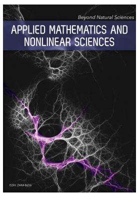
Applied Mathematics and Nonlinear Sciences 6(1) (2021) 43–56



Applied Mathematics and Nonlinear Sciences

https://www.sciendo.com

Nonlinear optimization generating the Tomb Mural Blocks by GANS

Meng Wu[1]†,Adim Payshanbiev[1], Qing Zhao[2],Wenzong Yang[3].

[1] School of Information and Control Engineering, Xi'an University of Architecture and Technology, Xi'an 710055, China

[2] School of Literature, Language and Law, Xi'an University of Architecture and Technology, Xi'an 710055, China

[3] Shaanxi History Museum, Xi'an 710061, Shaanxi, China

Submission Info

Communicated by Juan Luis García Guirao
Received September 25th 2020
Accepted December 2nd 2020
Available online December 31st 2020

Abstract

Tomb murals are the special kind of murals that are buried underground. Due to the narrow exit of the tomb passage, the tomb murals were excavated by dividing the whole mural into blocks, which made lots of information missing between the blocks.The digital restoration technology Image inpainting uses the edge information around the missing parts to spread the information inside of the defect area and fills the information from the outside to the inside. But it is not suitable for filling the missing parts between the tomb mural blocks. Because these parts are large for exemplar-based inpainting which may make texture dislocation and for PDE which may make cartoon blur. It is a need to generate the information outwards to complete the information.The generative adversarial network uses deep learning training by the murals remains to generate the information from inside to outside, but the typical GAN doesn't have a good nonlinear feature.This paper provided a generating technology based on the deep convolution generative adversarial network to rebuild the missing information between the tomb mural blocks. It built the training data set of the simulation platform with Keras and designed a whole mural generation scheme based on DCGAN. In order to get better generated results to avoid the bad artifacts; it adds the nonlinear layers by choosing 13 layers convolution and 2 deconvolution layers of the generator and contained 5 layers convolution discriminator; it designed a new phased nonlinear loss function by using Pycharm pretreatment for Numpy array file data sets; finally, it completed the generate tomb mural information to obtain the good simulation effect.

Keywords: nonlinear optimization, tomb mural, generative adversarial network, loss function.
AMS 2010 codes: 65J15,47J06,94A08,90C35.

†Corresponding author.
Email address: wumeng@xauat.edu.cn

ISSN 2444-8656 doi:10.2478/AMNS.2020.2.00072

Open Access. © 2021 Meng Wu et al., published by Sciendo.
This work is licensed under the Creative Commons Attribution alone 4.0 License.

1 Introduction

The tomb murals buried underground thousands of years reflect the living conditions, social customs, and artistic tastes of the ancient royal vivid. Because the tomb murals are different from these kinds of murals such as the cave murals and temple murals, the cave murals are mainly effected by windy, dusty, and ultraviolet light, and the temple murals are mainly effected by human-made damage. The tomb murals are completely enclosed spaces before being excavated, and the residual information is highly reliable. But they are too huge to excavate once for a whole mural. So, the archaeologists always make the whole mural into several blocks to move to the museum. It caused a lack of information between the mural blocks, which affected the coherence and integrity of the entire mural. Figure 1 is the blocks of playing polo mural from Crown Prince Zhanghuai.

