A Generative Adversarial Network Model For Different Image generation Tasks

ZHANG, Zhong ZHOU, Yanting

Hong Kong University of Science and Techonology

{zzhangfc,yzhoucf}@connect.ust.hk

Abstract

Image synthesis has been an active research area in computer vision. Many existing models can generate plausible and high-quality results in different generation tasks. However, these models are usually task-specific and computationally expansive, requiring large datasets and excessive effort to train. For many developers with limited resources, a simpler model that requires a smaller dataset and less computational cost is preferred. We tried to construct a simple network combining GAN and U-net that is applicable to different generation tasks without imposing training overhead or modifying the original network structures. To validate our network performance, we evaluated it qualitatively on two image synthesis tasks including image colorization and image inpainting.

1. Introduction

Many problems in computer vision aim at image-to-image translation, including image colorization, style transformation, inpainting, and super-resolution. These translation problems have received significant attention due to their inherent potential in a variety of industries. While most of the architectures are task-specific and some are heavily relied on post-processing work to guarantee performance, we seek to construct a simple and generalizable architecture. To dive into the problem, we restricted our task to the domain of deterministic mapping, with a dataset consisting of paired preprocessed images.

We proposed a simple self-supervised framework for image-to-image translation problems based on a GAN model. In addition, with notice of the semantic capture ability and low data size requirement of U-net [12], we employ it with minor adjustments as the generator of the whole architecture. Also, to further curtail the training time and the number of trainable parameters, we implement another partition-integration feature in the network. Such feature is composed of a partitioning layer and a concatenating layer, it partitions a high-resolution image into small pieces and feeds it to the generator separately. The output images are then being integrated by the learned concatenating layer. The output result, however, sometimes lacks a logical correlation between separate parts, but the overall performance is satisfactory.

To evaluate the model, we trained it for two classic image synthesis tasks: colorization and inpainting. We compare the results qualitatively with several other early works addressing the same problem. The experiments are conducted on the place365 dataset, and the result shows an acceptable performance of our network.

In summary, the major works we have done are as follows:

- We proposed an image synthesis network based on GAN architecture that is generalizable across different tasks. This model is designed to be simple and effective, without excessive training cost and a large dataset.
- To leverage the feature capture ability and the simplicity of the U-net, we employ this framework in the generator of the GAN. Also, we used additional layers to partition and integrate the input image to keep the network on a small scale.
- Experiments of image colorization and image inpainting are conducted to validate the proposed network. The experiment result demonstrates the satisfactory level of our model in comparison to several other existing models.

2. Related work

2.1. Generative adversarial networks (GANs)

The GAN framework [4] has demonstrated impressive results on different image synthesis tasks and has been wildly used in numerous applications. With an adversarial approach of joint training conducted between two separate networks, the model is able to generate images under

