


Article

# Image Super-Resolution Based on CNN Using Multilabel Gene Expression Programming

Jiali Tang <sup>1</sup> , Chenrong Huang <sup>2,\*</sup>, Jian Liu <sup>1</sup> and Hongjin Zhu <sup>1</sup>

<sup>1</sup> College of Computer Engineering, Jiangsu University of Technology, Changzhou 213001, Jiangsu, China; tangjl@jsut.edu.cn (J.T.); lj18816276124@hotmail.com (J.L.); zhuhongjin@jsut.edu.cn (H.Z.)

<sup>2</sup> School of Computer Engineering, Nanjing Institute of Technology, Nanjing 211167, Jiangsu, China

\* Correspondence: huangcr@njit.edu.cn

Received: 25 December 2019; Accepted: 23 January 2020; Published: 25 January 2020



**Abstract:** Current mainstream super-resolution algorithms based on deep learning use a deep convolution neural network (CNN) framework to realize end-to-end learning from low-resolution (LR) image to high-resolution (HR) images, and have achieved good image restoration effects. However, as the number of layers in the network is increased, better results are not necessarily obtained, and there will be problems such as slow training convergence, mismatched sample blocks, and unstable image restoration results. We propose a preclassified deep-learning algorithm (MGEP-SRCNN) using Multilabel Gene Expression Programming (MGEP), which screens out a sample sub-bank with high relevance to the target image before image block extraction, preclassifies samples in a multilabel framework, and then performs nonlinear mapping and image reconstruction. The algorithm is verified through standard images, and better objective image quality is obtained. The restoration effect under different magnification conditions is also better.

**Keywords:** super-resolution (SR); convolution neural network (CNN); Gene Expression Programming (GEP); deep learning; image preclassification

## 1. Introduction

Aiming at addressing image degradation during digital image acquisition and processing, Single-Image Super-Resolution (SISR) technology [1,2] enables high-resolution (HR) images to be recovered from low-resolution (LR) ones with high-frequency texture details and edge structure information observations of images to meet people's image quality needs.

Early reconstruction-based SR methods mainly modeled the acquisition process of low-resolution observation images, used the regularization method to construct the prior constraints of high-resolution images, estimated the HR images from the LR observation images, and finally restored the super-resolution image. The problem turns into an optimization problem of a cost function under a constraint [3]. Learning-based super-resolution restoration technology was first proposed by Freeman [4]. This type of algorithm uses the similarity of different images in high-frequency details to obtain the relationship between the HR and LR images through the algorithm to guide the reconstruction of the output image. Prior knowledge can be obtained through learning, instead of defining the prior knowledge in the model-based reconstruction method.

In recent years, deep learning has been a hot topic in the field of machine learning, and its related theories have attracted widespread attention from researchers. It has been gradually used in the field of image super-resolution reconstruction and has better effects than shallow learning [5–13]. The most classic of them is an end-to-end deep learning model, the super-resolution convolution neural network (SRCNN), constructed by a three-layer convolutional neural network proposed by Dong et al. [5]. The image is input after Bicubic preprocessing, and it is trained on LR and HR images in pairs.

The gradient descent method continuously adjusts the weights to obtain a mapping from LR to HR. Its simple network structure design and excellent image restoration results created a precedent for deep learning in super-resolution image reconstruction. However, the SRCNN method does not obtain better results when the number of layers in the network is increased. The training convergence rate is slow and it is not suitable for multiscale amplification.

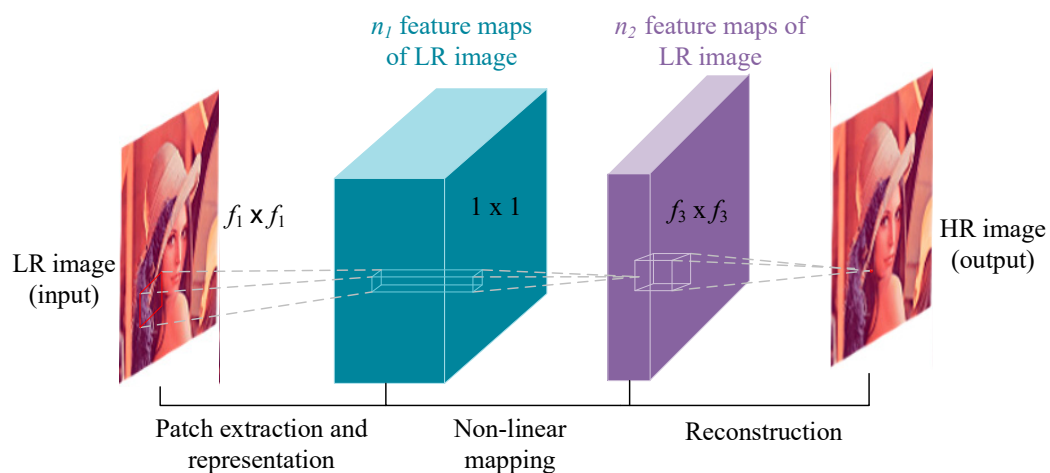
Based on the above problems, we propose an improved deep learning algorithm for Gene Expression Programming (GEP) preclassification. This method uses the GEP multilabeling algorithm to classify the trained image set and select a subset of samples that are related in color and texture feature categories, thereby reducing the complexity of the convolutional neural network parameters. We compare the performance of our approach to that of other state-of-the-art methods and obtain improved objective image quality.

## 2. Related Work

In recent years, deep learning and artificial intelligence have been widely used in various industries, especially in the field of computer vision, and have achieved better results than traditional methods [14,15]. Using the feed-forward depth network methods of CNN is the mainstream of the current super-resolution reconstruction field after sparse representation. Such methods focus less on the reconstruction speed and more on whether the high-resolution map can be better restored at a large magnification. They have better generalization ability and ability to characterize the high-level characteristics, compared with traditional shallow learning algorithms.

### 2.1. SRCNN/Fast SRCNN (FSRCNN)

Dong et al. [5] first proposed the use of a convolutional neural network for image super-resolution reconstruction. An LR image was first enlarged to the target size using Bicubic interpolation, and then a nonlinear mapping was performed through a three-layer convolutional network. The obtained results were output as high-resolution images, and good results were obtained. As shown in Figure 1, the network structure design of this method is simple. Compared with previous learning algorithms, it saves a lot of artificial feature extraction steps and post-integration, thus opening up the era of deep CNN super-resolution image processing problems.



