Supplementary Material for Patchwise Generative ConvNet: Training Energy-Based Models from a Single Natural Image for Internal Learning

Zilong Zheng [1], Jianwen Xie [2], Ping Li [2]
[1] University of California, Los Angeles, CA
[2] Cognitive Computing Lab, Baidu Research, Bellevue, WA
z.zheng@ucla.edu, {jianwenxie,liping11}@baidu.com

In this supplementary material, we will provide full descriptions of the training and sampling algorithms and details about architecture design of the energy function to support the paper.

A. Multi-Scale Training and Sampling

A.1. Algorithm Description

We provide the descriptions of the proposed learning and sampling algorithms in Algorithm 1 (illustrated by Figure 1) and Algorithm 2 (illustrated by Figure 2), respectively. Algorithm 1 presents the multi-scale sequential training of the pyramid of energy-based models, where the multi-scale sequential sampling presented in Algorithm 2 is used for efficient MCMC generation to compute the update gradients.

Algorithm 1 Multi-scale sequential training

Input:
(1) A single training image $\mathbf{I}$
(2) Numbers of Langevin steps at different scales $\{K^{(s)}, s = 0, ..., S\}$
Output:
(1) Model parameters $\{\theta^{(s)}, s = 0, ..., S\}$
(2) Different scales of synthesized images $\{\tilde{\mathbf{I}}^{(s)}, s = 0, ..., S\}$

Create multi-scale versions of the training image $\{\mathbf{I}^{(s)}, s = 0, ..., S\}$ by downsampling operation.
for $s = 0$ to S **do**
  repeat
    Sample $\{\tilde{\mathbf{I}}_i^{(s)}, i = 1, ..., n\}$ from the model at scale s by Algorithm 2
    Update θ_s according to Eq.(5) using Adam optimizer.
  until converged.
end for

A.2. Model Architecture

Table 1 shows the network structures of EBMs at different scales. Each model consists of five Conv2D layers with 3×3 kernel size. We add spatial zero paddings to the input and use padding size 0 for all convolutional layers. We use the Spectral Normalization to regularize the Conv2D parameters and ELU as the activation function. Parameters are initialized from a Gaussian distribution $\mathcal{N}(0, 0.005)$.

Algorithm 2 Multi-scale sequential sampling

Input:
(1) The scale s' of the model that need to be sampled
(2) Numbers of Langvein steps $\{K^{(s)}, s = 0, ..., s'\}$
(3) Learned model parameters $\{\theta^{(s)}, s = 0, ..., s'\}$
Output:
(1) Synthesized image $\tilde{\mathbf{I}}^{(s')}$ at scale s'

for $s = 0$ to s' **do**
  if $s = 0$ **then**
    Initialize $\tilde{\mathbf{I}}_0^{(s)}$ with $\mathcal{U}_d((-1, 1)^d)$
  else
    Initialize $\tilde{\mathbf{I}}_0^{(s)}$ with Upsample$(\tilde{\mathbf{I}}_{K^{(s-1)}}^{(s-1)})$
  end if
  for $t = 0$ to $K^{(s)} - 1$ **do**
    Update $\tilde{\mathbf{I}}_{t+1}^{(s)}$ according to Eq.(7).
  end for
end for

Table 1: Model architectures of various image scales. w and h correspond to the width and the height of the scaled training image, respectively.

(a) $\max(w, h) < 64$.	(b) $\max(w, h) \geq 64$.
ZeroPadding2D((5, 5))	ZeroPadding2D((5, 5))
3×3 Conv2D, 64, ELU	3×3 Conv2D, 128, ELU
3×3 Conv2D, 32, ELU	3×3 Conv2D, 64, ELU
3×3 Conv2D, 32, ELU	3×3 Conv2D, 64, ELU
3×3 Conv2D, 32, ELU	3×3 Conv2D, 64, ELU
3×3 Conv2D, 1	3×3 Conv2D, 1

