Shuffling-SinGAN: Improvement on Generative Model from a Single Image

Ming Zheng¹ ²*, Pengyuan Zhang¹ ², Yining Gao¹ ², Hang Zou¹ ²
¹Wuhan Research Institute of Posts and Telecommunications, Wuhan, Hubei, China; ²Nanjing Fenghuo Tiandi Communication Technology Co., Ltd; *Corresponding author’s e-mail: 724849494@qq.com

Abstract. Recently, SinGAN takes the use of GANs into a new realm – unconditional generation learned from a single natural image. Following the SinGAN architecture, we propose Shuffling-SinGAN, an efficient unconditional generative model that trained on a single natural image for general image manipulation. Our new network includes a pyramid of fully convolutional GANs, in which each layer is responsible for learning the patch distribution at a different scale of the image. We can generate new samples with variability through our network, which have the function of maintaining the texture and global structure of the original image. New random image generated by the model after multiple training is different from the original image in detail. Inspired by sinIR, we decided to add random pixel shuffling to the network. After experimentation, we found that the changed model generated more random new samples. Shuffling-SinGAN allows generating new samples of arbitrary size and aspect ratio, that have significant variability, yet maintain both the global structure and the fine textures of the training image. User tests confirm that the generated samples are commonly confused to be real images. With quantitative evaluation, we show that Shuffling-SinGAN has competitive performance on random image generation.

1. Introduction
Nowadays, Generative Adversarial Networks (GANs) [1] have achieved impressive results in many visual processing tasks, such as image super-resolution [2], inpainting [3], image editing [4, 5, 6, 7, 8] and image style Migration etc. From the actual results, GAN seems to produce better generated samples. Now researchers are interested in using a single image for training, and then using the training model for image processing operations. The strict requirements for the GPU memory size are not need and the code that need to be preprocessed for large-scale datasets has been reduced. Training on a single image is reasonable because it has abundant information that can be used as a powerful prior for solving various problems. Several internal deep learning methods have been proposed and achieved excellent performance comparable to external training methods on large-scale datasets, such as super resolution, restoration [9], reflection removal [10], deblur, segmentation and dehazing, denoising and inpainting tasks.

However, these methods have serious problems and do not apply deep internal learning to practice. First, with the exception of MGANs [11], most of these methods are image-specific in terms of operation. This means that the trained model can only operate on the original training image, and for other images, a separate model must be trained. Second, most of these methods are task-specific, which means that the trained model can only perform a specific image manipulation. SinGAN [12] takes the use of GANs into a new realm – unconditional generation learned from a single natural image. The internal statistics
of patches within a single natural image typically carry enough information for learning a powerful generative model. It does not need to rely on same type of images, and it allows us to process general natural images containing complex structures and textures. Because unconditional image generation is a relatively difficult problem, a complex loss function [13] is used in the original SinGAN to better converge and train multiple for a large number of iterations GAN.

The improvement in this article is to add an attention mechanism to each layer. Because there is only random noise input at the bottom layer, the pictures generated at the bottom layer are often weird, and there are multiple items mixed together. Therefore, we thought of the attention mechanism, which really ensures that there are not many kinds of new pictures generated mixed objects appear. We also found a problem at the same time. At high scales of sinGAN, the new generate image is the same as the training image. This violates the idea of sinGAN to generate a new random image from an original image, so we added the random pixels shuffling from sinIR [14] which does not significantly increase the computational cost, and random pixel shuffling provides additional control over the operation.

2. Materials and Methods

2.1 Model

The structure of resnet [15] can speed up the training of neural network, and improve the accuracy of the model. At the same time, resnet can be directly used in the Inception network [16]. The main idea of resnet is to add a direct connection channel to the network which is Highway Network. The generators structure in Our Shuffling-SinGAN Network (SSGAN) is resnet. It is composed of two main modules, generator \( \{G_0, \ldots, G_n\} \) and discriminator \( \{D_0, \ldots, D_n\} \). This model is similar to the traditional GAN, except that the training samples here are patches of a single image instead of the whole image samples in the database. We process common natural images.

Our model trained in the order of each scale, from the coarsest scale to the finest scale. The parameters of each scale are independent and will remain fixed after being trained. The nth GAN training loss is comprised of an adversarial term and a reconstruction term.

\[
\text{Loss} = \max_{D_n} \min_{G_n} \mathcal{L}_{adv}(G_n, D_n) + \alpha \mathcal{L}_{rec}(G_n). \tag{1}
\]

