Hindawi
Complexity
Volume 2020, Article ID 6180317, 13 pages
https://doi.org/10.1155/2020/6180317

WILEY | Hindawi

Research Article

Image Recognition Based on Multiscale Pooling Deep Convolution Neural Networks

Haitao Sang,[1] **Li Xiang**,[1] **Shifeng Chen**,[1] **Bo Chen**,[1] **and Li Yan**[2]

[1]*College of Information Engineering, Lingnan Normal University, Zhanjiang 524048, China*
[2]*College of Science, Guangdong University of Petrochemical Technology, Maoming 525000, China*

Correspondence should be addressed to Li Xiang; xiangl@lingnan.edu.cn

Received 5 June 2020; Revised 16 July 2020; Accepted 20 July 2020; Published 7 September 2020

Guest Editor: Kailong Liu

Depth neural network (DNN) has become a research hotspot in the field of image recognition. Developing a suitable solution to introduce effective operations and layers into DNN model is of great significance to improve the performance of image and video recognition. To achieve this, through making full use of block information of different sizes and scales in the image, a multiscale pooling deep convolution neural network model is designed in this paper. No matter how large the feature map is, multiscale sampling layer will output three fixed-size character matrices. Experimental results demonstrate that this method greatly improves the performance of the current single training image, which is suitable for solving the image generation, style migration, image editing, and other issues. It provides an effective solution for further industrial practice in the fields of medical image, remote sensing, and satellite imaging.

1. Introduction

Edge computing extends computing, network, storage and other capabilities to the edge side of the network near the Internet of Things devices, while the artificial intelligence technology represented by deep learning enables each edge computing node to have the ability of computing and decision making, allowing some complex intelligent applications to be processed at the local edge, meeting the needs of agile connection, real-time data optimization business, application intelligence, security, and privacy protection. Intelligent edge computing collects data and analyses computing by means of the edge device of the Internet of Things so as to realize the flow of intelligence between the cloud and the edge. Now that new demands on artificial intelligence algorithms, terminals, and chips have to be met, it appeals more and more artificial intelligence (AI) enterprises [1–3]. However, most current edge devices today cannot support deep learning applications in a low-latency, low-power, high-precision manner due to resource constraints; for example, deep learning model inference requires a large amount of computational resources.

Deep neural networks, like many other machine learning models, can be divided into two stages of training and reasoning. The training phase learns the parameters in the model according to data (for the neural network, it is mainly the weight in the network). The inference phase enters the new data into the model and calculates the results. Over-parameterization refers to the training phase. The network needs a large number of parameters to capture the small information in the data. And once the training has reached the reasoning stage, it does not need so many parameters. Based on this assumption, the model can be simplified before deployment. However, deep neural network model that exists in the process of removing specific fuzzy kernel and noise usually needs to set up different training models according to different noise levels, which does not only lack flexibility, but also cannot cope with more general image recognition tasks. Therefore, how to use DNN model flexibly in image recognition task has more important research value [4].

Although the purpose of neural network is to solve the general machine learning problem, domain knowledge also plays an important role in the design of depth model.

Among the applications related to image and video, the most successful one is the deep convolution network, whose design is only used in the special structure of image. Two of the most important operations, convolution and pooling, come from domain knowledge related to images. How to introduce new effective operation and layer in the depth model through the knowledge of research field is of great significance to improve the performance of image and video recognition. For example, the pool layer brings about local translation invariance, and the deformation pool layer proposed in [5–7] can better describe the geometric deformation of various parts of the object. At present, it can be further extended to achieve scale invariance, rotation invariance, and robustness to occlusion.

In this paper, we put forward Multiscale Pooling Deep Convolution Neural Networks model that aims to train a set of fast and effective schemes. Rather than learning MAP inference-guided discriminative models, we instead adopt plain DNN to learn the denoisers, taking advantage of recent progress in DNN as well as the merit of GPU computation. Meanwhile providing good performance for image recognition, the learned set of multiscale pooling is plugged in a model-based optimization method to tackle various problems.

The contribution of this work is summarized as follows:

(i) We trained a set of fast and effective Multiscale Pooling Deep Convolution Neural Networks (MSDNN) model. Using the core idea of pyramid method for reference, the powerful multiscale expression ability is applied to the deep level of network

(ii) Extensive experiments on classical feature extraction problems, including superresolution and deblurring, have demonstrated great improvement in the performance of the current single training image and can be applied to image generation, style migration, image editing, and other issues.

2. Related Work

2.1. Feature Learning. The biggest difference between traditional pattern recognition method and deep learning lies in its characteristics; the feature is automatically learned from big data, rather than using manual design. Good features can improve the performance of pattern recognition system [8–10]. In the past few decades, in various applications of pattern recognition, the characteristics of manual design have been in domination level. Manual design mainly depends on the prior knowledge of designers, so it is difficult to take advantages of the advantages of big data. Due to the dependence on manual parameter adjustment, the number of parameters allowed in the design of the feature is very limited. Deep learning can automatically learn representations of features from big data and can contain thousands of parameters [11–16]. It takes five to ten years to design effective features by hand, and deep learning can quickly learn new and effective feature representations from training data for new applications.

A pattern recognition system includes two parts: feature and classifier. In traditional methods, the optimization of features and classifiers is separated. Under the framework of neural network, feature representation and classifier are jointly optimized, which can give full play to the performance of joint cooperation [17–21].

In 2012, Hinton participated in the ImageNet competition and adopted the convolution network model, which contains 60 million parameters learned from millions of samples [22]. The feature representation learned from ImageNet has very strong generalization ability and can be successfully applied to other datasets and tasks, such as object detection, tracking, and inspection. Another famous competition in the field of computer vision is PSACAL VOC. However, its training set is small and is not suitable for training deep learning models. Some scholars use the feature representation learned from ImageNet for physical examination on PSACAL VOC; the detection rate increased by 20% [23].

Since feature learning is so important, what are good features? In an image, various complex factors are often combined in a nonlinear way. For example, the face image contains a variety of information such as identity, posture, age, expression, light, and so on. The key to deep learning is to successfully separate these factors through multilevel nonlinear mapping; for example, in the last hidden layer of the depth model, different neurons represent different factors. If you take this hidden layer as a feature representation, face recognition, pose estimation, expression recognition, and age estimation will become very simple, because the various factors become a single linear relationship, no longer interfere with each other.

