PetsGAN: Rethinking Priors for Single Image Generation

Paper ID: 2865

Contents

1 Proof

Symbol description. In this appendix, we follow the notations in the submitted paper to simplify the expression. The bold symbol represents vector or matrix, the roman symbol represents constant and the italic symbol represents variable.

Proposition 1. *Considering image* $\mathbf{I}$ *and its sub-sampling version* $\mathbf{c_I}$. *An image restoration network (IR) can be well designed to satisfy* $\mathbf{I} = IR(\mathbf{c_I})$, *and*

(i) for $v \sim \mathbb{P}^s_{\mathbf{I}}$, *there exists* s^*, $u \sim \mathbb{P}^{s^*}_{\mathbf{c_I}}$, *and a deterministic mapping* r *induced by* IR *from* u *to* v *(in our context it means* $\mathbb{P}^{s^*}_{\mathbf{c_I}}$ *and* IR *determine* $\mathbb{P}^s_{\mathbf{I}}$*).*

(ii) If $\rho(\mathbb{P}^{s^*}_{\mathbf{c}}, \mathbb{P}^{s^*}_{\mathbf{c_I}}) = 0$, *then* $\rho(\mathbb{P}^s_{IR(\boldsymbol{c})}, \mathbb{P}^s_{\mathbf{I}}) = 0$.

Proof. To simplify the expression, without loss of generality we only consider the d-dimensional data, this analysis also holds to matrix cases. $\mathbf{I} = IR(\boldsymbol{c_I})$ can be always satisfied if the network is complex enough [1]. Hence, we look at the simple models and use them to build complex models. Assuming that $\mathbf{c_I} = [x_1, x_2, \ldots, x_n]^T$ is $2\times$ donwsampling of image $\mathbf{I} = [y_1, y_2, \ldots, y_{2n}]^T$, $(d = 2n)$.

Case 1. IR is only composed of a $2\times$ pixelshuffle block ps with kernel size 1, that means

$$\begin{aligned} \mathbf{I} &= [y_1, y_2, \ldots, y_{2n}]^T \\ &= [f(x_1), g(x_1), \ldots, f(x_n), g(x_n)]^T. \end{aligned} \tag{1}$$

For patch with odd size s,

$$\boldsymbol{\omega}_j^{\mathbf{I},s} = [y_{\mathrm{mod}(j-\lfloor s/2 \rfloor, d)}), \ldots, y_{\mathrm{mod}(j+\lfloor s/2 \rfloor, d)}]^T,$$

IR determines two types mapping ps_1 and ps_2 from preimage $\hat{\boldsymbol{\omega}}^{\mathbf{c_I},\lfloor s/2 \rfloor+1}_{\lceil j/2 \rceil}$ to $\boldsymbol{\omega}_j^{\mathbf{I},s}$:

Firstly, if j is even, the patch is

$$\boldsymbol{\omega}_j^{\mathbf{I},s} = [\ldots, g(x_{\lceil j/2 \rceil}), \ldots]^T,$$

and the preimage of $\boldsymbol{\omega}_j^{\mathbf{I},s}$ is

$$\hat{\boldsymbol{\omega}}^{\mathbf{c_I},\lfloor s/2 \rfloor+1}_{\lceil j/2 \rceil} = [x_{\mathrm{mod}(\lceil j/2 \rceil - \lfloor s \lfloor s/2 \rfloor/2 \rfloor, d)}, \ldots, x_{\lceil j/2 \rceil}, \ldots, x_{\mathrm{mod}(\lceil j/2 \rceil + \lceil s \lfloor s/2 \rfloor/2 \rceil, d)}]^T, \tag{2}$$

which meets $ps_1(\hat{\boldsymbol{\omega}}^{\mathbf{c_I},\lfloor s/2 \rfloor+1}_{\lceil j/2 \rceil}) = \boldsymbol{\omega}_j^{\mathbf{I},s}$.