**Figure 1.** Shuffling-SinGAN’s multi-scale struct. Our model training and inference are done in a coarse-to-fine fashion using a pyramid of GANs. At each scale, the input to \( G_n \) is a random noise image \( z_N \), and image \( x_N \) generated from previous scale after random pixel shuffling and upsampled to the current resolution. \( G_n \) learn these images to generate image samples. \( G_n \) learns to fool an associated discriminator \( D_n \). \( D_n \) try to distinguish patches in the generated samples \( x_n \) and the real image \( r_n \).
2.1.1 CBAM
CBAM [17] includes two independent sub-modules, the Channel Attention Module (CAM) and the Spatial Attention Module (SAM), which perform channel and spatial attention respectively. This not only saves parameters and computing power, but also ensures that it can be integrated into the existing network architecture as a plug-and-play module. After the introduction of CBAM, the features cover more parts of the object to be recognized, and the probability of finally discriminating the object is also higher. CBAM allows the network to learn to focus on key information. CAM: Use the input feature map $F$ ($H \times W \times C$) through global max pooling and global average pooling based on width and height respectively, obtain two $1 \times 1 \times C$ features Figure. Then send them to a two-layer neural network (MLP), the number of neurons in the first layer is $C/l$ ($l$ is the reduction rate), the activation function is Relu. The number of neurons in the second layer is $C$, this two-layer neural network is shared. The MLP output features are subjected to an element-wise addition operation, and through the sigmoid activation operation is performed to generate the final channel attention feature, namely $M_c$. Finally, $M_c$ and the input feature map $F$ are subjected to an element-wise multiplication operation to generate the input features required by the Spatial attention module. SAM: The feature map output by the Channel attention module is used as the input feature map of this module. First do a global max pooling base on channel and global average pooling to get two $H \times W \times 1$ feature maps, then make channel splicing on these two feature maps based on the channel. After a 7×7 convolution (7×7 is better than 3×3) operation, the dimensionality is reduced to 1 channel ($H \times W \times 1$). Through sigmoid, the spatial attention feature named $M_s$ generated. Finally, the feature and the input feature of the module are multiplied to get the final generated feature. The struct of CBAM can be seen in Figure 2.

![Convolutional Block Attention Module](image)

**Figure 2.** The overview of CBAM. This module has two sequential sub-modules: channel and spatial. Input feature map will be refined through this module. This module can be used in every convolutional block of deep networks.

2.1.2 Random pixel shuffling
We added random pixel shuffling [18] module before upsampling. We did not randomly set some pixels to be black like the denoising autoencoder, because we used more complex natural images. In addition, we randomly shuffled some pixels in a given single image so that the network can learn a more robust relationship between adjacent pixels. We used random pixel shuffling to constrain the images generated at the previous scale, then as the input of the next layer. During the training process, by adjusting the random pixel shuffling wiped parameter $p$, the generated images can be adjusted. In the training process, we refered to sinIR to set the random pixel shuffling wiped parameter $p$ to 0.005. At the same time, in order to confirm the function of the random pixel shuffling module, we also made a training that $p$ is 0.3 as a reference case. When the $p$ value is too high, there will be distortion of the generated picture. Figure 3 shows the influence of parameter $p$ on the network. Figure 4 shows the generate network after we improved.
Figure 3. This figure shows the effect of different p on the generated image. The generate images (a) are which p is 0.3. Color distortion is obvious. The p in generate images (b) is 0.05. These images are similar to the training image.

Figure 4. Generation network in each scale. The image from the previous scale, $x_{n+1}$ and the input noise map at scale n ($z_n$). After $x_{n+1}$ is upsampled, it added to $z_n$. The result is sent into network. After calculation through 5 conv layers and CBAM, whose output is a residual image. Then the output added back to $(x_{n+1})^\uparrow$. We get the nth scale result, $x_n$. Finally send the result after random pixel shuffling to next scale.

2.1.3 Adversarial loss
Each of the generators $G_n$ is coupled with a discriminator $D_n$. The role of the discriminator is to classify each overlapping block of its input as true or false. In the training process, we use the same WGAN-GP loss as sinGAN, which can improve training stability. In the WGAN-GP where the final discriminant score is the average over the patch discrimination map. We define the loss over the whole image. This way allows the net to learn boundary conditions. The patch size in $D_n$ is the same with $G_n$ because the architecture of $D_n$ is the same as the net $G_n$. The net’s receptive field (patch size) is $11 \times 11$.

2.1.4 Reconstruction loss
The purpose of reconstruction loss is to hope that there will be a set of random noise input, and the final output image is the original image. At the beggning, we follow sinGAN use MSE (mean squared error)
loss. But then we realized that MSE loss will produce blurred images [19], and considering that the meaning of the model is to generate new images on the basis of maintaining the integrity of the original image information, we modified the original loss, SSIM (structural similarity) loss is added to the MSE loss:

\[
L_{\text{rec}}(A, B) = \text{MSE}(A, B) + (1 - \text{SSIM}(A, B))
\]

3. Results & Discussion

3.1 Datasets
We have tested our method qualitatively and quantitatively on a wide range of images covering a large number of scenes including urban and natural scenery as well as artistic and texture images. The images we used are taken from Berkeley Segmentation Database (BSD), Places and the Web.