2.2. ImageNet Image Classification. The most important development of deep learning in object recognition is image classification task in ImageNet ILSVRC 3 challenge. The lowest error rate of traditional computer vision method in this test set is 26.172%. In 2012, Hinton's team used convolutional networks to reduce the error rate to 15.315%. This network structure is called AlexNet [24]. Compared with the traditional convolution network, it has three differences. First of all, AlexNet adopts dropout training strategy. During the training process, some neurons in the input layer and the middle layer are randomly set to zero. This simulates the situation that noise interferes with input data so that some neurons fail to detect some visual patterns. Dropout makes the training process converge more slowly, but the resulting network model is more robust. Secondly, AlexNet uses the rectifier linear element as the nonlinear excitation function. This not only greatly reduces the computational complexity, but also makes the output of neurons sparse and more robust to various interferences. Thirdly, AlexNet generates more training samples and reduces overfitting by mapping the image of training samples and adding random translation disturbance.

In the ImageNet ILSVRC 2013 competition, the top 20 groups use deep learning techniques. The winner is Rob Fergus's research group of New York University. The depth model adopted is convolutional network, and the network structure has been further optimized. The error rate is

11.197%. Its model is called clarifai [25]. In the ILSVRC 2014 competition, the winner GoogLeNet [25] reduced the error rate to 6.656%. The outstanding feature of GoogLeNet is that it greatly increases the depth of convolution network, which is more than 20 layers, which was unimaginable before. The deep network structure makes the backpropagation of prediction error difficult, because the prediction error is transmitted from the top layer to the bottom layer, and the error transmitted to the bottom layer is very small, so it is difficult to drive the updating of the bottom-layer parameters. GoogLeNet's strategy is to add the monitoring signal directly to multiple middle layers, which means that the feature representation of middle layer and bottom layer should also be able to accurately classify the training data. In the last competition of ILSVRC 2017, the "DPN dual-channel network + basic aggregation" deep learning model proposed by 360 Artificial Intelligence Research Institute and the National University of Singapore (NUS) team achieved the lowest positioning error rate of 0.062263 and 0.061941, respectively, setting a world record. Among them, the DPN-92 costs about 15% fewer parameters than ResNeXt-101, while the DPN-98 costs about 26% fewer parameters than ResNeXt-101. In addition, the DPN-92 consumes about 19% less FLOPs than ResNeXt-101, and the DPN-98 consumes about 25% less FLOPs than ResNeXt-101. How to train the deep network model effectively is still an important research topic in the future.

2.3. Face Recognition. Another important breakthrough of deep learning in object recognition is face recognition. The biggest challenge of face recognition is how to distinguish the intraclass changes caused by light, posture, expression, and other factors from the interclass changes caused by different identities. The distribution of these two kinds of changes is nonlinear and extremely complex, which cannot be effectively distinguished by the traditional linear model. The purpose of deep learning is to get new feature representation through multilayer nonlinear transformation. These new features need to remove as many intraclass changes as possible, while retaining interclass changes.

Face recognition includes two tasks: face confirmation and face recognition. Face confirmation is to judge whether two face photos belong to the same person, which is a binary classification problem. The accuracy of random guessing is 50%. Face recognition is to divide a face image into one of N categories, which are defined by the identity of the face. This is a multiclassification problem, which is more challenging. The difficulty increases with the number of categories. The accuracy of random guess is $1/n$. Both tasks can learn the facial expressions through deep models.

Using convolution network to learn face features, literature [26–28] used face recognition task as a supervisory signal; the recognition rate is 92.52% on LFW. Although this result is lower than the follow-up deep learning method, it also exceeds most of the nondeep learning algorithms. Because face recognition has two classified problems, the efficiency of using it to learn face features is relatively low, and it is easy to have fitting on the training set. Face recognition is a more challenging multiclassification problem, which is not easy to be fitted, and it is more suitable to learn face features through depth model [29–31]. On the other hand, in face recognition, each pair of training samples is manually labelled into one of two categories, which contains less information. In face recognition, each training sample is labelled as one of N classes, which has a large amount of information.

Some people think that the success of deep learning is due to the use of complex modules with a large number of parameters [32]. In fact, it is far from so simple to fit the dataset. For example, the success of DeepID2+ lies in its many important and interesting characteristic [33]: its top neurons respond moderately sparsely; it has strong selectivity to face identity and various face attributes and strong robustness to local occlusion. In previous studies, in order to get these attributes, we often need to add various display constraints to the model. DeepID2+ has these attributes automatically through large-scale learning, and the theoretical analysis behind it is worth further study in the future.

2.4. Deep Learning for Video Analysis. The application of deep learning in video classification is still in its infancy; there is still a lot of work to be done in the future. The depth model that can be learned from ImageNet can be used to describe the static image features of video. The difficulty is how to describe the dynamic features. In the past, the description of dynamic features in visual research methods often depends on optical flow estimation, key point tracking, and dynamic texture. How to embody this information in depth model is a difficulty. The most direct way is to treat video as a three-dimensional image and apply convolution network directly [34–36]; three-dimensional filters are learned at each level. But this idea obviously does not take into account the differences of time and space dimensions. Another simple but more effective way is to calculate the spatial field distribution of optical flow field or other dynamic characteristics by preprocessing, as an input channel of convolution network [37–39]. There are also research works using deep autoencoder to extract dynamic texture in a nonlinear way [38]. In the latest research work [40], long short-term memory (LSTM) has been widely concerned, which can capture long-term dependence and complex dynamic modelling in video.

In 2018, Ulyanov et al. proposed a depth image prior (Deep Image Prior (DIP)) model [1], which considers that the neural network structure is a priori that does not require learning and pretraining to a large number of datasets; only learning network superparameters adaptively can realize image conversion and it is verified in the tasks of denoising, superresolution, and filling. After dip was proposed, it was successively used in image decomposition-related tasks [41] and blind deconvolution [42]. Double-Dip [43] is an unsupervised layer decomposition of a single image based on a coupled dip network, which is used for image segmentation (foreground layer and background layer), image defogging, watermarking, and other issues. Ren et al. [42] proposed a blind deconvolution framework based on depth

prior—SelfDeblur, using dip and full convolution network, respectively, and fuzzy kernel.

Compared with the image processing, the sound signal processing is another direction of successful application in deep learning, especially speech recognition, which many large companies are going for. Unlike images, sound signals are one-dimensional sequence data, and although they can be converted to two-dimensional spectrum by frequency domain conversion algorithms such as FFT, their two dimensions also have specific meanings (vertical axis represents frequency; horizontal axis represents timeframe) and cannot be processed directly in the form of an image, requiring a specific processing method in a specific area. Deep neural network has the ability to extract features automatically, which can jointly optimize the separation of feature representation and classification model, especially after 2016. Deep learning has made great progress in the field of speech recognition; for example, RNN, CRN, and other structures have made a breakthrough in speech recognition performance [44–46]. With the research developing and the hardware updating, some end-to-end models have achieved the performance in the state of the art, for example, as the previously mentioned, it is sequence-to-sequence model (CTC). As speech recognition performance continues to improve, industry has produced many applications, such as virtual assistants: Google Home, Amazon Alexa, and Microsoft Cortana.