Then, if j is odd, the patch is

$$\boldsymbol{\omega}_j^{\mathbf{I},s} = [\ldots, f(x_{\lceil j/2 \rceil}), \ldots]^T,$$

and the preimage of $\boldsymbol{\omega}_j^{\mathbf{I},s}$ is

$$\hat{\boldsymbol{\omega}}^{\mathbf{c_I},\lfloor s/2 \rfloor+1}_{\lceil j/2 \rceil} = [x_{\mathrm{mod}(\lceil j/2 \rceil - \lceil s \lfloor s/2 \rfloor/2 \rceil, d)}, \ldots, x_{\lceil j/2 \rceil}, \ldots, x_{\mathrm{mod}(\lceil j/2 \rceil + \lfloor s \lfloor s/2 \rfloor/2 \rfloor, d)}]^T. \tag{3}$$

which meets $ps_2(\hat{\boldsymbol{\omega}}^{\mathbf{c_I},\lfloor s/2\rfloor+1}_{\lceil j/2\rceil}) = \boldsymbol{\omega}^{\mathbf{I},s}_j$.

(For example, if $d = 32$, $s = 3$, $j = 10$, $\boldsymbol{\omega}^{\mathbf{I},3}_{10} = [f(x_5), g(x_5), f(x_6)]^T$,thus the preimage is $\hat{\boldsymbol{\omega}}^{\mathbf{c_I},2}_5 = [x_5, x_6]^T$ and $ps_1 : [x_5, x_6]^T \to [f(x_5), g(x_5), f(x_6)]^T$. If $j = 11$, $\boldsymbol{\omega}^{\mathbf{I},3}_{11} = [g(x_5), f(x_6), g(x_6)]^T$,thus the preimage is $\hat{\boldsymbol{\omega}}^{\mathbf{c_I},2}_5 = [x_5, x_6]^T$ and $ps_2 : [x_5, x_6]^T \to [g(x_5), f(x_5), g(x_6)]^T$)).

Therefore, if $v \sim \mathbb{P}^s_{\mathbf{I}}$, $s^* = \lfloor s/2\rfloor + 1$, $u \sim \mathbb{P}^{s^*}_{\mathbf{c}_0}$, $ps_1(u) \sim \mathbb{P}_{ps_1(u)}$ with all samples $\boldsymbol{\omega}_1$, and $ps_2(u) \sim \mathbb{P}_{ps_2(u)}$ with all samples $\boldsymbol{\omega}_2$. It is obvious that $\boldsymbol{\omega}_1$ and $\boldsymbol{\omega}_2$ are of equal probability since the odd numbers and even numbers in $\{1, \ldots, 2n\}$ are of same numbers. Hence, we have

$$\begin{aligned}\int_A d\mathbb{P}_v &= \mathbb{P}^s_{\mathbf{I}}(A)\\ &= \mathbb{P}(A|\boldsymbol{\omega}_1)\mathbb{P}(\boldsymbol{\omega}_1) + \mathbb{P}(A|\boldsymbol{\omega}_2)\mathbb{P}(\boldsymbol{\omega}_2)\\ &= \frac{1}{2}\mathbb{P}(A|\boldsymbol{\omega}_1) + \frac{1}{2}\mathbb{P}(A|\boldsymbol{\omega}_2)\\ &= \frac{1}{2}\int_A d\mathbb{P}_{ps_1(u)} + \frac{1}{2}\int_A d\mathbb{P}_{ps_2(u)}\end{aligned} \tag{4}$$

Thus, $\mathbb{P}^{s^*}_{\mathbf{c_I}}$, ps_1 and ps_2 determine $\mathbb{P}^s_{\mathbf{I}}$. (i) has been proven and the mapping is defined by Eq. (4). Moreover,

$$\rho(\mathbb{P}^{s^*}_{\boldsymbol{c}}, \mathbb{P}^{s^*}_{\mathbf{c_I}}) = 0 \Rightarrow \rho(\mathbb{P}^s_{IR(\boldsymbol{c})}, \mathbb{P}^s_{IR(\mathbf{c_I})}) = 0 \Rightarrow \rho(\mathbb{P}^s_{IR(\boldsymbol{c})}, \mathbb{P}^s_{\mathbf{I}}) = 0. \tag{5}$$

Hence (ii) has been proven.

Case 2. Note that we do not require the linearity of g and f, hence same conclusion can be obtained in the case of IR containing activation and convolutional blocks by using the same analytical method in Case 1. Since $n\times$ upsampling can be decomposed into multi-$2\times$ upsampling, the conclusion also holds in general cases.

