Journal of New Media
DOI:10.32604/jnm.2020.010062
Article

Edge Detection Based on Generative Adversarial Networks

Xiaoyan Chen, Jiahuan Chen* and Zhongcheng Sha

Nanjing University of Information Science and Technology, Nanjing, 210044, China
*Corresponding Author: Jiahuan Chen. Email: 20171308065@nuist.edu.cn
Received: 10 February 2020; Accepted: 21 March 2020

Abstract: Aiming at the problem that the detection effect of traditional edge detection algorithm is not good, and the problem that the existing edge detection algorithm based on convolution network cannot solve the thick edge problem from the model itself, this paper proposes a new edge detection method based on the generative adversarial network. The confrontation network consists of generator network and discriminator network, generator network is composed of U-net network and discriminator network is composed of five-layer convolution network. In this paper, we use BSDS500 training data set to train the model. Finally, several images are randomly selected from BSDS500 test set to compare with the results of traditional edge detection algorithm and HED algorithm. The results of BSDS500 benchmark test show that the ODS and OIS indices of the proposed method are 0.779 and 0.782 respectively, which are much higher than those of traditional edge detection algorithms, and the indices of HED algorithm using non-maximum suppression are similar.

Keywords: Edge detection; generative adversarial network; computer vision; image processing

1 Introduction

The edge of the image is the boundary that distinguishes the content of the image, and is also one of the important structural information of the image. The edge information can be extracted by edge detection while retaining the important structural properties of the image. Accurate edge detection is the basis for image processing tasks such as salient object detection [1], image segmentation [2], object detection and recognition [3], tracking and motion analysis [4], medical imaging [5], action structure [6] and 3D reconstruction [7].

Classic edge detection extraction algorithms include Canny [8], Sobel [9], Scharr [10], Laplacian [11], Roberts [12] and DoG [13]. These algorithms for detecting image edges are based on image pixel gradient calculations and image edge information is detected and extracted by first-order or second-order gradient operations. However, in these classical algorithms, both the Sobel operator and the Canny operator and the Scharr operator exist in the existence of the problem of artificially adjusting the threshold. The adjustment effect of the threshold is directly related to the final edge detection result, which has no universal applicability and brings additional cumbersome threshold adjustment operation to the final detection. In addition, the calculation limited to the pixel gradient also causes such algorithms to fail to detect the edges of the image, the final effect is much different from the ideal effect.

Subsequently, researchers also studied the method of edge detection based on image features extracted manually. For example, the method of extracting image edges based on feature classification of edge brightness, color and texture by DR Martin et al. [14] and P. Arbelaez et al. [15] by combining multiple local clue features by detector to extract image edges Method. Although the effect of this algorithm is improved compared to traditional edge detection operators. However, due to the need to make a large number of image features for the data set in advance for subsequent image edge extraction, the use of such methods is too cumbersome, which limits its further development.

In recent years, with the rapid development of convolutional networks [16] in the field of image processing, and the outstanding achievements of this method in the field of computer vision. For example, the LeNet network model for reading and verifying numbers in postal codes by Y. LeCun et al. [17], and the outstanding performance of AlexNet [18], VGGNet [19], GoogleNet [20] in the field of image classification and recognition, ResNet [21], etc. These all show the excellent ability of neural network in image feature extraction. The edge detection algorithm based on deep convolution network has gradually become the mainstream.

In the application of convolutional networks for edge detection, G. Bertasius et al. [22] proposed a detection model DeepEdge consisting of a fixed pre-trained five-layer convolutional layer stacking a bifurcated sub-network; J. J. Hwang et al. [23] proposed a detection model DeepCoutour consisting of four fully convolved layers stacked with four layers of convolutional layers; Y. Ganin et al. [24] proposed a detection model N 4 field based on a combination of convolutional neural networks and nearest neighbor search; H. Chen et al. [25] proposed a stack of three layers of convolutional layers consisting of five layers of convolutional layers. The full convolutional network model DCAN; Y. Liu et al. [26] proposed a model RCF modified by the VGG network layer and fused to multiple stages of edge images and S. Xie et al. [27] Scale and multi-level feature fusion for edge detection model HED. Compared with traditional edge detection algorithms and edge detection based on manual features, this method is more effective. It avoids the problem that the traditional edge detection algorithm performs edge extraction on the stepwise gradient, and does not need to perform threshold-like adjustment when detecting pictures. At the same time, the operation is relatively simple, and it is not necessary to make features in advance, and has good real-time performance. However, the edges of such algorithms are often thicker and require non-maximum suppression to achieve better results. The model itself does not avoid this problem well.

Since Ian Goodfellow proposed Generative Adversarial Networks [28], a probabilistic model derived from zero-sum game ideas in 2014, GAN has been widely used in computer vision, for example, image style conversion [29], super-resolution reconstruction [30], image generation [31], text-to-image [32], etc. These all demonstrate GAN's excellent ability in image generation and provide another way to handle edge detection problems.

In order to solve the problem that the traditional edge detection algorithm is not good, and the thick edge problem of edge detection algorithm based on convolutional network, this paper proposes an edge detection algorithm based on generator-based anti-neural network. The generator network of the confrontation network is composed of a contraction path composed of an 8-layer convolution network and an extension path composed of an 8-layer deconvolution network, and the discriminator network is composed of a five-layer convolution network. Both the generator network and the discriminator network use the Adam optimizer to train parameters while using a moving average to eliminate occasional fluctuations during each training session [33]. The model was trained using the BSDS500 dataset and the BSDS500 benchmark was used to evaluate the model results.

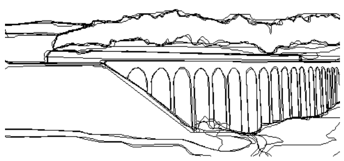
2 Introduction of Edge Detection and generated antagonistic Neural Network

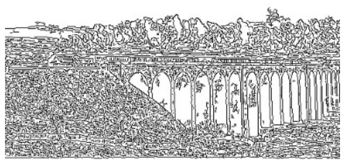
2.1 Edge Detection

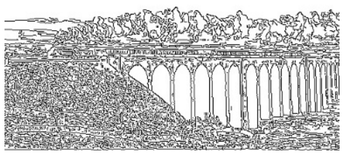
2.1.1 Traditional Edge Detection Algorithm

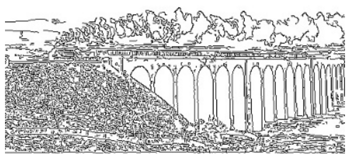
The traditional edge detection mainly realizes the detection of the image edge by calculating the gradient value of the image pixel. Taking Canny detection algorithm as an example, the algorithm flow is as follows: (1) First of all, it is necessary to filter and reduce the noise of the gray image of the input image. (2) The next step is to calculate the gradient strength and direction of each pixel in the graph. (3) Then, in order to eliminate the previous step of multiple gradient responses to the same edge, it is necessary to apply non-maximum suppression. (4) Then the double threshold algorithm is used to detect the real and potential edges. (5) Finally, suppress the isolated weak edge. By studying the whole process of Canny, we can find that the algorithm has inevitable limitations in principle. Edge detection is not performed from the whole because the edge detection is limited to the pixel gradient value. The results of Canny edge detection are shown in Fig. 1.

