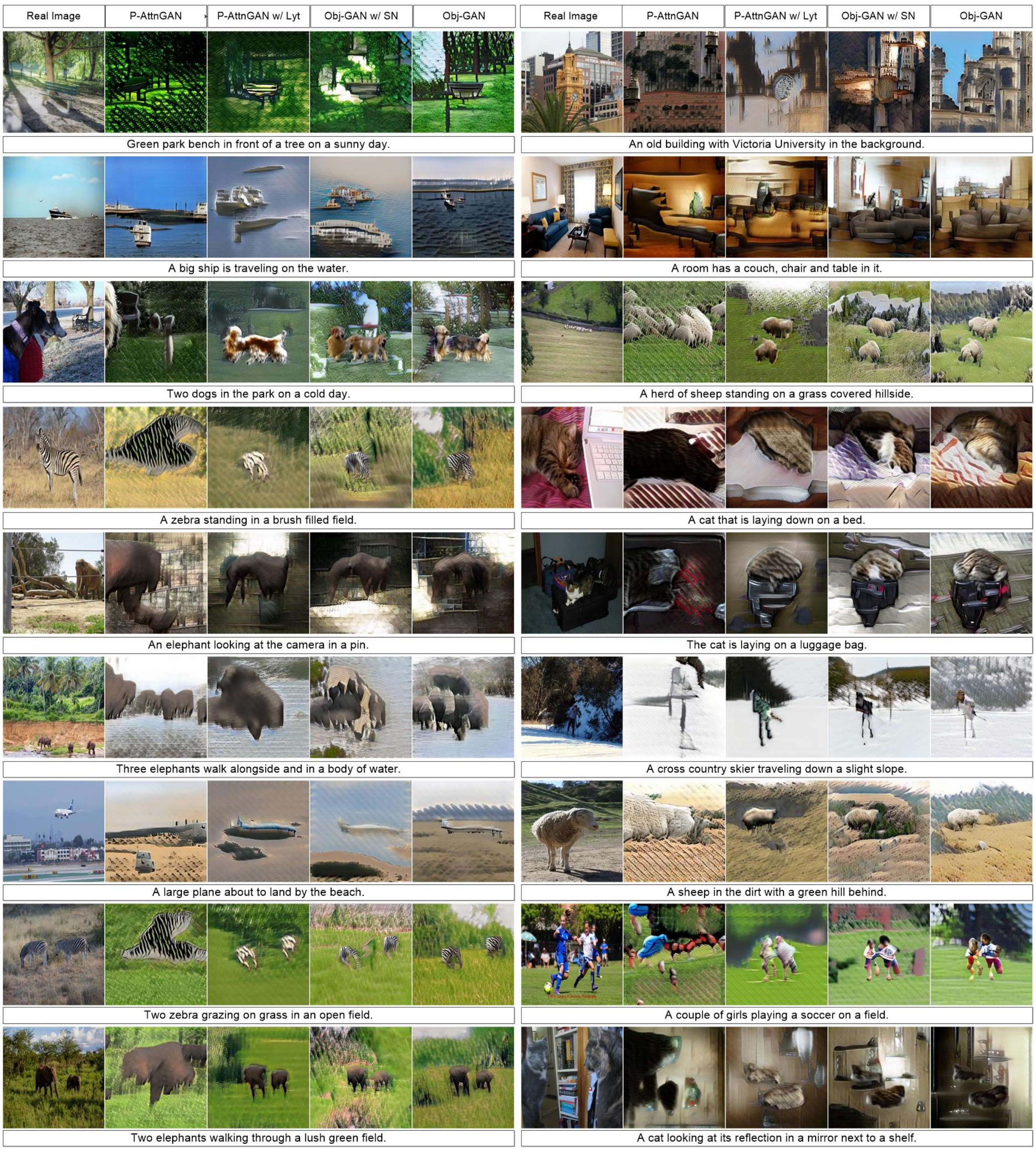
Object-driven Text-to-Image Synthesis via Adversarial Training

1. Comparison between Obj-GAN and the Ablative Versions

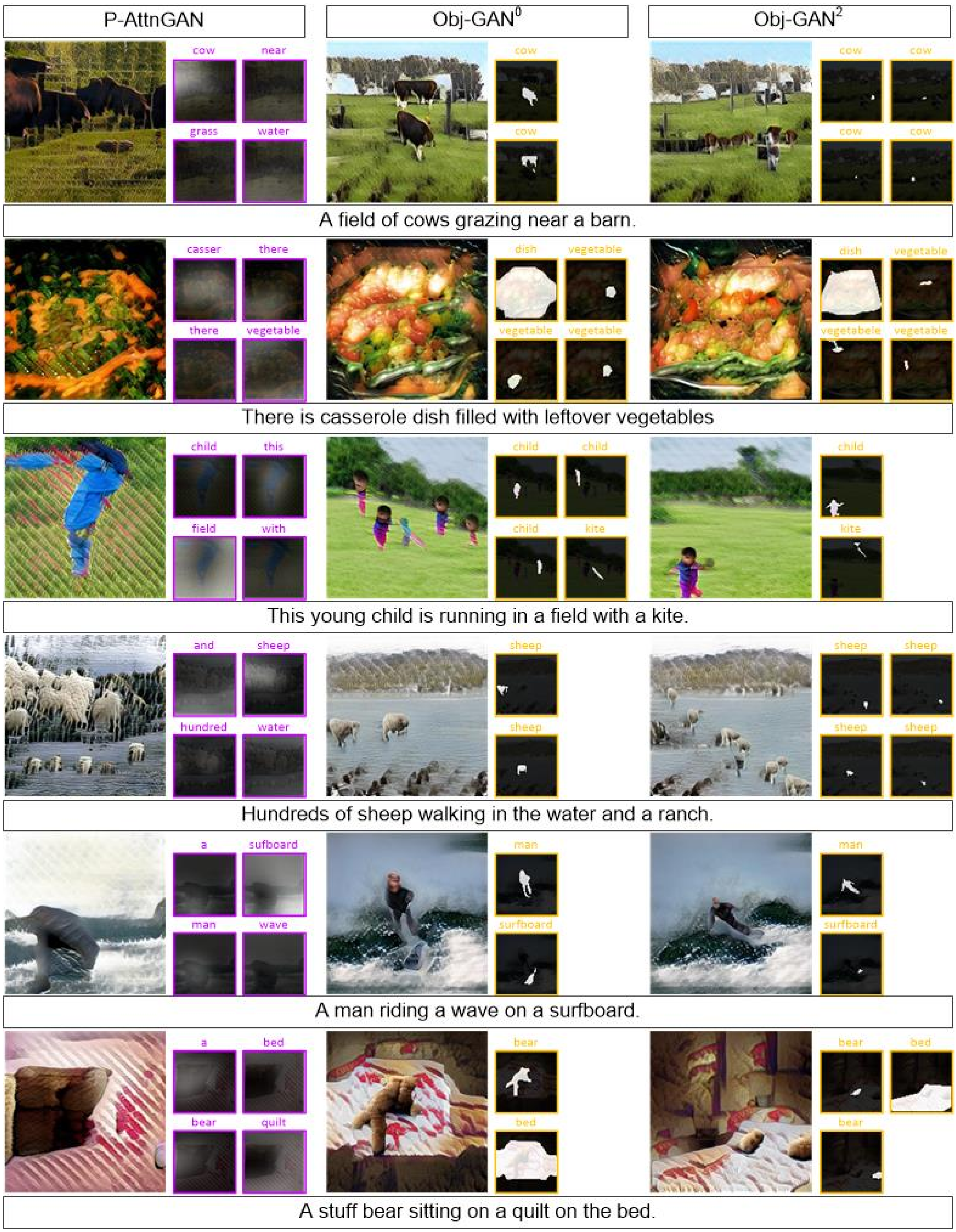
In this section, we show more images generated by our Obj-GAN and its ablative versions on the COCO dataset. There are more comparisons between the cases with or without spectral normalization in the discriminators, which can be found that there are no obvious improvement on the visual quality when we choose to use the spectral normalization.