Although deep learning has achieved great success in practice, moreover, the characteristics (such as sparsity, selectivity, and robustness to occlusion [29]) of the depth model obtained by big data training are striking, but the theoretical analysis behind it still needs to be completed.

3. Method

In the field of target detection, the manual extraction of characteristics has been replaced by the conventional multiple layer NN, which has saved a great deal of costs of background research. But as the number of network layers increases, the corresponding training methods become lacking. Moreover, the conventional NNs are sensitive to the translation and scale change of the target, which leads to the relatively low detection rate of the target. At present, CNN can realize the simulation of the large-scale neural network by the following three techniques: local receptive field, weight sharing, and pooled sampling. Therefore, we put forward a Multiscale Pooling Deep Convolution Neural Networks method, to overcome the restriction during the image detection that the input image must be of a fixed size.

3.1. The Basic Structure of Convolutional Neural Network. A multilayer NN has too many parameters and is difficult to train. To overcome this limitation, the CNN emerges, which is based on ANN and includes convolution and sampling operations. The characteristics extracted by CNN are spatial invariant to certain degree, which makes CNN more suitable for the target detection of an image. Although there are many kinds of CNN, the basic structure is largely the same (Figure 1): the input layer I, the convolution layer C, the sampling layer S, the output layer O, and sometimes the full-connection layer F.

The main purpose of convolution layer is to realize local feature perception through convolution operation and then realize the same kind of feature extraction through weight sharing and realize different types of feature extraction by using different convolution kernels. They should be referred to as follows:

$$X_j^l = f\left(\sum_{i\in M_j} X_i^{l-1} * K_{i,j}^l + b_j^l\right), \tag{1}$$

where X_j^l represents the convolutional output of the jth characteristic map of the lth layer; $f(\ )$ represents the activation function; l represents the number of the layer, K is the kernel; M_i represents one choice of the input characteristic map; and b is an offset.

The sampling layer is an important layer of convolution neural network, which performs convolution to get the average or maximum value of the local space of the feature map. They should be referred to as follows:

$$X_j^l = f\left(\beta_j^l \mathrm{down}\left(X_j^{l-1}\right) + b_j^l\right), \tag{2}$$

where down$(\cdot)$ is a sampling function obtaining the average or maximum of a rectangle area.

The full-connection layer is the penultimate layer of convolution neural network, which means that the input feature map and the output feature map adopt the full-connection method. Its function is to reduce the dimension effectively and to facilitate the classification processing of the output layer. They should be referred to as follows:

$$X_j^l = f\left(\sum_{i=1}^{N_{\mathrm{in}}} a_{i,j}\left(X_j^{l-1} * K_i^l\right) + b_j^l\right), \tag{3}$$

where the constraint $\sum_i \alpha_{ij} = 1$ and $0 \le \alpha_{ij} \le 1$ must be satisfied.

The constraint to the variable a_{ij} can be strengthened by turning the variable a_{ij} into an unconstrained Softmax function of the implicit weight c_{ij}, They should be referred to as follows:

$$a_{ij} = \frac{\exp\left(c_{ij}\right)}{\sum_K \exp\left(c_{ij}\right)}. \tag{4}$$

For a given j, every group of weight values c_{ij} is independent of those in the other groups. We therefore can omit the subscript j, for the sake of convenience. In other words, one needs only to consider updating a map, because the updating of the other maps is the same process. The only difference is the index j of the map. The details of the regression process about Softmax are not explained here. It should be noted that the input to the FCL needs to have a fixed dimensionality for an easier training of the BP algorithm, which necessitates the consistency between the size of the input image and the size of the training network. This requirement represents one shortcoming of the network

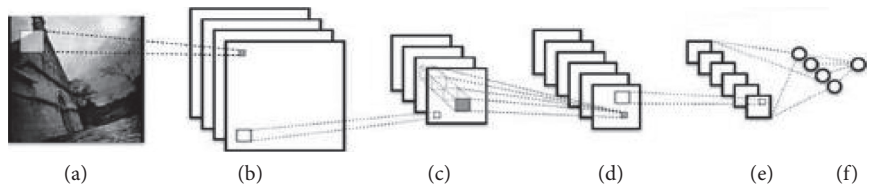


Figure 1: The basic structure of CNN. (a) Input layer I. (b) Layer C1. (c) Layer S1. (d) Layer C2. (e) Layer S2. (f) Output.

structure. Therefore, to solve the problem that the size of the input image must be fixed, the multiscale sampling method is to be used.

3.2. The Multiscale Sampling

3.2.1. Shortcomings of the Conventional CNN. The conventional CNN has a fixed window size of the input image, which restricts the size of the input image and length-to-width ratio. When the input image is of arbitrary size, one often has to first fix the image to a size suitable for the training network by clipping the image or deformation along the horizontal and vertical directions. This manual aligning method cannot preserve the size and length-to-width ratio of the original image and cannot guarantee that the clipped image subblocks encompass the whole image, which leads to the loss of some important information and geometrical distortion due to rotation and stretching of the image. When the image size changes to a large extent, the whole network has to be trained again to adapt to the new image, even if the network was trained by images of a given input size. Otherwise the accuracy would reduce of detecting the whole image or the subblocks of the other arbitrary sizes. The fixed input dimensionality greatly restricts the development of CNN in multiscale image detection.

The convolutional and sampling layers extract image characteristics by using sliding windows. The FCL, as the hidden intermediate between the sampling and output layers, plays a pivotal role in the whole network structure. In fact, the convolutional layer can generate characteristic maps of different sizes and ratios without inputting images of fixed size. On the contrary, according to the definition of CNN, the input to FCL needs to be images of fixed size. The reason is that only the FCL has a fixed input dimensionality, which further fixes the dimensionality of the parameters used in the training and thus allows for the use of the BP algorithm for the training.

To solve the problem, one can use multiscale sampling to integrate the deep characteristics generated by the previous convolutional–sampling layers to a characteristic expression of fixed length and then send it to the FCL. In this way, one needs not to clip images of different sizes and length-to-width ratio before sending them to the network. At the previous layer of the FCL, namely, the last sampling layer, the multiscale sampling is used to fix the images' sizes so that they can be input to the FCL. Figure 2 shows how the CNN structure is changed by introducing the MSL.

Now, one can see that the MSL integrates the image characteristics between the convolutional–pooled layers and the FCL. The integration at deep layers is reasonable because it conforms well with bionic topology: human learning starts from the overall classification but not from the postclipping parts; thus to detect a target, the brain needs not to first process the target and adapt to the fixed size. Some authors proposed the solution of inputting images of different sizes to CNN, but used images of different sizes for both training and testing. They found no significant improvement of the detection rate. For MSDNN, the training and testing are performed on the same network.