**Figure 1.** Network architecture of the super-resolution convolution neural network (SRCNN).

After that, Dong et al. [6] improved upon the SRCNN, and the fast SRCNN (FSRCNN) was proposed, which increases the depth of the network and introduces a deconvolution layer to restore features. The deconvolution layer can realize the conversion from low-resolution space to high-resolution space. This feature allows the FSRCNN to directly use the low-resolution image instead of the interpolation result as the network input. Directly using LR images as input can not only

reduce the calculation amount of the model, but also avoid the obvious artificial traces introduced by interpolation. FSRCNN offers a great improvement in speed, without any preprocessing, to achieve end-to-end input and output of the network, but the restoration accuracy is somewhat insufficient.

### 2.2. Sparse-Coding-Based Network (SCN)

The SCN method [9] firstly obtains the sparse prior information of the image through the feature extraction layer, then establishes a feed-forward neural network which can implement sparse encoding and decoding of the image, and finally uses a cascade network to complete the image enlargement. This method can improve the Peak Signal-to-Noise Ratio (PSNR) at a higher magnification, and the algorithm running speed is further improved. Moreover, with the correct understanding of each layer's physical meaning, the SCN method offers a more principled way to initialize the parameters, which helps to improve optimization speed and quality.

### 2.3. Very Deep Convolutional Networks for Image Super-Resolution (VDSR)

The ultra-deep super-resolution network proposed by Kim et al. [8] extended the SRCNN from a 3-layer shallow network structure to a 20-layer ultra-deep network, and they concluded that the reconstruction effect will be improved as the number of layers increases. Compared to SRCNN, which only depends on the image context in a small area, VDSR, by exploring more contextual information through a larger receptive field, helps to better restore the detailed structure, especially in super-resolution applications with large magnification factors. In addition, in order to solve the problem of slow convergence in SRCNN, VDSR residual learning was introduced, which greatly increased the learning rate.

VDSR accepts image features of different scales by adjusting the size of the filter to produce a fixed feature output. Although VDSR can achieve specific-scale magnification, it cannot achieve free-scale, multiscale magnification, and its parameter storage and retrieval also have obvious shortcomings.

## 3. The Proposed Method

### 3.1. Multifeature Representation of Images

Image features include color, texture, shape, and spatial relationships. Features can be extracted from the image after being detected by the computer. The result is called a feature description or feature vector. In this paper, the Multilabel Gene Expression Programming (MGEP) algorithm mainly uses image color and texture features for multilabel recognition.

#### 3.1.1. Color Feature

The color feature is a global feature that describes the surface properties of the scene corresponding to the image or image area, and is also the most direct visual feature in the physical characteristics of the image. Compared with various other image features, the color feature has two obvious advantages: one is stability, low sensitivity to various changes in the image such as translation, scaling, rotation, etc., and strong robustness; the second is that its complicated calculation degree is low. The pixel values in the image are converted, and the corresponding numerical expression is used to obtain the image characteristics [16]. Because of the stability and simple calculation, the color feature has become a widely used image feature.

Shown in Figure 2, the color histogram is widely used in many image retrieval systems. It describes the proportion of different colors in the entire image, and does not pay attention to the spatial location of each color; that is, it cannot describe the objects or objects in the image.

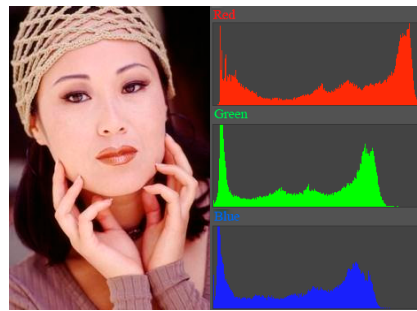


Figure 2. Color histogram of an image (RGB channel).

The color histogram can be defined as the joint probability density function of the three color channels (RGB) in the image:

$$h_{R,G,B}(a,b,c) = N \bullet P(R = a, G = b, B = c) \tag{1}$$

where  $R$ ,  $G$ , and  $B$  represent the RGB color channels of the image,  $N$  indicates the number of image pixels,  $P$  represents the probability density function, and  $h$  represents a histogram function, defined as a four-dimensional eigenvector  $H(H_R, H_G, H_B, \mu)$ . The first three dimensions  $H_R$ ,  $H_G$ , and  $H_B$  correspond to the three color channels, and the last dimension  $\mu$  indicates the proportion of the color in the entire image.

### 3.1.2. Texture Feature

Texture is a visual feature that reflects homogeneous phenomena in an image, and it reflects the surface structure organization and arrangement properties of an object surface with slow or periodic changes. Texture is a pattern produced by the gray or color of the target image in space in a certain form [17]. From the perspective of texture, the image can be roughly divided into three cases: first, the gray distribution has a certain periodicity (even if the gray change is random, it also has certain statistical characteristics, and may be in a larger area repeatedly); second, the basic components that make up the sequence are regular rather than random; third, the texture of each part in the texture area shows roughly the same size, structure, and image, and is uniformly distributed as a whole.

The Fourier power spectrum method is used to measure the texture characteristics of the image. Let the texture image be  $f(x, y)$ ; its Fourier transform can be expressed by Equation (2).

$$F(u, v) = \int \int_{-\infty}^{\infty} f(x, y) \exp\{-j2\pi(ux + vy)\} dx dy \tag{2}$$

The definition of the power spectrum of the two-dimensional Fourier transform is shown in Equation (3):

$$|F|^2 = FF^* \tag{3}$$

where  $F^*$  stands for the conjugate of  $F$ . The power spectrum  $|F|^2$  reflects the nature of the entire image. If the Fourier transform is expressed in polar form, i.e.,  $F(r, \theta)$  form, then the energy on the circle  $r$  from the origin is

$$\Phi_r = \int_0^{2\pi} [F(r, \theta)]^2 d\theta. \tag{4}$$

From research on the energy in the small fan-shaped region in the angle  $\theta$  direction, the law of this energy changing with the angle can be obtained by Equation (5):

$$\Phi_\theta = \int_0^{\infty} |F(r, \theta)|^2 dr. \tag{5}$$

When a texture image runs along  $\theta$  and there are many lines, edges, etc., in the direction  $\theta + \frac{\pi}{2}$ , i.e., in a right-angle direction to  $\theta$ , the energy is concentrated. If the texture does not show directionality, there is no directionality in the power spectrum. Therefore, the  $|F|^2$  value reflects the directionality of the texture.

