(12) INNOVATION PATENT (11) Application No. AU 2017101166 A4
(19) AUSTRALIAN PATENT OFFICE

(54) Title
A Method For Real-Time Image Style Transfer Based On Conditional Generative Adversarial Networks

(51) International Patent Classification(s)
G06T 1/40 (2006.01)

(21) Application No: **2017101166** (22) Date of Filing: **2017.08.25**

(45) Publication Date: **2017.11.02**
(45) Publication Journal Date: **2017.11.02**
(45) Granted Journal Date: **2017.11.02**

(71) Applicant(s)
Haolun Wu;Haodong Lai;Yutao Liu;Wentao Zhou;Qinlai Li

(72) Inventor(s)
Wu, Haolun;Lai, Haodong;Liu, Yutao;Zhou, Wentao;Li, Qinlai

(74) Agent / Attorney
Gloria Li, 65b hattaway avenue, buckland beach, Auckland, 2012, NZ

2017101166 25 Aug 2017

ABSTRACT

The present invention discloses a method for achieving image style transfer in real time by using conditional Generative Adversarial Networks(cGAN). By using these networks, the invention achieves a good performance at high-level image information. The cGAN networks not only learn the mapping frominput images to output images, but also learn a loss function to train this mapping. The main steps include: using a traditional way to generate training datasets to get input and target pairs of images; puttingimages into the encoder of cGAN to extract the features and decoding them by using a "U-Net" architecture which is an advanced convolutional network; optimizing the values of generator and discriminator simultaneously by using a gradient descent approach based on the established loss functions and ultimately getting our results. The cGAN makes it possible to applythe same generic approach to problems that traditionallywould require very different loss formulations.In addition to the advantage of the fast speed, this invention is effective especially for dealing with high level image features and synthesis which can have more explicit results.

2017101166 25 Aug 2017

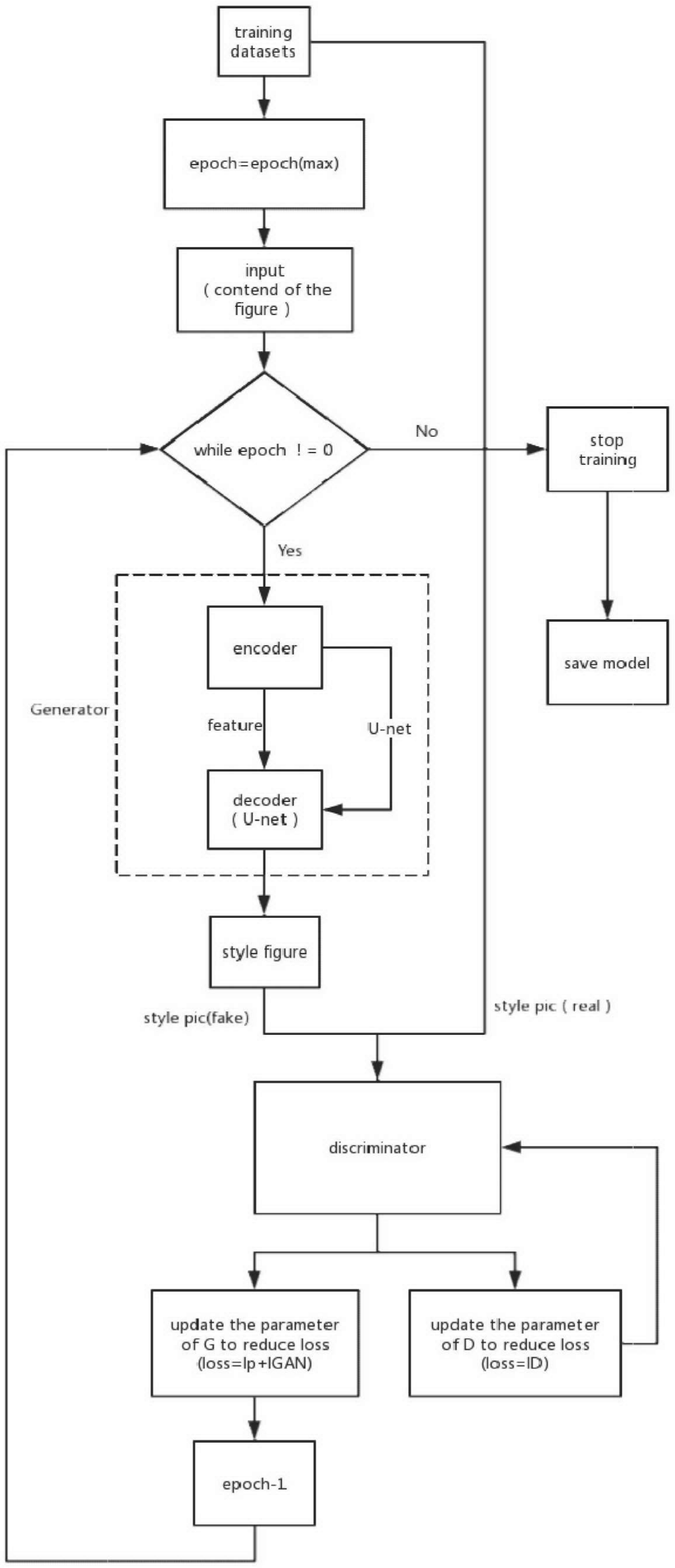

Fig. 1.