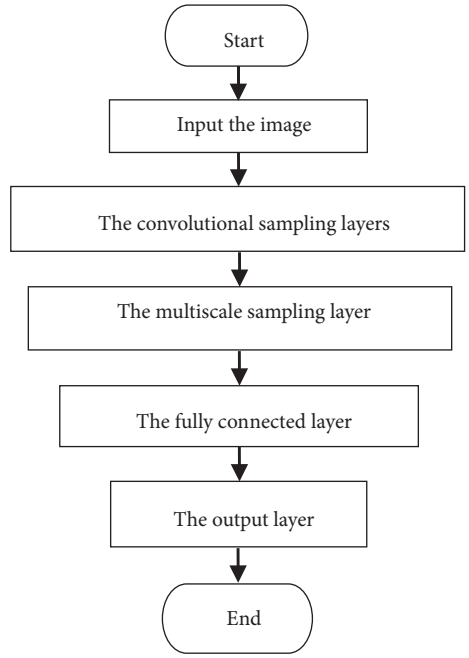


Figure 2: The MSDNN structure.

The idea of multiscale sampling was primarily inspired by the method of pyramid, which preserves the spatial information of the local parts during the integration. The pioneers applied the pyramid method to all the levels of CNN and extracted multiscale image characteristics by sliding windows. These image characteristics were then used to train or test the networks of the respective sizes. This actually enriched the scale of the input image and extended the training set, but the training became more difficult with more network parameters that can easily lead to the overfitting problem. The MSL design makes it unnecessary to use different networks to train and test images of different size. The MSL does not simply use the pyramid method to extract the image's multiscale characteristics, but applies the core idea of the pyramid method, namely, multiscale expression, to the deep layers of the network. These enable not only the

training of NN with images of different sizes, which enriches the characteristic expressions, but also the output of characteristic vectors of the fixed size.

3.2.2. Methods of Multiscale Sampling. Multiscale expression is applied to the deep layers of NN. We propose to replace the normal sampling with the multiscale sampling at the final sampling layer, where "multiscale" lies in the use of multiple sampling sizes and strides. Figure 3 shows no matter how large the previous characteristic map, the MSL outputs three characteristic matrices with sizes $1 \times m$, $4 \times m$, and $9 \times m$, where m is the number of the previous convolutional layer's characteristic maps. The three characteristic matrices are concatenated in the column-major order to form a column vector with fixed dimensionality $14 \times m \times 1$. The column vector is the input to the FCL.

The sizes of these sampling windows are matched with those of the input images. That is, the sampling size and sampling stride change as the size of the input changes. In this way, the number of sampling windows, namely, the size of the output characteristic column vector, becomes fixed, no matter how large the input image is. This makes it easy to send the vector to FCL. For example, if the size of the input characteristic map is $r \times s$ and we use the maximum sampling method with three sampling sizes and strides, then the computation formulas are as follows:

$$\text{size} = r \times s, \lceil (r/2) \times (s/2) \rceil, \lceil (r/3) \times (s/3) \rceil, \tag{5}$$

$$\begin{aligned} \text{stride} &= r \times s, \lceil (r/2) \times (s/2) \rceil, \lceil (r/3) \times (s/3) \rceil, \\ y_j^1 &= \max(x_i), \quad i \in \mathfrak{R}_{r \times s}, j \in \mathfrak{R}_{1 \times 1}, \\ y_j^2 &= \max(x_i), \quad i \in \mathfrak{R}_{[(r/2) \times (s/2)]}, j \in \mathfrak{R}_{2 \times 2}, \end{aligned} \tag{6}$$

$$y_j^3 = \max(x_i), \quad i \in \mathfrak{R}_{[(r/3) \times (s/3)]}, j \in \mathfrak{R}_{3 \times 3}, \tag{7}$$

where y_j^1, y_j^2, and y_j^3 represent the output of every characteristic map after the multiscale maximum sampling. These operations finally obtain three output characteristic matrices with fixed sizes: $1 \times 1 \times m$, $2 \times 2 \times m$, and $3 \times 3 \times m$, which can be converted into characteristic matrices $1 \times m$, $4 \times m$, and $9 \times m$ by unfolding in the column-major order. Finally, these characteristic matrices are concatenated in sequence, becoming a characteristic column vector of a fixed size $14 \times m \times 1$. Take two different size input images as an example (Figure 4). The sizes of the characteristic maps are 16×16 and 13×9, respectively. The target is a 3×3 output matrix. In Figure 4(a), the sampling size is 6×6 and sampling stride is 5×5. By (3) and (4), one obtains the maximum value of the 6×6 area and uses it as the output, which finally obtains a 3×3 output characteristic matrix. In Figure 4(b), the sampling size is 5×3 and sampling stride is 4×3. By (3) and (4), one obtains the maximum value of the 5×3 area and uses it as the output, which finally obtains a 3×3 output characteristic matrix.

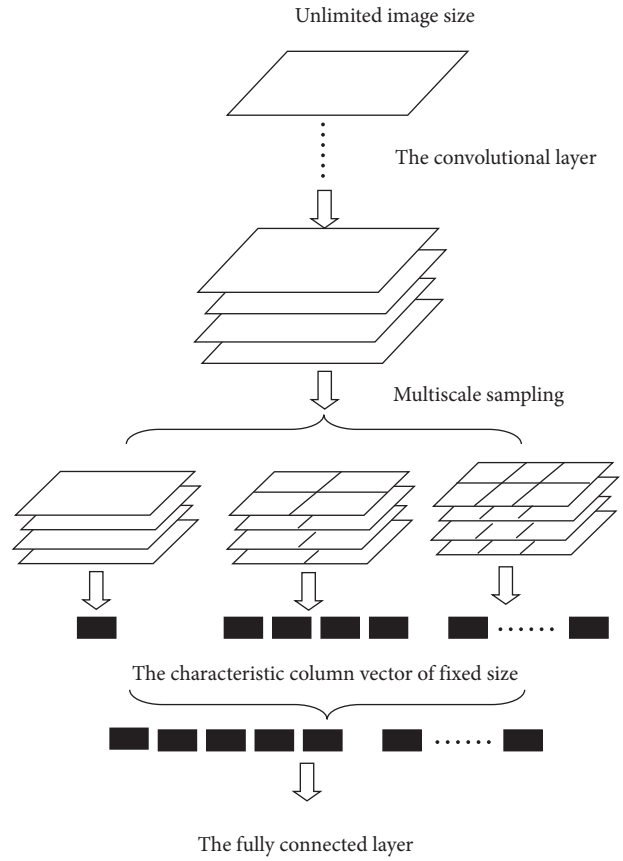


Figure 3: Illustration of the multiscale sampling.

