAN2. As it is

commonly done in GANs, we train ours using a mini-batch of size m.

In our approach, at each iteration of training, we randomly draw m latent vectors from the prior distribution in latent space, i.e., $\mathbf{z}_i \sim p_{\mathbf{z}}$, $i = 1, \ldots, m$. We then compute $\mathcal{Y} = \{G(\mathbf{z}_i)\}_{i=1}^{m}$.

Next, we do a forward pass using the whole-image discriminator $D_0(.)$. That is, D_0.forward_pass$(\widehat{\mathcal{X}}_0, \mathcal{Y})$, where $\widehat{\mathcal{X}}_0$ is a randomly selected subset of m of the images in $\mathcal{X}_0$, i.e., $\widehat{\mathcal{X}}_0 \sim \mathcal{X}_0$ s.t. $|\widehat{\mathcal{X}}_0| = |\mathcal{Y}|$.

We can now compute the loss function $\mathcal{L}_0$. This allows us to do the backward pass on $D_0(.)$ to train the discriminator, i.e., D_0.backward_pass$(\nabla \mathcal{L}_0)$.

Then, we want to identify where each fine-grained discriminators needs to be applied. This is done by the broker module defined in the previous section. Specifically, we apply $f(\widehat{Y}_i)$, where $\widehat{Y}_i = G(\mathbf{z}_i)$. This yields the set $\{(\widehat{Y}_{ij}, k_{ij})\}_{j=1}^{q_i}$, which allows us to compute the p sets $\widehat{\mathcal{Y}}_k$, as described above.

We train each fine-grained discriminator using a forward pass, computing its loss, and doing a backward pass on the gradient of the loss. That is, D_k.forward_pass$(\widehat{\mathcal{X}}_k, \widehat{\mathcal{Y}}_k)$, compute $\mathcal{L}_k$, then D_k.backward_pass$(\nabla \mathcal{L}_k)$, where $\widehat{\mathcal{X}}_k$ is a randomly drawn subset of m images of $\mathcal{X}_k$.

As stated above, our approach is general and works with any loss function. Nonetheless, the loss does need to be extended to work with multiple discriminators. For example, we can readily extend the original minimax loss as,

$$\mathcal{L}_k = \mathbb{E}_{\mathbf{x} \sim p_{\mathbf{x}_k}} [\log D_k(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}} [\log(1 - [D_k(f(G(\mathbf{z})))]_k)] \quad (3)$$

where $p_{\mathbf{x}_k}$ is the distribution of the k^{th} fine-grained class, and $[f(D_k(G(\mathbf{z})))]_k$ is an image patch of the fine-grained class k. Note that when $k = 0$ this equation reduces to the original loss. For $k > 0$, the loss corresponds to each of the fine-grained local discriminators. A possible alternative is to use a discriminator loss where at each iteration the contribution of each discriminator is given by a probability [29]. This and other approaches [11, 1] are claimed to stabilize training and avoid mode collapse. We did not experience these problems with our approach and thus decided not to use them. Our approach focuses on how to improve photo-realism not on how to improve training, but our formulation also has this advantage.

Finally, we need to do a forward and a backward pass on the generator. As in [22], we find that resampling the latent vectors improves training. Thus, we generate a new set $\widehat{\mathcal{Z}} = \{\hat{\mathbf{z}}_i\}_{i=1}^{m}$, where $\hat{\mathbf{z}}_i \sim p_{\mathbf{z}}$. We then do the forward pass, compute the loss, and do the backward pass on its gradient. That is, G.forward_pass $(\widehat{\mathcal{Z}})$ and G.backward_pass$(\nabla \mathcal{L}_G)$, where we use G.forward_pass$(\widehat{\mathcal{Z}})$ and $G(\widehat{\mathcal{Z}})$ interchangeably.

The generator loss $\mathcal{L}_G$ needs to be modified to include the gradients of the global and fine-grained discriminators. For instance, extending the minimax loss yields

$$\mathcal{L}_G = -\mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}} \left[\lambda_0 \log(D(G(\mathbf{z}))) + \sum_{k=1}^{p} \lambda_i \log(D_k(f(G(\mathbf{z})))) \right] \quad (4)$$

where $\sum_{i=0}^{p} \lambda_i = 1$. The overall algorithm is summarized in Algorithm 1.