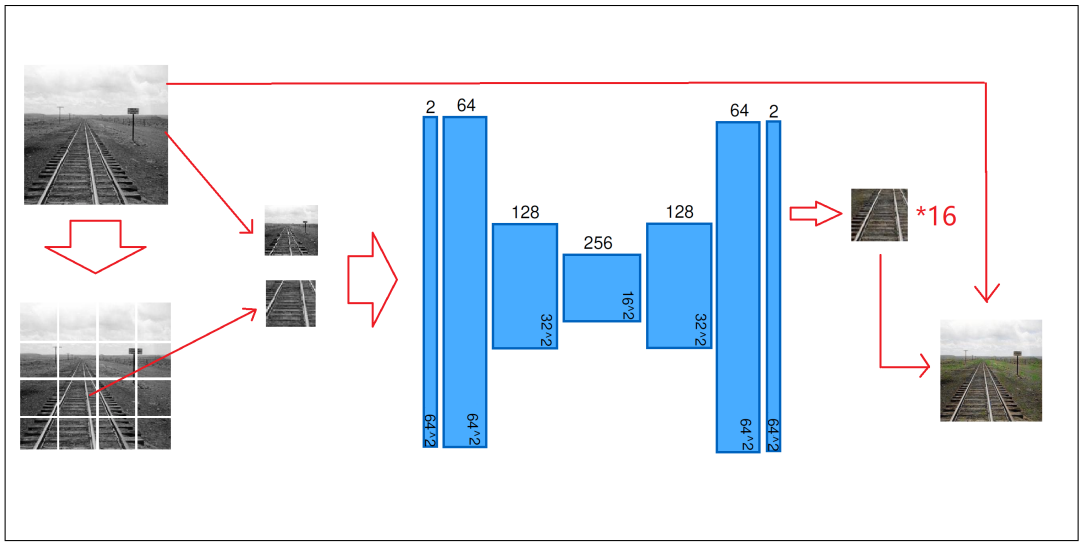

Figure 1. The simplest overall model

a self-supervised setting. Based on GANs, different models are proposed as an extension, including DCGAN [11], cGAN [9], and iGAN [19]. DCGAN stacks deep convolutional layers as generators and discriminators, which can learn hierarchical visual patterns. It easily outperforms the original GAN architecture and has gradually become an industrial standard. cGAN gives an additional condition to generate images other than random noise. As an extension of cGAN, iGAN can take the user input scribble-like data to generate output images accordingly.

In recent years, GAN has been applied to various image synthesis applications, such as image colorization [10], super-resolution [7] and image inpainting [16]. Nevertheless, these models usually involve a large number of parameters and high training costs.

2.2. Image colorization

Image colorization is a popular image-to-image translation problem and has great potential in applications like image restoration. Models to colorize greyscale are generally based on two approaches: hint-guided colorization [8] and automatic colorization [2] [10]. The hint-guided approach requires the user to provide a scribble-like mask or reference images as a hint on the input greyscale image and is heavily depends on human supervision. Cheng *et al*. [2] proposed the first automatic colorization architecture with a deep neural network. In recent years, there are also multiple self-supervised colorization models based on GAN and cGAN being proposed [10] [6]. However, these strategies with GAN generally require a very large dataset and suffer from a long training time.

2.3. Image inpainting

Image inpainting is a reconstruction technology aims at refilling the missing holes in the damaged images. In recent research, many of the frameworks proposed for this image synthesis task is based on adversarial training [3]. To better retain the semantic meaning and coherency of the original image, many additional methods are applied to the original structure. Besides the global discriminator, [1] [3] introduced a PatchGAN discriminator which gives an array output with each value evaluating a corresponding image patch. This strategy effectively alleviates the blind side of GAN's preference for the most "realistic" image instead of the most "well-matched" image. Similarly, [5] proposed a local discriminator focus on the region around the damaged area in addition to the global discriminator. Both approaches can generate good predictions on this task, however, these adjustments might not perform equally well in other image synthesis tasks and the training cost increased to a certain extent.

3. Method

3.1. Network Architectures

Inspired by [20] [6] [11], the overall architecture (Figure 1) for our GAN network will be similar.

The input and output in our task are rather similar - they have the same shape but possibly a different number of channels. Therefore we use a U-net [12] for our generator, which has a symmetrical encoder-decoder architecture. They have the same shape at layer i and layer n-i (n is the total number of layers), as Figure 1 shows. Each down-sampling layer contains a Convolution-BatchNorm-ReLU module while the up-sampling layers contain a Transposed

Convolution instead of Convolution layer.

We apply BatchNorm to prevent the generator from always generating the same image that happens to fool the discriminator as suggested by [13]. To further solve the problem, we trained the discriminator for more than once in each iteration. This helps a lot when we just began the training process. We do not put any fully connected layer in the model, since it often introduces too many parameters but does not perform well in understanding the image.

To retain the meaning of data, we use stride convolution layers instead of pooling layers for down-sampling. It gives the network more space to learn how to downsampling instead of using less meaningful max pooling.[13] Nevertheless, we also tested by how much will pooling layers affect the result in the next section.

Since the discriminator here only needs to tell a single number indicates whether it thinks the graph is generated by the generator or not, we just simply stack Convolution-BatchNorm-LeakyRelu modules many times until it reaches 1*1 shape. It is straightforward and can compute fast. For the same reason, we do not use pooling layers and add BatchNorm layers for each module in the discriminator. Furthermore, Radford *et al*. [11] have shown that using LeakyRelu instead of Relu would increase training speed and performance.

Currently, our model requires around 4M parameters to process 256*256 images. It can be further reduced by using fewer layers in the discriminator. The drawback is that the output of the discriminator is no longer 1*1. Therefore we tried to average all numbers as the only output. It is likewise effective as expected because discriminating part of the graph or discriminating the whole graph sounds equally reasonable. We will use the full 4M-parameter model in this report.

As we are trying to minimize the number of parameters while obtaining acceptable results, making the model more powerful is easy. Simply making the generator U-net deeper, with more shape-holding convolution layers is very likely to make the results better. This deviates from our original goals to introduce a simple, general model but could be an intrinsic potential improvement when one has adequate resources for his specific task.