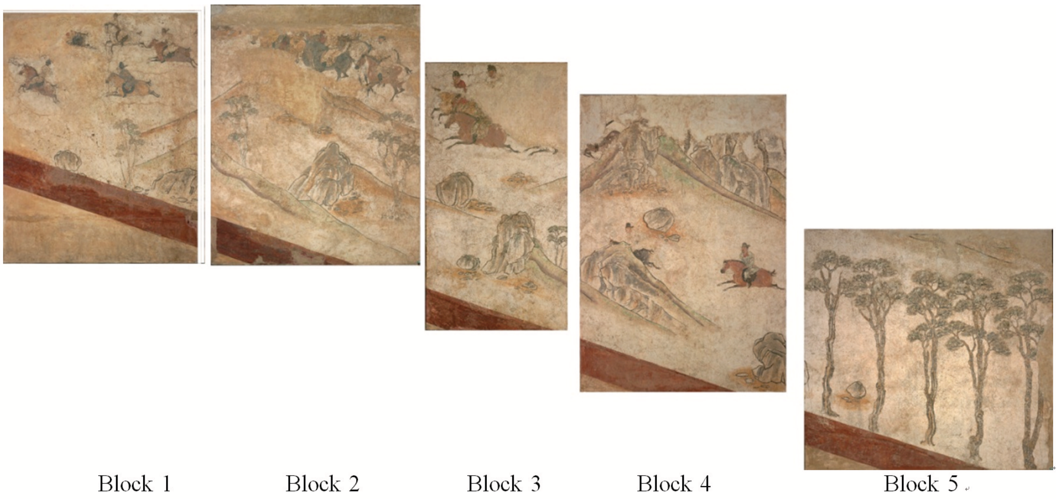


Fig. 1 Blocks of the tomb mural in Zhanghuai‘s tomb

This mural is 6.88 m long and 2.29 m high and dived into 5 blocks. There is some information missing between the neighbours. It is difficult to restore the mural directly because it may lead to stitching dislocation. The first thing should do is that using information extension get the missing parts to help reconstruct the whole mural. The current method of computer-aided ancient murals for digital information restoration is digital image inpainting. It uses the residual mural information to diffuse to the filling edges and fills the information to the missing parts like peeling onions, and finally completes the mural information reconstruction. This kind of technology is mainly completed from two technical directions.On the one hand, the inpainting algorithms based on partial differential equations (PDE) use different diffusion equations to spread the residual information to the missing parts.Such as J Shen and TF Chan [1] proposed a diffusion equation of Total Variation (TV) to solve the diffusion limited optimization problem. Chan et al. [2] added Euler-Lagrange equation to Total Variation to constrain the diffusion direction.Humphrey et al. [3] added auxiliary variables and replaced a function of one variable with a function of two variables to improve inpainting efficiency. Barbu T. et al. [4] proposed a novel TV model based on the second order PDE using the nonlinear second order the Euler-Lagrange diffusion model. They are all belonging to the PDE kind of algorithm. The biggest advantage of these kinds of algorithms is that it is convenient and quick to use, but the disadvantage is that it makes blurry when the missing part is large as the interspace between the mural blocks. On the other hand, it is a texture synthesis model based on exemplar filling. Such as Criminisi et al. [5] proposed an algorithm based on exemplar filling that compares the priority values of each exemplar. [6] changed the exemplars source from global to local using the content perception Markova model. Siadati S Z et al. [7] redefined the priority value of Criminisi‘s algorithm by adding the weight of structure tensor. They are all belonging to the exemplar-based algorithm. The disadvantage is that the filling of a large number of similar exemplars will produce a mosaic effect. Whether the inpainting algorithm based on PDE or exemplar filling does not help to rebuild the whole mural information because the limit residual information and the large missing parts between the blocks. It is a more challenging task than recovering the deleted part of the image in the past. It needs to deduce the texture information of the extension part based on the information inside, and the generated information needs to be as real as possible. There are some new techniques to help generate the information using deep learning, such as S. Iizuka et al. [8] set up a CGAN to generate the information by adding predefined conditions, and Sabini M et al. [9]set up a DCGAN to generate the information out of the image. That kind of information generation algorithm is good for getting the losing information between the blocks. The generative adversarial network is suitable as a deep learning technique compared with the inpainting methods, and it has an excellent data fitting effect, which is manifested in higher fitting efficiency and sharper effect of generated image samples. But the traditional GAN is difficult to train and unstable, so it needs to be modified to achieve better results for the huge tomb mural restoration. So in this paper, we set up a mural generating method based on DCGAN to restore the missing information, and change the construction of the layers in convolution and deconvution by optimizing the nonlinear activation function. After adjusting the pooling layer of the structure in GANs, the tomb mural blocks extend to the two sides of the mural and get the missing information.

2 Set up the mural information generating network

The generative adversarial network belongs to the typical deep learning network [10]. The main parts are the generator module *G* and the discriminator module *D* occupied two pillar positions respectively in the network architecture. The generator constantly captures input image samples and generates samples similar to frame number images itself. The discriminator makes accurate judgments and scores the input data.The work of the generator and discriminator is iterated repeatedly, and the final result is more and more realistic samples generated by the generator, but it is more and more difficult for the discriminator to distinguish between true and false input data.The aim is to get the Nash equilibrium which is reached that the image generated by the generator is infinitely close to the distribution of the real image, while the discriminator can‘t distinguish the new image with the original image, and the prediction probability of the image output by the generator is close to 1/2 [11]. The following figure 2 is the essential GAN structure.

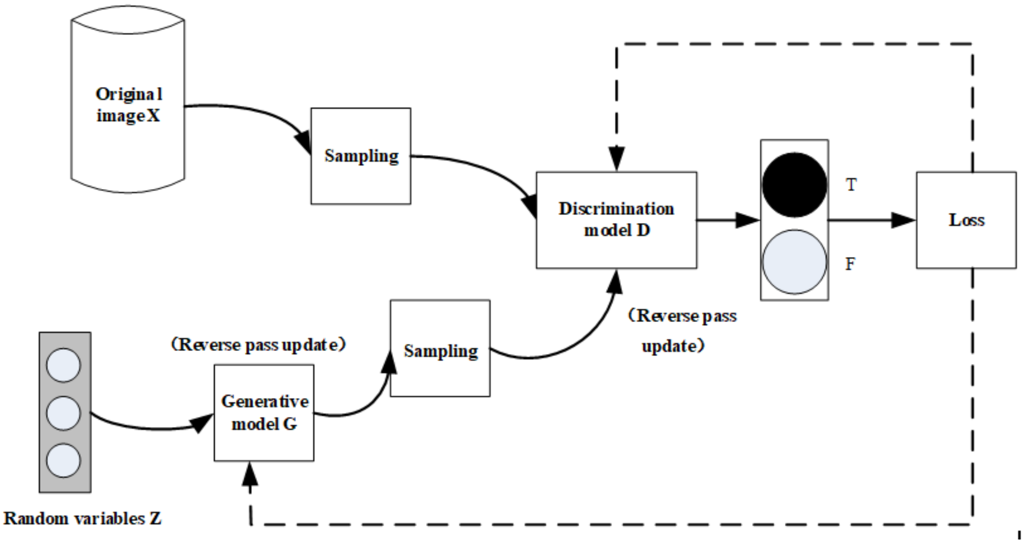


Fig. 2 GAN Network Structure Diagram

Both the generator model and the discriminator model shown in Figure 2 can be constructed with differentiable functions. The sampling of the original image is used as the input data of the identification model, and the random variable is used as the input data of the generation model. The output is as close as possible to the sample $G(Z)$ of the real data distribution set. The goal of the discriminator is to realize the dichotomy identification of data sources. If the input of the discriminator comes from real data, the judgment is true; if the input source is sample $G(Z)$, the judgment is false. The goal of the generator is to make the $D(G(Z))$ displayed on the discriminator of the fake data $G(Z)$ generated by itself be the same as the $D(G(Z))$ displayed on the discriminator of the real data x.

There are more and more extension models based on GANs, and different models have different improvements [12]. The comparison of the different typical GANs has different characteristics, and they are shown in table 1.