### 3.2. GEP Network

Gene Expression Programming (GEP) was proposed by Ferreira in 2001. It is a new evolutionary model and belongs to the family of genetic algorithms [18,19]. The GEP algorithm, like the Genetic Algorithm (GA) and Genetic Programming (GP), is a computational model that simulates the evolutionary process of living things. Because GEP chromosomes are simple, linear, and compact; make it easy to carry out genetic operations, etc.; and have stronger problem-solving capabilities, they are 2 to 4 orders of magnitude faster than GA and GP [19]. Because of these advantages, GEP technology has attracted the attention of many researchers and has been used in machine learning fields such as function discovery, symbol regression, classification, clustering, and association rule analysis.

$(S, F, T)$  represents a GEP gene as a 3-tuple, of which  $S$  is a fixed-length string,  $F$  is the set of calculation functions, and  $T$  is the basic terminal set. Sometimes, for convenience, the fixed-length string  $S$  is called a gene. The gene is divided into two parts, the head and the tail. The former symbol can be taken from  $F$  and  $T$ , and the latter must be taken from  $T$ . GEP gene coding rules ensure that it can be decoded into an expression tree corresponding to a legal mathematical expression. Suppose its head length is  $h$ , tail length is  $t$ , and  $n_{\max}$  is the maximum number of parameters of a function in the function set; then the relationship between  $h$  and  $t$  can be expressed by Equation (6).

$$t = h(n_{\max} - 1) + 1 \tag{6}$$

GEP's neural network selective integration process is divided into two stages.

**Stage 1:** Network group generation.

We use existing methods such as Boosting and Bagging to generate network groups. Assume the output vector of the network population is  $Y = (y_1, y_2, \dots, y_n), y_i \in \{0, 1\}$ .

**Stage 2:** Individual network selection and conclusion generation based on GEP.

For input  $x$ , there are the following integrated classification results:

$$y(x) = \text{sign}(f(Y')). \tag{7}$$

Among them,  $Y' \subseteq Y, Y' = (y_{i_1}, y_{i_2}, \dots, y_{i_m})$ . That is, the final classification result is synthesized from some network outputs in the network group in some way. Because GEP has powerful function discovery and parameter selection functions, it can be discovered using the GEP method  $f(Y')$ .

Taking the threshold  $\lambda = 0.5$ , the integrated classification result  $y(x)$  is calculated as follows:

$$y(x) = \begin{cases} 1 & f(Y') \geq 0.5 \\ 0 & f(Y') < 0.5 \end{cases} \tag{8}$$

### 3.3. Fitness Function Design

The fitness function is the guideline for the evolution of the GEP algorithm. For feed-forward neural networks, the topology selection and training of weights can be regarded as an optimization process. The purpose of optimization is to design the network so that the fitness function value reaches the maximum value. The performance of the current network for a given training data set is described by a least squares error function.

We use category-based multilabel evaluation indicators. We first measure the classifier's corresponding two-class classification performance on a single class, and then calculate the average performance of the classifier on all classes as the evaluation index value of the classifier. Suppose we

have a multilabeled test set with  $p$ -many sample data  $S = \{(x_i, Y_i) | 1 \leq i \leq p\}$  for the  $j$ th category  $y_j$  ( $1 \leq j \leq q$ ). In terms of the multilabel classifier, the two-class classification performance of  $h(\cdot)$  in this category can be described by the four statistics given by Equations (9)–(12).

1.  $TP_j$  (#true positive instances)

$$TP_j = |\{x_i | y_j \in Y_i \wedge y_j \in h(x_i), (x_i, Y_i) \in S\}| \tag{9}$$

2.  $FP_j$  (#false positive instances)

$$FP_j = |\{x_i | y_j \notin Y_i \wedge y_j \in h(x_i), (x_i, Y_i) \in S\}| \tag{10}$$

3.  $TN_j$  (#true negative instances)

$$TN_j = |\{x_i | y_j \notin Y_i \wedge y_j \notin h(x_i), (x_i, Y_i) \in S\}| \tag{11}$$

4.  $FN_j$  (#false negative instances)

$$FN_j = |\{x_i | y_j \in Y_i \wedge y_j \notin h(x_i), (x_i, Y_i) \in S\}| \tag{12}$$

From Equations (9)–(12),  $TP_j + FP_j + TN_j + FN_j = p$  is established. Most classification performance indicators, such as accuracy, precision, and recall, can be derived from the above four statistics—see Equations (13)–(15).

$$\text{Accuracy} = B(TP_j, FP_j, TN_j, FN_j) = \frac{TP_j + TN_j}{TP_j + FP_j + TN_j + FN_j} \tag{13}$$

$$\text{Precision} = B(TP_j, FP_j, TN_j, FN_j) = \frac{TP_j}{TP_j + FP_j} \tag{14}$$

$$\text{Recall} = B(TP_j, FP_j, TN_j, FN_j) = \frac{TP_j}{TP_j + FN_j} \tag{15}$$

Therefore, combined with category-based multilabel classification evaluation index, we design a fitness function for the MGEP classification algorithm.