2017101166 25 Aug 2017

TITLE

A Method For Real-Time Image Style Transfer Based On Conditional Generative Adversarial Networks

FIELD OF THE INVENTION

The present invention generally relates to an image-to-image processing technique for style transfer in a fast way, especially based on a method by using conditional Generative Adversarial Networks (cGAN).

BACKGROUND OF THE INVENTION

Transferring the style from one image onto another can be regarded as a problem of texture transfer, which is of practical significance. In daily life, human beings usually want to transfer their photographs and paintings to specific styles by post editing. Professional image editing requires a great artistic talent and rich experience, which is never easy for ordinary people to do so.

Most previous transfer algorithms rely on non-parametric methods to synthesize photo-realistic natural textures by resampling the pixels of a given source texture while using different ways to preserve the structure of the target image. More recently, plenty of image synthesis methods based on deep neural networks have emerged.The related reference references are:

[1]N. Ashikhmin. Fast texture transfer. `IEEE Computer Graphics and Applications`*, 23(4):38–43, July 2003.*

[2]A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pages 341–346. ACM, 2001.

[3]Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning withdeep conditional generative adversarial networks. CoRR abs/1511.06434 (2015).

Although these methods achieve remarkable results, there are some notable limitations. The non-parametric methods merely focus on the low-level features, so that they cannot perform well in generating an ideal image when considering advanced high-level style features. What's more, when using the methods on deep neural networks, the outputs of networks are difficult to be controlled because of too much freedom in the model. Therefore, a basic prerequisite is to find an independent model to synthesize detail features of images beyond what is contained in the input. The model should solve the above problems by learning both the pixel elements and the high-level style features.

SUMMARY OF THE INVENTION

The objects of present invention are to put forward a method for achieving an image-to-image style transfer in a fast way to avoid the above problems. The present invention disclosed an approach based on the conditional Generative Adversarial Networks(cGAN).

Image-to-image translation problems are often formulated as per-pixel classification or regression. These formulations treat the output space as "unstructured" in the sense that each output pixel is considered conditionally independent from all others given the input image. Conditional GANs instead learn a structured loss. Structured losses penalize the joint configuration of the output. Besides, Previous works have conditioned GANs on natural images. The image-conditional models have tackled inpainting, image prediction from a normal map, image manipulation guided by user constraints, future frame prediction, future state prediction, product photo generation, and style transfer. The related references are as follows:

[4]C. Li and M. Wand. Combining markov random fields and conditional neural networks for image synthesis. CVPR, 2016.

[5]C. Li and M. Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. ECCV, 2016.

[6]J.-Y. Zhu, P. Krahenbuhl, E. Shechtman, and A. A. Efros. ¨Generative visual manipulation on the natural image manifold. In ECCV, 2016.

The conditional GAN in this invention is different in that the loss is learned, and can, in theory, penalize any possible structure that differs between output and target. The method also differs from other works in some choices. Unlike previous works, this invention uses a "U-Net" architecture, and for a discriminator it uses a conditional "PatchGAN" classifier.

GANs are generative models that learn a mapping from random noise vector *z* to output image $y:G:z \rightarrow y$. In contrast, conditional GANs learn a mapping from observed image *x* and random noise vector *z*, to $y:G:\{x,z\} \rightarrow y$. The generator *G* is trained to produce outputs that cannot be distinguished from "real" images by an adversarially trained discriminator*D*, which is trained to do as well as possible at detecting the generator's "fake". Therefore, we use a traditional model to train an abundant of pictures and get the style images which are regarded as the "real", while we input the same training set into the model in this invention and what we get is the "fake". Our ideal goal is to let the discriminator *D* cannot distinguish the "real" and the "fake".

Besides, we provide a Gaussian noise *z* as an input of generator, in addition to *x*.

Each time we get an image by using the generator *G*, we send that output image to discriminator *D*. The discriminator *D* will try its best to judge the output is whether "fake" or "real" . If it is evaluated as "fake", a f*ull 0 matrix* will be used as an output. Otherwise, if it is evaluated as "real", a *full 1 matrix* will be considered as an output in this step.