Algorithm 1 Pseudo-code for training DMM-GAN.

Let $G(.)$ be the GAN's generator network.
Let $\mathcal{L}_G$ be the loss of $G(.)$.
Let $D_k(.)$ be the GAN's discriminator networks, $k = 0, \ldots, p$, with $D_0(.)$ the full-image discriminator and $D_1(.)$ to $D_p(.)$ the fine-grained discriminators.
Let $\mathcal{L}_k$ be the loss associated to $D_k(.)$.
Let $\mathcal{X}_k = \{X_{kj}\}_{j=1}^{n_k}$ be the n_k sample photos used to train $D_k(.)$.
Let m be the mini-batch size.
Let $p_{\mathbf{z}}$ be the latent space prior distribution.
Let $\widehat{\mathcal{Y}} = \left\{\widehat{Y}_i = G(\mathbf{z}_i)\right\}_{i=1}^{m}$ be a set of synthetic images, $\mathbf{z}_i \sim p_{\mathbf{z}}, i = 1, \ldots, m$.
Let $\{(\widehat{Y}_{ij}, k_{ij})\}_{j=1}^{q_i} = f(\widehat{Y}_i)$ be the q_i image regions on $\widehat{Y}_i$ where we wish to improve photo-realism with $D_{k_{ij}}$.
Let $\widehat{\mathcal{Y}}_k = \left\{\widehat{Y}_{ij}\right\}_{\forall(\widehat{Y}_{ij}, k_{ij} = k)}$.
while training **do**
    $\mathbf{z}_i \sim p_{\mathbf{z}}, i = 1, \ldots, m$.
    $\widehat{\mathcal{Y}} = \{G(\mathbf{z}_i)\}_{i=1}^{m}$.
    $\widehat{\mathcal{X}}_0 \sim \mathcal{X}_0$ s.t. $|\widehat{\mathcal{X}}_0| = |\widehat{\mathcal{Y}}|$.
    D_0.forward_pass$(\widehat{\mathcal{X}}_0, \widehat{\mathcal{Y}})$ and compute $\mathcal{L}_0$.
    D_0.backward_pass$(\nabla \mathcal{L}_0)$.
    Compute $\widehat{\mathcal{Y}}_k, k = 1, \ldots, p$.
    for $k = 1$ to p **do**
        $\widehat{\mathcal{X}}_k = \{\widehat{X}_{ki}\}_{i=1}^{m}, \widehat{X}_{ki} \sim \mathcal{X}_k, \widehat{X}_{ki} \neq \widehat{X}_{kj} \forall i \neq j$.
        D_k.forward_pass$(\widehat{\mathcal{X}}_k, \widehat{\mathcal{Y}}_k)$ and compute $\mathcal{L}_k$.
        D_k.backward_pass$(\nabla \mathcal{L}_k)$.
    Resample $\hat{\mathbf{z}}_i \sim p_{\mathbf{z}}, i = 1, \ldots, m$.
    $\widehat{\mathcal{Z}} = \{\hat{\mathbf{z}}_i\}_{i=1}^{m}$
    G.forward_pass$(\widehat{\mathcal{Z}})$ and compute $\mathcal{L}_G$.
    G.backward_pass$(\nabla \mathcal{L}_G)$.

2.5. Implementation details

In this section we provide practical implementation level details of the proposed algorithm.

In several of our experiments we use GANs to estimate the distribution of images of living rooms. A living

room is a collection of objects belonging to classes such as couch, coffee table, end table, lamp, etc. Thus, in this case, our fine-grained classes will focus on these types of objects. That is, $D_1(.)$ will be a fine-grained discriminator of couches, $D_2(.)$ a fine-grained discriminator of coffee tables, and so on.

This means that we need to detect these objects in real photos and synthetic images. We can achieve this with an object detection algorithm as $f(.)$. In this paper, we use YoLo V3 [35] due to its computational efficiency and high accuracy. We pre-train this object detector model using a set of 7,600 images of living rooms with manually annotated bounding boxes around each '*couch*', '*coffee table*', '*chair*', '*end table*' and '*table lamp*'. We use 6,100 images to train and 1,500 for validation purpose.

For experiments modeling the distribution of street scenes using the Cityscapes dataset [9], we use a detector tuned on COCO [24] and use it to detect people, cars, buses and trucks.