Thus, we can build a complex non-linear mapping to satisfy the proposition. □

Remark 1. *$\boldsymbol{a} = (a_1, \ldots, a_n)^T, a_i \geq 0$ and $\|\boldsymbol{a}\|_1 = 1$. Then $\|\boldsymbol{a}\|_2 \leq= 1$ and the equal sign holds when and only when $\|\boldsymbol{a}\|_\infty = 1$.*

2 Regularized Latent Variable Model

General SIG models can be divided into two categories, the first one is the SinGAN-based pyramid model and the other one is the inversion-based model. Both of them can be expressed as regularized latent variable models.

Pyramid model. Denote

$$M_n(G_0, \ldots, G_n) = \sum_{i=0}^{n} \rho_\tau(\delta_{\mathbf{c}_i}, \mathbb{P}_{G_i|G_{j<i}}) + \varphi_n(G_0, \ldots, G_{n-1}),$$

where $\mathbf{c}_i$, $i = 0, \ldots, n$ are the downsampled versions of image $\mathbf{I}$, the resolution of $\mathbf{c}_i$ increases with i and $\mathbf{c}_n = \mathbf{I}$. Pyramid SIG model learns

$$\begin{aligned} &\min_{G_0,\ldots,G_n} M_n(G_0, \ldots, G_n) \\ &s.t.\ \varphi_n(G_0, \ldots, G_{n-1}) = \begin{cases} 0 & \text{if } G_0, \ldots, G_{n-1} = \underset{G_0,\ldots,G_{n-1}}{\arg\min}\ M_{n-1}(G_0, \ldots, G_{n-1}) \\ \infty & \text{otherwise} \end{cases}. \end{aligned} \tag{6}$$

Inversion model. The inversion model essentially looks for the optimal latent code $\mathbf{c_I}$ of image $\mathbf{I}$ in the latent space of pre-trained generator G_{pre}, then disturbs $\mathbf{c_I}$ to generate new images.

$$\min_G \rho(\delta_{\mathbf{c_I},\mathbf{I},}, \mathbb{P}_{\mathbf{c},G}) + \varphi_1(\mathbb{P}_{G|\mathbf{c}}, \mathbb{P}_{G_{pre}|\mathbf{c}}) + \varphi_2(\mathbb{P}_{\mathbf{c}}, \mathbb{P}_{\mathbf{c_I}}). \tag{7}$$

The first term makes the generator learn the internal information, the second term makes the generator similar or same with the pre-trained generator, the third term constrains the disturb to some meaningful regions or semantic directions.

3 Architecture

Fig. 1 shows the details of PetsGAN architecture and parameter settings. The `Conv` and the numbers represents ConvBlock composed of `Conv(in_channels, out_channels, kernel_size, stride, padding, dilated)`, `InstanceNorm` and `LeakyReLU(0.2)`.

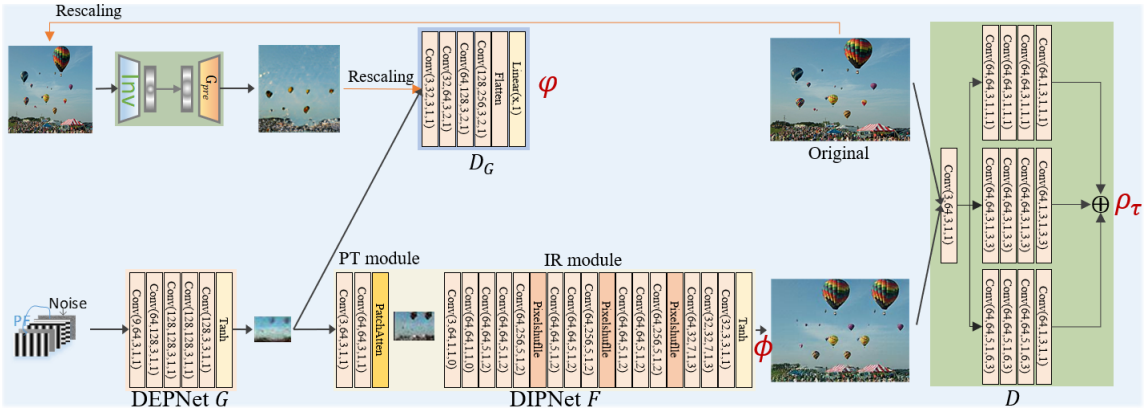


Figure 1: Architecture of PetsGAN. The red symbols represent the loss functions in paper.