(a) Artwork master


(b) True value


(c) The detection result of the threshold is 0 ≤ 50


(d) The detection result of the threshold is 50 ≤ 125


(e) The detection result of the threshold is 100 ≤ 175

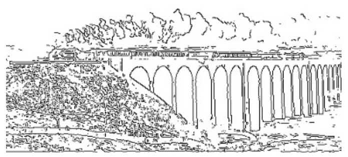

(f) The detection result of the threshold is 225 ≤ 255

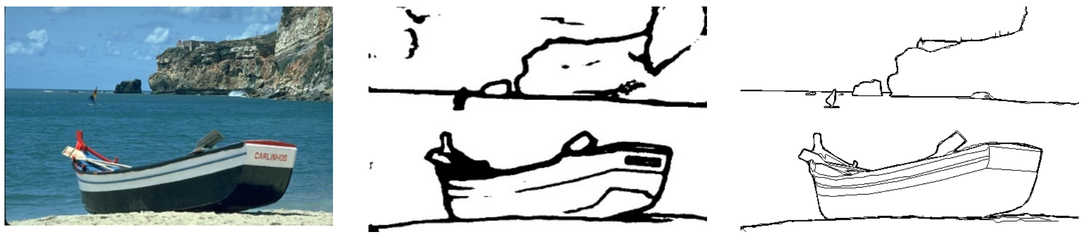
Figure 1: Canny edge detection results

We can see that the effect is not ideal, as a large number of image pixel differences are mistakenly detected as edges. Although if you carefully adjust the threshold and find the threshold that best suits the image, you may have better results. This brings tedious operation to this kind of method when edge detection is carried out, and the final effect is still far from the ideal result.

2.1.2 Summary of Edge Detection Based on Convolution Neural Network

In view of the limitations of traditional edge detection methods and the excellent performance of convolution network in image processing, researchers began to explore the use of deep neural networks to solve the problem of edge detection. In these studies, The global nesting edge detection method based on convolution neural network proposed by Xie et al. [27], the detection effect is much better than the traditional edge detection algorithm. The model initializes the network structure and parameters in the model by using the pre-trained VGGNet. The entire network can have multiple independent networks to detect the edge of the image across multiple scales, and finally, the images are fused to achieve the best results.

As a result of the HED edge detection algorithm, as shown in Fig. 2, we can see that the HED can better perform the task of detecting the edge of the image, the error is small, and the whole image can be detected without being limited to the pixel difference as in the conventional edge detection algorithm. Finally, compared with the results of Canny algorithm, while retaining the overall structural attributes of the image as much as possible, the false detection of edges is also better than that of Canny algorithm. However, if non-maximum suppression is not used in the end, the edge of the detection result will be thicker, and some of them will not be ideal. At the same time, the model modified by VGGNet requires too much memory during training and takes too long, and it takes a long time to test each picture.



(a) Artwork master (b) Result (c) True value

Figure 2: HED edge detection results

2.2 Generating Antagonistic Neural Network

2.2.1 Summary of Generative Antagonistic Neural Network

The inverse neural network of the generator type [28] is a generator model based on game theory. The model consists of two neural networks. These two networks are also referred to as generator networks and discriminator networks, depending on the functionality. The generator network maps the input data to the probability space of the real sample to generate the required data, and the discriminator network is a two-class network to classify and discriminate between real data and generator-generated data. Through this kind of opposite relationship, the model can theoretically approximate the probability space of any real sample distribution, so that it can be generated. Data for any results.

The whole network [34] is always in an antagonistic state during the training process, and the generator attempts to generate false data from the network that can deceive the discriminator, while the discriminator network attempts to distinguish between the false data generated by the generator and the real data. The objective function of this antagonistic relationship is shown in formula (1). At each training time, one of the neural networks is fixed, and then the other is trained. It is important to note that the purpose of the two training sessions is different. When training the discriminator, we need to make the discriminator have better ability to judge the true and false data, and when training the generator, we need to make the generator generate the data that is better used to deceive the discriminator. The graphical representation of such a model structure, as shown in Fig. 3, is also a further description of such a countermeasure relationship.

$$\underset{G}{min}\,\underset{D}{max}V(D,G) = Ex{\sim}Pdata(x)[log\,D\,(x)] + Ex{\sim}Pz(z)[log(\,1 - D(G(z)))] \tag{1}$$

In this case, D(x) means the test result which the discriminator gives to the true value. D(G(z)) means the test result which the discriminator gives to the generator's output value. These two results are used to calculate the loss function of the discriminator and the generator, so that the final result can be improved step by step. The model structure diagram is shown in Fig. 3.

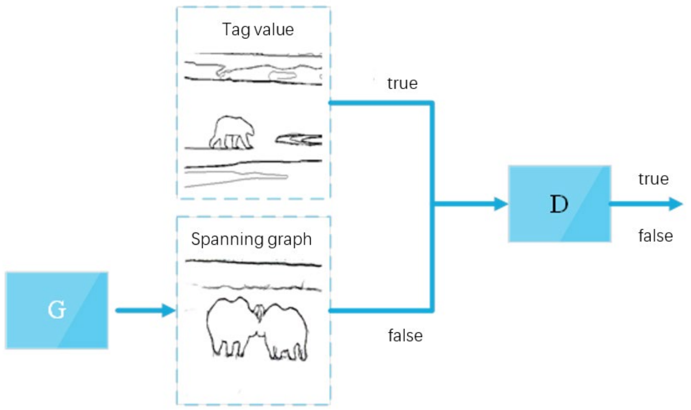


Figure 3: GAN network structure

But due to the lack of the limit of the final result, the result of this model may not be particularly precise. Thus, it will not work out very well as expected. Therefore, a variation type is taken in this paper, namely conditionally generated neural network [35]. This network model also uses this mechanism, while a new condition variable, y, is introduced in the model of both generator network and discriminator network. Meanwhile, by additional message, y brings additional constraint condition to the model, thus achieves the goal of guiding the generation of data, generating the final result. In the model the paper puts forward, the condition variable, y, is the original input image matrix that requires detection, and the objective function of this generative adversarial network is shown in formula (2). This model is an expansion of the original generative adversarial network. Both the generator and the discriminator have additional message, y, comparing to the original condition. Y can be any message like category information or other modal data, while the objective function of condition GAN is the adversarial result of the maximum and minimum value with the adding message.

$$\min_{G}\max_{D} V(D,G) = Ex{\sim}Pdata(x)[log\,D\,(x|y)] + Ex{\sim}Pz(z)[log(\,1 - D(G(z|y)))] \tag{2}$$

In this case, D(x|y) means the test result which the discriminator gives to the true value, D(G(z|y)) means the test result which the discriminator gives to the generator's output value. These two results are used to calculate the loss function of the discriminator and the generator, so that the final result can be improved step by step. The model structure diagram is shown in Fig. 4.