We set the detection probability threshold of each class at .4 on all datasets. Recall that this is the threshold used by the broker to mediate between the global and the fine-grained discriminators.

In our experiments, we estimate the median dimensions of the bounding boxes on the training set and use the size that is closest to the power of 2. Experiments with the minimum and maximum sizes of the fine-grained object classes yielded poorer results.

For example, to synthesize 1MP images of living rooms, we used squared crops of size 512, 256, 256, 128 and 128 for the classes representing couch, coffee table, chair, end table, and lamp, respectively. This means that the bounding box given by YoLo was reshaped to these sizes according to each fine-grained object class. To synthesize images at lower resolutions, we scaled these sizes by the appropriate scale factor.

We implement DMM-GAN in PyTorch [31] and extend commonly used codebases of BigGAN[1], ProGAN and StyleGAN[2], and StyleGAN2[3]. When using one of these networks in our approach we add DMM- in front to indicate that the algorithm now includes the broker module and the fine-grained discriminators described in this paper, i.e., DMM-BigGAN, DMM-ProGAN, DMM-StyleGAN, and DMM-StyleGAN2.

In all of our experiments we used default learning rates, optimizers, and resolution specific model capacities. To further stabilize the training of DMM-BigGAN on relatively small GPUs, we added the regularizer of [42]. We also provide comparative results with the extension of BigGAN described in [36] called Unet-BigGAN[4], and with the StyleGAN extension of [19] called MSG-StyleGAN[5].

Following [20, 21, 22, 19] we report the total number of real images seen by the GAN to indicate training time. We tune until 25M real images are seen. Similar to prior work, we choose the model with the lowest attained FID score during training. Experiments are performed on an AWS server with 8 V100−32GB GPUs.

3. Results

3.1. Datasets

We provide experimental results on three datasets: LSUN [43], Cityscapes [9], and a new dataset of high-resolution, high-quality images of indoor rooms called DeepRooms.

For the LSUN experiments, we use the living room images. The shortest side of these images is 256 pixels. To be consistent with prior work, we always crop the center window of 256×256 pixels. This dataset has been extensively used in generative models and allows for many comparisons to the state-of-the-art. Here, we use 1.3M living room images to train the GAN models defined in Section 2.5.

Additionally, we collected a dataset of 100K 1MP living room images from multiple sites. These are high-quality images of staged products. The high resolution and professional look of these images makes it a more challenging problem for generative models to attain imagery that is comparable to the real photos.

Finally, Cityscapes is a dataset of stereo video sequences of street scenes recorded from a car in 50 different cities. While the previous two datasets correspond to indoor scenes, Cityscapes provides a way to test the performance of the proposed algorithm with outdoor scenes. We use the provided 25K images of 1,024×1,024-pixels taken from centered crops.

For each dataset/experiment, all algorithms are trained using the same dataset.

3.2. Quantitative evaluation

A common way to evaluate the photo-realism of synthetic images is by computing the Frechet Inception distance (FID) [16]. In our experiments below, we report several FID scores obtained on the three datasets.

First, for each of the GAN algorithms described in Section 2.5, we compute the FID score on a set of 50K images. We refer to this score as FID-50K [5, 20, 21, 19].

Second, we report the FID-infinity score [8]. Some researchers prefer this metric because it is less likely to be biased by the number of samples used to compute the FID score.

[1] https://github.com/ajbrock/BigGAN-PyTorch
[2] https://github.com/genforce/genforce
[3] https://github.com/NVlabs/stylegan2-ada-pytorch
[4] https://github.com/boschresearch/unetgan
[5] https://github.com/akanimax/msg-stylegan-tf