4 Training Details

Algorithm 1: Training PetsGAN

Input: Image **I** longer side bound 256
Output: G, F, D_G, D
Initialization: Set $Warmup_G = 2000$, $Warmup_F = 5000$, $Epoch = 5000$, learning rate 1e-4 for both G and D_G, and 5e-4 for F. Adopt Adam optimizer. Construct Datasets $\mathcal{D}$ including 500 samples from DGP and 200 augmentation samples of **I** by thin plate spline interpolation and random horizontal flip.
begin Training
  for *i=1:*$Warmup_G$ **do**
    Sample a minibatch samples from Dataset $\mathcal{D}$ with 8× downsampling;
    Updating G and D by optimizing φ (Eq. (6) in paper);
  end
  for *i=1:*$Warmup_F$ **do**
    Sample Δc from $\mathcal{N}(0, 0.01)$;
    Updating F by optimizing ϕ (Eq. (8) in paper);
  end
  Reset Adam optimizer, learning rate 1e-6 for G, 1e-4 for D_G, IR module and freeze PT module.
  for *i=1:Epoch* **do**
    Updating G, D_G, F, D by optimizing weighted form of Eq. (5) in paper;
  end
end

5 User Studies

User studies are conducted to evaluate the realism and the diversity of generated results from SinGAN and PetsGAN. We recruit 50 Turkers for each triplet (reference, A, B), where A represents the synthesis of PetsGAN and B represents the synthesis of SinGAN. Turkers select the best preference from A and B in terms of realism and diversity respectively without time limitation. As Table 1 shows, In terms of generative realism, PetsGAN has obvious advantages over SinGAN. For diversity, as the difficulty of the dataset increases, so does the advantage of PetsGAN. The reason can be concluded as: the syntheses of SinGAN are diverse in terms of LPIPS, much of this diversity is due to the artifacts in the syntheses. Human judgment criteria will identify the diversity that artifacts bring.

	Places50	LSUN50	ImageNet50
	SinGAN/PetsGAN	SinGAN/PetsGAN	SinGAN/PetsGAN
Realism	$0.18/\mathbf{0.82}^{0.06}_{0.22}$	$0.26/\mathbf{0.74}^{0.04}_{0.15}$	$0.22/\mathbf{0.78}^{0.05}_{0.18}$
Diversity	$0.48/\mathbf{0.52}^{0.02}_{0.08}$	$0.33/\mathbf{0.67}^{0.03}_{0.12}$	$0.30/\mathbf{0.70}^{0.05}_{0.16}$

Table 1: Comparison of user preference, the left number is the percentage of preference for SinGAN, and the right number is the percentage of preference for PetsGAN. $mean^{ci}_{std}$(ci is the confidence interval at 95% confidence level).

6 Generation Results

We provide more generation results. Fig. 2∼Fig. 5 show some generation reults of the proposed PetsGAN and SinGAN on Places50, LSUN50 and Image50 datasets. Fig. 6∼Fig. 8 show some high resolution generation results.

References

[1] Nicolas Le Roux and Yoshua Bengio. Deep belief networks are compact universal approximators. *Neural Computation*, 22:2192–2207, 2010.

Figure 2: Syntheses on Places50. The first column is the original image. For each original image, the first row is generated by PetsGAN and the second row is generated by SinGAN.