Real Image
P-AttnGAN
P-AttnGAN w/ Lyt
Obj-GAN w/ SN
Obj-GAN
Real Image
P-AttnGAN
P-AttnGAN w/ Lyt
Obj-GAN w/ SN
Obj-GAN
A polo player on a horse following the ball on the ground.
Pizza with tomato and green olives being cut by a big knife.
A parking meter that reads "fail" and a white car across from it.
A cat is lying on top of a suitcase.
A group of scattered sheep move down a country path.
A slice of casserole with steamed broccoli on a plate.
A couple of people in a field flying kites.
Skier in mid jump on mountain with poles extended.
A train traveling down the rail road tracks.
A small two toned blue airplane flying.
The bed and desk in a modern hotel room.
A cat plays with grass outside in the garden.
A kitchen area with a sink, refrigerator, and stove.
Boys are playing a soccer game on a field.
A neatly made bed with a red sheet.
A large air plane on a run way.
Closeup of a cat in a bathroom sink.
A park bench on a grassy green hillside.
A living room has a beige couch, and carpet.
A coupe of people stand on a rocky hill top.

2. Attention Maps Generated by P-AttnGAN and Obj-GAN

We visualize more attention maps generated by P-AttnGAN and Obj-GAN as the supplementary for Figure 8 in the submission.



3. Results of Obj-GAN based on the Ground-truth Boxes and Shapes

(1) A glass table with a bottle and glass of wine next to a chair.

(2) A train sitting on some tracks next to a sidewalk.

(3) Soccer player wearing green and orange hitting soccer ball.

(4) A kitchen with a very messy counter space.

(5) The people are on the beach getting ready to surf.

(6) A jet airliner waits its turn on the runway.

(7) Two cows are grazing in a dirt field.

(8) A small lightweight airplane flying through the sky.

(9) A cow running in a field next to a dog.

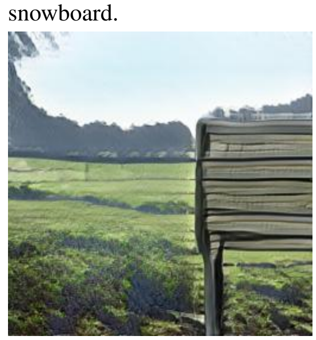
(10) Two people go into the water with their surfboards.

(11) A man in a helmet jumps a snowboard.

(12) A giraffe is standing all alone in a grassy area.

(13) The black dog is staring at the cat.

(14) A bunch of sheep are standing in a field.

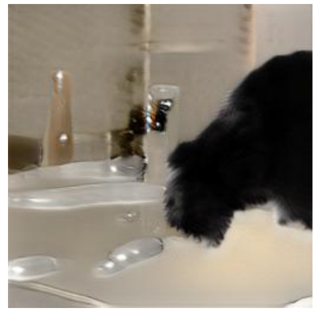


(15) A bench sitting on top of a lush green hillside.

(16) A polar bear playing in the water at a wild life enclosure.

(17) A man on a soccer field next to a ball.

(18) A dog sitting on a bench in front of a garden.



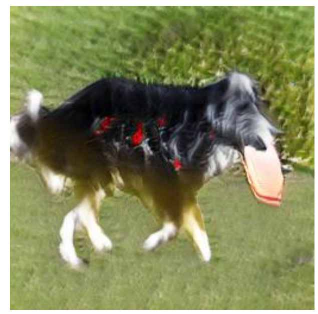
(19) A black cat drinking water out of a water faucet.

(20) A cat laying on a TV in the middle of the room.

(21) Four people on skis below a mountain taking a picture.

(22) A man in outdoor winter clothes holds a snowboard.

(23) A orange before and after it was cu.

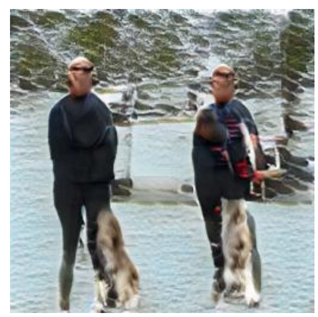


(24) A dog running with a frisbee in its mouth.

(25) A woman and a dog tussle over a frisbee.

(26) Man in a wetsuit on top of a blue and white surfboard.

(27) A white ship sails in the blue ocean water.

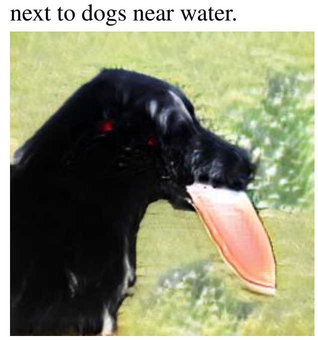


(28) A couple of men standing next to dogs near water.

(29) A man on a motorcycle in a carport.

(30) A group of people riding horses on a beach.

(31) A hipster wearing flood pants poses with his skateboard.



(32) A black dog holding a frisbee in its mouth.

(33) A big boat on the water near the shore.

(34) All the horses in the pen are grazing.

(35) A man riding a bike down the middle of a street.

(36) A bathroom with a sink and a toilet.

(37) A yellow school bus parked near a tree.

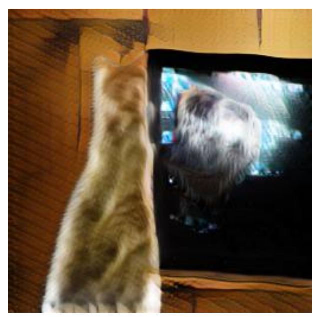
(38) A group of cows graze on some grass.

(39) A ship is sailing across an ocean filled with waves.

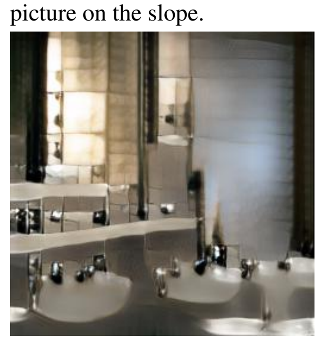
(40) Three skiers posing for a picture on the slope.

(41) A large green bus approaching a bus stop.

(42) A close view of a pizza, and a mug of beer.



(43) A cat is looking at a television displaying a dog in a cage.



(44) Three white sinks in a bathroom under mirrors.

(45) Three cranes standing on one leg in the water.

(46) A bear lying on a rock in its den, looking upward.

(47) Two bottles of soda sit near a sandwich.

(48) Someone on a snowboard coming to a stop.

4. Bi-LSTM Text Encoder, DAMSM and R-precision

We use the deep attentive multi-modal similarity model (DAMSM) proposed in [7], which learns a joint embedding of the image regions and words of a sentence in a common semantic space. The fine-grained conditional loss enforces the sub-region of the generated image to match the corresponding word in the sentence.