1. Design of fitness function based on macro-averaging:

$$Fit_i = \left| R - 100 \times \frac{1}{q} \sum_{j=1}^q B(TP_j, FP_j, TN_j, FN_j) \right| \tag{16}$$

2. Design of fitness function based on micro-averaging:

$$Fit_i = \left| R - B \left( \sum_{j=1}^q TP_j, \sum_{j=1}^q FP_j, \sum_{j=1}^q TN_j, \sum_{j=1}^q FN_j \right) \right| \tag{17}$$

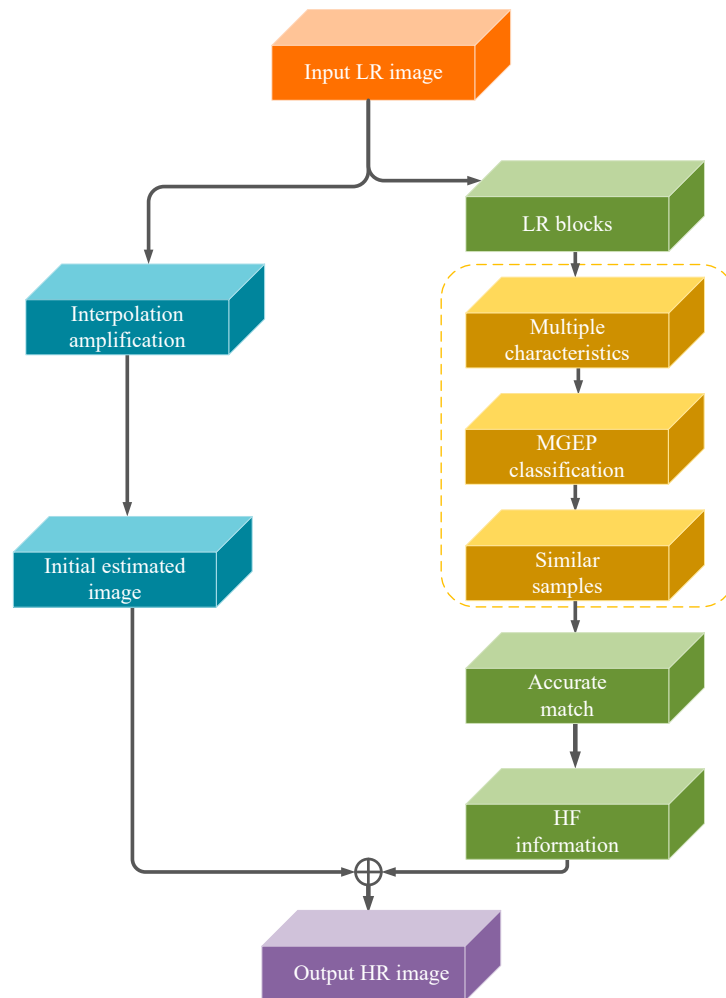
In Equations (16) and (17),  $Fit_i$  is the fitness value of the  $i$ th individual to the environment, and  $R$  is the selected bandwidth.

### 3.4. MGEP Classification before CNN Image Super-Resolution

In the process of image super-resolution reconstruction, the traditional K-means clustering algorithm [20] is used to classify the trained image set to improve the training effect and reduce the training time. In the K-means clustering algorithm, the value of  $K$  needs to be determined in advance,

and it cannot be changed during the entire course of the algorithm, which makes it difficult to accurately estimate the value of  $K$  when training high-dimensional data sets.

We used the MGEP classification algorithm instead of the K-means clustering algorithm, and its preclassification model is shown in Figure 3.



**Figure 3.** The Multilabel Gene Expression Programming (MGEP) preclassification model.

Let  $p$ -many sample pattern pairs  $(x_k, y_k)$  constitute the training set,  $k = 1, 2, \dots, p$ . According to the definition of the GEP classifier, for sample  $k$ , there are  $x_k = (x_{k1}, x_{k2}, \dots, x_{km})$  and  $y_k = (y_{k1}, y_{k2}, \dots, y_{kn})$ , where  $x_{kj}$  is the sample in the attribute  $A_j$  ( $j = 1, 2, \dots, m$ ) and  $y_{ki}$  is the degree of membership of the sample to the category  $C_i$  ( $i = 1, 2, \dots, n$ ). The training set constitutes the set of adaptive instances, and the set of adapted instances of a particular problem forms the adaptive environment of the MGEP algorithm. Under a certain adaptive environment, starting from the initial population, selection, replication, and various genetic operations are performed according to individual fitness to form a new population. This process is repeated until the optimal individual is evolved and decoded to obtain a GEP multilabel classifier. In the MGEP preclassification learning, we find similar samples for LR image blocks in terms of color and texture features, thereby shortening the time of the next precise matching and improving the efficiency and effect of SR image restoration.

In the training of the CNN, the MGEP algorithm is used to classify the trained image set, classify the approximate images into one category, and reduce the parameter scale of the convolutional neural network model, which can reduce the training time of the network to a certain extent and improve the training efficiency of the CNN. The improved algorithm flow based on SRCNN is shown in Figure 4.



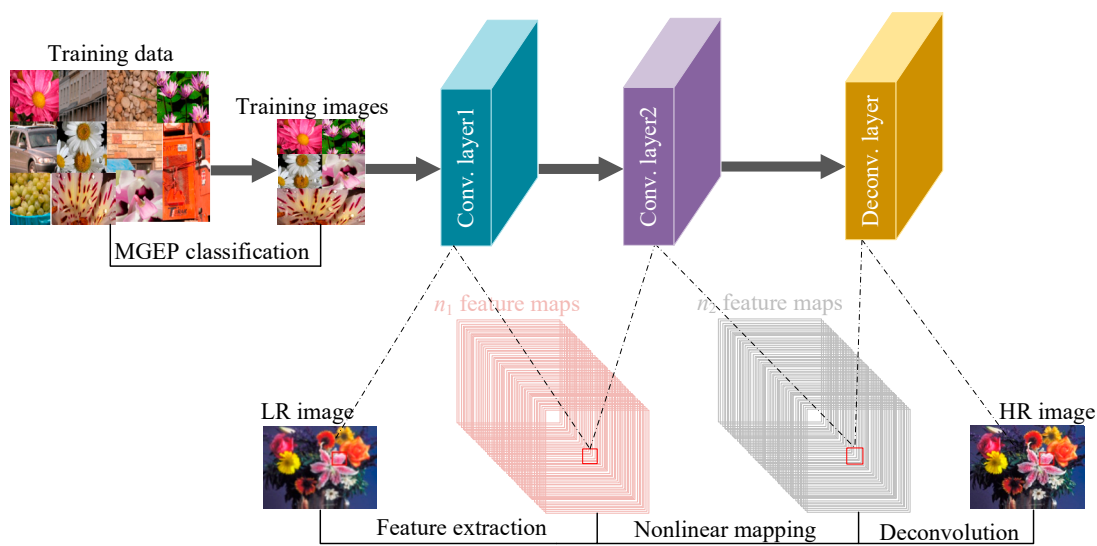


Figure 4. MGEP-SRCNN algorithm flow chart.