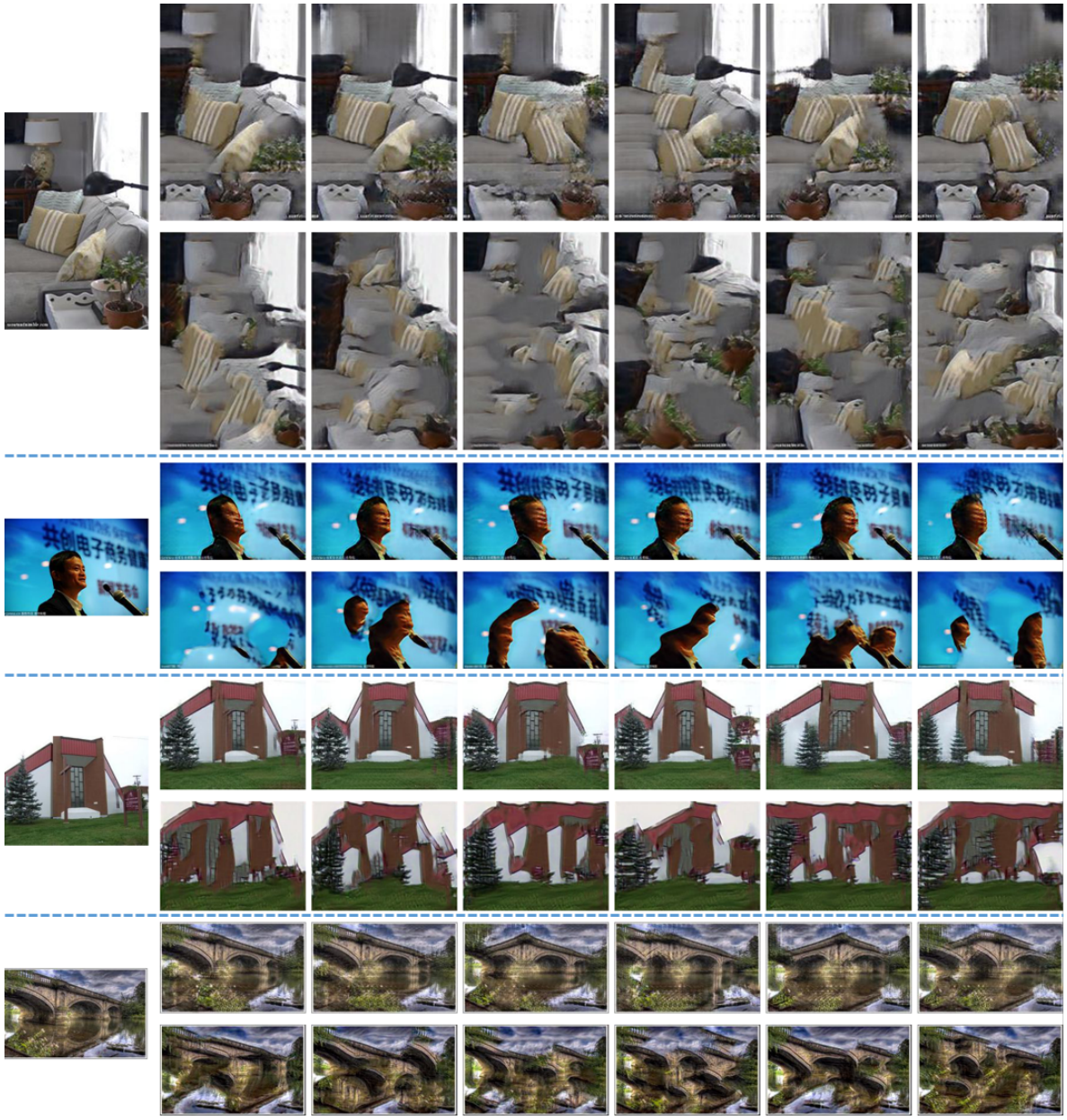


Figure 3: Syntheses on LSUN50. The first column is the original image. For each original image, the first row is generated by PetsGAN and the second row is generated by SinGAN.