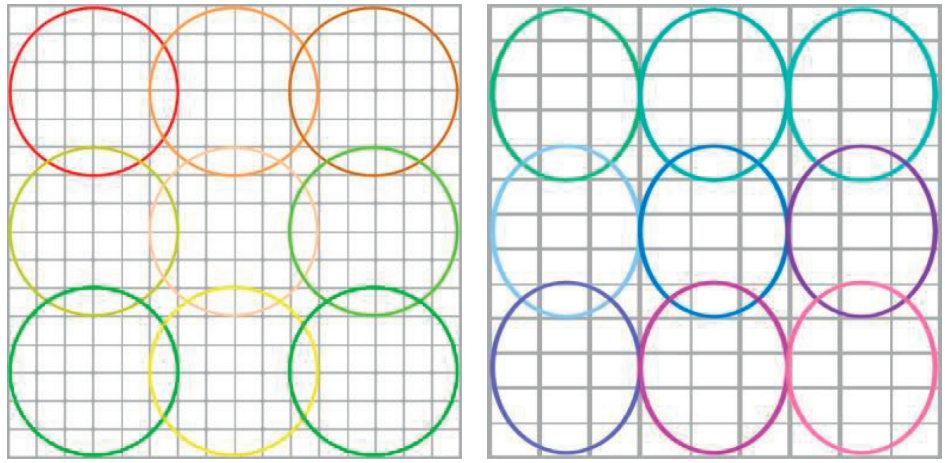
3.2.3. The Fully Connected Layer with Multilayer Characteristic Expression. The FCL, as the hidden intermediate between the sampling and output layers, plays a pivotal role in the whole network structure. Here the neurons of the FCL simultaneously, indirectly, and fully connect to the characteristic maps next to the convolutional layers so that the FCL can receive multiscale and multilevel feature representation input.

3.3. The Topological Structure of the Overall Network. Figure 5 shows the topological structure of the MSDNN put forward in this paper. Due to the page limit, taking a 64×64 input image as an example, we briefly introduce the main parameters and implementation methods of every network layer.

The input layer: the preprocessed grayscale image X_{in}.

The convolutional layer Cl : the layer consists of 20 different characteristic maps X_j^{C1}. Every characteristic map is convoluted with twenty 5×5 convolution kernels, respectively. The result is offset by b_j^{C1} and then used as the independent variable of the activation function ReLU to obtain the characteristic map X_j^{C1}. They should be referred to as follows:

$$\begin{aligned} X_j^{C1} &= \text{ReLU}\left(X_{\text{in}} \otimes K_{ij}^{C1} + b_j^{C1}\right) \\ &= \max\left(0, X_{in} \otimes K_{ij}^{C1} + b_j^{C1}\right), \quad i = 1,\ j = 1, 2, \ldots, 20, \end{aligned} \tag{8}$$



(a) (b)

Figure 4: Multiscale sampling with variable sampling size and stride. (a) 16×16 multiscale sampling. (b) 13×9 multiscale sampling.

where $\otimes$ represents the convolution operation with stride 1; the activation function is defined to be $\mathrm{ReLU}(x) = \max(0, x)$; and the size of the obtained X_j^C is 60×60.

The sampling layer S1 : perform statistical computation on the results obtained by the convolutional later C1, by using the maximum value sampling. After the sampling, the horizontal and vertical spatial resolutions become half of the original resolutions. The size is 30×30.

The convolutional layer C2 : by convoluting with convolutional kernels of size 3×3, the 20 characteristic maps X_j^{S1} are extended to 40 characteristic map X_j^{C2} of size 28×28:

$$X_j^{C2} = \max\left(0, \sum X_i^{S1} \otimes K_{ij}^{C2} + b_j^{C2}\right), \quad i = 1, 2, \ldots, 20, \; j = 1, 2, \ldots, 40. \tag{9}$$

The sampling layer S2 : perform the maximum value sampling with a sampling size 2 and without overlapping of the sampling area. After the sampling, the horizontal and vertical spatial resolutions become half of the original resolutions. The size is 14×14.

The convolutional layer C3 : by convoluting with convolutional kernels of size 3×3, the 40 characteristic maps X_j^{S2} are extended to 60 characteristic map X_j^{C3} of size 12×12:

$$X_j^{C3} = \max\left(0, \sum X_i^{S2} \otimes K_{ij}^{C3} + b_j^{C3}\right), \quad i = 1, 2, \ldots, 40, \; j = 1, 2, \ldots, 60. \tag{10}$$

The sampling layer S3 : perform the maximum value sampling with a sampling size 2 and without overlapping of the sampling area. After the sampling, the horizontal and vertical spatial resolutions become half of the original resolutions. The size is 6×6.

The convolutional layer C4 : by convoluting with convolutional kernels of size 3×3, the 60 characteristic maps X_j^{S3} are extended to 80 characteristic map X_j^{C4} of size 4×4:

$$X_j^{C4} = \max\left(0, \sum X_i^{S3} \otimes K_{ij}^{C4} + b_j^{C4}\right), \quad i = 1, 2, \ldots, 60, \; j = 1, 2, \ldots, 80. \tag{11}$$

The multiscale sampling layer MS4 : perform the maximum value sampling with three different scales on the 80 characteristic maps X_j^{C4}. The three sampling sizes and strides are as follows:

$$\begin{aligned} \text{size} &= 4 \times 4, 2 \times 2, 2 \times 2, \\ \text{stride} &= 4 \times 4, 2 \times 2, 1 \times 1. \end{aligned} \tag{12}$$

This obtains three output characteristic matrices of the fixed sizes $1 \times 1 \times 80$, $2 \times 2 \times 80$, and $3 \times 3 \times 80$. By extending the matrices in the column-major order, one obtains characteristic column vectors of sizes 1×80, 4×80, and 9×80. Finally, the vectors are concatenated in sequence to form a characteristic column vector x^{MP4} with a fixed size $14 \times 80 = 1120 \times 1$. For the input images of the other sizes, the sampling size and stride change adaptively, but the dimensionality of the final characteristic column vector does not change.