The algorithm uses Wiener filtering to construct a deconvolution layer, which is used to implement multiscale image reconstruction. The deconvolution network part adopts the mirror structure of the convolution network. The purpose is to reconstruct the shape of the input target, so the multilevel deconvolution structure can also capture the shape details of different levels like the convolution network. In the model of the convolutional network, low-level features can describe the rough information of the entire target, such as target position and approximate shape, while more complex high-level features have classification characteristics and also contain more detailed target information.

#### 4. Experiments

##### 4.1. Experimental Environment and Parameter Settings

The experimental software environment used was Ubuntu 14.04, Python 2.7, TensorFlow 1.4; the hardware environment was an Intel Core i7-6700K, RAM 16GB, and the GPU was an NVIDIA GTX1080.

As the training set, we used ImageNet-91 [5], and as the test set, we used Set5 [21], Set14 [22], BSD100 [23], and Urban100 [24]. We tested on three commonly used scale factors, 2, 3, and 4, and compared the results with those of the Bicubic, SCN [9], SRCNN [5], VDSR [8], and DRCN [10] algorithms. Two evaluation indexes, PSNR and Structural Similarity (SSIM), were selected as an objective reference basis for the superiority of algorithm reconstruction to measure the effect of image restoration.

##### 4.2. MGEP Preclassification Settings

We used the MGEP algorithm to preclassify the trained image set before deep learning. We selected a subset of samples that were related in color and texture feature categories. Equation (16) was used as the fitness function to implement the MGEP classification algorithm, which was then applied to sample preclassification in the SR image restoration process.

The function set was set to  $\{+, -, \times, /\}$ , the evolution algebra *gen* was set to 1000, and the population size *N* was set to 100. The total number of genes was set to 10, of which there were 7 common genes and 3 homologous genes. The three homologous genes respectively output the color category calculated according to Equation (1), the texture category calculated according to Equation (18), and the texture category calculated according to Equation (19) of the input LR image block.

$$T(j, k) = \sum_{\epsilon=-T}^j \sum_{n=-T}^k \epsilon^2 \eta^2 C(\epsilon, \eta, j, k) \tag{18}$$



$$\text{MEAN} = \frac{1}{m} \sum_i ip_{\Delta}(i) \quad (19)$$

Each gene had a head length of 5 and a chromosome length of 110. The genetic manipulation probability was set to 0.1.

The MGEP preclassification effect is shown in Figure 4. The training samples “Car”, “Building”, and “Cobblestone” in ImageNet-91 that have little correlation with the color texture features of the input image “Flowers” were excluded, thereby improving the training effect and matching accuracy.

### 4.3. Image Restoration Results

CNN deep learning was performed after MGEP preclassification. All convolutional layer filters were  $3 \times 3$  in size and the number of filters was 64. We used the method of He et al. [25] to initialize the convolutional layer. The convolution kernel moved in steps of 1. In order to keep the size of all feature maps the same as the input of each layer, 0 was filled around the boundary before applying the convolution. The learning rate of all layers was initialized to  $5 \times 10^{-4}$ , and the learning rate dropped by 2 times every 15 epochs until the learning rate was less than  $5 \times 10^{-9}$ .

Tables 1–4 show the PSNR/SSIM values and running times of the six algorithms on the Set5, Set14, BSD100, and Urban100 test sets when the upscale factors were 2, 3, and 4, respectively.

**Table 1.** Average Peak Signal-to-Noise Ratio (PSNR)/Structural Similarity (SSIM) values for 2× scale. Red color indicates the best and the blue color indicates the second-best performance.

Dataset	Bicubic	SRCNN [5]	SCN [9]	VDSR [8]	DRCN [10]	MGEP-SRCNN
Set5	33.66/0.9299	36.66/0.9542	36.93/0.9552	37.53/0.9587	37.63/0.9588	37.78/0.9594
Set14	30.24/0.8688	32.42/0.9063	32.56/0.9074	33.03/0.9124	33.04/0.9118	33.29/0.9135
BSD100	29.56/0.8431	31.36/0.8879	31.40/0.8884	31.90/0.8960	31.85/0.8942	32.01/0.8979
Urban100	26.88/0.8403	29.50/0.8946	29.50/0.8960	30.76/0.9140	30.75/0.9133	31.19/0.9181

**Table 2.** Average PSNR/SSIM values for 3× scale. Red color indicates the best and the blue color indicates the second-best performance.

Dataset	Bicubic	SRCNN [5]	SCN [9]	VDSR [8]	DRCN [10]	MGEP-SRCNN
Set5	30.39/0.8682	32.75/0.9090	33.10/0.9144	33.66/0.9213	33.82/0.9226	34.04/0.9250
Set14	27.55/0.7742	29.28/0.8209	29.41/0.8238	29.77/0.8314	29.76/0.8311	29.97/0.8333
BSD100	27.21/0.7385	28.41/0.7863	28.50/0.7885	28.82/0.7976	28.80/0.7963	28.92/0.7972
Urban100	24.46/0.7349	26.24/0.7989	26.21/0.8010	27.14/0.8279	27.15/0.8276	27.24/0.8330

**Table 3.** Average PSNR/SSIM values for 4× scale. Red color indicates the best and the blue color indicates the second-best performance.