The objective of a conditional GAN can be expressed as

$$L_{cGAN}(G,D) = E_{x,y\sim p_{data}(x,y)}[\log D(x,y)] + \\ E_{x\sim p_{data}(x),z\sim p_x(z)}[\log(1-D(x,G(x,z))]$$

Then we define L_p as the difference between generate image and the ground truth(the "real" and the "fake") at pixel level, which can be expressed as $L_{real} - L_{fake}$. More specifically, we choose the matrix norm to formulate it as

$$L_P(G) = E_{x,y\sim p_{data}(x,y),z\sim p_{data}(z)}[\|y-G(x,z)\|_1].$$

Based on the principle of cGAN, *G* tries to minimize this objective against an adversarial D that tries to maximize it, i.e.

$$G^* = \arg\min_G \max_D L_{cGAN}(G,D).$$

Thus, we use $L_{cGAN}(G,D)$ combined with $L_P(G)$ to train the generator to minimize the loss and our final objective is

$$G^* = \arg\min_G \max_D L_{cGAN}(G,D) + \lambda L_p(G)$$

Similarly, the discriminator is simultaneous trained to maximize the loss:

$$L_D = E_{x,y\sim p_{data}(x,y)}[\log D(x,y)] + \\ E_{x\sim p_{data}(x),z\sim p_x(z)}[\log(1-D(G(z))]$$

So, we update the weights of the encoder-unit in generator to minimize $V(G,D)$ and simultaneously update the weights of the unit in discriminator to maximize $V(G,D)$, while x is the input image, y is the target style image and $G(x,z)$ is the generated image.

Many previous solutions to problems in this area have used an encoder-decoder network, such as *[7]G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. Science,313(5786):504–507, 2006.* In such a network, the input is passed through a series of layers that progressively downsample, until a bottleneck layer, at which point the process is reversed. Such a network requires that all information flow pass through all the layers, including the bottleneck. For many image translation problems, there is a great deal of low-level information shared between the input and output, and it would be desirable to shuttle this information directly across the net.

To give the generator a means to circumvent the bottleneck for information like this, we add skip connections, following the general shape of a "U-Net". Specifically, we add skip connections between each layer i and layer n - i, where n is the total number of layers. In this invention, the n equals to 8. Each skip connection simply concatenates all channels at layer i with those at layer n - i.

Consequently, we avoid a massive calculation when transferring each picture for a specific style. Instead, we let our our model learn the implementation way in advance, which means each time we get an input, we can have our results in real time. Additionally, the use of "U-Net" architecture avoids the loss of information in every step of the encoder and decoder and offers more information during the deconvolution. Eventually, we have a good result especially for the semantic features in a high-level and obtain a real-time method to achieve image style transfer.

DESCRIPTION OF DRAWINGS

The following drawings are only for the purpose of description and explanation butnot for limitation, where in:

Fig.1 is the flow diagram of the core algorithm of our invention, which shows the specific step in our work step by step;

Fig.2 is the structure of patchGAN discriminator: inputof the original image (yellow block) and the "real" target image(red block), and the "fake" image (blue block);

Fig.3 shows two choices for the architecture of the generator. The"U-Net" is an encoder-decoder with skip connections between mirrored layers in the encoder and decoder stacks;

Fig.4 illustrates the "U-net" architecture (example for 32×32 pixels in the lowest resolution). Each bluebox corresponds to a multi-channel feature map. The number of channels is denotedon top of the box. The x-y-size is provided at the lower left edge of the box. Whiteboxes represent copied feature maps. The arrows denote the differentoperations;

Fig.5 shows the process of training a conditional GAN to predict aerial photos frommaps. The discriminator, *D*, learns to classify between real andsynthesized pairs. The generator learns to fool the discriminator. Both the generator and discriminator observe an input image.

Fig.6 shows the results of our invention. The first column lists the original images we used in our model, while the first row shows different styles. Each image at the cross point is our result after transferring the style of the input image(at the left of the row) to the specific style (at the top of the column).

DESCRIPTION OF PREFERRED EMBODIMENT

In order to make the present invention more readily understood, the embodiments of the present invention will be explained in details. The whole process is shown in *Fig. 1.*

Compared with the traditional way of image style transfer, we used cGAN to make the output more realistic in an efficient way. The generator of cGAN is a symmetric structure. Its encoder and decoder have *8* convolutional and deconvolutional layers respectively.

Step A: We use a traditional way to generate training sets. The related reference is *[8]**LA Gatys , AS Ecker , M Bethge, Image Style Transfer Using Convolutional Neural Networks, IEEE Conference on Computer Vision & Pattern Recognition** , 2016 :2414-2423* . We need to get enough images for training the model to generate realistic images, so we generate data sets which contain *1000* images. *800* of them are used to train the model and the rests are used to test. Generating an image needs *30* seconds by using this model.