The multiscale sampling layer MS3 : perform the maximum value sampling with three different scales on the 60 characteristic map X_j^{C3}. A characteristic column vector x^{MS3} with a fixed size $14 \times 60 = 840 \times 1$ is formed. The three sampling sizes and strides are as follows:

$$\begin{aligned} \text{size} &= 12 \times 12, 6 \times 6, 4 \times 4, \\ \text{stride} &= 12 \times 12, 6 \times 6, 4 \times 4. \end{aligned} \tag{13}$$

The multiscale sampling layer MS2 :

perform the maximum value sampling with three different scales on the 40 characteristic maps X_j^{C2}. A characteristic column vector x^{MS2} with a fixed size $14 \times 40 = 560 \times 1$ is formed. The three sampling sizes and strides are as follows:

$$\begin{aligned} \text{size} &= 28 \times 28, 14 \times 14, 10 \times 10, \\ \text{stride} &= 28 \times 28, 14 \times 14, 9 \times 9. \end{aligned} \tag{14}$$

The fully connected layer FC5 : the output expression column vector x^{FC5}, with the size 400×1, is obtained in the FCL from the three characteristic column vector obtained at the respective MSLs:

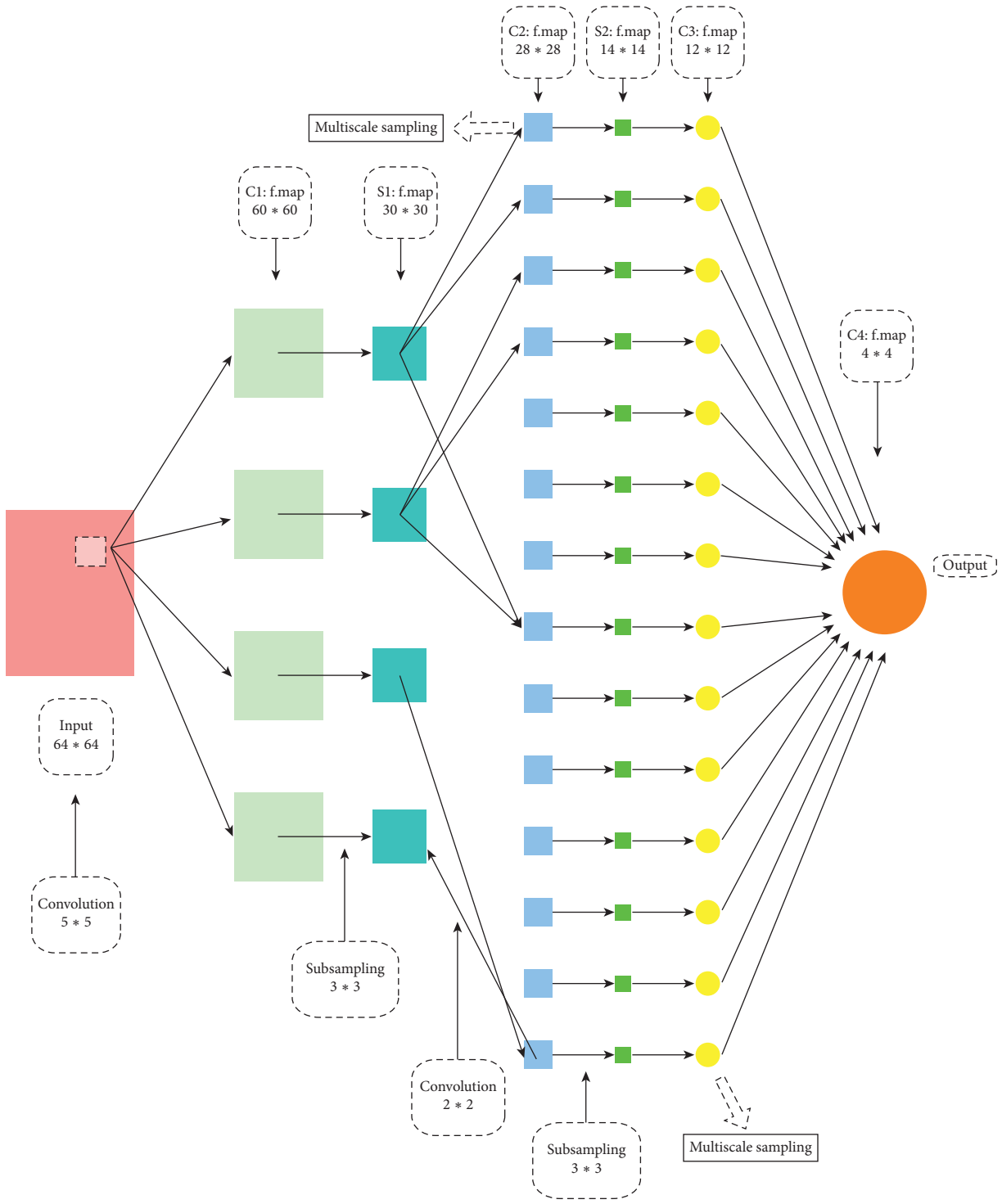


Figure 5: The MSDNN topology graph.

$$X^{FC5} = \max\left(0, W_2 \cdot x^{MS2} + W_3 \cdot x^{MS3} + W_4 \cdot x^{MS4} + b^{FC5}\right), \tag{15}$$

where W_2, W_3, and W_4 are the respective weight matrices that connect the three column vectors to the FCL; b^{FC5} is the offset matrix used in the FCL.

The output layer：a labor vector y_{output} is obtained from the output expression column vector x^{FC5} obtained in FC5.

3.4. Methods of Training and Learning. Before the training and testing, one should first build structures according to the degree of difficulty of the detection, the actual conditions of

the database, etc. In general, the larger the database and the more complex the classification are, the deeper the structure required. One then needs to do some preprocession on the database so that the training would be smoother. For small databases, one can expand the data capacity to reduce overfitting. The number of training samples can be increased by operations such as translation and random clipping.

Here the method of capacity expansion is used. The dataset is from the Internet, containing 1980 single human face images of different sizes, skin colors, directions, and positions. The same person may have different expressions; some are shielded; some with mustache; some with glasses, etc. To improve robustness of the network and increase multiplicity of the samples, we perform some transformation to a part of images in the initial human face image sample set, such as horizontal flipping, $\pm 20^{\circ}$ displacement along the horizontal and vertical directions, $\pm 20^{\circ}$ rotation, and the reduction of contrast. Of course, one can combine different transformations and then clip the corresponding image subblocks. In this way, tens of sample images can be obtained from a single image. A part of the human face image samples is illustrated in Figure 6.

The non–face image dataset can be a set of non–face images. In fact, any randomly clipped image can be used as a non–face sample. But it is impossible to train a network that can recognize all kinds of non–face samples. One can use the iterative bootstrapping program to collect non–face images and use methods including the iterative retraining system to update the training set.

The methods for training MSDNNs can be divided into the forward propagation stage and the backward propagation stage. They are not introduced here because they are similar to the BP algorithm of the conventional NN.

4. Experiments

4.1. Validity Analysis of the Multiscale Sampling. The innovation point of this paper is multiscale sampling, which improves the structure of the conventional CNN for the detection of human face. In the following, we perform experiments to compare their performance. To accelerate the speed of training and testing, we use the popular Caffe deep learning framework to realize GPU computation. The advantage of Caffe is that programming is not needed. Once the network structure is designed, Caffe can automatically help the user to train the NN. However, currently, there are no frameworks that accept input images of variable sizes. Therefore, the experiments focus on training the NN with input images of three fixed sizes.