Table 1 The comparison for different GANS

GANS type	Improvement points	Highlight improvement effect
f-GAN	Distance metric	The f-divergence metric was used for the discriminator model
WGAN	Distance metric	Wasserstein distance measurement is adopted to improve the stability of network training and prevent network crash effectively
WGAN-GP	Distance metric	Optimize the objective function of the identification model
EB-GAN	Energy model	There are more options for network structure and loss functions
PG-GAN	Add incentive convergence	The training efficiency is improved to obtain high-quality and diverse generated images
LAP-GAN	Add CGAN layer to the Laplace pyramid	Increase the number of pixels in the resulting image
CVAE-GAN	Fused VAE and GAN	Assign a specific label to the image
SGAN	Learn both a single generation model and a single semi-supervised classification	The classification efficiency of experimental data set is improved and the training time scheduling is more flexible
CGAN	Add additional information to the target function in the network	Customize the type of image generation
DCGAN	Add deep convolution layer and deconvolution layer	Recognize more advanced image features to improve network stability performance

The proposal of deep convolution generating confrontation network (DCGAN) largely overcomes the technical difficulties of unstable training of original generative adversarial network and prone model collapse. Therefore, this paper adopts this network model to carry out mural information restoration. The whole structure of the generator is a convolutional neural network structure [13]. Its input is different from that of an ordinary convolutional neural network, which uses images to replace random noise. The convolutional layer mainly uses deconvolution to replace convolution operation. The activation function for the output layer is set to Tanh and the activation functions for the other layers are set to nonlinear ReLU. The discriminative model eliminates the full connection layer and uses another functional layer to replace the full connection layer. The convolutional layer set up in this way can also achieve a good sampling effect. Similar to the generative model, a batch normalization operation is added at all the layers except for the first input, setting the generative function for each layer to nonlinear LeakyReLU. Figure 3 is the network structure of DCGAN.

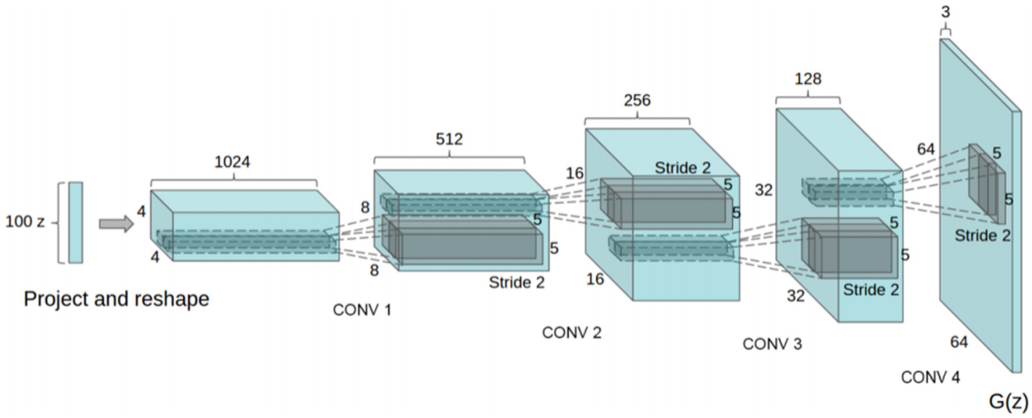


Fig. 3 The network structure of DCGAN

The proposal of this network has a great promotion effect on the development of GAN, and the organic combination of convolutional neural network (CNN) and generative adversarial network (GAN) ensures the quality and diversity of images generated by this kind of structure. In the training process of DCGAN, the special technique of batch normalization (BN) is adopted to make the training process more stable and reliable. After each convolutional layer or deconvolution layer, the corresponding activation function is adopted to effectively solve the problem of gradient disappearance. But the (BN) may generate bad artifact, so it is needed to and more nonlinear layers to make the network more complex. So it is needed to normalize the image batches to a typical Gaussian noise which values from zero to one. By removing the pooling layer, the convolutional neural network can retain the advanced feature information of the image more effectively. The most important thing is that it should take balance to the generator and discriminator training throughout the training process.

3 The nonlinear optimization for generating mural information

3.1 The training scheme for the tomb mural blocks

This paper chooses DCGAN as the essential (D,G)construction, the generative model G uses the encoder-decoder convolutional neural network, and the discriminative model D uses marching convolution repeatedly to down-sampling the image for binary classification. It has a training heart CNN to capture the features of the mural blocks. The way of dimensionality reduction in the deep convolutional networks is similar to that of Galerkin reduced order model (ROM) [14], Variational Autoencoder (VAE) [15] and cluster-based reduced order model (CROM) [16]. Nevertheless, while CNN is able to capture the nonlinear image features since the effective loss functions are non-linear.

The following figure 4 is the flow chart for mural information generation based on DCGAN.

For each training image data I_d , we pre-processed them to Numpy array to get I_n and I_p as the testing group and training group. By running generator G to load I_p to obtain the output image $I_o = G(I_p) \in [0,1]^{128\times 128\times 3}$, and the inside of the generator G includes 13 layers convolution and 2 layers deconvolution. The activation function of convolution is chosen nonlinear LeakyReLU and the activation function of deconvolution is chosen nonlinear ReLU, and the activation function of the final output layer is chosen nonlinear Tanh. The inside of the