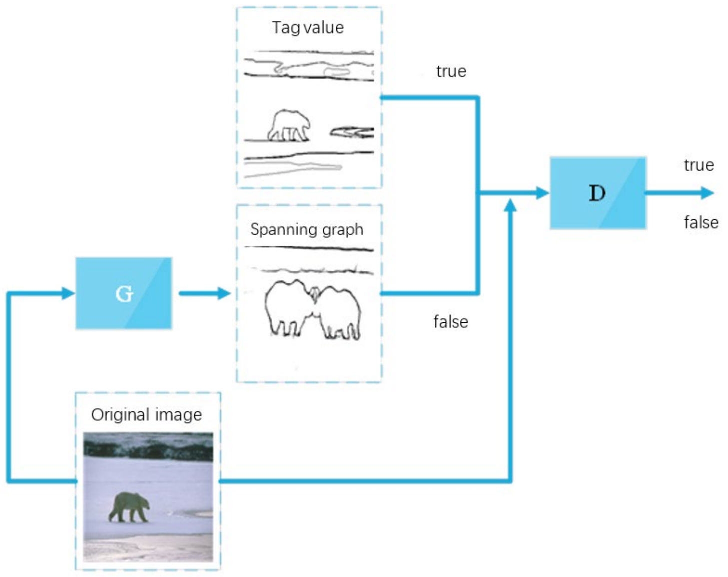


Figure 4: CGAN network structure

2.2.2 An Overview of the Network Layer in Generator Network and Discriminator Network

GAN has two neural networks to accomplish the process of the adversarial generation and fit the objective function effectively, so that the final result can be presented as expected. In creation of the generator, the network layers mentioned in this paper are Convolution layer, Deconvolution layer, Batch normalization layer and Dropout layer. Some activation function like Tanh function, Relu function and LRelu function are also used in them. In the creation of the discriminator, Convolution layer, LRelu activation function and Sigmoid function which is used ultimately to compress the result to between 0-1 are used likewise, in order to discriminate the authenticity of the result.

(1) Convolution layer [36]

The aim of Convolution layer is feature extraction. Continuous convolution operation is used in the creation of DNN (deep neural network) to obtain a feature map of higher latitude. The convolution process of Convolution layer is shown in Fig. 5. It is a picture in respect of one-layer 2D plane graph but in fact, it is far more than one layer. For example, there are three channels in color map. Therefore, 3D Convolution kernels is used to the convolution operation. In order to extract the required characteristics in each operation, there must be some loss to the result of the picture. To reduce the loss to the minimum, more Convolution kernels are used in the system, obtaining pictures of more layers. Ultimately, the it can lead to smaller size, more precise feature yet more layers. Similar to the hierarchical perception of human vision, the bottom layer only processes the simple linear message. However, the higher it goes, the more structure attributes it can present by feature extraction.

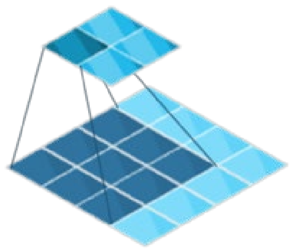


Figure 5: Convolution operation

(2) Deconvolution [37]

Deconvolution is an inverse process of convolution operation. The principle is shown in Fig. 6. The features generated after inputting the convolution and the data generated before outputting the convolution plays a role of restoring. By transposing convolution matrix, matrix of 2*2 can be mapping to matrix of 4*4. The first layer is taken for an example in this paper. When operating, deconvolution kernel is seldom used in one layer. That's because the result of the upper layer contains messages of different layers. Provided that only one layer was used to the model, the relation information between layers and layers would have been lost. Generally speaking, multilayer deconvolution kernel is used to deconvolution the result of smaller size of multilayers after convolution. On account of this, a better effect of feature extraction can be accomplished.

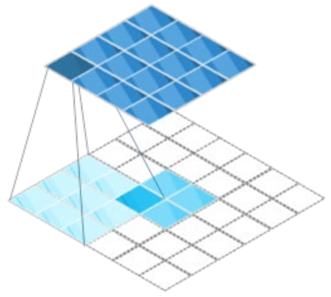


Figure 6: Deconvolution operation

(3) Batch Normalization [38]

Batch normalization means proceeding normalization before inputting data to any layer of the network. The formula of Batch normalization is shown in formula 3.

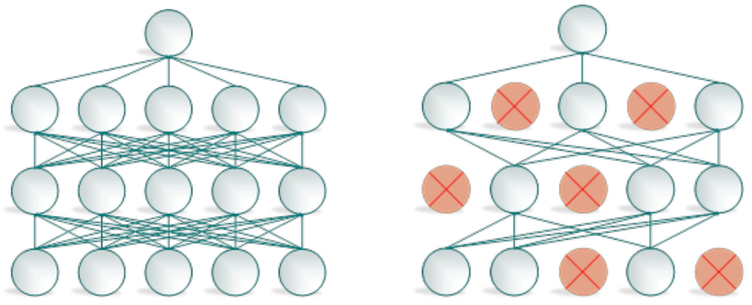
$$\hat{x}^{(k)} \leftarrow \frac{x_i^{(k)} - \mu_\beta^{(k)}}{\sqrt{{\sigma_\beta^{(k)}}^2 + \epsilon}} \tag{3}$$

In this case, $x_i^{(k)}$ represents for the input of the k-layer, μ_β represents for the means for the input of the k-layer, σ_β represents for the standard deviation for the input of the k-layer, ε represents for the additional value preventing the divisor from becoming zero (Generally, ε is assigned the value 0).

Independent identically distributed hypothesis is an essential hypothesis in Machine Learning. It is a basic guarantee to a relatively accurate result after training the test data sets. Nevertheless, Batch normalization is what keeps every layer of neural network's input in the same distribution. By operating Batch normalization, not only does the training speed improve while the convergence process is accelerated dramatically, but also makes the result of category more accurate by using Batch normalization to prevent overfitting regularization of expression. In addition, the process of parameter adjustment is simplified a lot. Not setting to much limit to initialization, the learning rate is higher too.

(4) Dropout layer [39]

This layer aims at solving the overfitting problem in the process of neural network training. Overfitting means neural network can only exert relatively good results in the test data sets while through testing the target data set it performs poorly. In respect of this problem, Hilton put forward a concept of Dropout layer [39]. The major principle is shown in the Fig. 7. A common fully connected structure is shown in first figure (a) in Fig. 7 while a fully connected structure with Dropout layer is shown in the second figure (b) in Fig. 7. When training in the Dropout layer, some random nodes in the network will not participate in the calculation (setting the output to value 0) while other processes remain the same. On account of this, it can be prevented that some features can only take effect on a certain condition. Thus, the network can learn some commonalities instead of a certain feature, preventing the overfitting problem efficiently. Using Dropout layer can also train multiple models simultaneously and using the means of models as the output improves the qualification of the result.



(a) Network without Dropout layer (b) Network with Dropout layer

Figure 7: Principle of Dropout layer

(5) Activation function

Activation function is used to nonlinear the output, making the result closer to the target after multilayer nonlinear mapping. The typical activation functions are Sigmoid function, Tanh function [40], Relu function [41], LRelu function [42] and so on. The formula of each function is shown in Tab. 1.