To verify effectiveness of multiscale sampling, we use a single-layer cascade network structure and connect it to the final multiscale layer MS4 (Figure 5). Two conditions are analyzed in the following.

4.1.1. The Training of MSDNN with a Single Input Size. Take a 64×64 input image as example. The three structures are as follows:

Figure 6: A part of face images.

Table 1: Temperature and wildlife count in the three areas covered by the study.

Methods	Detection rates (%)
The ordinary CNN	97.7
One sampling size	95.4
Two sampling sizes	97.5
Three sampling sizes	98.3

(1) Without multiscale sampling, i.e., the ordinary CNN structure

(2) One sampling size (1×80 characteristic matrix)

(3) Two sampling sizes (4×80, 9×80 characteristic matrices)

According to the final testing results obtained by averaging 100 times of experiments (Table 1), MSDNN has a higher detection rate than the ordinary CNN. As far as the number of characteristics is concerned, the dimensionalities of the ordinary CNN and the MSDNN are 1280 and 1120, respectively, which demonstrates that the improvement of the detection rate by MSDNN does not require the increase of the dimensionality. It is the change of the spatial deployment that brings about the improvement of target detection.

4.1.2. The Training of MSDNN with Multiple Input Sizes. Because the Caffe framework can only accept input of fixed size during one period of training, the trained network is applied to process images of arbitrary sizes in the testing stage. Taking into account the quality of the human face images in the library and the quality of hardware, we train the network with three input image sizes 64×64, 80×80, and 100×100. The images only differ in the resolution, sampling size, and sampling stride, but the other properties (the content, layout, and output characteristic column vector) are the same. In this way, one can first randomly choose any one of the three network sizes (64×64) to

Figure 7: The original image.

Figure 8: The binary image after skin-color segmentation.

Figure 9: The binary image after morphological operation.

complete one training period, at the end of which all the parameters are saved. Then, one switches to the other two network sizes to continue the training, until the termination condition is satisfied.

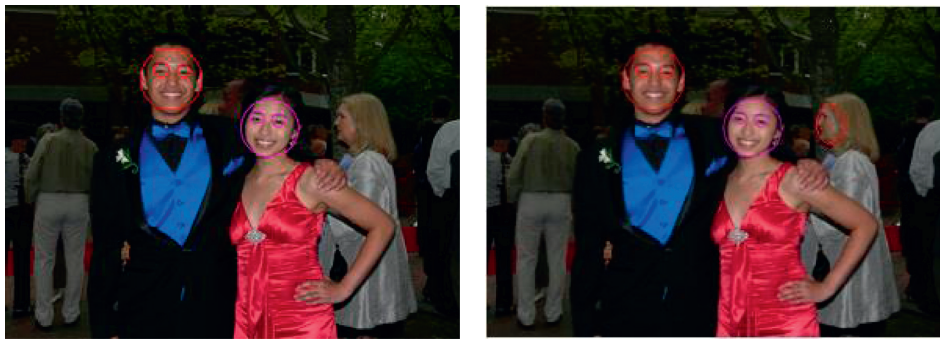
In the experiments, the detection rate of the multiscale sampling training with multiple input sizes is higher than that with a single input size. This is because the network is trained by the characteristics of images of different sizes and is thus more suitable for the real detection. The final detection rates of the networks trained by one, two, and three input sizes are 98.25%, 98.46%, and 98.59%, respectively.

4.2. Comparative Experiments. The experiments follow the scheme of first performing segmentation of skin color from the background color in the YCbCr space to remove the wide background area so that the computation speed can be improved and then using MSDNN classifier to screen the extracted skin-color area in order to determine whether or not a human face exists in the image, and if yes, labelling the human face.

We used MATLAB 2014 as the platform to validate the detection algorithm. The data were from two kinds of sources. The first kind of data was from Internet download or life photos, including human faces of different illumination, different complex background, and different sizes. The other data were some human face images from the CMU database. We performed the detection and analyzed the results both qualitatively and quantitatively.

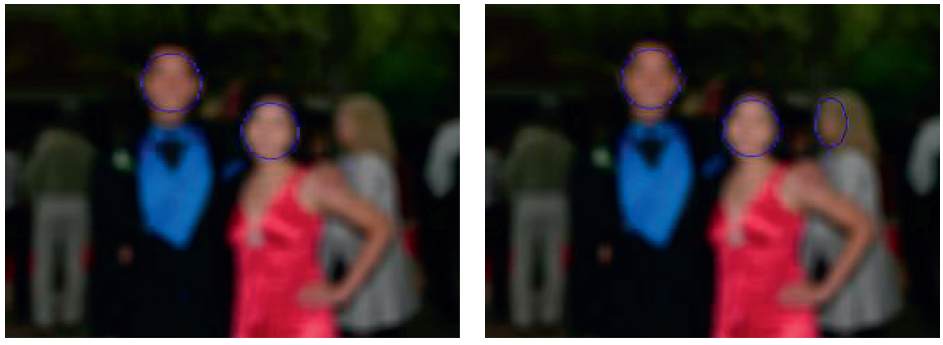
We performed the simulations by using MATLAB 2014 installed in an ordinary PC with Intel Core i5 CPU, 3.20 GHZ, 8G RAM. Figure 7 is an image of multiple front-view human faces obtained from the Internet. The human faces are different in the size and direction. Figures 8 and 9 present the result of skin-color segmentation by the Gaussian mixture model. Therefore, it was necessary to

Figure 10: The detection results.



(a) (b)

Figure 11: Detection on the original image by the CNN and MSDNN methods. (a) Results by the CNN method. (b) Results by the MSDNN method.



(a) (b)

Figure 12: Detection on the blurred image by the CNN and MSDNN methods. (a) Results by the CNN method. (b) Results by the MSDNN method.

locate human faces by using MSDNN. The characteristic matrices after the three scales sampling were: 1×80, 4×80, and 9×80. The concatenated characteristic column vector, which was sent to the FCL, was 1120×1. The results are presented in Figure 10.

To test the validity of the new method in detecting human face images with certain degree of rotation and with different resolutions, we performed human face detection on the segmented original image and the segmented fussy fixated image, respectively, by the conventional CNN and MSDNN methods. We used two sampling scales and obtained 1×80 and 4×80 characteristic matrices. The circled areas are the detected human faces (Figures 11 and 12).

From Figures 11 and 12, one sees that the conventional CNN can only detect two human faces, while the MSDNN can detect three faces: not only the two front-view faces, but also the accurate positioning of the side-view face. Even if the image is fuzzy and of low resolution, the MSDNN method can still locate the human face. These results further demonstrate the correctness of the methods put forward in this paper and that multiscale information plays an important role in image detection.

Table 2: Comparison of the three methods.