3.2. Loss function

We tried out two different loss functions, one is BCE loss:

$$\begin{aligned}\mathcal{L}_{GAN}(G,D) =& \mathbb{E}_{x,y}[\log D(x,y)] + \\ & \mathbb{E}_{x,y}[\log(1 - D(x, G(x)))]\end{aligned}$$

and L2 loss:

$$\begin{aligned}\mathcal{L}_{GAN}(G,D) =& \mathbb{E}_{x,y}[(1 - D(x,y))^2] + \\ & \mathbb{E}_{x,y}[(D(x, G(x)))^2]\end{aligned}$$

where x is a input vector, y corresponding to a output vecotr, Generator maps a input to an output $G : x \to y$. D tries to maximize the the loss while G tries to minimize the loss.

To encourage smoothness of the graph, it is appropriate to add an L1 loss:

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y}[||G(x) - y||_1]$$

Therefore the final object is:

$$G^* = \arg\min_G \max_D \mathcal{L}_{GAN}(G,D) + \lambda \mathcal{L}_{L1}(G)$$

where λ is a parameter. We find out that letting $\lambda \geq 1$ at the beginning of the training and reducing λ in the course of training could accelerate training speed.

As for the two different GAN losses, they perform equally well in our tasks.

3.3. Partitioning Graph

We find out that generating a quality high-resolution graph with such a simple model requires a great many parameters and computation, which deviate from our original intention. Therefore we tried to partition an image into smaller low-resolution images, generate results separately, and combine everything afterward. Additionally, we compute another vector from shrunk image and concatenate it with the one from partitioned images before feeding it into the decoder.

4. Experiments

We will introduce how we procees data, compare and analysis results on two tasks separately.

4.1. Image Colorization

4.1.1 Data

Since colorization is a self-supervised task, every set of colorful images can be the dataset. We choose places365 because it contains 256*256 images which is small enough to train with little computing power. The RGB image is translated to lab color space, where l channel represents a greyscale image, ab channels are with color information. Then we normalized the input data. Since we have partitioned the image, l channel of one small sub-image and l channel of the rescaled whole image are fed to the generator. It should output the predicted ab channels. Therefore both input and output number of channels are 2. Before training the GAN, we pretrained the generator using L1 loss between predicted ab channels and ground-truth ab channels. This quickens the training process significantly.