Table 1: Activation function

Function Name	Formula
Sigmoid function	$\mathrm{f(z)} = \frac{1}{1+\mathrm{e}^{-\mathrm{z}}}$
Tanh function	$tanh(x) = \frac{e^{X} - e^{-x}}{e^{x} + e^{-x}}$
Relu function	$Relu = max(0, x)$
LReLu function	$L\,Re\,L\,u(xi) = \begin{cases} xi & if\ xi > 0 \\ aixi & if\ xi < 0 \end{cases}$

Through these activation function, the input and output of each layer can be nonlinear, making the deep neural network expression more powerful. Provided that there was no activation function, both the input and the output would have been linear which has much limit to the approximation capability of the network regardless of the layers of neural network, according to the researches of neural network nowadays. The model of the convolution layer is shown in Fig. 8. Every node in the neural network takes the output value, X_{11}-X_{nn}, as the input of the current neuron. Then, the input multiples the Convolution kernel parameter, W_{11}-W_{nn}, summing all the current results, and calculates with activation function F(x) to obtain the output H_{11}. The output is given to the next layer later. The model of convolution network can finally map the input to the target result, making the complicated network approach the target function through calculation from layer to layer.

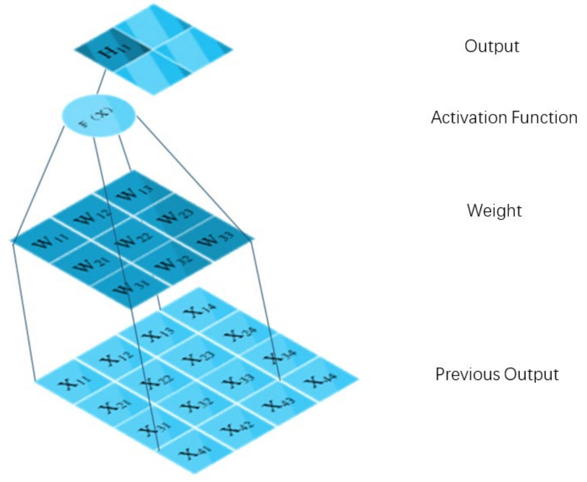


Figure 8: Neuron model of Convolution layer

Among these functions, Function Sigmoid was proposed earlier, and is used as an activation function in LeNet. The function has a great advantage in the two-class problem, it will obtain the results by compressing the input from the previous layer between 0 and 1. However, if this function is used between network layers, the gradient disappearance problem will occur in the training optimization process of the deep neural network, and the final training fails. For this reason, it is not generally to use Function Sigmoid as an activation function, but to select it to handle the two-class problem at the end.

Function Tanh proposed next also cannot avoid the gradient disappearance in the back propagation of deep neural networks, because it also compresses the value into the space of -1~1, which eventually leads to the model not being able to converge, training failed and did not achieve an expected result. But the advantage of Function Tanh is that it Symmetrically centered at 0, so it will be better than Function Sigmoid in actual use and is also generally used as a final classification.

The proposal of Function Relu solve the problem of gradient disappearance in the back propagation of deep neural networks, which cannot be avoided by Function Sigmoid and Function Tanh. This makes the deep neural network converge during the training, the gradient does not disappear within the positive range, and has an effect of optimizing parameter in the back propagation, leading the model to have an ideal output. Though Function Relu is very simple, it play an important role in neural network training. It is generally to use Function Relu as a default activation function. This function has many advantages: (1) solving the problem of gradient disappearance (within the positive range); (2) Greatly fast calculation, only need to judge whether the input is greater than 0; 3) faster convergence speed than Sigmoid and Tanh.

In addition to the Relu function, many researchers have proposed a large number of variants to meet other needs in the training, of which LReLu is one of them. Given that gradient disappears if the value of the Function Relu appears in the negative interval, Function LReLu choose a small fixed value as a multiplier instead of 0 in the negative interval. In this way, even if there is a negative value, gradient will still exist and continued to have an effect of optimizing parameter in the back propagation. Like Function Relu, Function LReLu is also used as an activation function.

3 Method of Edge Detection Based on Generative Adversarial Network

3.1 Model Instruction

In this paper, the adversarial model for edge detection is the generated adversarial network. The discriminator is trained with the input value of 6 × 256 × 256 three-dimensional data connected by the image and the tag value, so that the test result is true. Meanwhile image is input to the generator, and finally the output value of 3 × 256 × 256 comes into being. The generator connects this result with image generating the output value of 6 × 256 × 256, which trains the discriminator and make the test result false. In this dynamic process, not only the result of generator becomes more better, but also the discrimination ability of the discriminator stronger, ultimately leading to the goal of producing better results.

In the construction of the generator network model, this paper uses the U-Net architecture [43], which has strong ability of image detail extraction. The whole neural network consists of two parts: the shrink path and the extended path. Meanwhile, in order to prevent the loss of image details after the convolution and deconvolution multiple times during the process, the shrink path will input convolution result to the next layer and the corresponding layer of the extended path, ensuring the image detail reserved when extract image structure. The shrink path, consists of 8 convolution layers, is mainly to catch the context information in the image. By contrast, the extended path, consists of 8 deconvolution layers, is to precisely locate the parts of the image that need to be segmented. The model of this network is shown in Fig. 9 and Fig. 10. By the extraction of low-latitude features and the integration of high-dimensional features, model has a good testing effect of the edge of image.

In generative network, the convolution operation of the 8-layer contraction path uses a 4 × 4 convolution kernel with a sliding step length of 2 while filling the boundary. The first layer has 64 convolution kernels, followed by the second layer of 128, 256 kernels in the third layer, and the subsequent five layers of network layers each have 512 convolution kernels. The parameters of each convolution layer are initialized by a Gaussian distribution with a mean of 0 and a standard deviation of 0.02. Each layer after the first layer uses LReLu with a fixed parameter of 0.2 as the activation function for convolution operation mentioned before. In order to fasten the convergence speed of training, the next 7 layers all used batch standardization, remapping the input to a range that fits the normal distribution. The result of the first 7 layers in the shrink path is input to the next layer, and is also to the deconvolution layer of the extension path of the corresponding height. The deconvolution layer will connect the result of the previous deconvolution layer with the corresponding height convolution layer in the third dimension to obtaining the double data of the previous layer, and then carry on the operation of each layer. To be exact, each layer will firstly deal with the result of the previous layer by Activation Function Relu, then deconvolution and convolution kernels of each layer are 4 × 4, the sliding step is 1, and fill the boundary. The convolution kernel parameter is initialized with a Gaussian distribution with a mean of 0 and a standard deviation of 0.02. There are 512 convolution kernels in the first 3 layers , 512 kernels in the fifth layer, 256 kernels in the fourth layer, 128 in third layer,64 in the second, 3 in the first, at the same time, the Function Tanh is used to compress the range of result, and finally 3 × 256 × 256 data is output as the result of mapping the true distribution.