Bi-LSTM text encoder serves as the text encoder for both DAMSM and the box generator (see § 5). Bi-LSTM text encoder is a bi-directional LSTM [5] that extracts semantic vectors from the text description. In the Bi-LSTM, each word corresponds to two hidden states, one for each direction. Thus, we concatenate its two hidden states to represent the semantic meaning of a word. The feature matrix of all words is indicated by $\dot{e} \in \mathbb{R}^{D \times T_s}$. Its i^{th} column $\dot{e}_i$ is the feature vector for the i^{th} word. D is the dimension of the word vector and T_s is the number of words. Meanwhile, the last hidden states of the bi-directional LSTM are concatenated to be the global sentence vector, denoted by $\widehat{e} \in \mathbb{R}^D$. We present the network architectures for the Bi-LSTM text encoder in Table 1.

Table 1: The architecture of Bi-LSTM text encoder.

Layer Name	Hyperparameters
Embedding	num embeddings $=$ vocab size, embedding dim $= 300$
Dropout	prob $= 0.5$
LSTM	input size $= 300$, hidden size $(\frac{D}{2}) = 128$, num layers $= 1$, dropout prob $= 0.5$, bidirectional $=$ True

The image encoder is a convolutional neural network that maps images to semantic vectors. The intermediate layers of the CNN model learns local features of different regions of the image, while the later layers learn global features of the image. More specifically, the image encoder is built upon Inception-v3 model [6] pre-trained on ImageNet [4]. We first rescale the input image to be 299×299 pixels. And then, we extract the local feature matrix $f \in \mathbb{R}^{768 \times 289}$ (reshaped from 768×17×17) from "$mixed\_6e$" layer of Inception-v3. Each column of f is the feature vector of a local image region. 768 is the dimension of the local feature vector, and 289 is the number of regions in the image. Meanwhile, the global feature vector $\overline{f} \in \mathbb{R}^{2048}$ is extracted from the last average pooling layer of Inception-v3. Finally, we convert the image features to the common semantic space of text features by adding a new layer perceptron as shown in Eq. (1),

$$v = Wf; \quad \overline{v} = \overline{W}\,\overline{f}, \tag{1}$$

where $v \in \mathbb{R}^{D \times 289}$ and its i^{th} column v_i is the visual feature vector for the i^{th} image region; $\overline{v} \in \mathbb{R}^D$ is the visual feature vector for the whole image. While v_i is the local image feature vector that corresponds to the word embedding, $\overline{v}$ is the global feature vector that is related to the sentence embedding. D is the dimension of the multimodal (*i.e.*, image and text modalities) feature space. For efficiency, all parameters in layers built from Inception-v3 model are fixed, and the parameters in newly added layers are jointly learned with the rest of networks.

The fine-grained conditional loss is designed to learn the correspondence between image regions and words. However, it is difficult to obtain manual annotations. Actually, many words relate to concepts that may not easily be visually defined, such as *open* or *old*. One possible solution is to learn word-image correspondence in a semi-supervised manner, in which the only supervision is the correspondence between the entire image and the whole text description (a sequence of words).

We can first calculate the similarity matrix between all possible pairs of word and image region by Eq. (2),

$$s = \dot{e}^T\, v, \tag{2}$$

where $s \in \mathbb{R}^{T \times 289}$ and $s_{i,j}$ means the similarity between the i^{th} word and the j^{th} image region.

Generally, a sub-region of the image is described by none or several words of the text description, and it is not likely to be described by the whole sentence. Therefore, we normalize the similarity matrix by Eq. (3),

$$\overline{s}_{i,j} = \frac{\exp(s_{i,j})}{\sum_{k=0}^{T-1} \exp(s_{k,j})} \tag{3}$$

Second, we build an attention model to compute a context vector for each word (query). The context vector c_i is a dynamic representation of image regions related to the i^{th} word of the text description. It is computed as the weighted sum over all visual feature vectors,

$$c_i = \sum_{j=0}^{288} \alpha_j v_j, \tag{4}$$

where we define the weight α_j via Eq. (5),

$$\alpha_j = \frac{\exp(\gamma_1 \overline{s}_{i,j})}{\sum_{k=0}^{288} \exp(\gamma_1 \overline{s}_{i,k})} \tag{5}$$

Here, γ_1 is a factor that decides how much more attention is paid to features of its relevant regions when computing the context vector for a word.

Finally, we define the relevance between the i^{th} word and the image using the cosine similarity between c_i and $\dot{e}_i$, *i.e.*, $R(c_i, \dot{e}_i) = (c_i^T \dot{e}_i)/(||c_i||||\dot{e}_i||)$. The relevance between the entire image (Q) and the whole text description (U) is computed by Eq. (6),

$$R(Q,U) = \log\Big(\sum_{i=1}^{T-1} \exp(\gamma_2 R(c_i, \dot{e}_i))\Big)^{\frac{1}{\gamma_2}}, \tag{6}$$

where γ_2 is a factor that determines how much to magnify the importance of the most relevant word-image pair. When $\gamma_2 \to \infty$, $R(Q,U)$ approximates to $\max_{i=1}^{T-1} R(c_i, \dot{e}_i)$.

For a text-image pair, we can compute the posterior probability of the text description (U) being matching with the image (Q) via,

$$P(U|Q) = \frac{\exp(\gamma_3 R(Q,U))}{\sum_{U' \in \mathbb{U}} \exp(\gamma_3 R(Q,U'))}, \tag{7}$$

where γ_3 is a smoothing factor determined by experiments. $\mathbb{U}$ denotes a minibatch of M text descriptions, in which only one description U^+ matches the image Q. Thus, for each image, there are $M-1$ mismatching text descriptions. The objective function is to learn the model parameters Λ by minimizing the negative log posterior probability that the images are matched with their corresponding text descriptions (ground truth),

$$\mathcal{L}_1^w(\Lambda) = -\log \prod_{Q \in \mathbb{Q}} P(U^+|Q), \tag{8}$$

where 'w' stands for "word".

Symmetrically, we can compute,

$$\mathcal{L}_2^w(\Lambda) = -\log \prod_{U \in \mathbb{U}} P(Q^+|U), \tag{9}$$

where $P(Q|U) = \frac{\exp(\gamma_3 R(Q,U))}{\sum_{Q' \in \mathbb{Q}} \exp(\gamma_3 R(Q',U))}$.

If we redefine Eq. (6) by $R(Q,U) = \big(\overline{v}^T \widehat{e}\big)/\big(||\overline{v}||||\widehat{e}||\big)$ and substitute it to Eq. (7),Eq. (8), Eq. (9), we can obtain loss functions $\mathcal{L}_1^s$ and $\mathcal{L}_2^s$ (where 's' stands for "sentence") using the sentence embedding $\widehat{e}$ and the global visual vector $\overline{v}$.

The fine-grained conditional loss is defined via Eq. (10),

$$\mathcal{L}_{DAMSM} = \mathcal{L}_1^w + \mathcal{L}_2^w + \mathcal{L}_1^s + \mathcal{L}_2^s \tag{10}$$

The DAMSM is pre-trained by minimizing $\mathcal{L}_{DAMSM}$ using real image-text pairs. Since the size of images for pre-training DAMSM is not limited by the size of images that can be generated, real images of size 299×299 are utilized. Furthermore, the pre-trained DAMSM can provide visually-discriminative word features and a stable fine-grained conditional loss for the attention generative network.

The R-precision score. The DAMSM model is also used to compute the R-precision score. If there are R relevant documents for a query, we examine the top R ranked retrieval results of a system, and find that r are relevant, and then by definition, the R-precision (and also the precision and recall) is r/R. More specifically, we use generated images to query their corresponding text descriptions. First, the image encoder and Bi-LSTM text encoder learned in our pre-trained DAMSM are utilized to extract features of the generated images and the given text descriptions. Then, we compute cosine similarities between the image features and the text features. Finally, we rank candidates text descriptions for each image in descending similarity and find the top r relevant descriptions for computing the R-precision.