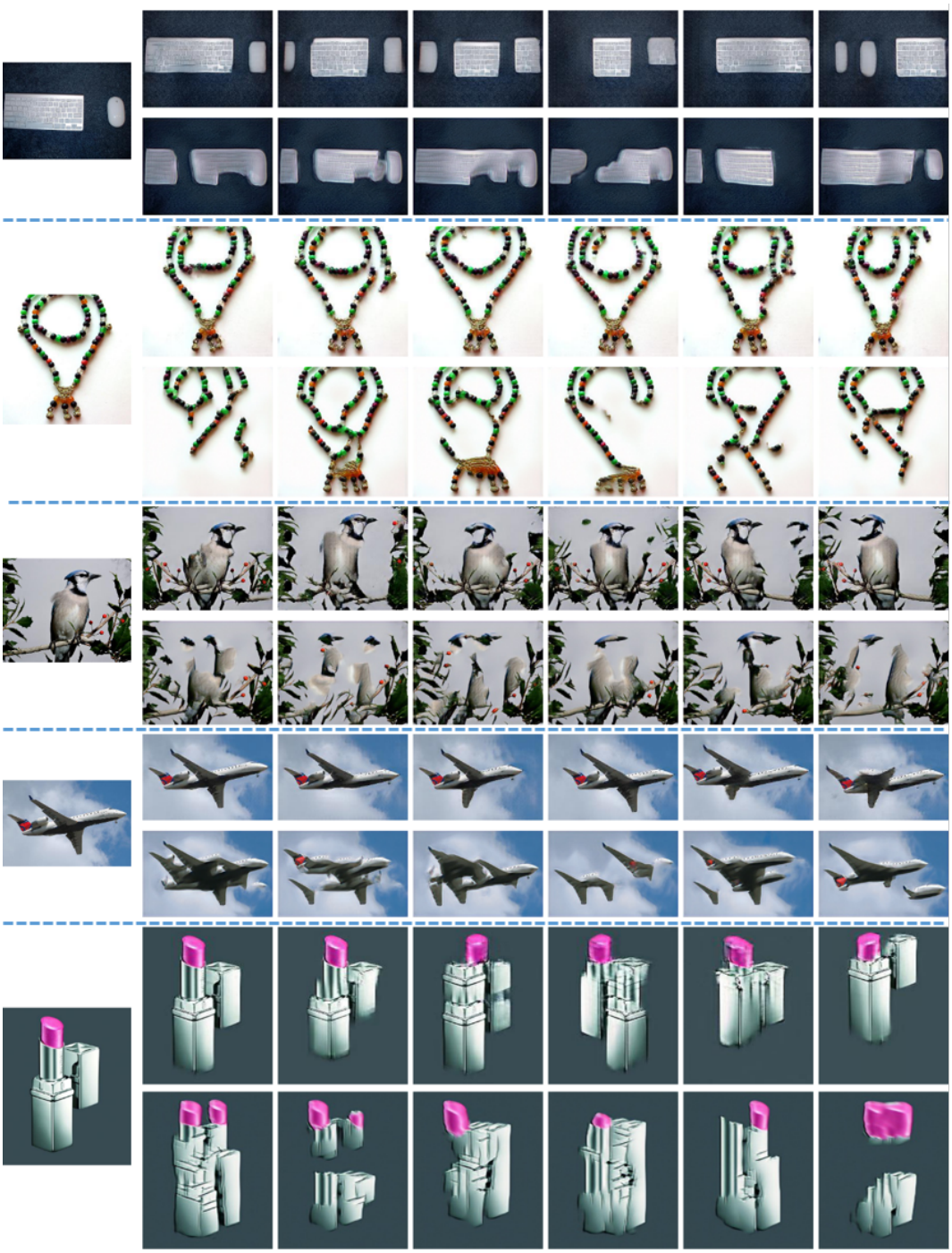


Figure 4: Syntheses on ImageNet50. The first column is the original image. For each original image, the first row is generated by PetsGAN and the second row is generated by SinGAN.