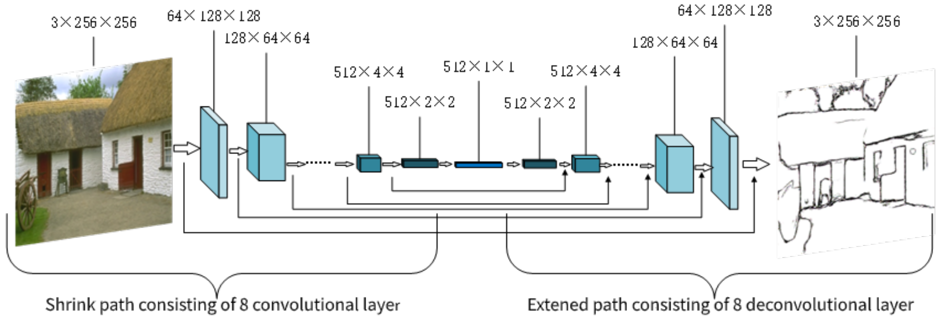

3×256×256
64×128×128
128×64×64
512×4×4
512×2×2
512×1×1
512×2×2
512×4×4
128×64×64
64×128×128
3×256×256
Shrink path consisting of 8 convolutional layer
Extened path consisting of 8 deconvolutional layer

Figure 9: Generator model

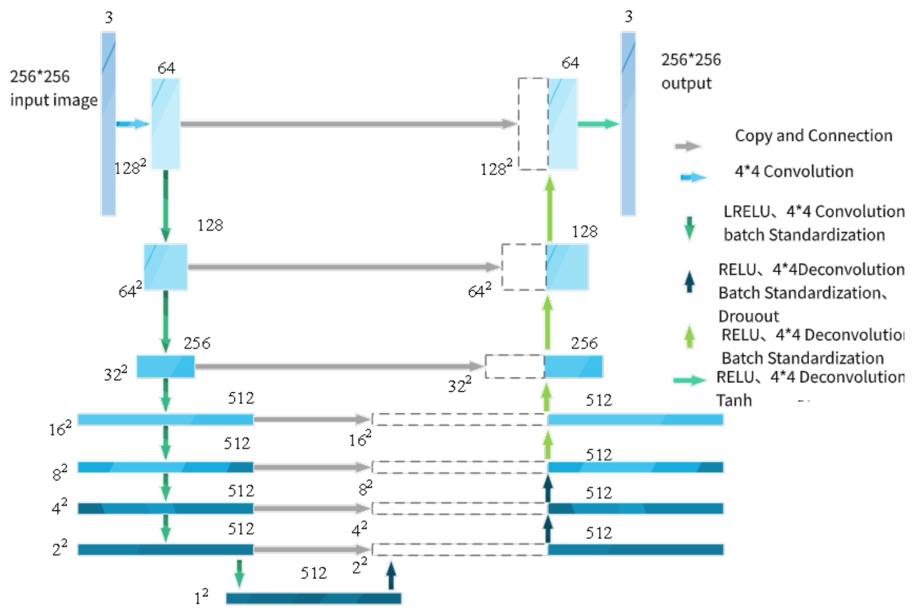

256*256 input image
3
64
128²
128
64²
256
32²
512
16²
8²
4²
2²
1²
256*256 output
Copy and Connection
4*4 Convolution
LRELU、4*4 Convolution batch Standardization
RELU、4*4Deconvolution Batch Standardization、Drouout
RELU、4*4 Deconvolutio Batch Standardization
RELU、4*4 Deconvolution Tanh

Figure 10: Generator network architecture model

The discriminator network used in the method is as shown in Fig. 11, and a convolution network of 5 layers is used to extract image features. To be exact, data of two inputting 3 × 256 × 256 image will join together to 6 × 256 × 256, then the first layer of convolution layer uses 64 4 × 4 convolution kernels, the sliding step size is 2, and the data boundary is filled at the same time. The second layer uses 128 4 × 4 convolution kernels, the third uses 256 convolution kernels, and the fourth uses 512 convolution kernels while the sliding step size is 1. The fifth layer uses 1 kernel, and the sliding step size is 1. The first to the fourth layer will all use the Function LReLu as the activation function, the second to the fourth layer will handle the input from the previous layer with batch standardization, and the fifth layer will use activation function Sigmoid for the convolution results. Finally, output 30 × 30 data is the detection result, which is used to calculate the loss value of the discriminator and the generator later.

Sigmoid activation function, 30 × 30 data is output as the test result in the end, which is used to calculate the loss value of the discriminator and generator later.

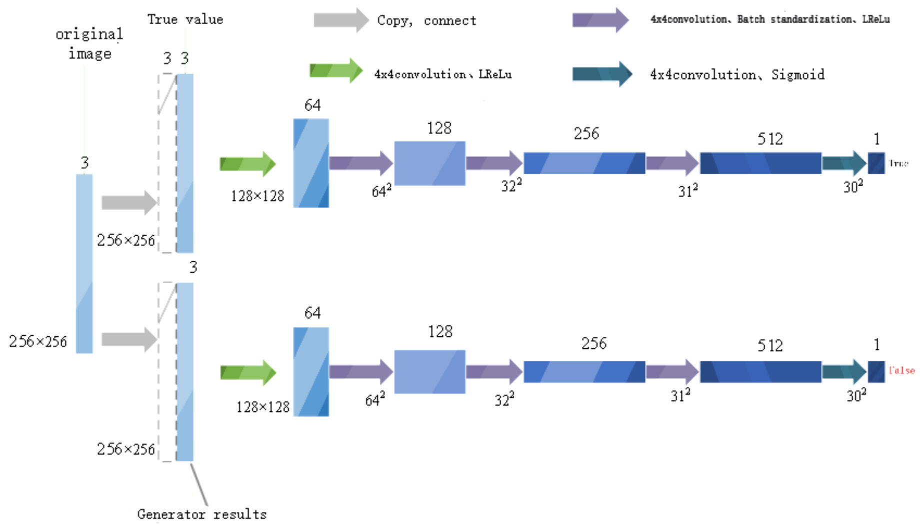


Figure 11: Network model of discriminator

3.2 Model Training

In the introduction of the model structure, it is mentioned that the model will output the predicted value of the real result $\mathrm{D}(x|y)$ and the predicted value of the generator result $\mathrm{D}(\mathrm{G}(z|y))$. In the model training process, in order to make the model get better effect in the confrontation gradually, the generator parameters will be optimized in the direction of maximization $\mathrm{D}(\mathrm{G}(\mathrm{z}|y))$, while the discriminator will maximize the real value and minimize the generated result of the generator in the process Direction optimization of. In the model of the method proposed in this paper, the loss functions of generator and discriminator are shown in formula (4) and formula (5).

$$L_D = \frac{1}{n}\sum_{i=1}^{n}(\text{-logD }(x|y)) + \frac{1}{n}\sum_{i=1}^{n}(\text{log(1-D }(G(z|y))) \tag{4}$$

$$L_G = \frac{1}{n}\sum_{i=1}^{n}(\text{-log(D }(G(z|y))) \tag{5}$$

In the training process, in order to prevent the generator over fitting from affecting the final generation effect, the mean value of the absolute value of the difference between the generated value and the tag value is used as the penalty term of the generator. The specific formula is shown in formula (6).

$$L_1 = \frac{1}{n}\sum_{i=1}^{n}(|G(z|y)\text{-x}|) \tag{6}$$

When optimizing the parameters, the Adam optimizer with learning rate of 0.002, b1 is 0.5 and b2 is the default value of 0.999 is used to optimize the parameters. This algorithm is a combination of the advantages of momentum learning algorithm and learning algorithm. It is simple to realize, and can automatically adjust the learning rate. The updating of parameters will not be affected by the scaling transformation of gradient. At the same time, the step length of each updating can be limited to a certain range, and finally better results can be obtained during training. The calculation formula is shown in (7):

$$m = b1 \times m + (1 - b1) \times dx$$

$$v = b2 \times v + (1 - b2) \times dx^2$$

$$W = W - Learning\ \text{rate}\times m/\sqrt{v} \tag{7}$$

Additionally, in the process of parameter optimization, in order to enhance the generalization ability of the model better, when optimizing the parameters of generator and discriminator, a sliding average method with a attenuation rate ($decay$) of 0.99 is also used [44]. In the process of model optimization, this method will record average values of all parameters w and b in the model for a period of time, and then assign values to variables, so as to make the update of parameters more smooth, not make the sliding average fluctuate greatly because of some abnormal values, and enhance the robustness in the training process. The attenuation rate (decay) determines the update speed of the sliding average, and the larger the value is, the more stable it is. The calculation formula of each sliding average is as shown in formula (8).

$$shadow\_variable = decay * shadow\_variable + (1 - decay) * variable \tag{8}$$

4 Analysis of the Experimental Results

The experimental environment of this paper is based on a PC with 16 g memory, NVIDIA gtx1060 as the graphics card, TensorFlow-GPU 1.12 as the application deep learning framework, and python 3.6 as the programming language.