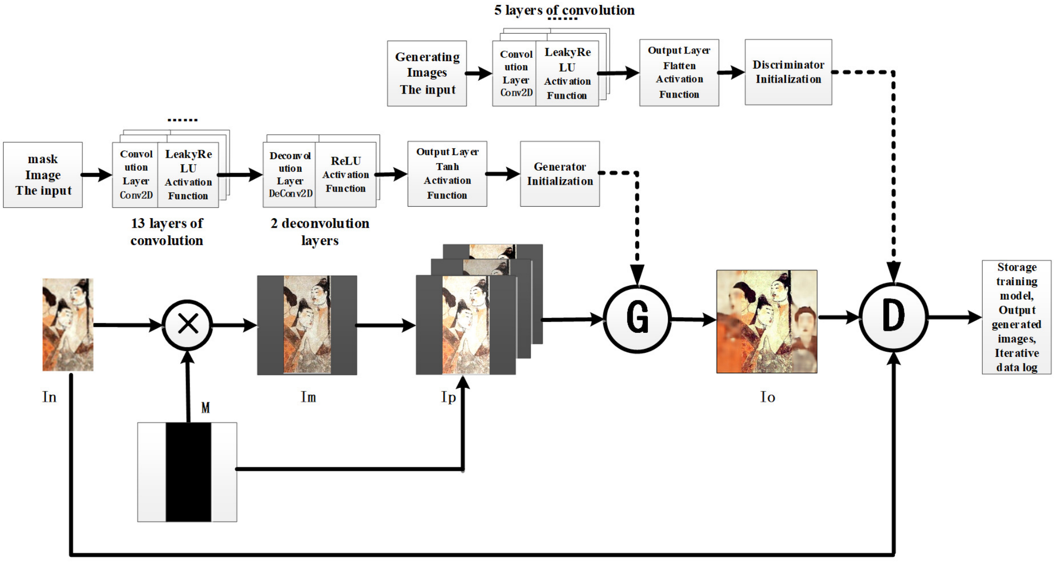

5 layers of convolution
Generating Images The input
Convolution Layer Conv2D
LeakyReLU Activation Function
Output Layer Flatten Activation Function
Discriminator Initialization
mask Image The input
Convolution Layer Conv2D
LeakyReLU Activation Function
13 layers of convolution
Deconvolution Layer DeConv2D
ReLU Activation Function
2 deconvolution layers
Output Layer Tanh Activation Function
Generator Initialization
G
D
Storage training model, Output generated images, Iterative data log
In
M
Im
Ip
Io

Fig. 4 The information generation flow chart on tomb mural

discriminator *D* just includes 5 layers convolution and its activation function is nonlinear LeakyReLU, and the output layer is Flatten. This GAN will generate new mural information constantly to discriminate.

3.2 The nonlinear optimization based on DCGAN

3.2.1 The design of the generator *G*

Since the aim is to predict the original size of the tomb mural, and the convolution often reduces the size of the mural images, a 2-layer transposed convolution is used to restore the original size, and the transposed convolution can generate images from specific input data.

Convolution function

Conv is to calculate the sum of N-dimensional convolution:The Layer-input is the input of each layer;Filter-size is the size of the filter; and Kernel -size choose 4; strides choose 2; activation function is LeakyReLU; the dilation of the convolution kernel expands is 1.

AtrousConvolution2-D is to calculate four-dimensional input and the 4-dimensional convolution kernel operations, and do a 2-D convolution for the input data. All the parameters are the same as Conv. The definition of the convolution function for the generator *G* shows in figure 5.

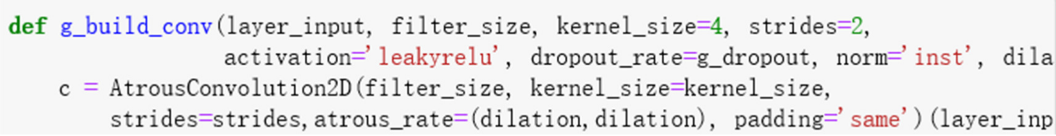
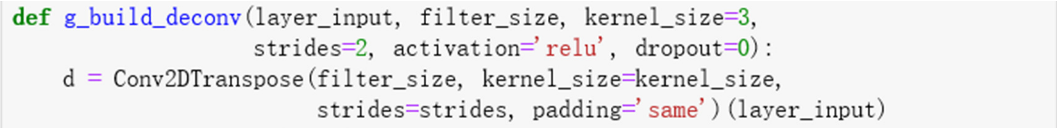

```
def g_build_conv(layer_input, filter_size, kernel_size=4, strides=2,
                 activation='leakyrelu', dropout_rate=g_dropout, norm='inst', dila
    c = AtrousConvolution2D(filter_size, kernel_size=kernel_size,
        strides=strides, atrous_rate=(dilation, dilation), padding='same')(layer_inp
```

Fig. 5 The setting of convolution layers

Deconvolution function

Deconv is to calculate the sum of N-dimensional deconvolution: The Layer-input is the input of each layer; Filter-size is the size of the filter; and Kernel-size choose 3; strides choose 2; activation function is ReLU. The define of the deconvolution function for the generator *G* shows in figure 6.

For the generator *G* , the original encoder-decoder structure is still maintained, and the convolution is expanded to increase the receiving field of neurons and improve authenticity. By doing 13 times convolutions and 2 times deconvolutions for the original mural image data the total parameter construction shows below.



```
def g_build_deconv(layer_input, filter_size, kernel_size=3,
                   strides=2, activation='relu', dropout=0):
    d = Conv2DTranspose(filter_size, kernel_size=kernel_size,
                        strides=strides, padding='same')(layer_input)
```

Fig. 6 The setting of deconvolution layers

The detailed parameters of the generator are shown in Table 2 below.

Table 2 Construction parameter list of generator G

	input	filter	kemel	strides	dilation	output
Conv1	g_shape	64	4	1	1	g1
Conv2	g1	128	4	2	1	g2
Conv3	g2	256	4	2	1	g3
Conv4	g3	512	4	1	1	g4
Conv5	g4	512	4	1	1	g5
Conv6	g5	512	4	1	2	g6
Conv7	g6	512	4	1	4	g7
Conv8	g7	512	4	1	8	g8
Conv9	g8	512	4	1	16	g9
Conv10	g9	512	4	1	1	g10
Conv11	g10	512	4	1	1	g11
Deconv12	g11	256	4	2	1	g12
Deconv13	g12	128	4	2	1	g13
Conv14	g13	128	4	1	1	g14
Conv15	g14	64	4	1	1	g15

3.2.2 The design of the discriminator D Convolution function

The convolution function for discriminator D is the same as the generator G .