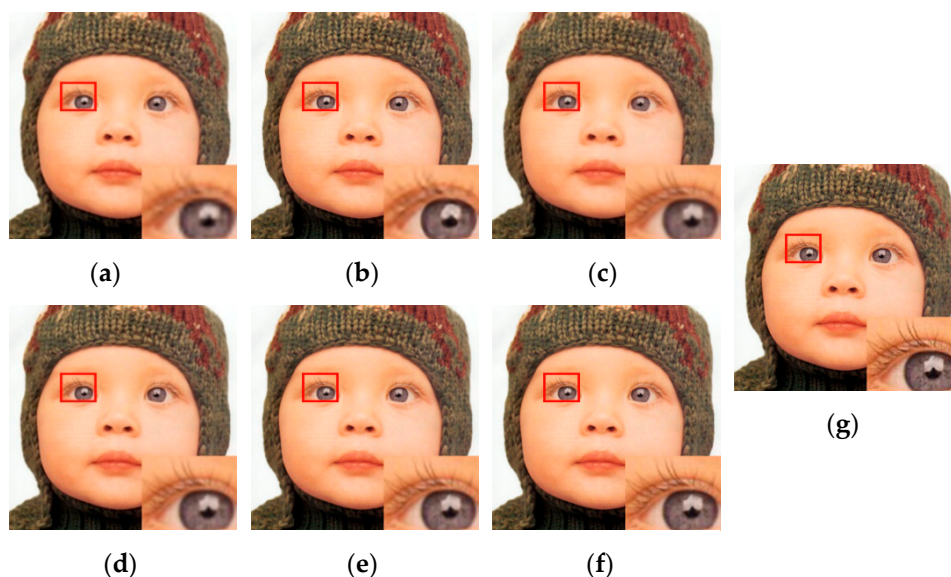
Dataset	Bicubic	SRCNN [5]	SCN [9]	VDSR [8]	DRCN [10]	MGEP-SRCNN
Set5	28.42/0.8104	30.48/0.8628	30.86/0.8732	31.35/0.8838	31.53/0.8854	31.66/0.8899
Set14	26.00/0.7027	27.49/0.7503	27.64/0.7578	28.01/0.7674	28.02/0.7670	28.19/0.8359
BSD100	25.96/0.6675	26.90/0.7101	27.03/0.7161	27.29/0.7251	27.23/0.7233	27.34/0.7238
Urban100	23.14/0.6577	24.52/0.7221	24.52/0.7250	25.18/0.7524	25.14/0.7510	25.21/0.7837

**Table 4.** Comparison of the running times (sec) for scales 2×, 3×, and 4×. Red color indicates the best performance.

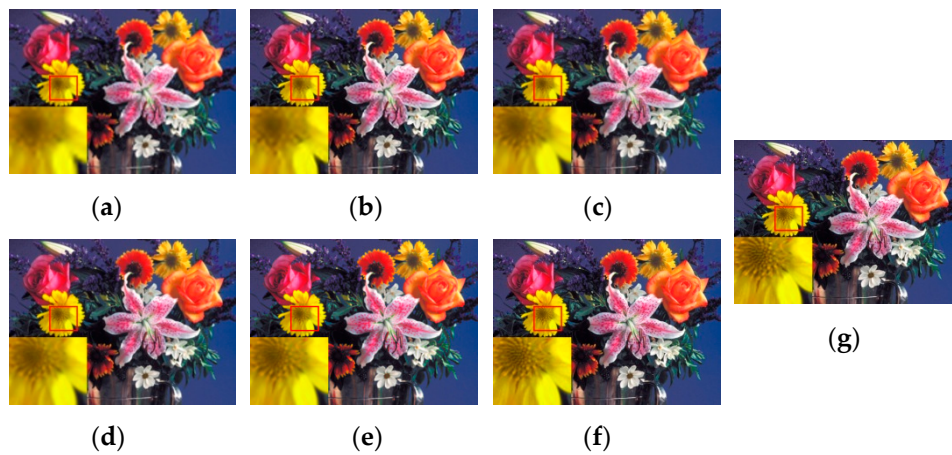
Dataset	Scale	Bicubic	SRCNN [5]	SCN [9]	VDSR [8]	DRCN [10]	MGEP-SRCNN
Set5	×2	-	2.191	0.941	0.054	0.735	0.039
	×3	-	2.235	1.829	0.062	0.748	0.052
	×4	-	2.193	1.245	0.054	0.735	0.044
Set14	×2	-	4.324	1.709	0.113	1.579	0.097
	×3	-	4.402	3.611	0.122	1.569	0.113
	×4	-	4.397	2.377	0.112	1.526	0.101
BSD100	×2	-	2.517	1.015	0.071	0.983	0.069
	×3	-	2.583	2.138	0.071	0.996	0.067
	×4	-	2.516	1.290	0.071	0.984	0.061
Urban100	×2	-	22.123	4.779	0.451	5.010	0.401
	×3	-	19.358	4.012	0.514	5.054	0.484
	×4	-	18.462	3.199	0.448	5.048	0.407

As can be seen from Tables 1–3, the MGEP-SRCNN algorithm achieved the best PSNR effect at different magnifications on the four test sets. Compared with SRCNN, the PSNR evaluation index increased by 0.44–1.69 dB, and the improvement effect obtained in the Set5 dataset was the best. In terms of SSIM indicators, except for the suboptimal data obtained under the 2× and 3× conditions on the BSD100 dataset, other SSIM indicators were all optimal. Compared with SRCNN, the SSIM evaluation index improved by about 0.005–0.062, where 4× showed the best improvement. In addition, from Table 4 we can see that the MGEP-SRCNN algorithm achieved the best performance of running time on the precondition of accuracy.

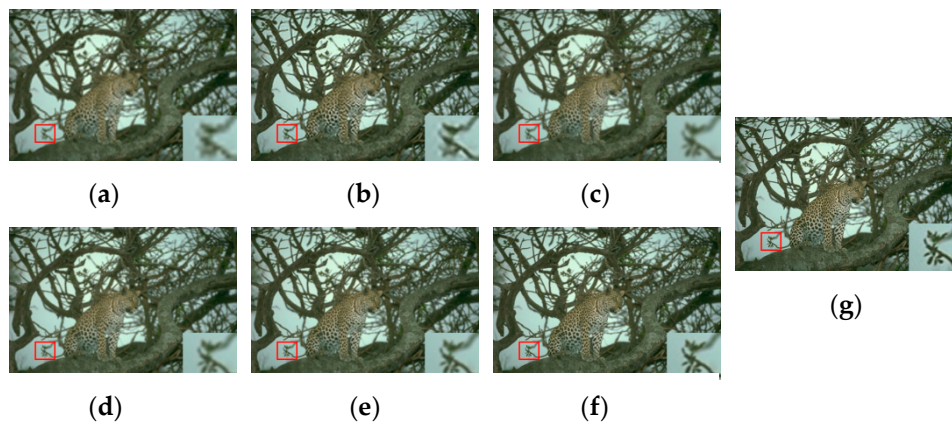
Then, we subjectively determined the quality of the output image and compared the performance of the six SR algorithms by observing the visual effects of the restored image. For comparison, given a 3× upscale factor, the restoration effects of the different SR algorithms used on the Set5, Set14, BSD100, and Urban100 test sets are shown in Figures 5–8.