4.1 Training Data and Test Data

The data set used in this experiment comes from bsds500 data set provided by Berkeley University. This data set is an extension of bsds300 data set, including 300 images for training and verification, and 200 images for testing. Finally, the performance is evaluated by benchmark tests such as accuracy and recall rate of detected boundary. When training, the image will be normalized first, and all of them will be scaled to a unified size of 256 × 256, so as to facilitate batch operation. After a period of training, after the model has a better result, then run the bsds500 benchmark to make a more detailed analysis and comparison of the model results.

4.2 Experimental Results

The output results of the model trained for about 2 hours are shown in Fig. 12 below. The left side is the image to be edge detected, the middle is the operation results of the experimental model, and the right side is the results of manual annotation. It can be seen that the result of the model is relatively ideal, which is basically similar to the annotation result.

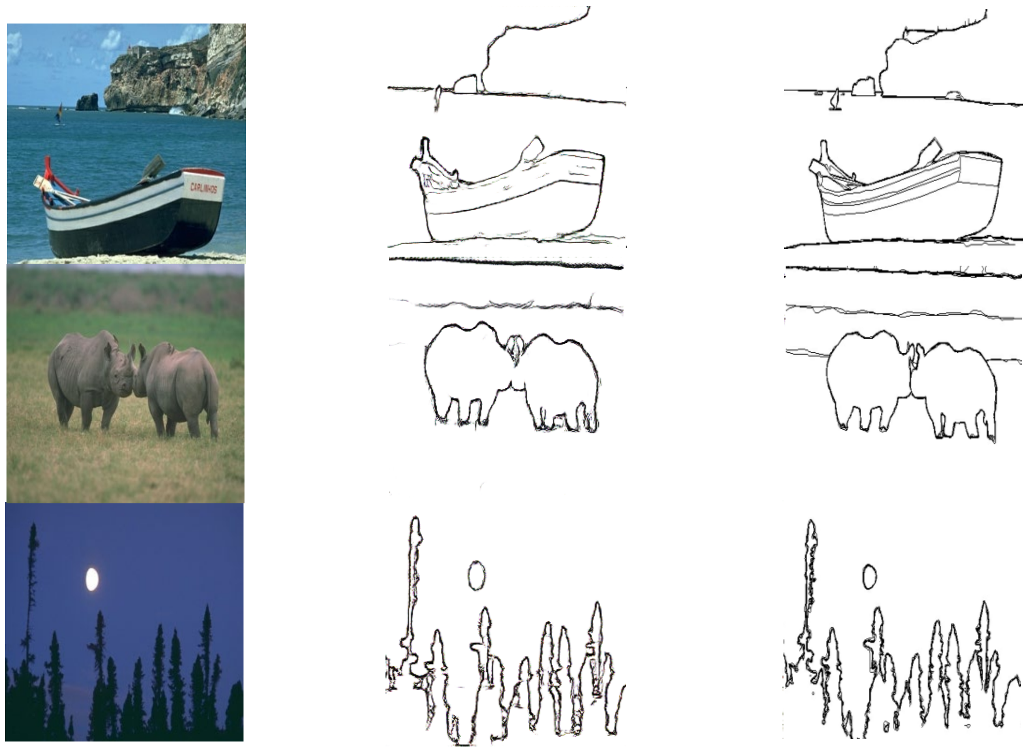
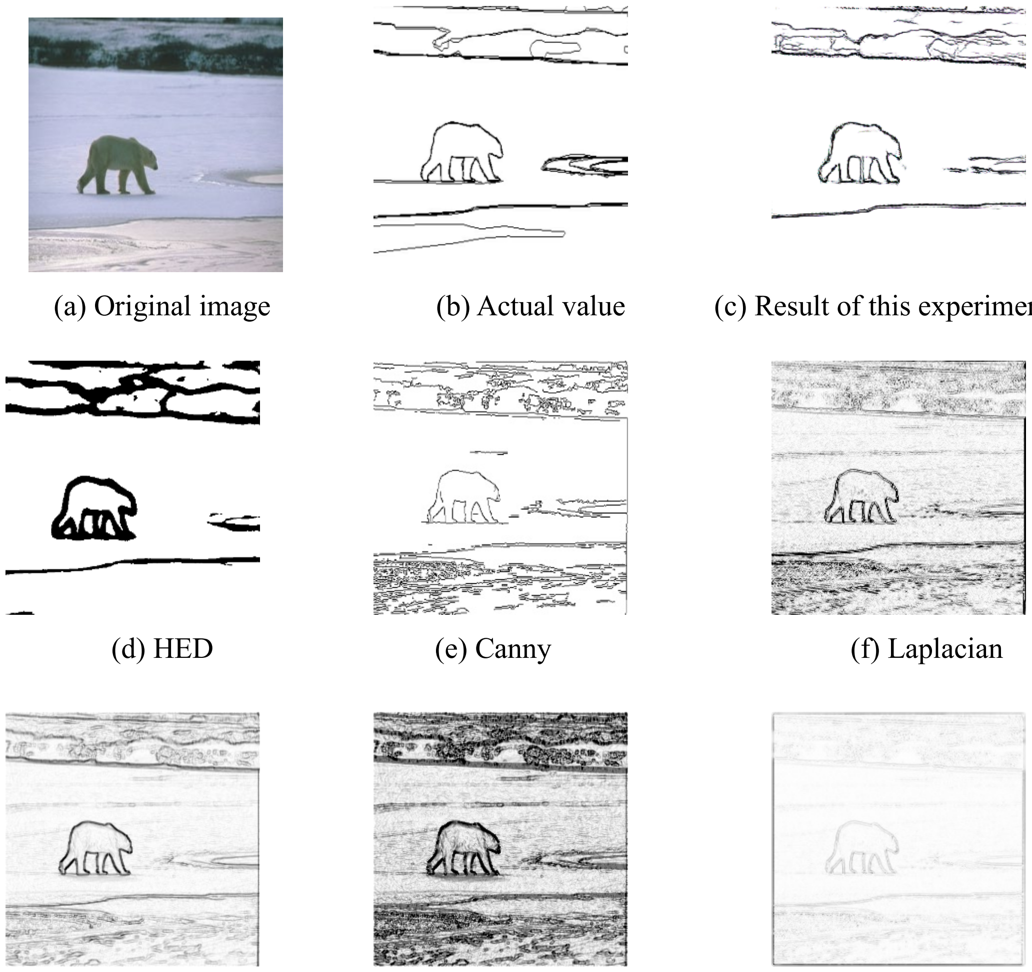