		Scene	Couch	Coffee table	Chair	End table	Lamp
FID-50K	BigGAN	114.2±1.56	–	–	–	–	–
	Unet-BigGAN	47.3±0.33	54.1±0.44	68.3±0.81	71.0±0.62	–	–
	DMM-BigGAN	**29.8±0.26**	**18.1±0.06**	**20.8±0.09**	**21.4±0.11**	**35.3±0.31**	**22.6±0.14**
	ProGAN	12.5±0.11	12.3±0.19	12.6±0.15	18.2±0.16	22.5±0.48	20.3±0.21
	DMM-ProGAN	**10.4±0.04**	**11.1±0.11**	**11.1±0.09**	**16.9±0.11**	**21.5±0.31**	**17.8±0.16**
	StyleGAN	5.7±0.02	9.5±0.07	8.0±0.08	10.3±0.07	26.1±0.31	12.9±0.11
	MSG-StyleGAN	4.6±0.01	7.6±0.04	7.1±0.06	9.1±0.06	24.9±0.41	11.6±0.19
	DMM-StyleGAN	**3.7±0.02**	**5.3±0.05**	**5.5±0.03**	**7.7±0.09**	**19.0±0.26**	**9.2±0.24**
	StyleGAN2	4.3±0.03	6.5±0.04	6.4±0.03	8.2±0.08	21.7±0.27	10.5±0.29
	DMM-StyleGAN2	**3.2±0.02**	**4.9±0.02**	**5.3±0.04**	**6.9±0.03**	**17.1±0.11**	**8.1±0.05**
FID-Inf	BigGAN	105.1±1.08	–	–	–	–	–
	UnetGAN	43.1±0.20	49.9±0.21	62.5±0.37	68.1±0.33	–	–
	DMM-BigGAN	**24.9±0.19**	**16.2±0.06**	**17.9±0.08**	**18.6±0.1**	**32.1±0.17**	**19.6±0.13**
	ProGAN	9.8±0.05	9.9±0.07	9.8±0.08	15.1±0.05	19.2±0.13	17.0±0.12
	DMM-ProGAN	**7.1±0.03**	**7.3±0.05**	**7.4±0.06**	**13.3±0.04**	**17.1±0.11**	**14.9±0.10**
	StyleGAN	2.3±0.02	4.4±0.04	8.1±0.08	9.2±0.06	20.1±0.19	12.6±0.09
	MSG-StyleGAN	2.1±0.01	4.1±0.03	7.6±0.06	8.0±0.05	18.9±0.15	11.1±0.09
	DMM-StyleGAN	**1.7±0.01**	**3.6±0.03**	**6.5±0.04**	**7.1±0.03**	**16.9±0.08**	**9.7±0.08**
	StyleGAN2	2.1±0.01	4.0±0.02	6.9±0.02	7.5±0.01	17.6±0.07	9.7±0.03
	DMM-StyleGAN2	**1.5±0.01**	**3.4±0.02**	**6.1±0.03**	**6.9±0.02**	**16.2±0.13**	**8.4±0.02**

Table 1: **Results on the LSUN living room synthetic images of 256×256 pixels.** FID↓: lower values are better.

		Scene	Couch	Coffee table	Chair	End table	Lamp
FID-50K	ProGAN	12.0±0.11	20.8±0.31	54.5±0.47	62.9±0.42	131.1±1.51	134.5±1.63
	StyleGAN	9.2±0.07	16.8±0.18	42.6±0.31	43.5±0.21	59.8±0.19	50.3±0.71
	MSG-StyleGAN	12.8±0.08	21.1±0.15	51.2±0.39	57.3±0.29	93.4±0.64	104.3±1.01
	StyleGAN2	6.3±0.04	8.8±0.09	26.1±0.11	26.9±0.23	36.7±0.31	25.2±0.24
	DMM-StyleGAN2	**5.1±0.04**	**5.2±0.04**	**17.8±0.08**	**19.1±012**	**31.2±0.16**	**19.8±0.11**
FID-Inf	ProGAN	9.2±0.08	18.6±0.11	56.5±0.19	59.1±0.41	138.2±0.77	137.6±0.59
	StyleGAN	6.4±0.05	13.1±0.11	39.8±0.13	40.2±0.23	56.7±0.44	48.4±0.30
	MSG-StyleGAN	9.7±0.10	18.3±0.09	53.1±0.26	55.1±0.31	96.9±0.67	102.2±0.88
	StyleGAN2	3.7±0.02	6.4±0.05	21.2±0.07	22.8±0.11	33.5±0.19	22.1±0.09
	DMM-StyleGAN2	**3.1±0.03**	**4.1±0.02**	**12.9±0.07**	**15.4±0.07**	**28.7±0.12**	**16.2±0.14**

Table 2: **Results on the DeepRooms living room synthetic images of 1,024×1,024 pixels.** FID↓: lower values are better.