3.2 Implementation Details

3.2.1 Architecture
We used SinGAN as baseline. We added two new modules as the generator in network. They are random pixel shuffling and Convolutional Block Attention Module (CBAM). The size of the training picture is adjusted to 250px.

3.2.2 Training.
To make a fair comparison, the parameters are same with sinGAN. We set scale factor \( r \) to 4/3, the minimum and maximum dimension to 25px, 250px, and the number of scale \( N \) is calculated by these parameters. Shuffling-SinGAN is trained for 2000 iterations at every scale, The optimizer uses Adam, set the momentum parameter beta1=0.5, beta2=0.999, the learning rate of the generator and the discriminator is 0.0005 (decrease after 1600 iterations 0.1 times), reconstruction loss weight is 10, gradient penalty weight of WGAN-GP loss is 0.1. Our codes use the PyTorch environment, and the training process used a single NVIDIA RTX 3070 GPU.

3.2.3 Testing
Our testing process is similar to training process, putting the input image into the trained network to get the result. The advantage of our model is that we can set the different scales to control the generated pictures. The results have been evaluated qualitatively and quantitatively. The following section shows results.

The purpose of our improvement to sinGAN is to optimize it, so our goal is the same as it. AMT perceptual study indicator is used in sinGAN, and we use a similar method. We followed the protocol of [20,21] and set experiments: real vs fake. We conducted user research using randomly sampled images from a dedicated dataset provided by (Luan et al., 2017). 20 subjects with computer vision experience to help us complete the experiment. Workers were asked to select fake images. (1) Unpaired (either real or fake): Workers were presented with a single image for 1 second, and were asked if it was fake. In total, 50 real images, other 50 fake images generated by sinGAN were presented in random order to each worker. (2) Paired: Workers were presented with a sequence of 50 trials, in each of fake images which generated by SinGAN was presented against real training image for 1 second. Workers were asked to pick the fake image. We repeated these two protocols for two types of generation processes: Starting the generation from the \( N \) scale, and from scale \( N - 1 \). In this way, we evaluate the authenticity of the results on two different scales. We have done the same test using the fake images generate by SSinGAN. In figure5, (a), (b) and (c) are the random samples generated by SinGAN and SSinGAN from the scale of \( n=N \), \( n=N-1 \) and \( n=N-2 \) respectively.
In order to ensure that our model can guarantee the generation of diverse images, we have selected images different from each other in the database, such as Mountains, Hills, Desert, and Sky. We performed a calculation with a parameter called diversity of the generated images: for each training example we calculated the standard deviation (std) of the intensity values of each pixel over 100 generated images, averaged it over all pixels, and normalized by the std of the intensity values of the training image. For the N-1th scale picture, we have achieved better results, 50% of workers think the pictures we generate are real. Table 1 shows the result.

Table 1. “Real/Fake” test. This is the result of confusion rates for two generation processes: Starting from the coarsest scale, and starting from the second coarsest scale N−1. In each case, we conducted a paired study (showing a pair of real and fake images) and an unpaired study (showing either fake or real image). SinGAN and SSinGAN also shown as a comparison.

| Model     | Scale | Diversity | Survey | Confusion |
|-----------|-------|-----------|--------|-----------|
| SinGAN    | N     | 0.5       | paired | 22%       |
|           | N-1   | 0.38      | unpaired | 43%       |
|           |       |           | paired | 32%       |
|           | N     | 0.5       | unpaired | 47%       |
|           |       |           | paired | 25%       |
| SSinGAN   | N     | 0.5       | unpaired | 45%       |
|           | N-1   | 0.38      | paired | 35%       |
|           |       |           | unpaired | 50%       |

A common metric for GAN evaluation is the Frechet Inception Distance (FID) [22], which measures the deviation between the distribution of deep features of generated images and that of real images. In our Shuffling-SinGAN, there is only one single real image, so we used SIFID be proposed in SinGAN, it use the internal distribution of deep features at the output of the convolutional layer before the second pooling layer. SIFID is the FID between the statistics of those features in the real image and in the generated sample. We repeated these two protocols for two types of generation processes: starting with the Nth(coarsest) scale, and starting with the N-1 scale. In this way, we assess the realism of the results in two different levels. The SIFID value is smaller, the generated image is closer to the real image. Table 2 is the result of comparing two models with two different sizes. The result shows that the image samples generated by the method in this paper is closer to that of the real image samples, and can effectively capture the detailed information in the image and the dependence between each feature channel, and generate high-quality images.
Table 2: Single Image FID (SIFID). This is the result of the average score for 50 images. It consists of two scales, the coarsest scale (Nth) and the second coarsest scale (N-1th). We also show the differences between SinGAN and SSinGAN.

| Model  | Scale | SIFID |
|--------|-------|-------|
| SinGAN | N     | 0.09  |
|        | N-1   | 0.05  |
| SSinGAN| N     | 0.08  |
|        | N-1   | 0.04  |

We show different samples generated by different kinds of images with big gaps in Nth. Figure 6 shows the result.