Step B:Afterobtaining the content image (input) and transferred image (target) pairs, we need to map the input domain to the output domain. We put the content image into the encoder of cGAN. An image can be regarded as atensor of three dimention, or threematrices since one image consist of three color channel red, blue and green. The convolutional layer is the core building block of a CNN, of which the outputs are feature maps. Parameters of each layer consist of filters and biases. For a certain layer, each feature map is calculated by convolving all feature maps in former layer by its own filter and summed together, then a bias is added. Thus every filter can produce a feature map, and then we stack the series of feature maps along the depth dimension forms the full output volume of the convolution layer. The encoder is a CNN which consists of *8* convolutional layers and for each layer we use convolutional kernels to extract the features of each image channel. The first layer of the encoder can extract low-level features of the image, such as the edges in the image. The subsequent convolutional layers obtain higher level features such as the object parts, then the semantic features. We also need to resize the image to make it smaller, because an original image has many parameters so it is a complex way to get its features. Three shrinking layers are stacked in the middle of the convolutional layers. These three shrinking layers can also be regarded as performing linear combination within the learned features. The traditional way to shrink the image channel is pooling, which is a form of non-linear down-sampling. It divides an input image into a set of non-overlapping rectanglesand for each such sub-region, outputs the maximum. However, this way may cause the loss of important information. Instead of using pooling, we use stride to control the filter convolved across each part of the image, it can learn the downsampling process by itself, it will know the most important feature of each part of the image. It would also reduce the number of layers, so we can get the most useful and sharp features of the image as well as shrink the spatial size. The stride reduces the parameter of the image by shrink the image size, thus the encoder can get the feature vectors faster. For example, when the image passes each convolutional layer, the size of the image can be from $256*256$ to be $128*128$, and then to be $64*64$. Finally, the image becomes small with many extracted features, which can keep more useful information, and that means we get the feature vectors in the encoder parts.

Step C: Now we have feature vectors and we use decoder to transform those vectors into a new image. It is like a reverse process of encoder and has *8*deconvolutional layers as well. The deconvolutional block is an upsampling process. It is an algorithm-based process used to reverse the effects of convolution on recorded data. The concept of deconvolution is widely used in the techniques of signal processing and image processing.Eventually the output is an image with different style. That is how the traditional way to transfer the style of the image. However, the input goes through a series of down sampling steps to a lower dimension, then gets some non-linear transformation with a fully connected layer and finally gets up sampling to the present output size. Such a structure may cause information loss when passing through layers. We need to keep the edge as important part to keep the output high quality. So we use the "U-Net" to concatenate layers in encoder to the corresponding layers of decoder. This way can help us keep the edge and other important information of the image, the output would be clearer instead of blurry. The "U-Net" architecture is shown in *Fig. 3* and *Fig. 4* in detail.

Step D:We simultaneously train the generator and the discriminator, consequently the generator has the ability to produce image whose content is consistent with the input image (i.e. the original image whose style needs to be transferred), and at the same time the style is consistent with the target style which is provided by the style images in training set. The traditional way to improve the quality of image is that the generator adjusts its parameter based on the content loss and feature loss, and for every style transformation it adjusts the parameter layer by layer from the output one to the input, and it is a slow and inefficient way. We instead generate the image in a feed-forward way. Firstly, the model is trained by Generator-Discriminator to adjust its parameters to train the model and finally get realistic images, after the finish of training the generator of cGAN can generate image in real time without any adjustment, which is efficient. For the output image, the discriminator of cGAN can identify the loss of the image. The structure of the discriminator is shown in *Fig. 2.*In image synthesis with GAN, the generator attempts to produce a realistic image from an input random vector to fool the simultaneously adversarial trained discriminator, which needs to distinguish the authentic and fake pairs of images. It is like two-player game, and players improve their ability through training, which means the ability of generator and discriminator are both improved through the training. The goal of the training is the synthesized image generated by G which is indistinguishable from actual distribution by the discriminator D. The process of this part can be illustrated in *Fig. 5*. The objective (loss) function of general cGAN can be expressed by:

$$\min_G \max_D V(G,D) = E_{x,y\sim p_{data}(x,y)}[\log D(x,y)] + E_{x\sim p_{data}(x),z\sim p_{data}(z)}[\log(1-D(x,G(x,z)))]$$

where the generator G tries to minimize the objective function while an adversarial D tries to maximize it. To understand the difference between the fake and real images better, we defined a loss function which can be expressed by:

$$L = \lambda_p L_p + \lambda_a L_{GAN}$$

Where L_p is pix loss which refers to thestandard deviation of the pixel between the generated image and the authentic image. L_{GAN} is adversarial loss which is from discriminator. λ_p and λ_a are pre-defined weights for perceptual loss and adversarial loss. If we set both λ_p and λ_a to be *0*, then the network reduces to a normal CNN configuration, which aims to minimize only the Euclidean loss between output image and ground truth. If λ_p is set to *0*, then the network reduces to a normal GAN. If λ_a set to *0*, then the network reduces to the structure. Given an image pair $\{x, y_b\}$ with C channels, width W and height H (i.e. $C \times W \times H$), where x is the input, image and y_b is the corresponding ground truth. The L_p can be expressed by:

$$L_p(G) = \frac{1}{C\cdot W\cdot H}\sum_{c=1}^{C}\sum_{x=1}^{W}\sum_{y=1}^{H}\left\|\phi_E(x^{c,w,h}) - (y_b^{c,w,h})\right\|_2^2,$$

Given a set of *N* input images (ground truth or de-rained result) $\{y_i\}_{i=1}^{N}$ with the corresponding labels l_i (*1* represents “real”, *0* represents “fake”), the binary cross-entropy loss for the discriminator *D* is defined as:

$$L_A = -\frac{1}{N}\sum_{i=1}^{N}(l_i \log(D(y_i) - (1-l_i)\log(1-D(y_i))))$$

L can express the difference between the fake and real image, so it can help *G* and *D* train better. After the training process, the generator can produce almost authentic images with transferred style.

Step E:We have got trained parameters from the training model, which can help the model generate transferred images in real time. We create a model used to test and restore the parameters, so the model can generate images directly. We use content image as input and output per image with different style less than *1s*, and that is much faster than the traditional way which needs *30*s to generate an image.

Ultimately, we write ahypertext mark-up language(*html*) to show our results. Part of our results are shown in *Fig. 6.*

There are several key advantages in this invention:

By using the cGAN, we obtain substantial diversity which extract the feature of the input picture more explicitly. And the test results prove that these networks have a better performance than the previous related works particularly in dealing with the high-level features.

Besides, for each style transfer, we let our our model learn the implementation way in advance, which means each time we get an input, we can have our results in real time. Conversely, by using a traditional way, a massive calculation at every time is unavoidable even for transferring the same style.

What's more, the "U-Net" architecture avoids the loss of information in every step of the encoder and decoder. We use a "U-Net" architecture to splice the two feature map together in order to offer more information during the deconvolution; thus we have a good result especially for the semantic features in a high-level.

2017101166 25 Aug 2017

CLAIMS

1. This method for real-time image style transfer isbased onconditional Generative Adversarial Networks(cGAN), which includes following steps: using a traditional way to generate training datasets to get input and target pairs of images; puttingimages into the encoder of cGAN to extract the features and decoding them by using a “U-Net” architecture which is an advanced convolutional network; optimizing the values of generator and discriminator simultaneously by using a gradient descent approach based on the established loss functions and ultimately getting our results; a PatchGAN structure is also used in this model for evaluation.