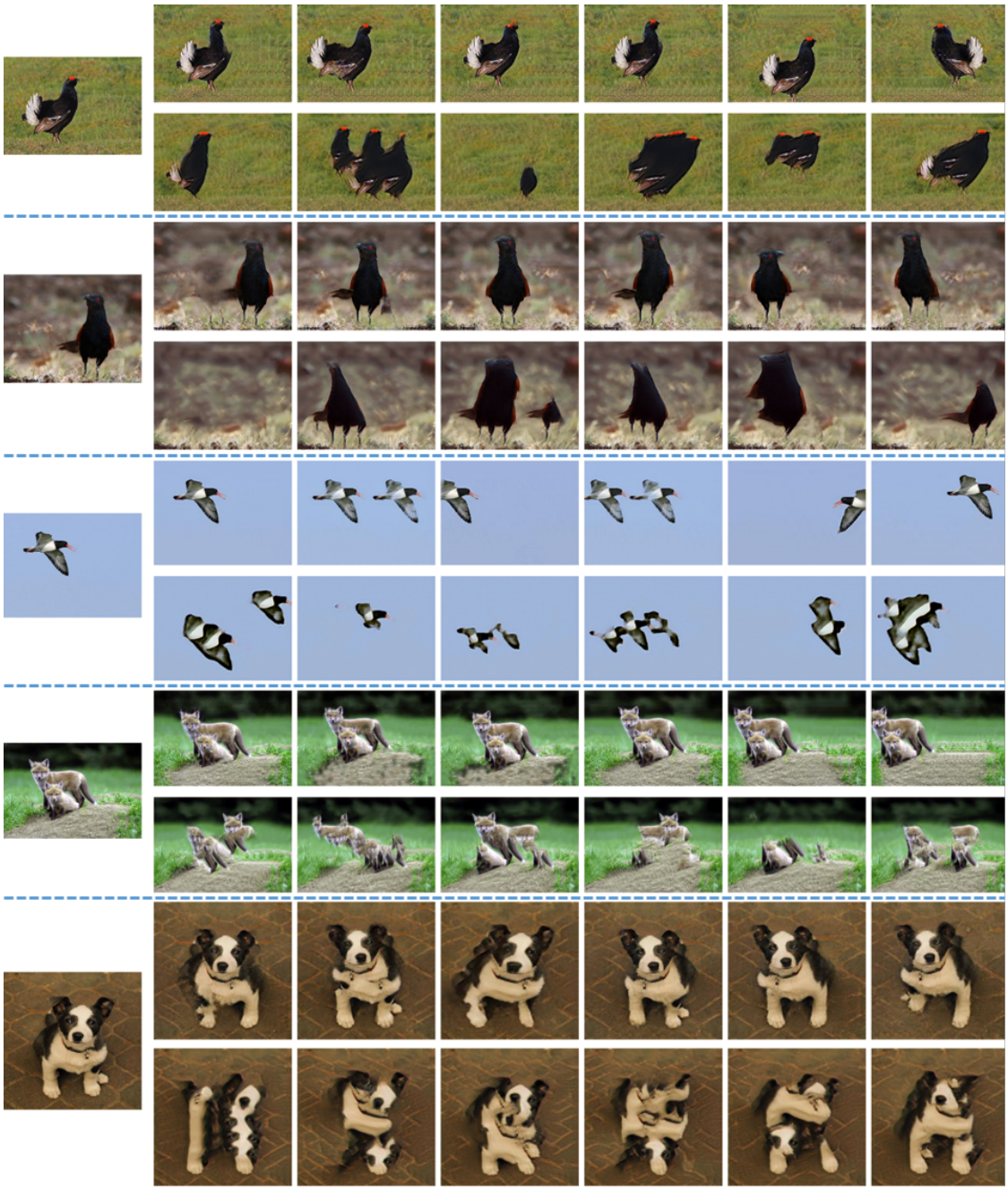
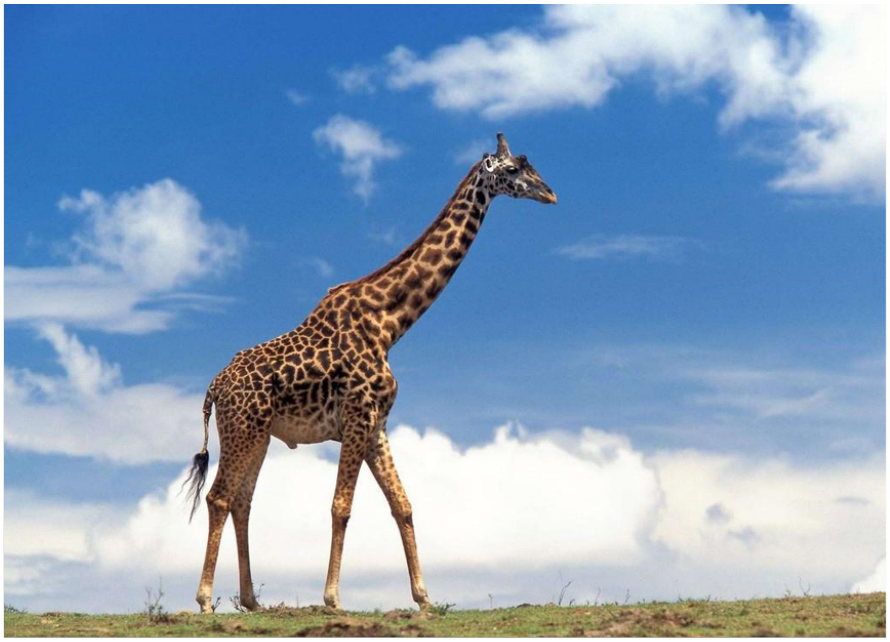


Figure 5: Syntheses on ImageNet50. The first column is the original image. For each original image, the first row is generated by PetsGAN and the second row is generated by SinGAN.