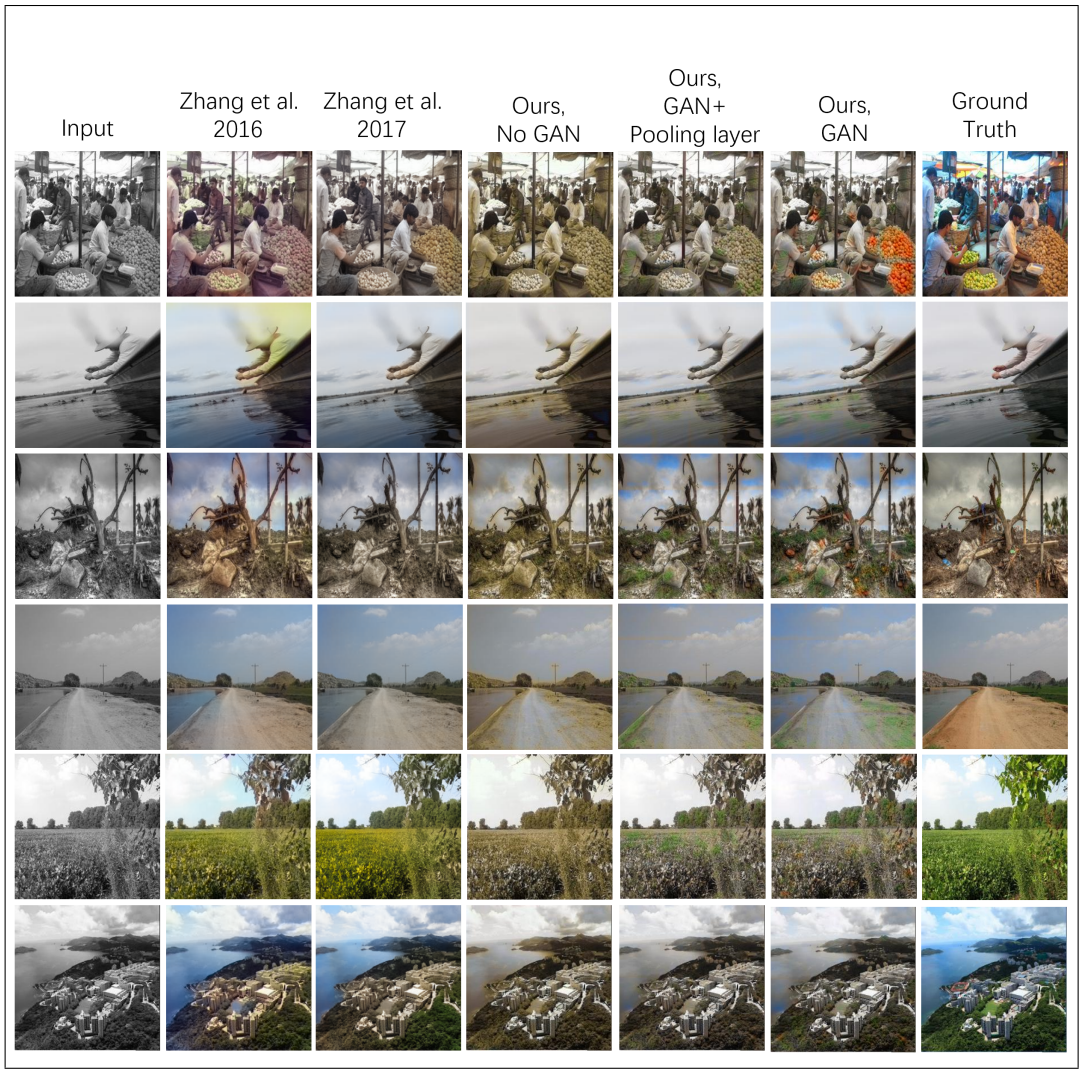

Figure 2. Comparison of colorized image.

4.1.2 Results

The results are shown in Figure 2. We compare our results, both using GAN, and only go through L1 loss pretrained with results by Zhang *et al*. [17] [18], they do not use a GAN but utilize deep convolutional neural networks that focus more on the semantic meaning of the image.

The overall performance of our model is satisfactory. Our model is the only one that succeeds in generating bright orange color in the first example. Though the results are not very close to the ground truth, they are colorful and realistic. It also does not produce "wrong" colors as Zhang 2016 did in the second example. Our model shows its favor of blue and green partly because a large part of our dataset contains sky, trees, and grass. In the case of ambiguous objects, blue or green are also colors that could fool the discriminator easier. However, our model failed in generating bright colors in the last two examples, probably because the low contrast image is harder for the discriminator either. As for the one with pooling layers, it fears to use bright colors even more. The edge of colorful objects is not as sharp as that in the GAN version.

Method	Number of parameters
Zhang *et al*. 2016	32,236,011
Zhang *et al*. 2017	34,187,027
Ours	4,081,027

Table 1. Number of parameters comparison

Our greatest strength is that our model is significantly more efficient, with less trainable parameters (Table 1), a smaller dataset (we use only around 8000 images as our training data), and consequently shorter training time.

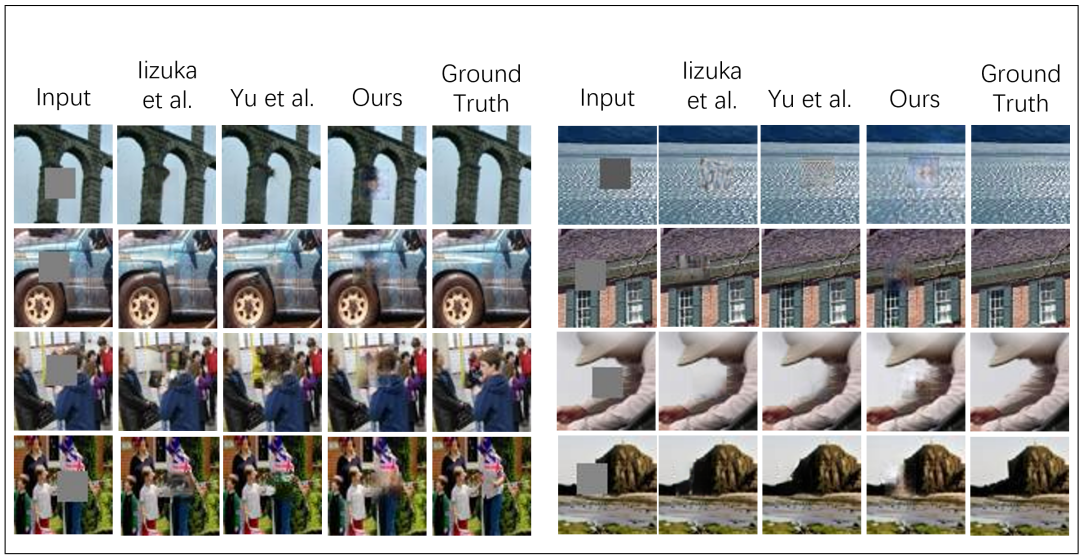

Figure 3. Comparison of image inpainting result.