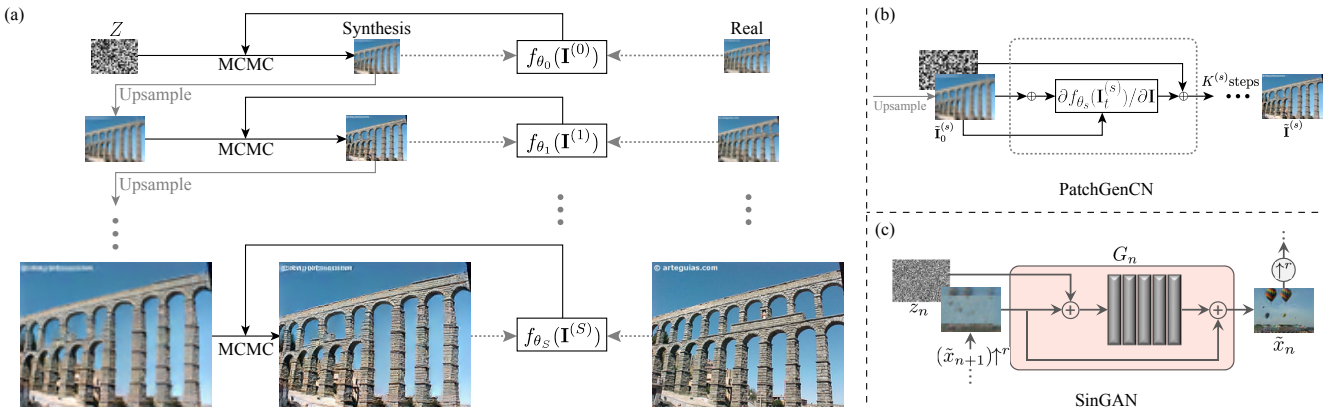

Figure 1: Learning framework of the multi-scale Patchwise Generative ConvNet (PatchGenCN). (a) Illustration of coarse-to-fine multi-scale learning and sampling procedure. Our model parameterizes the energy function by a convolutional network f_{θ_s} at each scale s. Z indicates an image initialized from the uniform white noise. The solid arrows in black indicate the multi-scale MCMC sampling paradigm; the dashed arrows in grey indicate the parameter updates; and the solid arrows in grey indicate the image upsampling operations. (b) Illustration of $K^{(s)}$-step Langevin sampling at scale s. $\oplus$ indicates the elementwise addition operation. (c) Illustration of single-scale generation of SinGAN, where the image synthesis is performed by the top-down generator G. Compared with (c), the sampling process in (b) is derived from the bottom-up energy function f_{θ_s}, and performed in an iterative way. Such a sampling process can be interpreted as a noise-injected $K^{(s)}$-layer residual generator network.

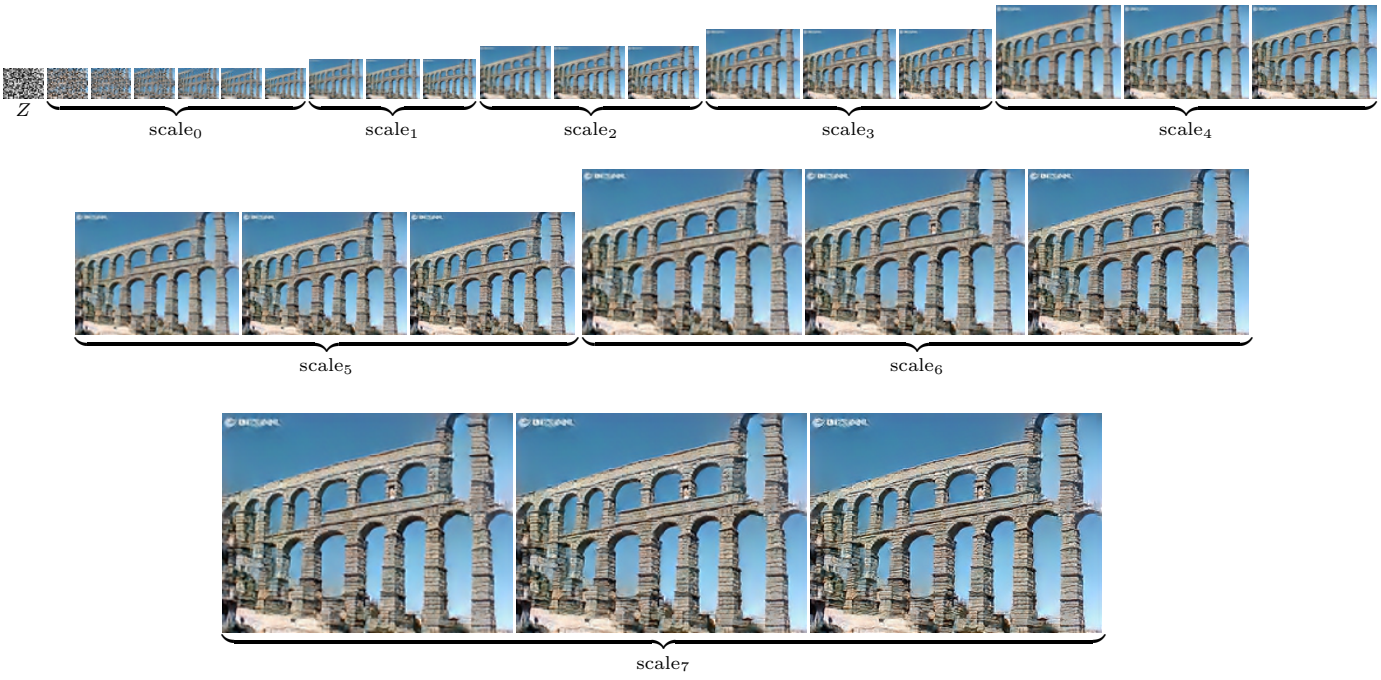

Figure 2: Multi-scale sequential sampling process starting from a randomly initialized noise image Z with the minimum scale. For each scale s, the initial synthesis is updated by $K^{(s)}$ steps of Langevin revision. We visualize a sampled image every 10 Langevin steps for each scale. Except that the initial synthesis at scale 0 is from uniform distribution, the Langevin dynamics of any other scale is initialized from the upsampled version of the Langevin output at its previous scale.