		Scene	Person	Car	Bus	Truck
FID-50K	ProGAN	28.4±0.18	94.1±0.48	57.3±0.19	96.8±0.61	61.4±0.41
	StyleGAN	17.4±0.12	68.8±0.31	39.5±0.13	79.8±0.38	53.2±0.33
	MSG-StyleGAN	9.6±0.07	51.1±0.21	28.8±0.14	56.5±0.26	39.1±0.27
	StyleGAN2	6.7±0.04	37.1±0.26	17.3±0.11	38.4±0.23	22.4±0.29
	DMM-StyleGAN2	**5.3±0.03**	**31.3±0.09**	**11.2±0.05**	**34.4±0.18**	**19.3±0.11**
FID-Inf	ProGAN	26.1±0.18	98.2±0.34	54.1±0.23	93.2±0.19	58.9±0.61
	StyleGAN	13.2±0.08	66.1±0.41	33.4±0.22	76.1±0.43	51.3±0.23
	MSG-StyleGAN	6.1±0.04	48.6±0.38	23.1±0.16	49.1±0.21	34.8±0.32
	StyleGAN2	4.8±0.02	23.1±0.13	8.9±0.09	31.1±0.18	17.2±0.23
	DMM-StyleGAN2	**3.5±0.02**	**20.2±0.08**	**5.8±0.03**	**27.3±0.12**	**16.1±0.05**

Table 3: **Results on the Cityscapes synthetic images of 1,024×1,024 pixels.** FID↓: lower values are better.

		Scene
FID-50K	StyleGAN2	6.6±0.02
	DMM-StyleGAN2 w/ 1 fgd	6.0±0.02
	DMM-StyleGAN2 w/ 2 fgd	5.7±0.03
	DMM-StyleGAN2 w/ 5 fgd	5.2±0.02
FID-Inf	StyleGAN2	3.2±0.01
	DMM-StyleGAN2 w/ 1 fgd	2.7±0.02
	DMM-StyleGAN2 w/ 2 fgd	2.4±0.02
	DMM-StyleGAN2 w/ 5 fgd	2.2±0.01

Table 4: **Ablation study on DMM-StyleGAN2 at 512×512 resolution.** fgd = # fine-grained discriminators.

	Couch	Lamp
Real photos	.63	.56
ProGAN	.41	.22
StyleGAN	.48	.33
MSG-StyleGAN	.46	.31
StyleGAN2	.53	.40
DMM-StyleGAN2	.55	.45

Table 5: **Training an object detector.** mAP results when training with photos vs synthetic images as given by the listed algorithms and object categories.

Third, and new to this paper, we report the FID scores defined in the previous two paragraphs on the crops of the objects detected by the YoLo algorithm. For example, when generating images of living rooms, we use YoLo to detect couches, chairs, coffee tables, end tables, and lamps. We use the pixels inside the bounding box given by YoLo to compute the FID-50K and FID-infinity scores of the cropped couches, chairs, coffee tables, end tables, and lamps. We then report the mean and standard deviation of these FID scores for each of the five object categories.

This yields 6 FID-50k and 6 FID-infinity scores, two per object category and two for the global scene, Table 1. While the scene's two FID scores give a global measure of the quality of the whole image, the FID scores of the objects evaluate the photo-realism of the rendered objects in the scene. Note that the FID scores of some object categories are not available for some GANs. This is the case when the YoLo algorithms fails to detect any object of that category in the rendered synthetic images.

3.3. Results on LSUN

Quantitative results on the synthetic living room images are shown in Table 1. This table shows the results of multiple GAN algorithms and the results of these GANs with the addition of the broker module and fine-grained discriminators introduced in this paper. We see that, in every instance, the addition of the herein derived broker module yields a statistically significant improvement of the FID score. Recall that lower FID scores are preferred.

Qualitative results of the generation of synthetic living room images are given in Figure 3(a-b). The figure provides comparative results between StyleGAN2 in (a) and DMM-StyleGAN2 in (b).

3.4. Results on DeepRooms

Table 2 shows quantitative results on the generation of 1MP synthetic living rooms. As in the above, the table provides comparative results with and without the proposed broker module and fine-grained discriminators on existing GAN algorithms. Again, the addition of the broker module results in a statistically significant improvement of the FID score, i.e., significantly lower FID values.