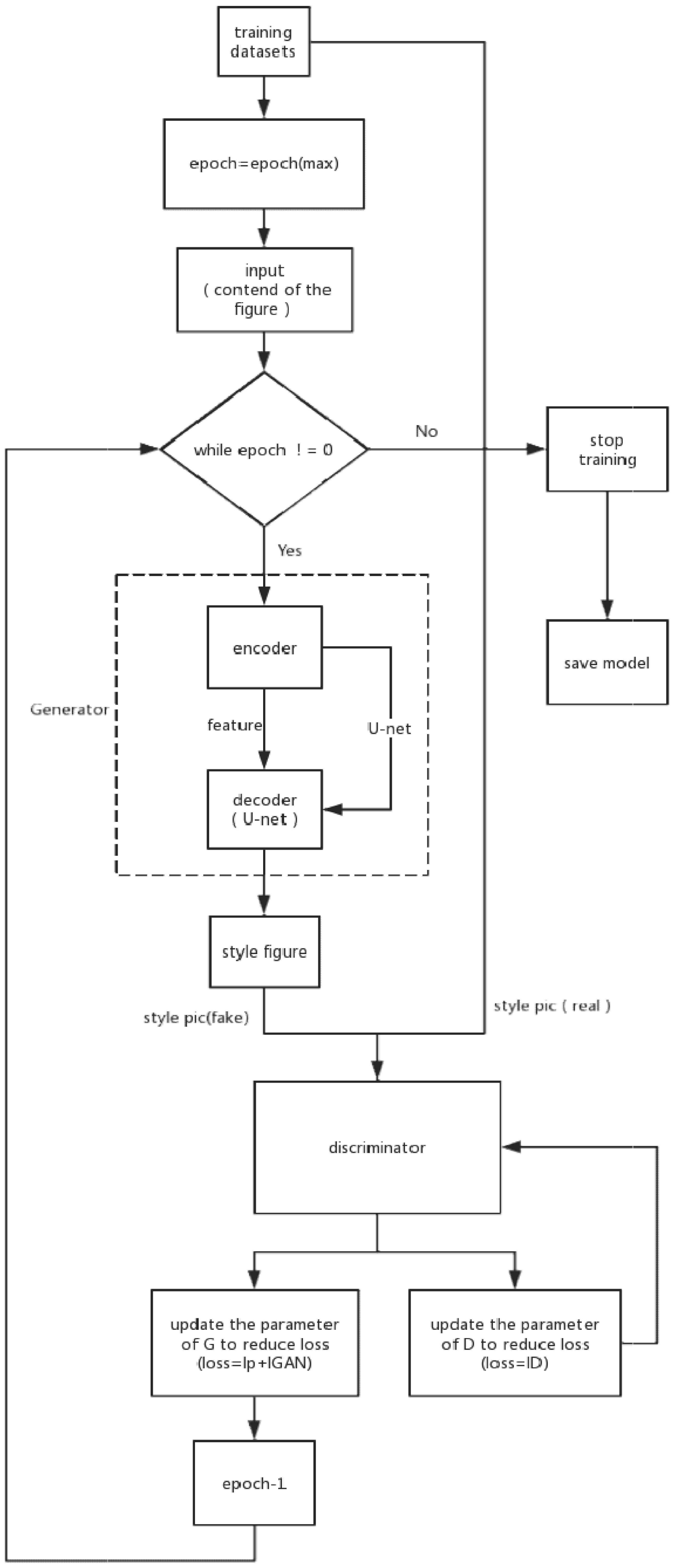
training datasets
epoch=epoch(max)
input (contend of the figure)
while epoch ! = 0
No
stop training
save model
Yes
Generator
encoder
feature
U-net
decoder (U-net)
style figure
style pic(fake)
style pic (real)
discriminator
update the parameter of G to reduce loss (loss=lp+lGAN)
update the parameter of D to reduce loss (loss=lD)
epoch-1

Fig. 1.

2017101166 25 Aug 2017

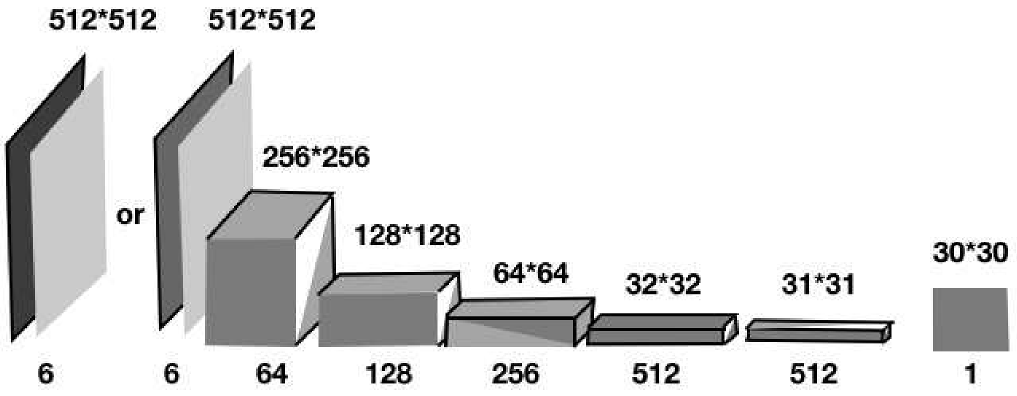

Fig. 2.