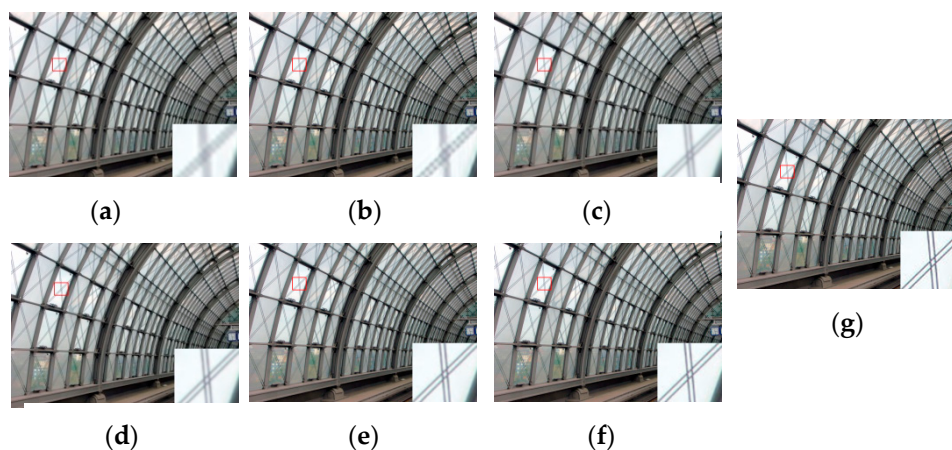
**Figure 5.** Super-resolution restoration results of the image “Baby” in Set5 [21]. (a) Bicubic; (b) SRCNN [5]; (c) SCN [9]; (d) VDSR [8]; (e) DRCN [10]; (f) MGEP-SRCNN; (g) Ground truth.



**Figure 6.** Super-resolution restoration results of the image “Flowers” in Set14 [22]. (a) Bicubic; (b) SRCNN [5]; (c) SCN [9]; (d) VDSR [8]; (e) DRCN [10]; (f) MGEP-SRCNN; (g) Ground truth.



**Figure 7.** Super-resolution restoration results of the image “016” in BSD100 [23]. (a) Bicubic; (b) SRCNN [5]; (c) SCN [9]; (d) VDSR [8]; (e) DRCN [10]; (f) MGEP-SRCNN; (g) Ground truth.



**Figure 8.** Super-resolution restoration results of the image “002” in Urban100 [24]. (a) Bicubic; (b) SRCNN [5]; (c) SCN [9]; (d) VDSR [8]; (e) DRCN [10]; (f) MGEP-SRCNN; (g) Ground truth.

It can be intuitively seen from the figure that the reconstructed images of both traditional Bicubic and SRCNN have aliasing, while the restored image provided by our algorithm is clearer and sharper, and the reconstruction quality is better. In the details, as seen in the baby eye in Figure 5, petal color

in Figure 6, branch texture in Figure 7, and building glass in Figure 8, MGEP-SRCNN reconstructed images have clearer features, have no sawtooth texture, and are more in line with the visual needs of the human eye.

Figure 9 shows the PSNR/SSIM values of the six algorithms on the Set5, Set14, BSD100, and Urban100 test sets when the upscale factor was 3. The MGEP-SRCNN algorithm achieved the best PSNR effects in all four datasets. Except for the suboptimal data obtained in BSD100, the SSIM indicators were all optimal.

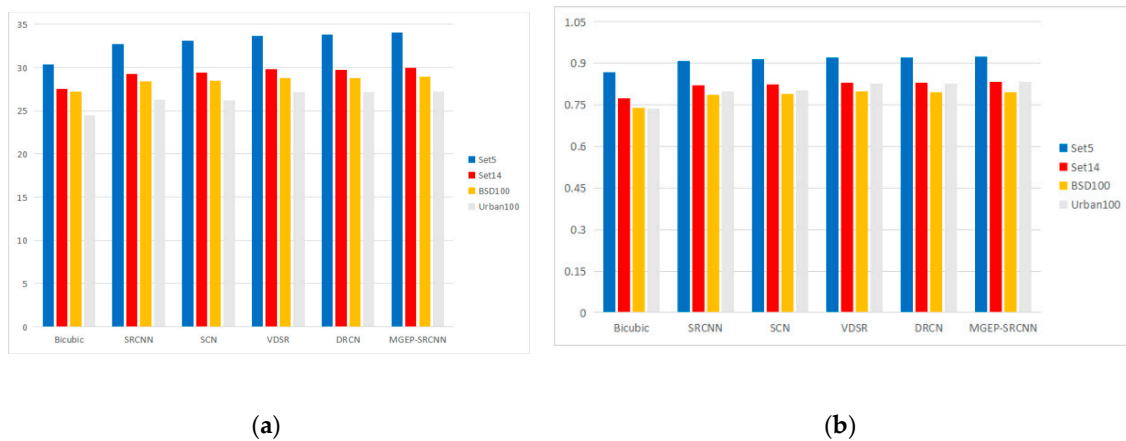


Figure 9. Image restoration results of the six SR algorithms. (a) PSNR; (b) SSIM.

## 5. Conclusions

In this work, we presented a super-resolution method using Multilabel Gene Expression Programming. This method uses MGEP to extract a subset of image samples related to color and texture features in advance, and then performs nonlinear mapping and image reconstruction, thereby reducing the complexity of the convolutional neural network parameters so as to avoid the SR problems of slow training convergence and unstable recovery results. It was experimentally verified that the image restoration effect of this method under different magnifications and on training sets is better than that of the commonly used deep learning algorithms, and it also performs well in terms of subjective visual effects.