Qualitative results of the generation of 1MP synthetic living room images were given in Figure 1. Comparative results are now given in Figure 3(c-d). In (c) we show results with StyleGAN2 and in (d) results obtained with DMM-StyleGAN2. Additional comparative results are in the Supplementary File.

3.5. Results on Cityscapes

Quantitative results on the generation of 1MP synthetic street scenes are in Table 3. Comparative results against other GAN algorithms are also provided. These results also show a statistically significance improved FID scores as compared to state-of-the-art algorithms.

Comparative qualitative results on the generation of 1MP synthetic street scene images are in Figure 3(e-f). Additional comparative results are in the Supplementary File.

3.6. The role of multiple discriminators

The best results in every experiment was attained with DMM-StyleGAN2 using 5 fine-grained discriminators. To better quantify the contribution of the broker module as a function of the number of discriminators, we ran an ablation study. In this study, we trained DMM-StyleGAN2 on DeepRooms using q fine-grained discriminators. We varied q from 1 to 5. We ran multiple experiments for each value of q, each time with a different set of discriminators. The mean and stanadrd deviation of the FID scores are in Table 4. As seen in these results, adding fine-grained discriminators improve photo-realism as measured by FID.

3.7. Relevance to downstream tasks

As a final quantitative evaluation, we investigated the usage of synthetic images in downstream tasks such as object detection as proposed in [37]. Specifically, we sampled 50,000 images from four GAN models considered in this work and trained an object detection algorithm on those images. The bounding boxes used to train the object detec-