Figure 6: Random image samples. After training, our model can generate realistic random images that include new structures and object configurations.

4. Conclusion
We introduced Shuffling-SinGAN, an unconditional generative model that is learned from a single natural image based on SinGAN. Our training objective is the same as SinGAN and the result proved our model is better than SinGAN in the coarsest scale. Although compared with the generation method of external training, internal learning is inherently limited in terms of semantic diversity. Our model does not surpass SinGAN in this respect, that cannot generate images that contain new information that does not exist in the training image. For example, the training image contains a zebra, our model cannot generate a cat. But we do surpass SinGAN while performing the same task. For future work, although we only used one training image in this work, we can further explore whether it is feasible to use multiple images for training. This way possible explores effective methods for dealing with extreme situations with fewer internal references in training images. We hope our work can bring valuable contribution and inspiration to the future research.

REFERENCES
[1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, 27.
[2] Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., ... & Shi, W. (2017). Photo-realistic single image super-resolution using a generative adversarial network. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4681-4690).
[3] Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., & Efros, A. A. (2016). Context encoders: Feature learning by inpainting. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2536-2544).

[4] Mechrez, R., Shechtman, E., & Zelnik-Manor, L. (2019). Saliency driven image manipulation. Machine Vision and Applications, 30(2), 189-202.

[5] He, K., & Sun, J. (2012, October). Statistics of patch offsets for image completion. In European conference on computer vision (pp. 16-29). Springer, Berlin, Heidelberg

[6] Cho, T. S., Butman, M., Avidan, S., & Freeman, W. T. (2008, June). The patch transform and its applications to image editing. In 2008 IEEE Conference on Computer Vision and Pattern Recognition (pp. 1-8). IEEE.

[7] Dekel, T., Michaeli, T., Irani, M., & Freeman, W. T. (2015). Revealing and modifying non-local variations in a single image. ACM Transactions on Graphics (TOG), 34(6), 1-11.

[8] Tlusty, T., Michaeli, T., Dekel, T., & Zelnik-Manor, L. (2018). Modifying non-local variations across multiple views. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 6276-6285).

[9] Zhang, L., Zhang, L., Liu, X., Shen, Y., Zhang, S., & Zhao, S. (2019, October). Zero-shot restoration of back-lit images using deep internal learning. In Proceedings of the 27th ACM International Conference on Multimedia (pp. 1623-1631).

[10] Yin, Y., Fan, Q., Chen, D., Wang, Y., Aviles-Rivero, A., Li, R., ... & Chen, B. (2019). Deep reflection prior. arXiv preprint arXiv:1912.03623.

[11] Li, C., & Wand, M. (2016, October). Precomputed real-time texture synthesis with markovian generative adversarial networks. In European conference on computer vision (pp. 702-716). Springer, Cham.

[12] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. (2017). Improved training of wasserstein gans. arXiv preprint arXiv:1704.00028.

[13] Shaham, T. R., Dekel, T., & Michaeli, T. (2019). Singan: Learning a generative model from a single natural image. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 4570-4580).

[14] Yoo, J., & Chen, Q. (2021). SinIR: Efficient General Image Manipulation with Single Image Reconstruction. arXiv preprint arXiv:2106.07140.

[15] Wu, Z., Shen, C., & Van Den Hengel, A. (2019). Wider or deeper: Revisiting the resnet model for visual recognition. Pattern Recognition, 90, 119-133.

[16] Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017, February). Inception-v4, inception-resnet and the impact of residual connections on learning. In Thirty-first AAAI conference on artificial intelligence.

[17] Woo, S., Park, J., Lee, J. Y., & Kweon, I. S. (2018). Cbam: Convolutional block attention module. In Proceedings of the European conference on computer vision (ECCV) (pp. 3-19).

[18] Vincent, P., Larochelle, H., Bengio, Y., & Manzagol, P. A. (2008, July). Extracting and composing robust features with denoising autoencoders. In Proceedings of the 25th international conference on Machine learning (pp. 1096-1103).

[19] Zhao, H., Gallo, O., Frosio, I., & Kautz, J. (2016). Loss functions for image restoration with neural networks. IEEE Transactions on computational imaging, 3(1), 47-57.

[20] Isola, P., Zhu, J. Y., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1125-1134).

[21] Zhang, R., Isola, P., & Efros, A. A. (2016, October). Colorful image colorization. In European conference on computer vision (pp. 649-666). Springer, Cham.

[22] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems, 30.