5. Network Architectures for Semantic Layout Generation

Box generator. We design our box generator by improving the one in [1] to be attentive. We denote the bounding box of the t-th object as $B_t = (b_t^x, b_t^y, b_t^w, b_t^h, \boldsymbol{l}_t)$. Then, we formulate the joint probability of sampling B_t from the box generator as

$$p(b_t^x, b_t^y, b_t^w, b_t^h, \boldsymbol{l}_t) = p(\boldsymbol{l}_t) p(b_t^x, b_t^y, b_t^w, b_t^h | \boldsymbol{l}_t). \tag{11}$$

We implement $p(\boldsymbol{l}_t)$ as a categorical distribution, and implement $p(b_t^x, b_t^y, b_t^w, b_t^h | \boldsymbol{l}_t)$ as a mixture of quadravariate Gaussians. As described in [1], in order to reduce the parameter space, we decompose the box coordinate probability as $p(b_t^x, b_t^y, b_t^w, b_t^h | \boldsymbol{l}_t) = p(b_t^x, b_t^y | \boldsymbol{l}_t) p(b_t^w, b_t^h | b_t^x, b_t^y, \boldsymbol{l}_t)$, and approximate it with two bivariate Gasussian mixtures by

$$p(b_t^x, b_t^y | \boldsymbol{l}_t) = \sum_{k=1}^{K} \pi_{t,k}^{xy} \mathcal{N}(b_t^x, b_t^y; \boldsymbol{\mu}_{t,k}^{xy}, \Sigma_{t,k}^{xy}), \tag{12}$$

$$p(b_t^w, b_t^h | b_t^x, b_t^y, \boldsymbol{l}_t) = \sum_{k=1}^{K} \pi_{t,k}^{wh} \mathcal{N}(b_t^w, b_t^h; \boldsymbol{\mu}_{t,k}^{wh}, \Sigma_{t,k}^{wh}). \tag{13}$$

In practice, as in [1], we implement the box generator within a encoder-decoder framework. The encoder is the Bi-LSTM text encoder as mentioned in § 4. The Gaussian Mixture Model (GMM) parameters for Eq. (11) are obtained from the decoder LSTM outputs. Given text encoder's final hidden state $h_{T_s}^{\text{Enc}} \in \mathbb{R}^D$ and output $H^{\text{Enc}} \in \mathbb{R}^{T_s \times D}$, we initialize the decoder's initial hidden state h_0 with $h_{T_s}^{\text{Enc}}$. As for H^{Enc}, we use it to compute the contextual input z_t for the decoder:

$$z_t = \sum_{i=1}^{T_s} \alpha_i h_i^{\text{Enc}}, \text{ with } \alpha_i = W_v \cdot (W_\alpha [h_{t-1}, h_i^{\text{Enc}}]), \tag{14}$$

where W_v is a learnable parameter, W_α is the parameter of a linear transformation, and $\cdot$ and $[\cdot, \cdot]$ represent the dot product and concatenation operation, respectively.

Then, the calculation of GMM parameters are shown as follows:

$$\begin{aligned} [h_t, c_t] &= \text{LSTM}([B_{t-1}, z_t]; [h_{t-1}, c_{t-1}]), &(15)\\ \boldsymbol{l}_t &= W^l h_t + \mathbf{b}^l, &(16)\\ \boldsymbol{\theta}_t^{xy} &= W^{xy} [h_t, \boldsymbol{l}_t] + \mathbf{b}^{xy}, &(17)\\ \boldsymbol{\theta}_t^{wh} &= W^{wh} [h_t, \boldsymbol{l}_t, b_x, b_y] + \mathbf{b}^{wh}, &(18) \end{aligned}$$

where $\boldsymbol{\theta}_t^{\cdot} = [\boldsymbol{\pi}_{t,1:K}^{\cdot}, \boldsymbol{\mu}_{t,1:K}^{\cdot}, \boldsymbol{\Sigma}_{t,1:K}^{\cdot}]$ are the parameters for GMM concatenated to a vector. We use the same Adam optimizer and training hyperparameters (*i.e.*, learning rate 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$) as in [1].

Shape generator. We implement the shape generator in [1] with almost the same architecture except the upsample block. In [1], the upsample block is designed as [convtranspose 4×4 (pad 1, stride 2) - Instance Normalization - ReLU]. We discovered that the usage of convtranspose would lead to unstable training which is reflected by the frequent severe grid artifacts. To this end, we replace this upsample block with that in our image generator (see Table 2) by switching the batch normalization to the instance one.

6. Network Architectures for Image Generation

We present the network architecture for image generators in Table 3 and the network architectures for discriminators in Table 4, Table 5 and Table 6. They are built with basic blocks defined in Table 2. We set the hyperparameters of the network structures as: $N_g = 48$, $N_d = 96$, $N_c = 80$, $N_e = 256$, $N_l = 50$, $m_0 = 7$, $m_1 = 3$, and $m_2 = 3$.

We employ an Adam optimizer for the generators with learning rate 0.0002, $\beta_1 = 0.5$ and $\beta_2 = 0.999$. For each discriminator, we also employ an Adam optimizer with the same hyperparameters.

We design the object-wise discriminators for small objects and large objects, respectively. We specify that if the maximum of width or height of an object is greater than one-third of the image size, then this object is large; otherwise, it is small.

Table 2: The basic blocks for architecture design. ("-" connects two consecutive layers; "+" means element-wise addition between two layers.)

Name	Operations / Layers
Interpolating (k)	Nearest neighbor upsampling layer (up-scaling the spatial size by k)
Upsampling (k)	Interpolating (2) - convolution 3×3 (stride 1, padding 1, decreasing $\sharp$channels to k) - Batch Normalization (BN) - Gated Linear Unit (GLU).
Downsampling (k)	In Gs: convolution 3×3 (stride 2, increasing $\sharp$channels to k) - BN - LeakyReLU. In Ds, the convolutional kernel size is 4. In the first block of Ds, BN is not applied.
Downsampling w/ SN (k)	Convolution 4×4 (spectral normalized, stride 2, increasing $\sharp$channels to k) - BN - LeakyReLU. In the first block of Ds, BN is not applied.
Concat	Concatenate input tensors along the channel dimension.
Residual	Input + [Reflection Pad (RPad) 1 - convolution 3×3 (stride 1, doubling $\sharp$channels) - Instance Normalization (IN) - GLU - RPad 1 - convolution 3×3 (stride 1, keeping $\sharp$channels) - IN].
FC	At the beginning of Gs: fully connected layer - BN - GLU - reshape to 3D tensor.
FC w/ SN (k)	Fully connected layer (spectral normalized, changing $\sharp$channels to k).
Outlogits	Convolution 4×4 (stride 2, decreasing $\sharp$channels to 1) - sigmoid.
Repeat ($k \times k$)	Copy a vector $k \times k$ times.
Fmap Sum	Summing the two input feature maps element-wisely.
Fmap Mul	Multiplying the two input feature maps element-wisely.
Avg Pool (k)	Average pooling along the k-th dimension.
Conv 3×3 (k)	In Gs: convolution 3×3 (stride 1, padding 1, changing $\sharp$channels to k) - Tanh. In Ds, convolution 3×3 (stride 1, padding 1, changing $\sharp$channels to k) - BN - LeakyReLU.
Conv 4×4 w/ SN	Convolution 4×4 (spectral normalized, stride 2, keeping $\sharp$channels).
Conv 1×1 w/ SN	Convolution 1×1 (spectral normalized, stride 1, decreasing $\sharp$channels to 1).
F^{ca}	Conditioning augmentation that converts the sentence embedding $\widehat{e}$ to the conditioning vector $\overline{e}$: fully connected layer - ReLU.
$F^{\text{pat-attn}}$	Grid attention module. Refer to the paper for more details.
$F^{\text{obj-attn}}$	Object-driven attention module. Refer to the paper for more details.
$F^{\text{lab-distr}}$	Label distribution module. Refer to the paper for more details.
Shape Encoder (k)	RPad 1 - convolution 3×3 (stride 1, decreasing $\sharp$channels to k) - IN - LeakyReLU.
Shape Encoder w/ SN (k)	RPad 1 - convolution 3×3 (spectral normalized, stride 1, decreasing $\sharp$channels to k) - IN - LeakyReLU.
ROI Encoder	Convolution 4×4 (stride 1, padding 1, decreasing $\sharp$channels to $N_d * 4$) - LeakyReLU.
ROI Encoder w/ SN	Convolution 4×4 (spectral normalized, stride 1, padding 1, decreasing $\sharp$channels to $N_d * 4$) - LeakyReLU.
ROI Align (k)	Pooling $k \times k$ feature maps for ROI.