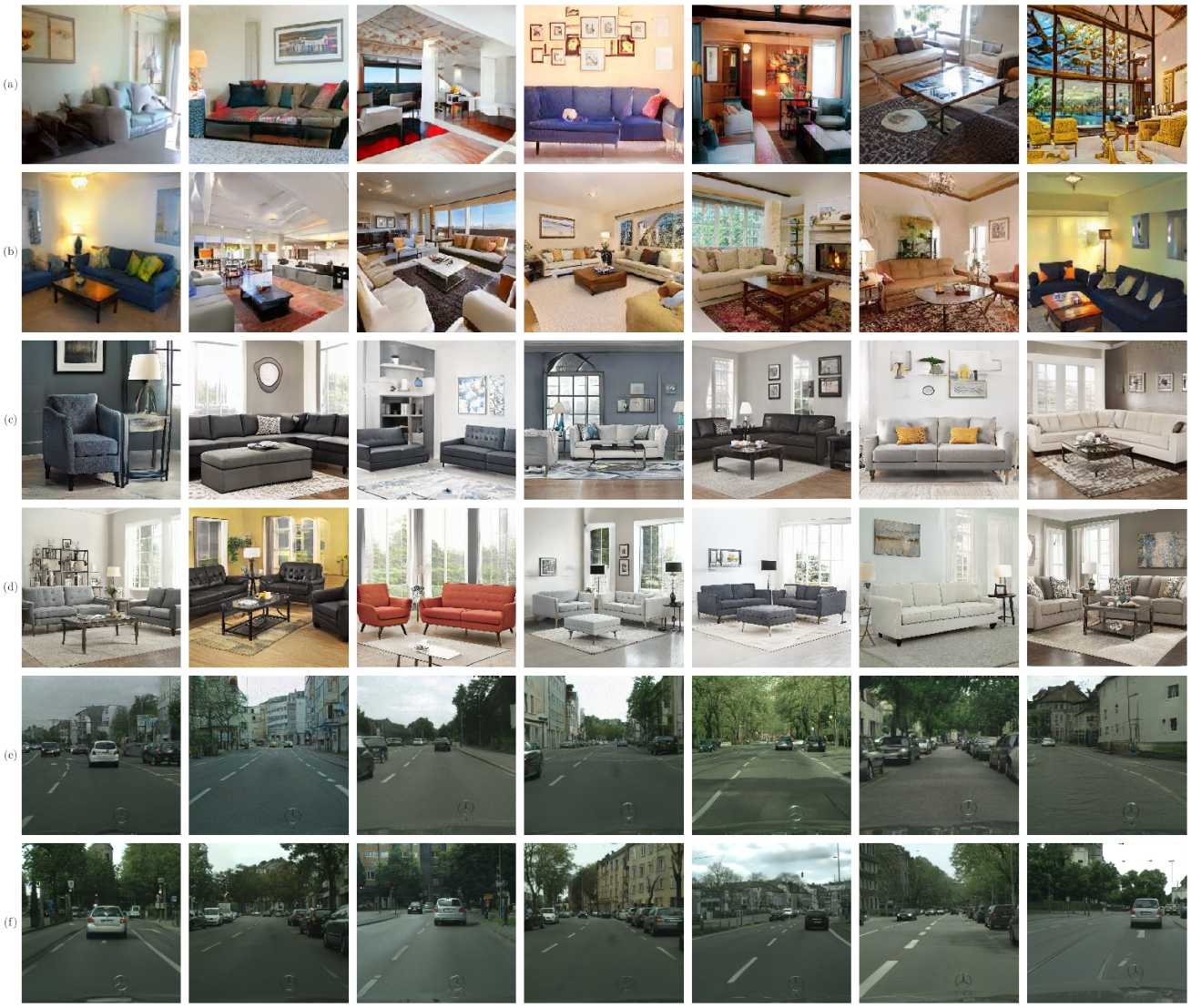

Figure 3: **Comparative results.** Results on (a-b) LSUN, (c-d) DeepRooms, and (e-f) Cityscapes. StyleGAN2 in (a,c,e) and DMM-StyleGAN2 (b,d,f). Zoom in to see improvements in the photo-realism of (b,d,f).

tor are obtained with the YoLo algorithm mentioned earlier. We compare these results to those obtained when training the object detector with real photos. Each trained model is tested on a set of 1,500 real photos with manually annotated bounding boxes. This task can thus be seen as a proxy for evaluating the quality of synthetic images as compared to actual photos. Table 5 shows the mean Absolute Precision (mAP) of the detection results on two object categories – couches and lamps. We see that as the quality of synthetic images improve so does mAP.

4. Conclusions

Generating synthetic images of complex scenes with GANs is still a difficult problem. While the generated images may look reasonable at first glance, these images contain gross errors that clearly identify them as synthetic. A major reason for these errors is given by the difficulty of modeling scenes of varying objects that can be or not be present in a large number of possible locations and poses. We solve this problem by adding a broker module that identifies where in the image to use different fine-grained discriminators. These fine-grained discriminators are then used to improve the photo-realism of these local regions of the image. We have provided extensive comparative results against the state-of-the-art and shown that the proposed approach yields superior results. We have shown this using a number of quantitative measure as well as a qualitative evaluation.

References

[1] Isabela Albuquerque, Joao Monteiro, Thang Doan, Breandan Considine, Tiago Falk, and Ioannis Mitliagkas. Multi-objective training of generative adversarial networks with multiple discriminators. In *International Conference on Machine Learning*, pages 202–211, 2019. 4

[2] Yazeed Alharbi and Peter Wonka. Disentangled image generation through structured noise injection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5134–5142, 2020. 1

[3] Dongsheng An, Yang Guo, Min Zhang, Xin Qi, Na Lei, and Xianfang Gu. Ae-ot-gan: Training gans from data specific latent distribution. In *European Conference on Computer Vision*, pages 548–564. Springer, 2020. 1

[4] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017. 2

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2018. 2, 5

[6] Arantxa Casanova, Michal Drozdzal, and Adriana Romero-Soriano. Generating unseen complex scenes: are we there yet? *arXiv preprint arXiv:2012.04027*, 2020. 1

[7] Ting Chen, Xiaohua Zhai, Marvin Ritter, Mario Lucic, and Neil Houlsby. Self-supervised gans via auxiliary rotation loss. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12154–12163, 2019. 2

[8] Min Jin Chong and David Forsyth. Effectively unbiased fid and inception score and where to find them. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6070–6079, 2020. 5

[9] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 2, 5

[10] Terrance DeVries, Michal Drozdzal, and Graham W Taylor. Instance selection for gans. *arXiv preprint arXiv:2007.15255*, 2020. 1

[11] Ishan Durugkar, Ian Gemp, and Sridhar Mahadevan. Generative multi-adversarial networks. *arXiv preprint arXiv:1611.01673*, 2016. 4

[12] Xinyu Gong, Shiyu Chang, Yifan Jiang, and Zhangyang Wang. Autogan: Neural architecture search for generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3224–3234, 2019. 2