Figure 12: Results of the experiment in the test set

4.3 Comparison of Common Edge Detection Methods

Before deep learning is applied to edge detection, there are many traditional methods for edge detection at home and abroad. In this section, we will compare the effects of several main edge detection algorithms, such as Canny operator [8], Sobel edge detection algorithm [9], Scharr edge detection algorithm [10], Laplacian edge detection calculation method [11], DoG edge detection algorithm [13] and HED algorithm [27], which are the main algorithms. The comparison between the algorithm results of this experiment and that of this method is shown in Fig. 13. The traditional edge detection algorithm can detect all the differences in pixel details, but this also leads to a very high error rate of edge detection, resulting in the final effect is not ideal, although better results can be obtained if the threshold value is carefully adjusted. However, every image needs to adjust the threshold manually, which brings more tedious manual operation to edge detection and loses the significance of machine edge detection. The HED algorithm based on convolution network can detect the edge of the whole image well, and it will not produce a large number of false detection edges in the running process of traditional edge detection. The effect is good, but its edge is obviously thicker, and the result compared with the annotation is also not ideal. It needs to use non maximum suppression to achieve better results. The algorithm implemented in this paper solves this problem well. While it can accurately respond to the edge, there will be no back-edge problem caused by algorithm like HEDS. The result is ideal.



(a) Original image (b) Actual value (c) Result of this experiment

(d) HED (e) Canny (f) Laplacian

(g) Sobel (h) Scharr (i) DoG

Figure 13: The results of edge detection for the same image with the proposed algorithm and HED, Canny, Laplacian, Sobel, Scharr, DoG

4.4 The Evaluation Result of BSDS500

This section gives an introduction about the evaluation results of BSDS500, which is a kind of edge detection evaluation test methods, and compares them with some results of popular edge detection algorithm. BSDS500, a high standard contour edge detection algorithm, which was proposed by P. Arbelaez and so on, has become a mainstream evaluation algorithm to evaluate detection performance in the field of edge detection at present. BSDS500 assesses the images, which are generated by edge detection algorithms from the aspects of edge detection precision and recall rate, and gets the evaluation indices ODS and OIS finally. Specifically, ODS refers to optimal data set ratio, which evaluates the detection results by fixed threshold. While OIS refers to optimal image ratio, which evaluates the detection results by optimal threshold that is best for the current image. The ODS and OIS indices of the proposed method are 0.779 and 0.782 respectively, which are far better than the results of traditional edge detection algorithms. Simultaneously, in contrast with some edge detection algorithm based on convolution network, the edge detection method based on the generative adversarial network has slightly advantages. What's more, the algorithm proposed by this paper has better effect than HED detection algorithm without non-maximum suppression, while is not as good as HED algorithm using non-maximum suppression. The final evaluation results are shown in Tab. 2.

Table 2: The score comparison of ODS and OIS with the proposed algorithm and HED, Canny, Laplacian, Sobel, Scharr, DoG by BSDS500 benchmark test

Algorithm	ODS	OIS
Roberts [12]	0.483	0.513
Sobel [9]	0.539	0.575
Canny [8]	0.611	0.676
DeepEdge [22]	0.753	0.772
DeepContour [23]	0.757	0.776
HED [27]	0.788	0.808
HED algorithm without non-maximum suppression [27]	0.724	0.728
This proposed algorithm	0.779	0.782

Although the results of the proposed algorithm are slightly inferior in the results of HED algorithm using non-maximum suppression, it can acquire the results of ideal effect in a shorter time. The whole training process trained on NVIDIA GTX 1060 just needed about two hours, while it took seven hours to train on NVIDIA K40 GPU. Obviously, the process trained on NVIDIA GTX 1060 had a distinct advantage. Meanwhile, it took a short time for generator network with U-Net as the core to generate test results when performing specific tests. Furthermore, the average time of calculating 200 images by HED algorithm using non-maximum suppression was roughly 53 ms when trained on NVIDIA GTX 1060, while the time with the algorithm proposed by the paper just was 36 ms. It can be seen from this that the algorithm proposed by the paper has a distinct advantage.

Next, the paper evaluated different results and got various accuracies and recall rates under different threshold settings. According to these data, a graph was drawn, as shown in Fig. 14. From the results, it observed that the algorithm proposed by the paper has good performance in accuracy and recall rate and has obvious advantage, compared with traditional algorithms. Besides, it is almost the same as HED algorithm.

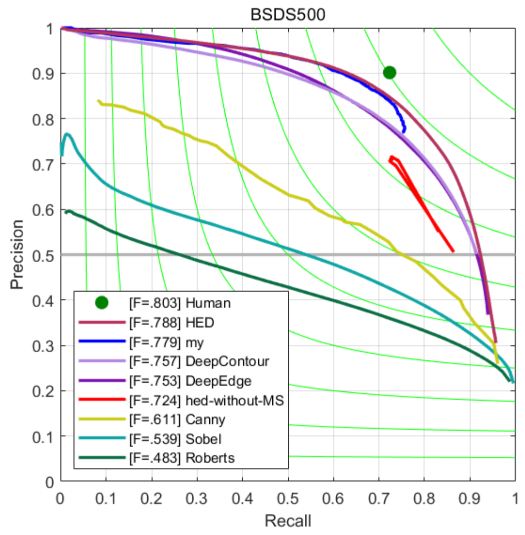


Figure 14: The accuracy and recall rate curve comparison with the proposed algorithm and HED, Canny, Laplacian, Sobel, Scharr, DoG by BSDS500 benchmark test

5 Conclusion

Aiming at the problem of edge detection, this paper proposes a new edge detection method based on the generative adversarial network. This algorithm can solve the problem of edge detection well. The proposed algorithm has distinct advantage, compared with traditional edge detection algorithms, and has better effect than some edge detection algorithm based on convolution network. Meanwhile, the algorithm proposed by the paper can deal with the thick edge problem in edge detection of convolution network and achieve better results without using non-maximum suppression. In contrast with HED algorithm, one of the best edge detection algorithms, the proposed method not only has shorter training time, but also detects the edge of images more quickly.

Nevertheless, the algorithm proposed by the paper also has some problems. The problems of GAN were reflected in the experiment in the process of train. For example, the model could not keep converging in a good direction. During the training process of discriminator, the advantages became greater and greater, which resulted in the consequence that builder could not learn anything and the whole model crashed. For this problem, intermediate results were chosen under a trade-off to stop the whole network, which is also a problem the next step to solve.

Funding Statement: The author(s) received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] W. Guan, T. Wang, J. Qi, L. Zhang and H. Lu, "Edge-Aware convolution neural network based salient object detection," *IEEE Signal Processing Letters*, vol. 26, no. 1, pp. 114-118, 2019.

[2] M. Seyedhosseini and T. Tasdizen, "Semantic image segmentation with contextual hierarchical models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 58, no. 5, pp. 951-964, 2016.