Dcrm-loss is the loss function for discriminator, set two tensors y-true and y-pred. Set I_l is the left part of I_d , and set I_l' is the right part of I_d . Let it fast flip around the ordinate axis to help the input D_l have the feedback in the left domain. For the aim of generating the prediction on I_d , the discriminator D calculate the $D_g(I_d)$, $D_l(I_l)$ and $D_l(I_r')$ separately. And put these three outputs into the connector C , and then get the final result of the discriminator D .

$$p = C(D_g(I_g)\|D_l(I_l)\|D_l(I_r')) \tag{1}$$

The discriminator uses 5 layers of 5×5 convolutional layers to repeatedly down-sample the mural images and performs dichotomy. The final parameters of the discriminator are shown in table 3 below.

Table 3 Construction parameter list of discriminator D

	input	filter	kemel	strides	output
Conv1	d_input	32	5	2	d1
Conv2	d1	64	5	2	d2
Conv3	d2	64	5	2	d3
Conv4	d3	128	5	2	d4
Conv5	d4	128	5	2	d5

The discriminator is used for distinguishing the real mural images I_d from the generated outputs mural images I_o . In the discriminator, $D_g(I_d) = 1$ if the targeted outputs are accepted while $D_g(I_d) = 0$ if they are rejected.

3.2.3 Training and define the loss function

In general, the training process in GAN can be considered as a minimization-maximization problem based on across entropy loss function $J(D,G)$:

$$\min_G \max_D E_{I_d \sim p_{data}(I_d)}[logD(I_d)] + E_{\mu \sim p_\mu}[log(1 - D(G(\mu)))] \tag{2}$$

Where $p_u(u)$ is a prior distribution for the random dataset$u$,and $p_{data}(I_d)$ is the corresponding probability data distribution for the targeted outputs I_d. In particular, the min-max process consists of two steps. Stept 1: Given a fixed generator, updating the discriminator by maximizing the discriminator function max $J^{(D)}((\theta)^{(D)},(\theta)^{(G)})$.

$$J^{(D)}(\theta^{(D)},\theta^{(G)})_{\max} = E_{I_d \sim P_{data}(I_d)}[logD(I_d,\theta^{(D)})] + E_{\mu \sim p_\mu(\mu)}[log(1 - D(G(\mu)),\theta^{(G)},\theta^{(D)})] \tag{3}$$

Stept 2: Updating the generator by minimizing the generator function min $J^{(G)}((\theta)^{(D)},(\theta)^{(G)})$.

$$J^{(G)}(\theta^{(D)},\theta^{(G)})_{\min} = E_{\mu \sim p_\mu(\mu)}[\log(1 - D(G(\mu)))] \tag{4}$$

The min-max game has a global optimum for$p_g(I) = p_{data}(I_d)$, if$D$ and G have enough capacity. The optimal discriminator for the$J^{(G)}(G,D)$ can be characterized by:

$$D^*(I_d) = \frac{P_{data}(I_d)}{P_{data}(I_d) + P_g(I)} \tag{5}$$

The min-imax game in(2)can be rewritten as

$$\max_D J^{(G)}(G,D) = E_{I_d \sim p_{data}}[\log \frac{p_{data(I_d)}}{p_{data(I_d + p_{g(I)})}}] + E_{I \sim P_g}[\frac{p_g(I)}{p_{data}(I_d) + p_g(I)}] \tag{6}$$

When the $D^*(I_d) = \frac{1}{2}$, the generator can use any random dataset u to synthesis the generated outputs mural image I , and the discriminator loses the ability to distinguish between the real dataset I_d and generated new mural image I.

In order to stabilise the training process, the three typical loss functions are defined.

$$J_D(I_n,I_p) = -|\log D(I_n) + \log(1 - D(G(I_p)))| \tag{7}$$

$$J_G(I_n,I_p) = J_{MSE}(I_n,I_p) - \alpha \cdot \log D(G(I_p)) \tag{8}$$

$$J_{MSE}(I_n,I_p) = \|M(G(I_p) - I_n)\|_2^2 \tag{9}$$

J_{MSE} refers to the mean square error of the original image I_n and the restored imageI_p ,the smaller the better.$J_D$is the loss function of the discriminator. It is hoped that the discriminator‘s ability to discriminate whether the image is generated by the generator or the real image, that is the probability of discrimination is infinitely close to 1/2. J_G is the generator loss function, and $\alpha$is an adjustable hyperparameter, which is used to weigh the error loss of the mean square and the generation of the standard generator against the network loss.Finally, the generator weight is updated by the training iteration J_{MSE} to adjust the generator. The weight of the discriminator is updated according to the iterative L_D to adjust the discriminator.

3.3 Set the training mural image database

3.3.1 Choose training images for pre-processing and post-processing

The tomb mural is huge for digital, so it is better to capture it by 114 sub-lens images which are divided into nine groups. Each mural image is divided into 4×416 non-overlapping blocks, and then $114\times 16=1824$ blocks are compressed into 256×256 images for unified numbering. Here we choose the tomb mural data set with a total number of 1487 images were created. The normalized data set is shown in Figure 7 below.