4.2. Image inpainting

4.2.1 Data

We use the same dataset as the previous task. The original 256*256 image is the ground truth. To create the input image, we assign 0 to a randomly chosen small square in the graph, these "0" worked like a mask. Noticed that the mask is randomly chosen and is not fed into the model. We work under RGB color space in this task, so both input and output has 3 channels.

4.2.2 Results

The results are shown in Figure 3. Our model can only generate a lower resolution image to fill the mask. However, the shape and color of the filler make much sense. In the second example, our model is the only one that produces a rounded wheel; in the third example, our model wisely imagined the backside of the head, which makes the image much more realistic than others. Moreover, our model does not require a mask image as input, instead, it guesses where is the mask from the only input image directly.

Compared to state-of-the-art methods, Iizuka *et al*. [5] used a similar GAN model to ours but it has a deeper generator, a local discriminator that focuses on the mask and a global discriminator that judges the whole image, a similar thought to our image partitioning but applied to the discriminator. Yu *et al*. [15] [14] follows the local-global discriminators but uses an even deeper generator with a refrainment network following a coarse network. It would generate a similar result if the refrainment network is omitted. Therefore it is reasonable to assert our model could achieve better results if we simply make the generator deeper, or add a refrainment u-net. As these methods have two generators and at least 4 times deeper generators, they outperform our model easily, but they require a large dataset and powerful computing.

5. Conclusion

We build a GAN model that could be used in different generative tasks without much modification. It has the advantage of being easy to build, requiring a small dataset, and training fast. We tried many methods to control its performance on training speed and testing, including adding BatchNorm layers, applying pure convolution layers, and using different activation functions. We then evaluated its performance on image colorization and inpainting tasks by comparing our model with others.

Though a deeper network with a more complex encoder architecture that could exploit the semantic meaning of the image could attain more favorable results, its simplicity sometimes makes it a better choice for a smaller project. It could also be the bottom line when compared to more pertinent models. In addition, many state-of-the-art models are developed based on this GAN architecture, which means our model could provide a starting point for any kind of deterministic mapping research.

References

[1] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein. The generalized patchmatch correspondence algorithm. In *European Conference on Computer Vision*, pages 29–43. Springer, 2010.

[2] Z. Cheng, Q. Yang, and B. Sheng. Deep colorization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 415–423, 2015.

[3] U. Demir and G. Unal. Patch-based image inpainting with generative adversarial networks. *arXiv preprint arXiv:1803.07422*, 2018.

[4] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[5] S. Iizuka, E. Simo-Serra, and H. Ishikawa. Globally and Locally Consistent Image Completion. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 36(4):107:1–107:14, 2017.

[6] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, 2017.

[7] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[8] Q. Luan, F. Wen, D. Cohen-Or, L. Liang, Y.-Q. Xu, and H.-Y. Shum. Natural Image Colorization. In J. Kautz and S. Pattanaik, editors, *Rendering Techniques*. The Eurographics Association, 2007.

[9] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[10] K. Nazeri, E. Ng, and M. Ebrahimi. Image colorization using generative adversarial networks. In *International conference on articulated motion and deformable objects*, pages 85–94. Springer, 2018.

[11] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[12] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[13] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[14] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang. Free-form image inpainting with gated convolution. *arXiv preprint arXiv:1806.03589*, 2018.

[15] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang. Generative image inpainting with contextual attention. *arXiv preprint arXiv:1801.07892*, 2018.

[16] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang. Free-form image inpainting with gated convolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[17] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016.

[18] R. Zhang, J.-Y. Zhu, P. Isola, X. Geng, A. S. Lin, T. Yu, and A. A. Efros. Real-time user-guided image colorization with learned deep priors. *ACM Transactions on Graphics (TOG)*, 9(4), 2017.

[19] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros. Generative visual manipulation on the natural image manifold. In *European conference on computer vision*, pages 597–613. Springer, 2016.

[20] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.