Figure 6: High resolution generation.

Original



PetsGAN

Figure 7: High resolution generation.

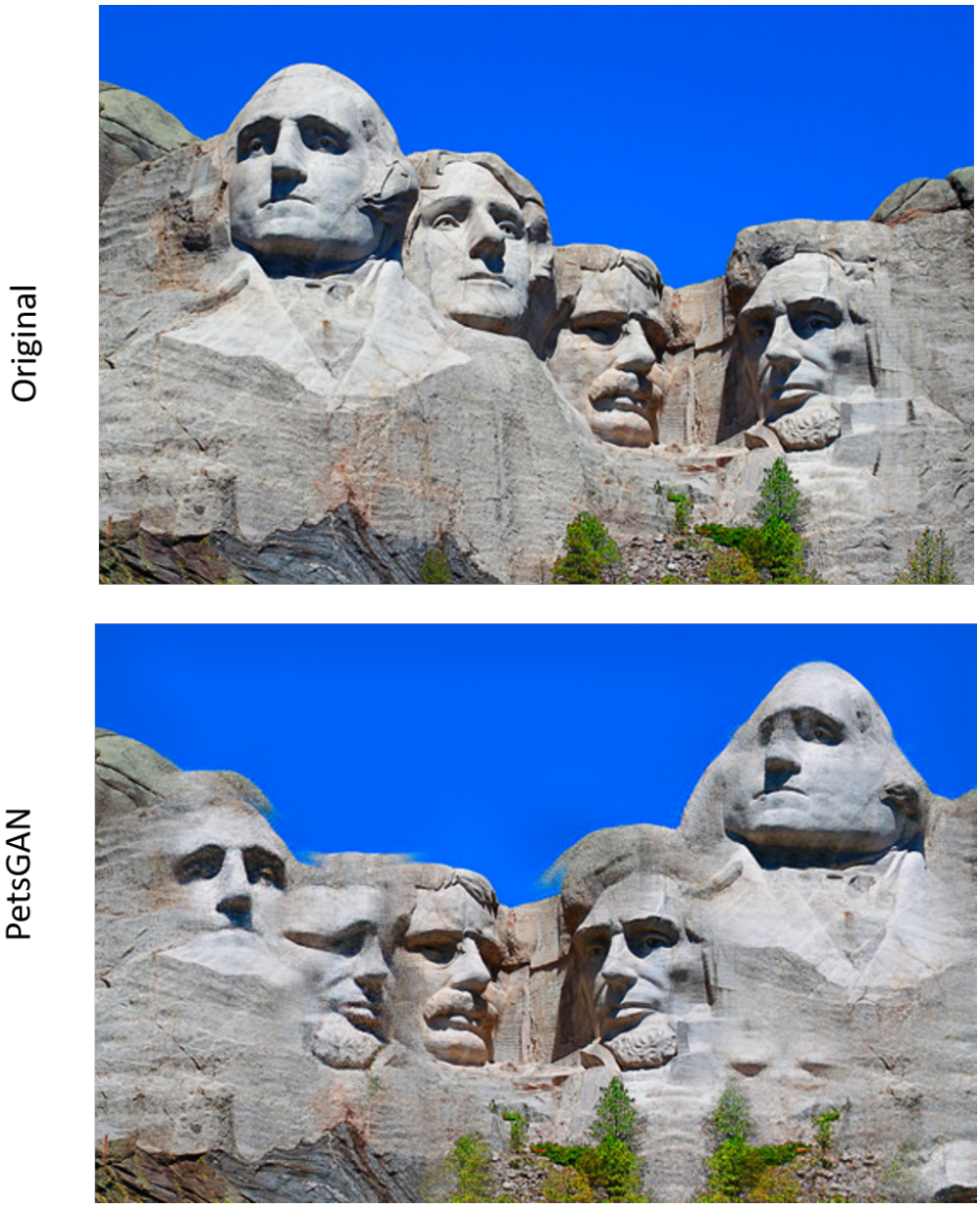


Figure 8: High resolution generation.