Table 3: The structure for generators of Obj-GAN.

Stage	Name	Input Tensors	Output Tensors
G_0	FC	100-dimensional z, and F^{ca}	$8 \times 8 \times 4N_g$
	Upsampling ($2N_g$)	$8 \times 8 \times 4N_g$	$16 \times 16 \times 2N_g$
	Upsampling (N_g)	$16 \times 16 \times 2N_g$	c ($32 \times 32 \times N_g$)
	Shape Encoder ($\frac{1}{2}N_g$)	M^0 ($64 \times 64 \times N_c$)	$64 \times 64 \times \frac{1}{2}N_g$
	Downsampling (N_g)	$64 \times 64 \times \frac{1}{2}N_g$	u_0 ($32 \times 32 \times N_g$)
	Concat	$c, u_0, F^{\text{obj-attn}}, F^{\text{lab-distr}}$	$32 \times 32 \times (3N_g + N_l)$
	m_0 Residual	$32 \times 32 \times (3N_g + N_l)$	$32 \times 32 \times (3N_g + N_l)$
	Upsampling (N_g)	$32 \times 32 \times (3N_g + N_l)$	h_0 ($64 \times 64 \times N_g$)
	Conv 3×3 (3)	h_0	x_0 ($64 \times 64 \times 3$)
G_1	Shape Encoder ($\frac{1}{2}N_g$)	M^1 ($128 \times 128 \times N_c$)	$128 \times 128 \times \frac{1}{2}N_g$
	Downsampling (N_g)	$128 \times 128 \times \frac{1}{2}N_g$	u_1 ($64 \times 64 \times N_g$)
	Fmap Sum	h_0, u_1	h_0 ($64 \times 64 \times N_g$)
	Concat	$F^{\text{pat-attn}}, h_0, F^{\text{obj-attn}}, F^{\text{lab-distr}}$	$64 \times 64 \times (3N_g + N_l)$
	m_1 Residual	$64 \times 64 \times (3N_g + N_l)$	$64 \times 64 \times (3N_g + N_l)$
	Upsampling (N_g)	$64 \times 64 \times (3N_g + N_l)$	h_1 ($128 \times 128 \times N_g$)
	Conv 3×3 (3)	h_1	x_1 ($128 \times 128 \times 3$)
G_2	Shape Encoder ($\frac{1}{2}N_g$)	M^2 ($256 \times 256 \times N_c$)	$256 \times 256 \times \frac{1}{2}N_g$
	Downsampling (N_g)	$256 \times 256 \times \frac{1}{2}N_g$	u_2 ($128 \times 128 \times N_g$)
	Fmap Sum	h_1, u_2	h_1 ($128 \times 128 \times N_g$)
	Concat	$F^{\text{pat-attn}}, h_1, F^{\text{obj-attn}}, F^{\text{lab-distr}}$	$128 \times 128 \times (3N_g + N_l)$
	m_2 Residual	$128 \times 128 \times (3N_g + N_l)$	$128 \times 128 \times (3N_g + N_l)$
	Upsampling (N_g)	$128 \times 128 \times (3N_g + N_l)$	h_2 ($256 \times 256 \times N_g$)
	Conv 3×3 (3)	h_2	x_2 ($256 \times 256 \times 3$)

Table 4: The structure for patch-wise discriminators of Obj-GAN. $\bar{e}$ is output by F^{ca}

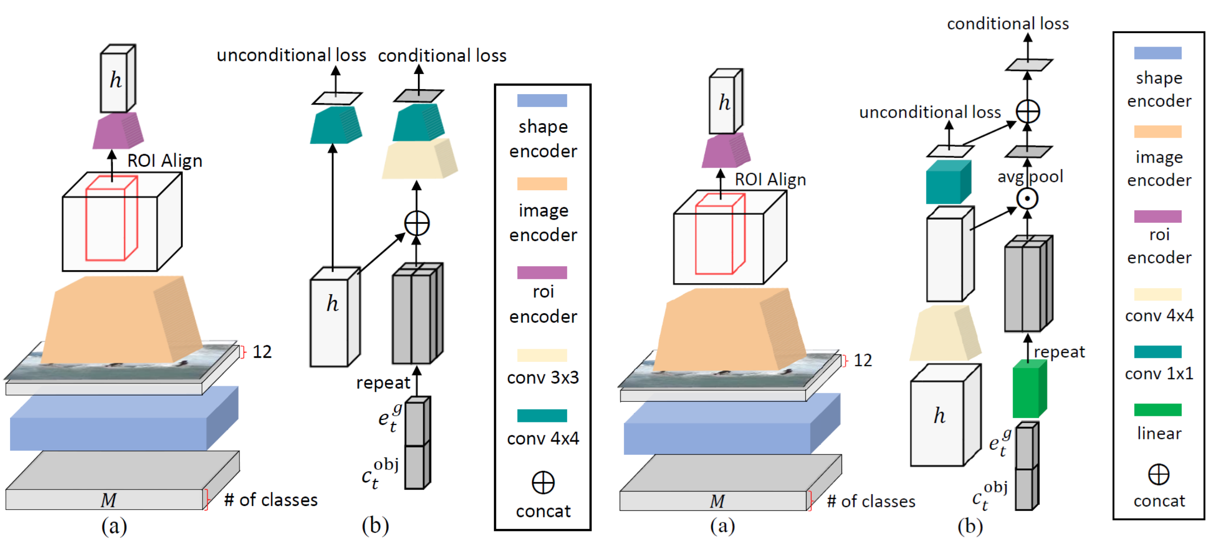
Stage	Name	Input Tensors	Output Tensors
D_0	Downsampling (N_d)	x_0 ($64 \times 64 \times 3$)	$32 \times 32 \times N_d$
	Downsampling ($2N_d$)	$32 \times 32 \times N_d$	$16 \times 16 \times 2N_d$
	Downsampling ($4N_d$)	$16 \times 16 \times 2N_d$	$8 \times 8 \times 4N_d$
	Downsampling ($8N_d$)	$8 \times 8 \times 4N_d$	h_0 ($4 \times 4 \times 8N_d$)
	Repeat (4×4)	$\bar{e}$ (N_e)	$4 \times 4 \times N_e$
	Concat - Conv 3×3 ($8N_d$)	$h_0, 4 \times 4 \times N_e$	he_0 ($4 \times 4 \times 8N_d$)
	Outlogits (unconditional loss)	h_0	1
	Outlogits (conditional loss)	he_0	1
D_1	Downsampling (N_d)	x_1 ($128 \times 128 \times 3$)	$64 \times 64 \times N_d$
	Downsampling ($2N_d$)	$64 \times 64 \times N_d$	$32 \times 32 \times 2N_d$
	Downsampling ($4N_d$)	$32 \times 32 \times 2N_d$	$16 \times 16 \times 4N_d$
	Downsampling ($8N_d$)	$16 \times 16 \times 4N_d$	h_1 ($8 \times 8 \times 8N_d$)
	Repeat (8×8)	$\bar{e}$ (N_e)	$8 \times 8 \times N_e$
	Concat - Conv 3×3 ($8N_d$)	$h_1, 8 \times 8 \times N_e$	he_1 ($8 \times 8 \times 8N_d$)
	Outlogits (unconditional loss)	h_1	3×3
	Outlogits (conditional loss)	he_1	3×3
D_2	Downsampling (N_d)	x_2 ($256 \times 256 \times 3$)	$128 \times 128 \times N_d$
	Downsampling ($2N_d$)	$128 \times 128 \times N_d$	$64 \times 64 \times 2N_d$
	Downsampling ($4N_d$)	$64 \times 64 \times 2N_d$	$32 \times 32 \times 4N_d$
	Downsampling ($8N_d$)	$32 \times 32 \times 4N_d$	h_2 ($16 \times 16 \times 8N_d$)
	Repeat (16×16)	$\bar{e}$ (N_e)	$16 \times 16 \times N_e$
	Concat - Conv 3×3 ($8N_d$)	$h_2, 16 \times 16 \times N_e$	he_2 ($16 \times 16 \times 8N_d$)
	Outlogits (unconditional loss)	h_2	7×7
	Outlogits (conditional loss)	he_2	7×7