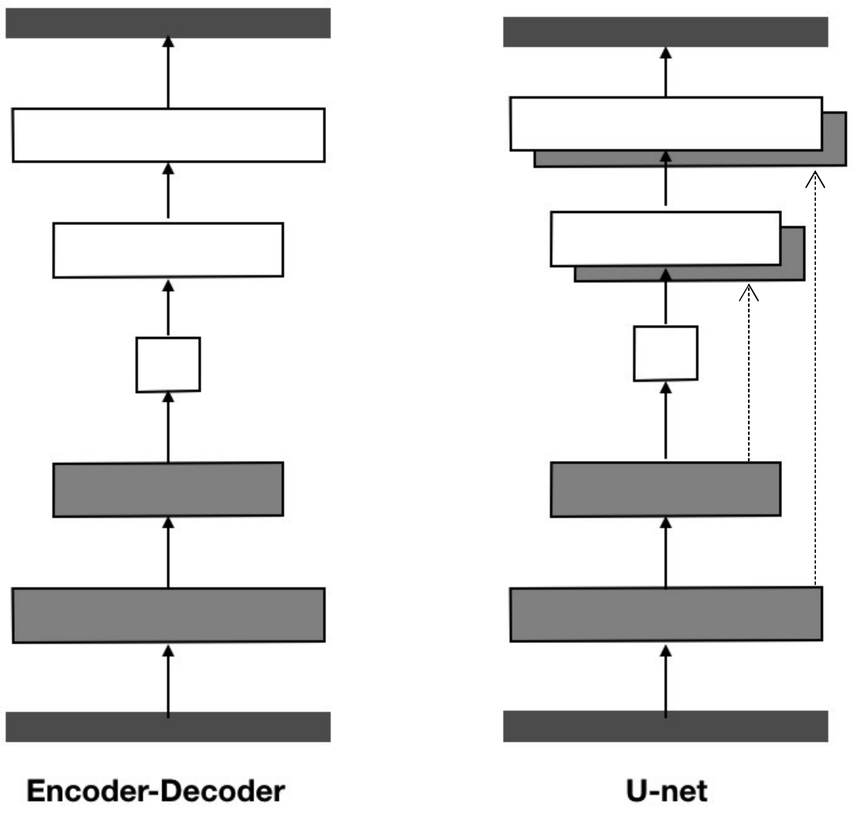

Fig. 3.

20171101166 25 Aug 2017

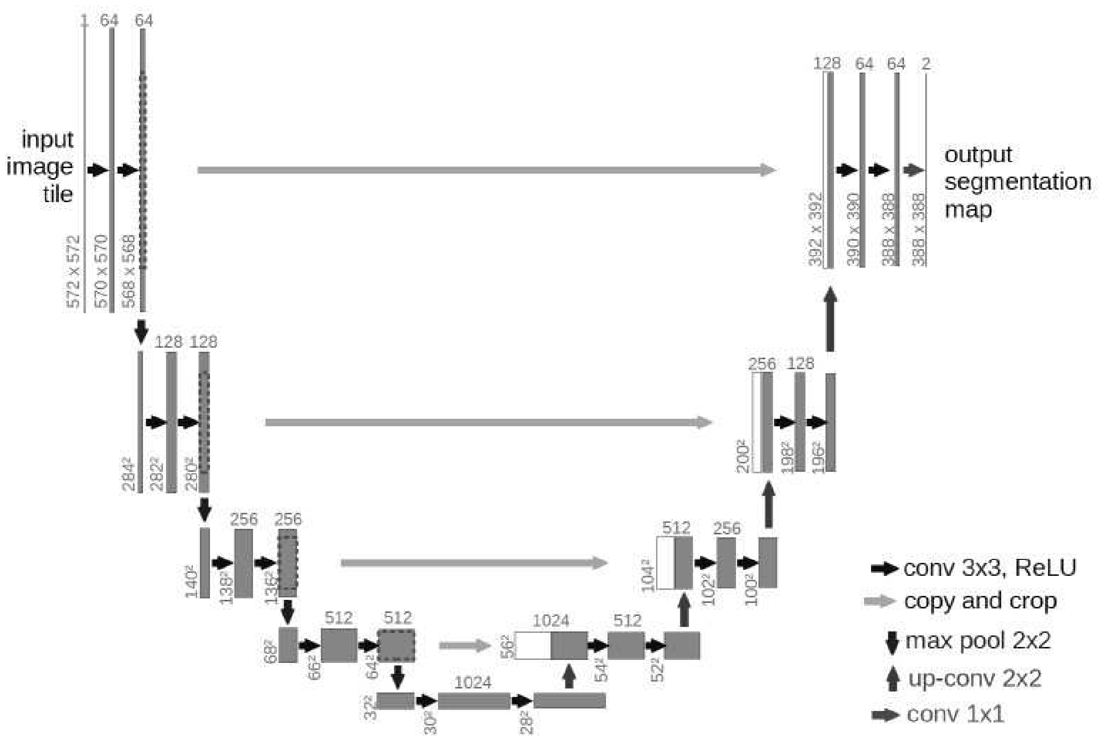

Fig. 4.

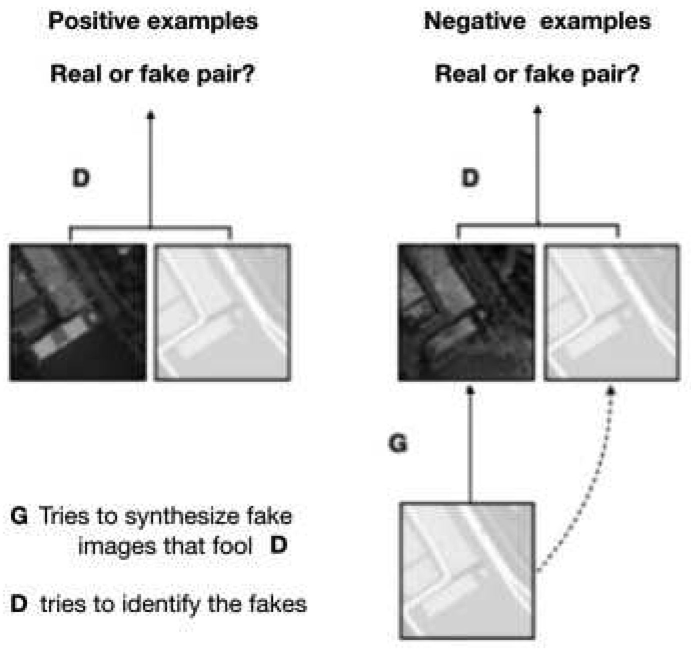
Positive examples
Real or fake pair?
D
Negative examples
Real or fake pair?
D
G
G Tries to synthesize fake images that fool D
D tries to identify the fakes