[3] Y. Zhu, C. Zhao, H. Guo, J. Wang, X. Zhao and H. Lu, "Attention CoupleNet: fully convolutional attention coupling network for object detection," *IEEE Transactions on Image Processing*, vol. 28, no. 1, pp. 113-126, 2018.

[4] Y. Li, M. L. Mekhalfi, M. M. Al Rahhal, E. Othman and H. Dhahri, "Encoding motion cues for pedestrian path prediction in dense crowd scenarios," *IEEE Access*, vol. 5, pp. 24368-24375, 2017.

[5] G. Wang et al., "Interactive medical image segmentation using Deep Learning with image-specific fine tuning," *IEEE Transactions on Medical Imaging*, vol. 37, no. 7, pp. 1562-1573, 2018.

[6] E. Ugur, Y. Nagai, E. Sahin and E. Oztop, "Staged development of robot skills: Behavior formation, affordance learning and imitation with motionese," *IEEE Transactions on Autonomous Mental Development*, vol. 7, no. 2, pp. 119-139, 2015.

[7] L. Fangmin, C. Ke and L. Xinhua, "3D Face reconstruction based on convolutional neural network," *10th International Conference on Intelligent Computation Technology and Automatio*n, Changsha, pp. 71-74, 2017.

[8] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679-698, 1986.

[9] H. Zhang, Q. Zhu and X. Guan, "Probe into image segmentation based on Sobel Operator and maximum entropy algorithm," *International Conference on Computer Science and Service System*, Nanjing, pp. 238-241, 2012.

[10] G. Levkine, "Prewitt, Sobel and Scharr gradient 5x5 convolution matrices". [Online]. Available:
http://www.hlevkin.com/articles/SobelScharrGradients5x5.pdf.

[11] A. Anand, S. S. Tripathy and R. S. Kumar, "An improved edge detection using morphological Laplacian of Gaussian operator," *2nd International Conference on Signal Processing and Integrated Networks*, Noida, pp. 532-536, 2015.

[12] G. M. H. Amer and A. M. Abushaala, "Edge detection methods," *2nd World Symposium on Web Applications and Networking*, Sousse, pp. 1-7, 2015.

[13] D. Marr and E. Hildreth, "Theory of edge detection," in *Proc. of the Royal Society of London. Series B, Containing Papers of a Biological Character*, Royal Society (Great Britain), vol. 207, no. 1167, pp. 187-217, 1980.

[14] D. R. Martin, C. C. Fowlkes and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 5, pp. 530-549, 2004.

[15] P. Arbelaez, M. Maire, C. Fowlkes and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, pp. 898-916, 2011.

[16] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, USA, pp. 1-224, 2015.

[17] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in *Proc. of the IEEE*, vol. 86, no. 11, pp. 2278-2324, 1998.

[18] A. Krizhevsky, I. Sutskever and G. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 1, no. 4, pp. 1-9, 2012.

[19] S. Liu and W. Deng, "Very deep convolutional neural network based on image classification using small training sample size," *3rd IAPR Asian Conference on Pattern Recognition*, Kuala Lumpur, pp. 730-734, 2015.

[20] C. Szegedy, L. Wei, Y. Q. Jia, P. Sermanet, S. Reed *et al.*, "Going deeper with convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-9, 2015.

[21] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770-778, 2016.

[22] G. Bertasius, J. Shi and L. Torresani, "DeepEdge: a multi-scale bifurcated deep network for top-down contour detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, pp. 4380-4389, 2015.

[23] J.J. Hwang and T. L. Liu, "Pixel-wise deep learning for contour detection," in *International Conference on Learning Representations*, pp. 1-2, 2015.

[24] Y. Ganin and V. Lempitsky, "N4-Fields: neural network nearest neighbor fields for image transforms," in *Asian Conference on Computer Vision*, pp. 536-551, 2014.

[25] H. Chen, X. Qi, L. Yu and P. Heng, "DCAN: deep contour-aware networks for accurate gland segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, pp. 2487-2496, 2016.

[26] Y. Liu, M. M. Cheng, X. W. Hu and K. Wang, "Richer convolutional features for edge detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3000-3009, 2017.

[27] S. Xie and Z. Tu, "Holistically-nested edge detection," in *The IEEE International Conference on Computer Vision*, pp. 1395-1403, 2015.

[28] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley *et al.*, "Generative adversarial nets," in *International Conference on Neural Information Processing Systems*, MIT Press, pp. 1-9, 2014.

[29] C. Zheng and Y. Zhang, "Two-stage color ink painting style transfer via convolution neural network," in *15th International Symposium on Pervasive Systems, Algorithms and Networks*, Yichang, China, pp. 193-200, 2018.

[30] M. S. M. Sajjadi, B. Schölkopf and M. Hirsch, "EnhanceNet: Single image super-resolution through automated texture synthesis," in *The IEEE International Conference on Computer Vision*, pp. 4491-4500, 2017.

[31] H. Heo and Y. Hwang, "Automatic sketch colorization using DCGAN," in *18th International Conference on Control, Automation and Systems*, Daegwallyeong, pp. 1316-1318, 2018.

[32] H. Zhang, T. Xu, H. S. Li and S. T. Zhang, "StackGAN: text to photo-realistic image synthesis with stacked generative adversarial networks," *IEEE International Conference on Computer Vision*, Venice, pp. 5908-5916, 2017.

[33] Z. Zhang, "Improved Adam Optimizer for deep neural networks" in *IEEE/ACM 26th International Symposium on Quality of Service*, Banff, AB, Canada, pp. 1-2, 2018.

[34] Z. Pan, W. Yu, X. Yi, A. Khan, F. Yuan and Y. Zheng, "Recent progress on generative adversarial networks (GANs): a survey" *IEEE Access*, no. 7, pp. 36322-36333, 2019.

[35] J. Deng, G. Pang, Z. Zhang, Z. Pang, H. Yang and G. Yang, "cGAN based facial expression recognition for human-robot interaction" *IEEE Access*, no. 7, pp. 9848-9859, 2019.

[36] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, USA, pp. 1-224, 2015.

[37] M. D. Zeiler, D. Krishnan, G. W. Taylor and R. Fergus, "Deconvolutional networks," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2528-2535, 2010.

[38] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. of the 32nd International Conference on Machine Learning*, no. 37, pp. 448-456, 2015.

[39] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1958, 2014.

[40] B. L. Kalman and S. C. Kwasny, “Why tanh? Choosing a sigmoidal function,” in *Proc. Int. Joint Conf. Neural Networks*, vol. 1, no. 4, pp. 578-581, 1992.

[41] X. Glorot, A. Bordes and Y. Bengio, “Deep sparse rectifier neural networks,” in *Proc. 14th International Conference on Artificial Intelligence and Statistics*, pp. 315-323, 2011.

[42] A. L. Maas and A. Y. Hannun, “Retifier nonlinearities improve neural network Acostic Models,” *Proceedings of Machine Learning Research*, pp. 30, 2013.

[43] O. Ronneberger, P. Fischer and T. Brox, “U-net: convolutional networks for biomedical image segmentation,” *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234-241, 2015.

[44] M. D. Awheda and H. M. Schwartz, “Exponential moving average Q-learning algorithm,” in *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, pp. 31-38, 2013.