Table 5: The structure for shape discriminators of Obj-GAN.

Stage	Name	Input Tensors	Output Tensors
D_0	Shape Encoder ($\frac{1}{8}N_d$)	M^0 ($64 \times 64 \times N_c$)	$64 \times 64 \times \frac{1}{8}N_d$
	Concat	x_0 $(64 \times 64 \times 3), 64 \times 64 \times \frac{1}{8}N_d$	$64 \times 64 \times (3 + \frac{1}{8}N_d)$
	Downsampling (N_d)	$64 \times 64 \times (3 + \frac{1}{8}N_d)$	$32 \times 32 \times N_d$
	Downsampling ($2N_d$)	$32 \times 32 \times N_d$	$16 \times 16 \times 2N_d$
	Downsampling ($4N_d$)	$16 \times 16 \times 2N_d$	$8 \times 8 \times 4N_d$
	Downsampling ($8N_d$)	$8 \times 8 \times 4N_d$	h_0 ($4 \times 4 \times 8N_d$)
	Outlogits (unconditional loss)	h_0	1
D_1	Shape Encoder ($\frac{1}{8}N_d$)	M^1 ($128 \times 128 \times N_c$)	$128 \times 128 \times \frac{1}{8}N_d$
	Concat	x_1 $(128 \times 128 \times 3), 128 \times 128 \times \frac{1}{8}N_d$	$128 \times 128 \times (3 + \frac{1}{8}N_d)$
	Downsampling (N_d)	$128 \times 128 \times (3 + \frac{1}{8}N_d)$	$64 \times 64 \times N_d$
	Downsampling ($2N_d$)	$64 \times 64 \times N_d$	$32 \times 32 \times 2N_d$
	Downsampling ($4N_d$)	$32 \times 32 \times 2N_d$	$16 \times 16 \times 4N_d$
	Downsampling ($8N_d$)	$16 \times 16 \times 4N_d$	h_1 ($8 \times 8 \times 8N_d$)
	Outlogits (unconditional loss)	h_1	3×3
D_2	Shape Encoder ($\frac{1}{8}N_d$)	M^2 ($256 \times 256 \times N_c$)	$256 \times 256 \times \frac{1}{8}N_d$
	Concat	x_2 $(256 \times 256 \times 3), 256 \times 256 \times \frac{1}{8}N_d$	$256 \times 256 \times (3 + \frac{1}{8}N_d)$
	Downsampling (N_d)	$256 \times 256 \times (3 + \frac{1}{8}N_d)$	$128 \times 128 \times N_d$
	Downsampling ($2N_d$)	$128 \times 128 \times N_d$	$64 \times 64 \times 2N_d$
	Downsampling ($4N_d$)	$64 \times 64 \times 2N_d$	$32 \times 32 \times 4N_d$
	Downsampling ($8N_d$)	$32 \times 32 \times 4N_d$	h_2 ($16 \times 16 \times 8N_d$)
	Outlogits (unconditional loss)	h_2	7×7

Table 6: The structure for object-wise discriminators of Obj-GAN. c^{obj} represents the intermediate context vectors of $F^{\text{obj-attn}}$, and e^{g} represents the embedding vectors the class labels.

Stage	Name	Input Tensors	Output Tensors
small	Interpolating (2)	M^2 $(256 \times 256 \times N_c)$	$512 \times 512 \times N_c$
	Interpolating (2)	x^2 $(256 \times 256 \times 3)$	$512 \times 512 \times 3$
	Shape Encoder ($\frac{1}{8}N_d$)	$512 \times 512 \times N_c$	$512 \times 512 \times \frac{1}{8}N_d$
	Concat	$512 \times 512 \times 3, 512 \times 512 \times \frac{1}{8}N_d$	$512 \times 512 \times (3 + \frac{1}{8}N_d)$
	Downsampling (N_d)	$512 \times 512 \times (3 + \frac{1}{8}N_d)$	$256 \times 256 \times N_d$
	Downsampling ($2N_d$)	$256 \times 256 \times N_d$	$128 \times 128 \times 2N_d$
	Downsampling ($4N_d$)	$128 \times 128 \times 2N_d$	$64 \times 64 \times 4N_d$
	ROI Align (5)	$64 \times 64 \times 4N_d$	$N_{\text{small}} \times 5 \times 5 \times 4N_d$
	ROI Encoder (5)	$N_{\text{small}} \times 5 \times 5 \times 4N_d$	h $(N_{\text{small}} \times 4 \times 4 \times 4N_d)$
	Repeat (4×4)	c^{obj} $(N_{\text{small}} \times N_g)$	$N_{\text{small}} \times 4 \times 4 \times N_g$
	Repeat (4×4)	e^{g} $(N_{\text{small}} \times N_l)$	$N_{\text{small}} \times 4 \times 4 \times N_l$
	Concat - Conv 3×3 ($4N_d$)	$h, N_{\text{small}} \times 4 \times 4 \times N_g, N_{\text{small}} \times 4 \times 4 \times N_l$	hc $(N_{\text{small}} \times 4 \times 4 \times 4N_d)$
	Outlogits (unconditional loss)	h	N_{small}
	Outlogits (conditional loss)	hc	N_{small}
large	Interpolating (2)	M^2 $(256 \times 256 \times N_c)$	$512 \times 512 \times N_c$
	Interpolating (2)	x^2 $(256 \times 256 \times 3)$	$512 \times 512 \times 3$
	Shape Encoder ($\frac{1}{8}N_d$)	$512 \times 512 \times N_c$	$512 \times 512 \times \frac{1}{8}N_d$
	Concat	$512 \times 512 \times 3, 512 \times 512 \times \frac{1}{8}N_d$	$512 \times 512 \times (3 + \frac{1}{8}N_d)$
	Downsampling (N_d)	$512 \times 512 \times (3 + \frac{1}{8}N_d)$	$256 \times 256 \times N_d$
	Downsampling ($2N_d$)	$256 \times 256 \times N_d$	$128 \times 128 \times 2N_d$
	Downsampling ($4N_d$)	$128 \times 128 \times 2N_d$	$64 \times 64 \times 4N_d$
	Downsampling ($8N_d$)	$64 \times 64 \times 4N_d$	$32 \times 32 \times 8N_d$
	ROI Align (5)	$32 \times 32 \times 8N_d$	$N_{\text{large}} \times 5 \times 5 \times 8N_d$
	ROI Encoder (5)	$N_{\text{large}} \times 5 \times 5 \times 8N_d$	h $(N_{\text{large}} \times 4 \times 4 \times 4N_d)$
	Repeat (4×4)	c^{obj} $(N_{\text{large}} \times N_g)$	$N_{\text{large}} \times 4 \times 4 \times N_g$
	Repeat (4×4)	e^{g} $(N_{\text{large}} \times N_l)$	$N_{\text{large}} \times 4 \times 4 \times N_l$
	Concat - Conv 3×3 ($4N_d$)	$h, N_{\text{large}} \times 4 \times 4 \times N_g, N_{\text{large}} \times 4 \times 4 \times N_l$	hc $(N_{\text{large}} \times 4 \times 4 \times 4N_d)$
	Outlogits (unconditional loss)	h	N_{large}
	Outlogits (conditional loss)	hc	N_{large}