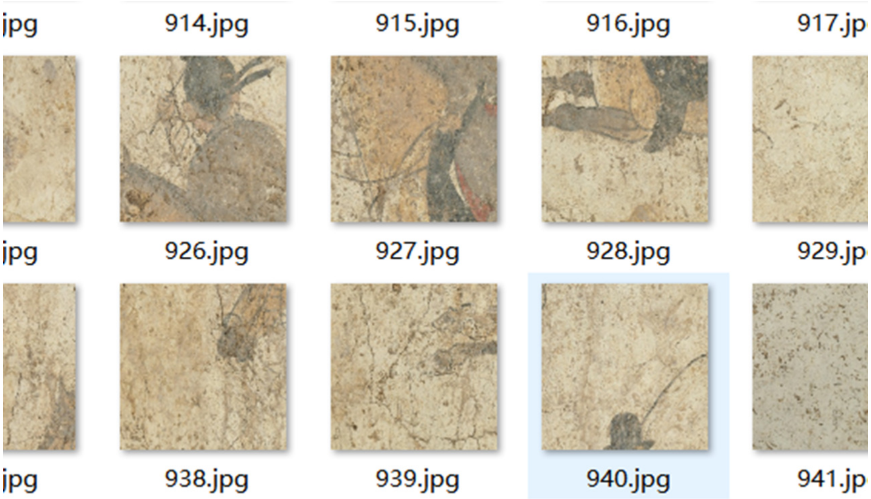


Fig. 7 The training mural image database

There are totally more than 20 GB data for training in the five big blocks in figure 1 shown.

Table 4 The training data in tomb mural blocks in Crown Prince Zhanghuai

	Block 1	Block 2	Block 3	Block 4	Block 5
Pixel	2210*1800	2220*1850	2270*1290	2510*1830	2060*2120
Number of blocks	384	367	362	324	387
$Data\setminus GB$	4.44	5.01	2.07	4.44	5.00

In order to prepare for the image training, this paper uses the following pre-processing to give a training image I_{tr}. Firstly the image is normalized to$I_n\in[0,1]^{128\times 128\times 3}$, and then defined a mask $M\in[0,1]^{128\times 128}$, and let M_{ij} can cover the middle part of the image. Secondly, it should calculate the average pixel strength u in the non-mask part $I_n\odot(1-M)$. Thirdly it set as the external pixies value for each channel. To improve the quality of the final output effect, it should do little post-processing for the final I_o . Using $I_o'=255\cdot I_o$ to reformatI_o , and using seamless clone to fuse I_{tr} and I_o', to output the final resultI_{op} .

3.3.2 Define the Mask

After preparing the data set, normalize the data set to $I_n\in[0,1]^{128\times 128\times 3}$, it should define a mask that uses a binary mask M (mask) with only two values of 0 and 1. The value of 1 means the effective information on the mural which needs to remain, and the value of 0 means the missing part which needs to be restored. Multiply the elements in I_n by the elements in M. The product of the multiplication of the elements at the corresponding positions of the two matrices is the Hadamard product. Define the I_m as bellow:

$$I_m=\mu\cdot M+I_n\odot(1-M) \tag{10}$$

Where the parameterμ is the average pixel value and M is the mask area. Area $I_n\odot(1-M)$ is an area without the mask. Add the two results toI_m , and use I_m and M to generate the $I_p \in [0,1]^{128\times128\times4}$. The program outputs (I_n , I_p) is the image for generating a confrontation network training.

4 The mural generation results and analysis

4.1 The information generation for mural clocks

The mural Polo in Crown Prince Zhanghuai tomb has a total of 1487 pieces of 256 × 256 images, which requires more than 40 hours for training. The continuity of the generated information is better, and the color matches with the initial background are higher, and the pattern texture of the mural can be well restored. And, as the training time increases, the more iterations of the training cycle, the training effect will gradually improve, and the overall image integration will be better.

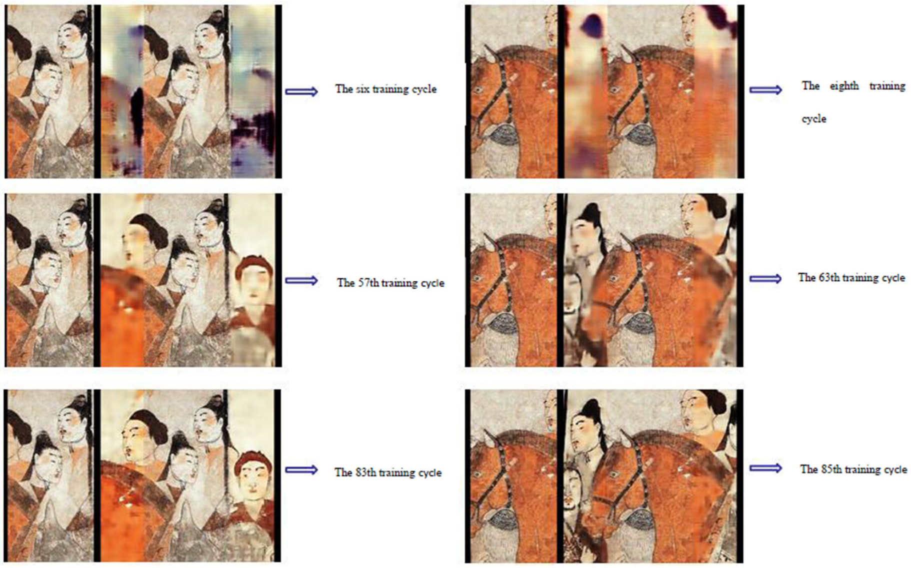


Fig. 8 The generations in different training cycles

From figure 8 it shows that different training cycle can lead to different generation results. The mural is too huge to training for a long time. It depends on the computer‘s operation capability. This experiment is output by a computer that has Intel(R) Xeon(R) CPU E5-1650v3@ 3.50 GHz and NVIDIA GeForce RTX 2080 Ti. The training module is set to save the log file during the repair process. The stored data is mainly the loss data of the generator and the discriminator and the recognition loss. The data file is exported and visualized statistical analyzing. The changes in the parameters of the generator loss and the discriminator loss are shown in figure 9.