**Author Contributions:** Conceptualization, J.T.; methodology, C.H.; software, J.L.; validation, H.Z.; formal analysis, J.T.; investigation, C.H.; resources, J.L.; data curation, J.L.; writing—original draft preparation, J.T.; writing—review and editing, H.Z.; supervision, C.H.; funding acquisition, J.T. and H.Z. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the National Natural Science Foundation of China (61806088), by the Opening Project of Jiangsu Key Laboratory of Advanced Numerical Control Technology (SYKJ201804), by the Project funded by Jiangsu Province Postdoctoral Science Foundation (2019K041), and by Changzhou Sci&Tech Program (CE20195030).

**Acknowledgments:** We are grateful to our anonymous referees for their useful comments and suggestions. The authors also thank Honghui Fan, Yijun Liu, Yan Wang, Wei Gao and Jie Zhang for their useful advice during this work.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

- Freeman, W.T.; Pasztor, E.C.; Carmichael, O.T. Learning Low-Level Vision. *Int. J. Comput. Vision* **2000**, *40*, 25–47. [[CrossRef](#)]
- Glasner, D.; Bagon, S.; Irani, M. Super-resolution from a single image. In Proceedings of the 2009 IEEE 12th International Conference on Computer Vision, Kyoto, Japan, 29 September–2 October 2009; pp. 349–356.



3. Zhang, X.L.; Lam, K.M.; Shen, L.S. Image magnification based on a blockwise adaptive Markov random field model. *Image Vis. Comput.* **2008**, *26*, 1277–1284. [[CrossRef](#)]
4. Freeman, W.T.; Jones, T.R.; Pasztor, E.C. Example-based super-resolution. *IEEE Comput. Graph. Appl.* **2002**, *22*, 56–65. [[CrossRef](#)]
5. Dong, C.; Loy, C.C.; He, K.M.; Tang, X.O. Image Super-Resolution Using Deep Convolutional Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 295–307. [[CrossRef](#)] [[PubMed](#)]
6. Dong, C.; Loy, C.C.; Tang, X. Accelerating the Super-Resolution Convolutional Neural Network. In Proceedings of the Computer Vision—ECCV 2016; Springer: Cham, Switzerland, 2016; pp. 391–407.
7. Hong, P.; Zhang, G. A Review of Super-Resolution Imaging through Optical High-Order Interference. *Appl. Sci.* **2019**, *9*, 1166. [[CrossRef](#)]
8. Kim, J.; Lee, J.K.; Lee, K.M. Accurate image super-resolution using very deep convolutional networks. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1646–1654.
9. Wang, Z.; Liu, D.; Yang, J.; Han, W.; Huang, T. Deep Networks for Image Super-Resolution with Sparse Prior. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 370–378.
10. Kim, J.; Lee, J.K.; Lee, K.M. Deeply-recursive convolutional network for image super-resolution. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1637–1645.
11. Zhang, K.; Zuo, W.; Chen, Y.; Meng, D.; Zhang, L. Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising. *IEEE Trans. Image Process.* **2017**, *26*, 3142–3155. [[CrossRef](#)] [[PubMed](#)]
12. Zhang, K.; Zuo, W.; Gu, S.; Zhang, L. Learning deep CNN denoiser prior for image restoration. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3929–3938.
13. Lim, B.; Son, S.; Kim, H.; Nah, S.; Lee, K.M. Enhanced deep residual networks for single image super-resolution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Honolulu, HI, USA, 21–26 July 2017; pp. 136–144.
14. Hayat, K. Multimedia super-resolution via deep learning: A survey. *Digit. Signal Process.* **2018**, *81*, 198–217. [[CrossRef](#)]
15. Yang, W.; Zhang, X.; Tian, Y.; Wang, W.; Xue, J.; Liao, Q. Deep Learning for Single Image Super-Resolution: A Brief Review. *IEEE Trans. Multimedia* **2019**, *21*, 3106–3121. [[CrossRef](#)]
16. Zhang, H.J.; Gong, Y.H.; Low, C.Y.; Smoliar, S.W. Image retrieval based on color features: An evaluation study. In Proceedings of the SPIE, Digital Image Storage and Archiving Systems, Philadelphia, PA, USA, 21 November 1995; pp. 212–220.
17. Haralick, R.M.; Shanmugam, K.; Dinstein, I. Textural Features for Image Classification. *IEEE Trans. Syst. Man. Cybern.* **1973**, *SMC-3*, 610–621. [[CrossRef](#)]
18. Ferreira, C. Gene Expression Programming: A New Adaptive Algorithm for Solving Problems. *Complex Syst.* **2001**, *13*, 87–129.
19. Zhou, C.; Xiao, W.M.; Tirpak, T.M.; Nelson, P.C. Evolving accurate and compact classification rules with gene expression programming. *IEEE Trans. Evol. Comput.* **2003**, *7*, 519–531. [[CrossRef](#)]
20. Choi, J.; Kim, M. Single Image Super-Resolution Using Global Regression Based on Multiple Local Linear Mappings. *IEEE Trans. Image Process.* **2017**, *26*, 1300–1314. [[CrossRef](#)] [[PubMed](#)]
21. Bevilacqua, M.; Roumy, A.; Guillemot, C.; Morel, M.L.A. Low-Complexity Single-Image Super-Resolution based on Nonnegative Neighbor Embedding. In Proceedings of the British Machine Vision Conference (BMVC), Guildford, Surrey, UK, 3–7 September 2012; pp. 1–10.
22. Zeyde, R.; Elad, M.; Protter, M. On Single Image Scale-Up Using Sparse-Representations. In *Proceedings of the International Conference on Curves and Surfaces*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 711–730.
23. Martin, D.; Fowlkes, C.; Tal, D.; Malik, J. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In Proceedings of the Eighth IEEE International Conference on Computer Vision (ICCV), Vancouver, BC, Canada, 7–14 July 2001; pp. 416–423.

24. Huang, J.B.; Singh, A.; Ahuja, N. Single image super-resolution from transformed self-exemplars. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 5197–5206.
25. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1026–1034.



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).