(49) Object-wise discriminator.

(50) Object-wise spectral normalized projection discriminator.

Figure 1: The comparison between the object-wise discriminator and its spectral normalized projection version. (a) extracts the region feature based on the Fast R-CNN model. (b) determines whether the t-th object is realistic (consistent with its label e_t^{g} and text context information c_t^{obj}) or not.

7. Network Architectures for Spectral Normalized Projection Discriminators

We combine our discriminators above with the spectral normalized projection discriminator in [2, 3]. The difference between the object-wise discriminator and the object-wise spectral normalized projection discriminator is illustrated in Figure 1. We present detailed network architectures of the spectral normalized projection discriminators in Table 7, Table 8 and

Table 9, with basic blocks defined in Table 2.

Table 7: The structure for patch-wise spectral normalized projection discriminators of Obj-GAN. $\overline{e}$ is output by F^{ca}

Stage	Name	Input Tensors	Output Tensors
D_0	Downsampling w/ SN (N_d)	x_0 ($64 \times 64 \times 3$)	$32 \times 32 \times N_d$
	Downsampling w/ SN ($2N_d$)	$32 \times 32 \times N_d$	$16 \times 16 \times 2N_d$
	Downsampling w/ SN ($4N_d$)	$16 \times 16\ 2N_d$	$8 \times 8 \times 4N_d$
	Downsampling w/ SN ($8N_d$)	$8 \times 8 \times 4N_d$	$4 \times 4 \times 8N_d$
	Conv 4×4 w/ SN	$4 \times 4 \times 8N_d$	h_0 ($8N_d$)
	FC w/ SN ($8N_d$)	$\overline{e}$ (N_e)	c_0 ($8N_d$)
	Fmap Mul - Avg Pool (0)	h_0, c_0	hc_0 (1)
	Conv 1×1 w/ SN (unconditional loss)	h_0	o_0^{uncond} (1)
	Fmap Sum (conditional loss)	$o_0^{\mathrm{uncond}}, hc_0$	o_0^{cond} (1)
D_1	Downsampling w/ SN (N_d)	x_1 ($128 \times 128 \times 3$)	$64 \times 64 \times N_d$
	Downsampling w/ SN ($2N_d$)	$64 \times 64 \times N_d$	$32 \times 32 \times 2N_d$
	Downsampling w/ SN ($4N_d$)	$32 \times 32 \times 2N_d$	$16 \times 16 \times 4N_d$
	Downsampling w/ SN ($8N_d$)	$16 \times 16 \times 4N_d$	$8 \times 8 \times 8N_d$
	Conv 4×4 w/ SN	$8 \times 8 \times 8N_d$	h_1 ($3 \times 3 \times 8N_d$)
	FC w/ SN ($8N_d$)	$\overline{e}$ (N_e)	$8N_d$
	Repeat (3×3)	$8N_d$	c_1 ($3 \times 3 \times 8N_d$)
	Fmap Mul - Avg Pool (2)	h_1, c_1	hc_1 (3×3)
	Conv 1×1 w/ SN (unconditional loss)	h_1	o_1^{uncond} (3×3)
	Fmap Sum (conditional loss)	$o_1^{\mathrm{uncond}}, hc_1$	o_1^{cond} (3×3)
D_2	Downsampling w/ SN (N_d)	x_2 ($256 \times 256 \times 3$)	$128 \times 128 \times N_d$
	Downsampling w/ SN ($2N_d$)	$128 \times 128 \times N_d$	$64 \times 64 \times 2N_d$
	Downsampling w/ SN ($4N_d$)	$64 \times 64 \times 2N_d$	$32 \times 32 \times 4N_d$
	Downsampling w/ SN ($8N_d$)	$32 \times 32 \times 4N_d$	$16 \times 16 \times 8N_d$
	Conv 4×4 w/ SN	$16 \times 16 \times 8N_d$	h_2 ($7 \times 7 \times 8N_d$)
	FC w/ SN ($8N_d$)	$\overline{e}$ (N_e)	$8N_d$
	Repeat (7×7)	$8N_d$	c_2 ($7 \times 7 \times 8N_d$)
	Fmap Mul - Avg Pool (2)	h_2, c_2	hc_2 (7×7)
	Conv 1×1 w/ SN (unconditional loss)	h_2	o_2^{uncond} (7×7)
	Fmap Sum (conditional loss)	$o_2^{\mathrm{uncond}}, hc_2$	o_2^{cond} (7×7)

Table 8: The structure for shape spectral normalized projection discriminators of Obj-GAN.