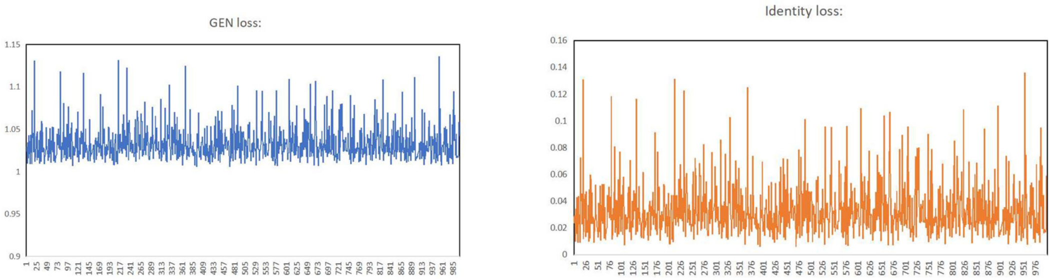


Fig. 9 The Generator loss and Discriminator loss

The generator loss parameters around 1.03 and the recognition loss parameters fluctuated around 0.02. The execution time of each step pre-sets as two minutes. With the superposition of the number of iterations, the fluctuation of the final loss function will become smaller and smaller, and eventually tend to be a stable value. The quality of the generated information is getting better and better, and the detailed features are restored. The following figures are the mural information extension on this platform.

4.2 The analysis of generated information

The experiment used Keras and Tensorflow to do the tomb mural blocks image extension. Keras is an open source library written by Python. Its biggest advantage is able to use the function library to complete the design of deep learning model debugging and successfully applied to solve specific problems. Tensorflow is a very good numerical open source library. It can clearly point to deal with the process of generating the training process. There are two groups of experimental effects. One is generated by the essential outpainting algorithm with 8 convolutions and 1 de deconvolutions DCGAN in the article [17], the other is our existential algorithm with 13 convolutions and 2 de deconvolutions DCGAN.

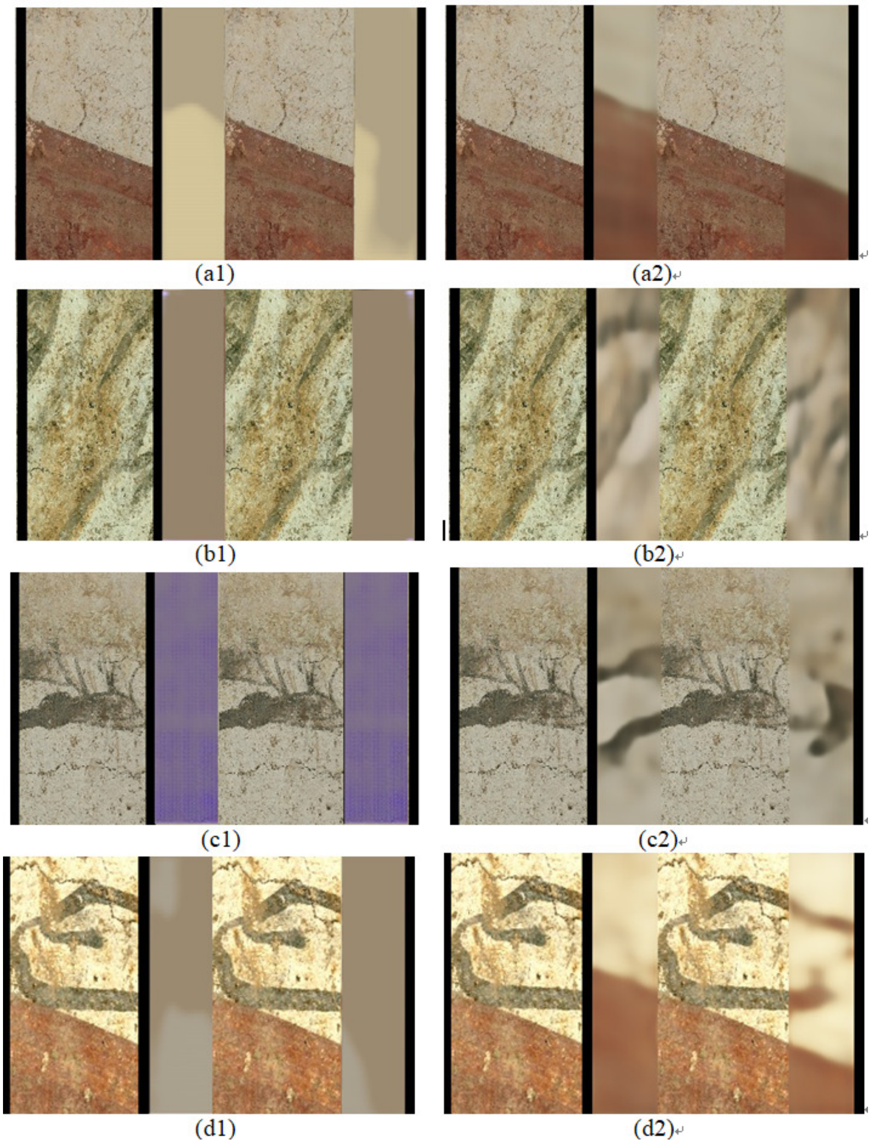


Fig. 10 The comparisons of information extension results

The generating results in figure10$(a1) \sim (d1)$ are generated by article [17], and our DCGAN generating results are shown in figure10$(a2) \sim (d2)$. We choose structural similarity (SSIM) and peak signal-to-noise ratio (PSNR) to evaluate the results.