Fig. 5.

2017101166 25 Aug 2017

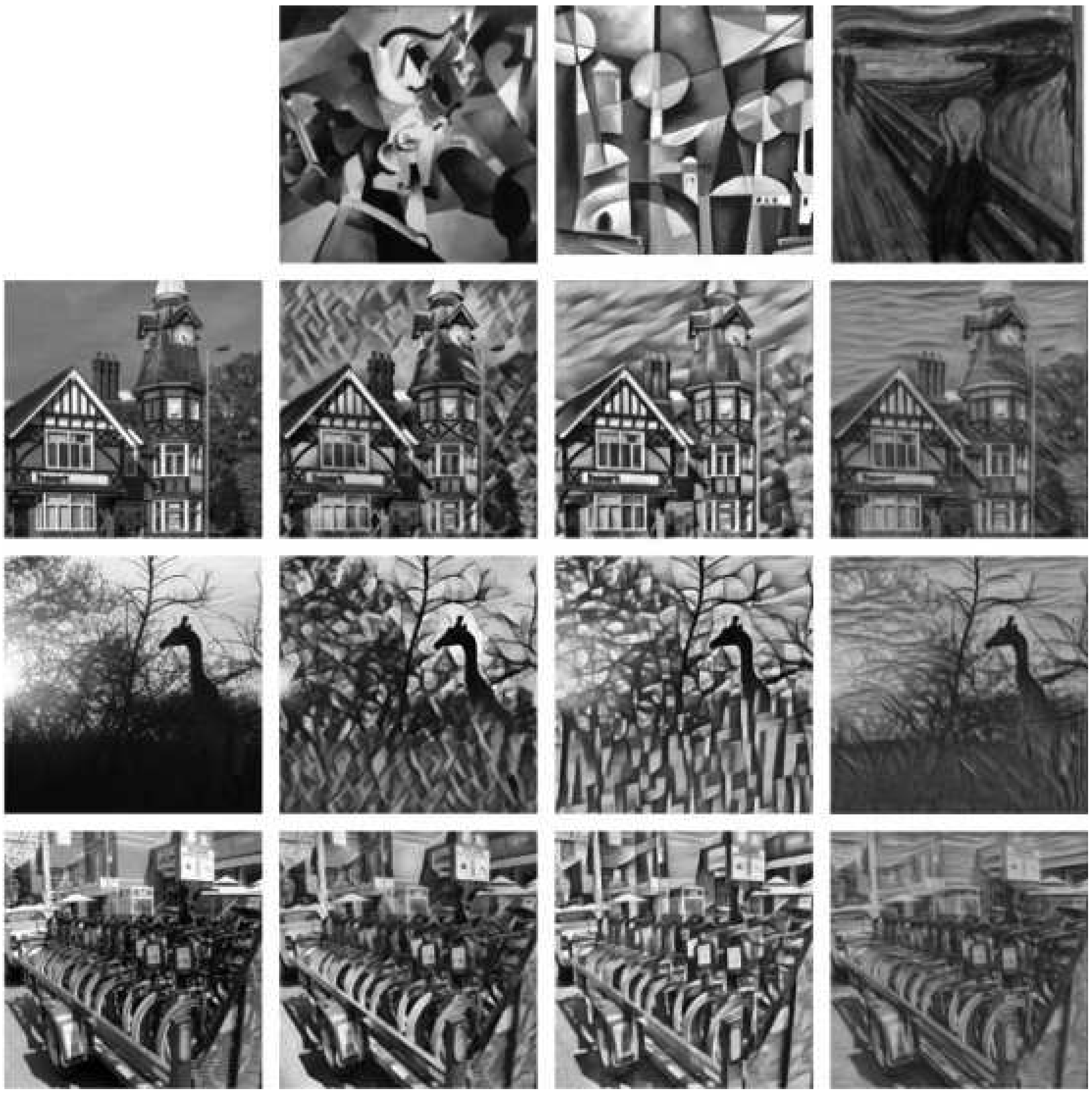
Fig. 6.