Stage	Name	Input Tensors	Output Tensors
D_0	Shape Encoder w/ SN ($\frac{1}{8}N_d$)	M^0 ($64 \times 64 \times N_c$)	$64 \times 64 \times \frac{1}{8}N_d$
	Concat	x_0 $(64 \times 64 \times 3), 64 \times 64 \times \frac{1}{8}N_d$	$64 \times 64 \times (3 + \frac{1}{8}N_d)$
	Downsampling w/ SN (N_d)	$64 \times 64 \times (3 + \frac{1}{8}N_d)$	$32 \times 32 \times N_d$
	Downsampling w/ SN ($2N_d$)	$32 \times 32 \times N_d$	$16 \times 16 \times 2N_d$
	Downsampling w/ SN ($4N_d$)	$16 \times 16 \times 2N_d$	$8 \times 8 \times 4N_d$
	Downsampling w/ SN ($8N_d$)	$8 \times 8 \times 4N_d$	$4 \times 4 \times 8N_d$
	Conv 4×4 w/ SN	$4 \times 4 \times 8N_d$	h_0 ($8N_d$)
	Conv 1×1 w/ SN (unconditional loss)	h_0	1
D_1	Shape Encoder w/ SN ($\frac{1}{8}N_d$)	M^1 ($128 \times 128 \times N_c$)	$128 \times 128 \times \frac{1}{8}N_d$
	Concat	x_1 $(128 \times 128 \times 3), 128 \times 128 \times \frac{1}{8}N_d$	$128 \times 128 \times (3 + \frac{1}{8}N_d)$
	Downsampling w/ SN (N_d)	$128 \times 128 \times (3 + \frac{1}{8}N_d)$	$64 \times 64 \times N_d$
	Downsampling w/ SN ($2N_d$)	$64 \times 64 \times N_d$	$32 \times 32 \times 2N_d$
	Downsampling w/ SN ($4N_d$)	$32 \times 32 \times 2N_d$	$16 \times 16 \times 4N_d$
	Downsampling w/ SN ($8N_d$)	$16 \times 16 \times 4N_d$	$8 \times 8 \times 8N_d$
	Conv 4×4 w/ SN	$8 \times 8 \times 8N_d$	h_1 ($3 \times 3 \times 8N_d$)
	Conv 1×1 w/ SN (unconditional loss)	h_1	3×3
D_2	Shape Encoder w/ SN ($\frac{1}{8}N_d$)	M^2 ($256 \times 256 \times N_c$)	$256 \times 256 \times \frac{1}{8}N_d$
	Concat	x_2 $(256 \times 256 \times 3), 256 \times 256 \times \frac{1}{8}N_d$	$256 \times 256 \times (3 + \frac{1}{8}N_d)$
	Downsampling w/ SN (N_d)	$256 \times 256 \times (3 + \frac{1}{8}N_d)$	$128 \times 128 \times N_d$
	Downsampling w/ SN ($2N_d$)	$128 \times 128 \times N_d$	$64 \times 64 \times 2N_d$
	Downsampling w/ SN ($4N_d$)	$64 \times 64 \times 2N_d$	$32 \times 32 \times 4N_d$
	Downsampling w/ SN ($8N_d$)	$32 \times 32 \times 4N_d$	$16 \times 16 \times 8N_d$
	Conv 4×4 w/ SN	$16 \times 16 \times 8N_d$	h_2 ($7 \times 7 \times 8N_d$)
	Conv 1×1 w/ SN (unconditional loss)	h_2	7×7

Table 9: The structure for object-wise spectral normalized projection discriminators of Obj-GAN. c^{obj} represents the intermediate context vectors of $F^{\text{obj-attn}}$, and e^{g} represents the embedding vectors the class labels.

Stage	Name	Input Tensors	Output Tensors
small	Interpolating (2)	M^2 $(256 \times 256 \times N_c)$	$512 \times 512 \times N_c$
	Interpolating (2)	x^2 $(256 \times 256 \times 3)$	$512 \times 512 \times 3$
	Shape Encoder w/ SN ($\frac{1}{8}N_d$)	$512 \times 512 \times N_c$	$512 \times 512 \times \frac{1}{8}N_d$
	Concat	$512 \times 512 \times 3, 512 \times 512 \times \frac{1}{8}N_d$	$512 \times 512 \times (3 + \frac{1}{8}N_d)$
	Downsampling w/ SN (N_d)	$512 \times 512 \times (3 + \frac{1}{8}N_d)$	$256 \times 256 \times N_d$
	Downsampling w/ SN ($2N_d$)	$256 \times 256 \times N_d$	$128 \times 128 \times 2N_d$
	Downsampling w/ SN ($4N_d$)	$128 \times 128 \times 2N_d$	$64 \times 64 \times 4N_d$
	ROI Align (5)	$64 \times 64 \times 4N_d$	$N_{\text{small}} \times 5 \times 5 \times 4N_d$
	ROI Encoder w/SN (5)	$N_{\text{small}} \times 5 \times 5 \times 4N_d$	$N_{\text{small}} \times 4 \times 4 \times 4N_d$
	Conv 4×4 w/ SN	$N_{\text{small}} \times 4 \times 4 \times 4N_d$	h $(N_{\text{small}} \times 4N_d)$
	Concat	c^{obj} $(N_{\text{small}} \times N_g), e^{\text{g}}$ $(N_{\text{small}} \times N_l)$	$N_{\text{small}} \times (N_g + N_l)$
	FC w/ SN ($4N_d$)	$N_{\text{small}} \times (N_g + N_l)$	c $(N_{\text{small}} \times 4N_d)$
	Fmap Mul - Avg Pool (1)	h, c	hc (N_{small})
	Conv 1×1 w/ SN (unconditional loss)	h	o^{uncond} (N_{small})
	Fmap Sum (conditional loss)	o^{uncond}, hc	o^{cond} (N_{small})
large	Interpolating (2)	M^2 $(256 \times 256 \times N_c)$	$512 \times 512 \times N_c$
	Interpolating (2)	x^2 $(256 \times 256 \times 3)$	$512 \times 512 \times 3$
	Shape Encoder w/ SN ($\frac{1}{8}N_d$)	$512 \times 512 \times N_c$	$512 \times 512 \times \frac{1}{8}N_d$
	Concat	$512 \times 512 \times 3, 512 \times 512 \times \frac{1}{8}N_d$	$512 \times 512 \times (3 + \frac{1}{8}N_d)$
	Downsampling w/ SN (N_d)	$512 \times 512 \times (3 + \frac{1}{8}N_d)$	$256 \times 256 \times N_d$
	Downsampling w/ SN ($2N_d$)	$256 \times 256 \times N_d$	$128 \times 128 \times 2N_d$
	Downsampling w/ SN ($4N_d$)	$128 \times 128 \times 2N_d$	$64 \times 64 \times 4N_d$
	Downsampling w/ SN ($8N_d$)	$64 \times 64 \times 4N_d$	$32 \times 32 \times 8N_d$
	ROI Align (5)	$32 \times 32 \times 8N_d$	$N_{\text{large}} \times 5 \times 5 \times 8N_d$
	ROI Encoder w/ SN (5)	$N_{\text{large}} \times 5 \times 5 \times 8N_d$	$N_{\text{large}} \times 4 \times 4 \times 4N_d$
	Conv 4×4 w/ SN	$N_{\text{large}} \times 4 \times 4 \times 4N_d$	h $(N_{\text{large}} \times 4N_d)$
	Concat	c^{obj} $(N_{\text{large}} \times N_g), e^{\text{g}}$ $(N_{\text{large}} \times N_l)$	$N_{\text{large}} \times (N_g + N_l)$
	FC w/ SN ($4N_d$)	$N_{\text{large}} \times (N_g + N_l)$	c $(N_{\text{large}} \times 4N_d)$
	Fmap Mul - Avg Pool (1)	h, c	hc (N_{large})
	Conv 1×1 w/ SN (unconditional loss)	h	o^{uncond} (N_{large})
	Fmap Sum (conditional loss)	o^{uncond}, hc	o^{cond} (N_{large})

References

[1] S. Hong, D. Yang, J. Choi, and H. Lee. Inferring semantic layout for hierarchical text-to-image synthesis. *arXiv preprint arXiv:1801.05091*, 2018. 9

[2] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018. 12

[3] T. Miyato and M. Koyama. cgans with projection discriminator. *arXiv preprint arXiv:1802.05637*, 2018. 12

[4] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 115(3):211–252, 2015. 7

[5] M. Schuster and K. K. Paliwal. Bidirectional recurrent neural networks. *IEEE Trans. Signal Processing*, 45(11):2673–2681, 1997. 7

[6] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 7

[7] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. *arXiv preprint arXiv:1711.10485*, 2017. 7