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 2

[14] Jiuxiang Gu, Handong Zhao, Zhe Lin, Sheng Li, Jianfei Cai, and Mingyang Ling. Scene graph generation with external knowledge and image reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1969–1978, 2019. 2

[15] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017. 2

[16] 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 *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6629–6640, 2017. 5

[17] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017. 2

[18] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1219–1228, 2018. 2

[19] Animesh Karnewar and Oliver Wang. MSG-GAN: Multi-scale gradients for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7799–7808, 2020. 1, 2, 5

[20] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018. 1, 2, 5

[21] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 1, 2, 5

[22] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8110–8119, 2020. 1, 2, 4, 5

[23] Karol Kurach, Mario Lučić, Xiaohua Zhai, Marcin Michalski, and Sylvain Gelly. A large-scale study on regularization and normalization in gans. In *International Conference on Machine Learning*, pages 3581–3590. PMLR, 2019. 2

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 5

[25] Mario Lucic, Karol Kurach, Marcin Michalski, Olivier Bousquet, and Sylvain Gelly. Are gans created equal? a large-scale study. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 698–707, 2018. 2

[26] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE inter-*

national conference on computer vision, pages 2794–2802, 2017. 2

[27] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International conference on machine learning*, pages 3481–3490. PMLR, 2018. 2

[28] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. 2

[29] Gonçalo Mordido, Haojin Yang, and Christoph Meinel. Dropout-gan: Learning from a dynamic ensemble of discriminators. *arXiv preprint arXiv:1807.11346*, 2018. 4

[30] Nelson Nauata, Sepidehsadat Hosseini, Kai-Hung Chang, Hang Chu, Chin-Yi Cheng, and Yasutaka Furukawa. House-gan++: Generative adversarial layout refinement networks. *arXiv preprint arXiv:2103.02574*, 2021. 2

[31] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32:8026–8037, 2019. 5

[32] Albert Pumarola, Antonio Agudo, Aleix M Martinez, Alberto Sanfeliu, and Francesc Moreno-Noguer. Ganimation: Anatomically-aware facial animation from a single image. In *Proceedings of the European conference on computer vision (ECCV)*, pages 818–833, 2018. 2

[33] Guo-Jun Qi. Loss-sensitive generative adversarial networks on lipschitz densities. *International Journal of Computer Vision*, 128(5):1118–1140, 2020. 2

[34] Yipeng Qin, Niloy Mitra, and Peter Wonka. How does lipschitz regularization influence gan training? In *European Conference on Computer Vision*, pages 310–326. Springer, 2020. 2

[35] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 5

[36] Edgar Schonfeld, Bernt Schiele, and Anna Khoreva. A u-net based discriminator for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8207–8216, 2020. 1, 2, 5

[37] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. How good is my gan? In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 213–229, 2018. 7

[38] Ivan Skorokhodov, Savva Ignatyev, and Mohamed Elhoseiny. Adversarial generation of continuous images. *arXiv preprint arXiv:2011.12026*, 2020. 1

[39] Mohammed Suhail, Abhay Mittal, Behjat Siddiquie, Chris Broaddus, Jayan Eledath, Gerard Medioni, and Leonid Sigal. Energy-based learning for scene graph generation. *arXiv preprint arXiv:2103.02221*, 2021. 2

[40] Tristan Sylvain, Pengchuan Zhang, Yoshua Bengio, R Devon Hjelm, and Shikhar Sharma. Object-centric image generation from layouts. *arXiv preprint arXiv:2003.07449*, 2020. 1

[41] Jiqing Wu, Zhiwu Huang, Janine Thoma, and Luc Van Gool. Energy-relaxed wasserstein gans (energywgan): Towards more stable and high resolution image generation. 2

[42] Dingdong Yang, Seunghoon Hong, Yunseok Jang, Tianchen Zhao, and Honglak Lee. Diversity-sensitive conditional generative adversarial networks. In *International Conference on Learning Representations*, 2018. 5

[43] 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. 2, 5

[44] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International conference on machine learning*, pages 7354–7363. PMLR, 2019. 2

[45] Zhiming Zhou, Yuxuan Song, Lantao Yu, Hongwei Wang, Jiadong Liang, Weinan Zhang, Zhihua Zhang, and Yong Yu. Understanding the effectiveness of lipschitz-continuity in generative adversarial nets. *arXiv preprint arXiv:1807.00751*, 2018. 2