SSIM is usually used to measure the structural similarity of the two images. The formula derivation mainly involves three measurement indicators: brightness, contrast, and overall image structure. Specifically, the average value can represent the brightness, the standard deviation represents the contrast, and the covariance represents the degree of structural similarity. It is shown in formula 11.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{({\mu_x}^2 + {\mu_y}^2 + c_1)({\sigma_x}^2 + {\sigma_y}^2 + c_2)} \tag{11}$$

μ_x is the average value of existing mural information x, μ_y is the average value of extended mural information y, σ_x^2 is the variance of x, σ_y^2 is the variance of y, and σ_{xy} is the covariance of x and y. c_1 and c_2 are constants. The range of structural similarity is 0 to 1. When generating information and existing information is exactly the same, the value of SSIM is equal to 1.

PSNR means the peak signal that reaches the noise ratio. It is generally used for evaluating the total visual result. It can measure the generated information that has the same quality as the original mural image. The definition of PSNR is shown as formula 12.

$$PSNR = 10{*}\log_{10}(\frac{{MAX_I}^2}{MSE}) \tag{12}$$

$MAX_I^2$is the maximum possible pixel value of the image. If each pixel is represented by an 8-bit binary, the value is 255. Different values of PSNR represent different levels of image quality.

It can be seen from Table 5 the different results in SSIM and PSNR.

Table 5 The evaluation of the generation results in article [17] and ours

Evaluation	figure a	figure b	figure c	figure d
SSIM value[17]	0.842	0.756	0.597	0.874
SSIM value(ours)	0.937	0.842	0.731	0.943
PSNR value[17]	26.505	24.272	23.174	25.224
PSNRvalue(ours)	34.852	31.758	28.226	31.741

By reconstructing the DCGAN for extending the inner information to the mural block side and generating the missing information to get the total mural. Our algorithm is better not only in the visual effect but also in the quantify evaluation.

5 Conclusion

Traditional image inpainting is to fill the holes in the image, but our restoration technique can help to join the separated mural block by generating the missing information between the mural blocks. The information training based on the modified DCGAN model using the tomb mural residual image can be used to generate the extension part of the mural blocks. It adds CNN to the typical GAN to have a better nonlinear feature. For the generator$G$, we keep the original encoder-decoder structure, and exchange convolution 13 layers to 8 layers and exchange deconvolution 2 layers to 1 layer to increase the reception of neurons and improve the authenticity. The more nonlinear convolution layers make it get better results in completing the entire screen.

However, due to the scarcity of the mural residual information, there are not enough images fit for training. The final image restoration results are not perfect. The mural data set can be expanded later to optimize the restoration effect. This program only can process images with 256×256pixels. It makes the pixels limit by the data size of the image and makes the generated information not be clear and blurry. So to increase the tomb mural training database is the next thing to do.

Acknowledgements: The authors acknowledge this work is supported by National Natural Science Foundation of China (No. 61701388), and Natural Science Foundation of Shaanxi Province of China (2018JM6080), and the Project of Xi'an Science and Technology Bureau (GXYD10.1)

References

[1] Chan T F, Shen J. Nontexture inpainting by curvature-driven diffusions [J]. Journal of Visual Communication and Image Representation, 2001, 12(4): 436-449.

[2] Chan T.F, Shen J. Variational image inpainting [J].Communications on pure and applied mathematics, 2005, 58(5):579-619.

[3] Humphrey D.Tauhman D.A filtering approach to edge preserving MAP estimation of image [J].IEEE Transactions on Singal Processing, 2011, 20(5):1234-1248.

[4] Barbu T. Variational image inpainting technique based on nonlinear second-order diffusions [J]. Computers and Electrical Engineering, 2016, 54: 345-353.

[5] Criminisi A, Perez P, Toyama K. Region filling and object removal by exemplar-based image inpaintin g[J]. Image Processing, IEEE Transactions on, 2004, 13(9): 1200-1212.

[6] Ruzic T, Pizurica A. Context-aware patch-based image inpainting using markov random field modeling[J]. Image Processing, IEEE Transactions on, 2015, 24(1): 444-456.

[7] Siadati S Z, Yaghmaee F, Mahdavi P. A new exemplar-based image inpainting algorithm using image structure tensors[C]//Electrical Engineering (ICEE), 2016 24th Iranian Conference on. IEEE, 2016: 995-1001.

[8] S. Iizuka, E. Simo-Serra, and H. Ishikawa. Globally and locally consistent image completion. ACM Transactions on Graphics (TOG), 36(4):107, 2017

[9] Sabini M, Rusak G. Painting outside the box: Image outpainting with GANs [J]. arXiv preprint arXiv:1808.08483, 2018.

[10] Hitaj B, Ateniese G, Perez-Cruz F. Deep models under the GAN: information leakage from collaborative deep learning[C]//Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 2017: 603-618.

[11] Oliehoek F A, Savani R, Gallego J, et al. Beyond local nash equilibria for adversarial networks[C]//Benelux Conference on Artificial Intelligence. Springer, Cham, 2018: 73-89.

[12] Kahng M, Thorat N, Chau D H P, et al. Gan lab: Understanding complex deep generative models using interactive visual experimentation [J]. IEEE transactions on visualization and computer graphics, 2018, 25(1): 1-11.

[13] Frid-Adar M, Diamant I, Klang E, et al. GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification[J]. Neurocomputing, 2018, 321: 321-331.

[14] Xiao D, Fang F, Zheng J, et al. Machine learning-based rapid response tools for regional air pollution modelling[J]. Atmospheric environment, 2019, 199: 463-473.

[15] Gonzalez F J, Balajewicz M. Learning low-dimensional feature dynamics using deep convolutional recurrent autoencoders[J]. arXiv preprint arXiv:1808.01346, 2018.

[16] Nair A G, Yeh C A, Kaiser E, et al. Cluster-based feedback control of turbulent post-stall separated flows[J]. Journal of Fluid Mechanics, 2019, 875: 345-375.

[17] Ying Z, Bovik A. 180-degree Outpainting from a Single Image [J]. arXiv preprint arXiv:2001.04568, 2020.

This page is intentionally left blank