Methods	Nondetected	Detected	Detection rate (%)	Time (s)
CNN	3	47	94	32.3
PCA	8	42	84	30.1
ANN	5	45	90	57.5
The present method	2	48	96	36.6

To demonstrate the merits of the new method, we compared the new method with the conventional CNN, PCA, and ANN methods by performing experiments on eight face images (50 human face windows) taken from the CMU database. The detection rate and time by these methods are presented in Table 2.

From Table 2, one sees that the detection rate of the new method was higher than that of the CNN, PCA, and ANN methods. Moreover, the computation time was relatively smaller (hardware configuration : Intel Core i5-6500 CPU, 3.20 GHz × 4, 16G RAM, with GUP acceleration). Therefore, the performance of the new method is better than the previous methods.

5. Conclusions

In this paper, a set of fast and effective MSDNN image recognition methods is to be designed and trained. In particular, instead of establishing a multiscale sampling layer to replace the ordinary sampling layer in the network structure, the network structure is improved, and the generalization ability of sample collection is improved to a certain extent. In the future, the depth model is not a black box; it is closely related to the traditional computer vision system, the various layers of neural network through joint learning, and overall optimization, so that the performance has been greatly improved. Applications related to image recognition are also driving the rapid development of deep learning in all aspects of network structure, layer design, and training methods. It can be expected that, in the next few years, deep learning will enter a period of rapid development in theory, algorithms, and applications.

Data Availability

All the data have been included in the paper; therefore, there are no other data available.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Competitive Allocation of Special Funds for Science and Technology Innovation Strategy in Guangdong Province of China (no. 2018A06001), Zhanjiang Science and Technology Project (Grant no. 2019B01076), Lingnan Normal University Natural Science Research Project (no. ZL2004), Science and Technology Program of Maoming (no.2019397), Scientific Research Foundation for Talents Introduction, and Guangdong University of Petrochemical Technology (no. 519025).

References

[1] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9446–9454, Salt lake city, UT, USA, June 2018.

[2] G. Yossi, A. Shocher, and M. Irani, "Unsupervised image decomposition via coupled deep-image-priors," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, June 2019.

[3] A. Krull, T. Buchholz, and F. Jug, "Noise2Void-learning denoising from single noisy images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2129–2137, Long Beach, CA, USA, June 2019.

[4] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[5] K. Liu, K. Li, Q. Peng, Y. Guo, and L. Zhang, "Data-driven hybrid internal temperature estimation approach for battery thermal management," *Complexity*, vol. 2018, Article ID 9642892, 15 pages, 2018.

[6] J. Lehtinen and M. Jacob, "Noise2Noise: learning image restoration without clean data," in *Proceedings of the 35th International Conference on Machine Learning*, Stockholm, Sweden, July 2018.

[7] Z. Yang, K. Liu, J. Fan, Y. Guo, Q. Niu, and J. Zhang, "A novel binary/real-valued pigeon-inspired optimization for economic/environment unit commitment with renewables and plug-in vehicles," *Science China Information Sciences*, vol. 62, no. 7, pp. 110–112, 2019.

[8] R. Jaroensri, C. Biscarrat, M. Aittala, and F. Durand, "Generating training data for denoising real rgb images via camera pipeline simulation," 2019, http://arxiv.org/abs/1904.08825.

[9] Y. Chen and T. Pock, "Trainable nonlinear reaction diffusion: a flexible framework for fast and effective image restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1256–1272, 2017.

[10] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," 2014, http://arxiv.org/abs/1406.2199.

[11] K. Liu, X. Hu, Z. Wei, Y. Li, and Y. Jiang, "Modified Gaussian process regression models for cyclic capacity prediction of lithium-ion batteries," *IEEE Transactions on Transportation Electrification*, vol. 5, no. 4, pp. 1225–1236, 2019.

[12] X. Tang, K. Liu, X. Wang et al., "Real-time aging trajectory prediction using a base model-oriented gradient-correction particle filter for Lithium-ion batteries," *Journal of Power Source*, vol. 440, 2019.

[13] Y. Li, K. Liu, A. M. Foley et al., "Data-driven based lithium-ion battery health diagnostic and prognostic technologies: a review," *Renewable and Sustainable Energy Reviews*, vol. 113, 2019.

[14] K. Liu, C. Zou, K. Li, and T. Wik, "Charging pattern optimization for lithium-ion batteries with an electrothermal-

aging model," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 12, pp. 5463–5474, 2018.

[15] W. Ouyang, P. Luo, X. Zeng et al., "Deepidnet: Multi-Stage and Deformable Deep Convolutional Neural Networks for Object Detection," 2014, https://arxiv.org/abs/1409.3505.

[16] K. Liu, Y. Li, X. Hu, M. Lucu, and W. D. Widanage, "Gaussian process regression with automatic relevance determination kernel for calendar aging prediction of lithium-ion batteries," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 3767–3777, 2020.

[17] X. Mao, C. Shen, and Y. Yang, "Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections," *Advances in Neural Information Processing Systems*, pp. 2802–2810, 2016.

[18] K. Liu, X. Hu, Z. Yang, Y. Xie, and S. Feng, "Lithium-ion battery charging management considering economic costs of electrical energy loss and battery degradation," *Energy Conversion and Management*, vol. 195, pp. 167–179, 2019.

[19] K. Xie, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: residual learning of deep CNN for image denoising," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142–3155, 2017.

[20] X. Meng, K. Liu, X. Wang et al., "Model migration neural network for predicting battery aging trajectories," *IEEE Transactions on Transportation Electrification*, vol. 6, no. 2, pp. 363–374, 2020.

[21] K. Liu, Y. Shang, Q. Ouyang, and W. D. Widanage, "A data-driven approach with uncertainty quantification for predicting future capacities and remaining useful life of lithium-ion battery," *IEEE Transactions on Industrial Electronics*, 2020.

[22] K. Zhang, W. Zuo, and L. Zhang, "FFDNet: toward a fast and flexible solution for CNN-based image denoising," *IEEE Transactions on Image Processing*, vol. 27, no. 9, pp. 4608–4622, 2018.

[23] B. Mildenhall, J. T. Barron, J. Chen, D. Sharlet, R. Ng, and R. Carroll, "Burst denoising with kernel prediction networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2502–2510, Salt lake city, UT, USA, June 2018.

[24] S. Guo, Z. Yan, K. Zhang, W. Zuo, and L. Zhang, "Toward convolutional blind denoising of real photographs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1712–1722, 2019.

[25] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, Boston, MA, USA, June 2015.

[26] T. Brooks, K. Mildenhall, T. Xue, J. Chen, D. Sharlet, and J. T. Barron, "Unprocessing images for learned raw denoising," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11036–11045, 2019.

[27] U. Schmidt, J. Jancsary, and S. Nowozin, "Cascades of regression tree fields for image restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 4, pp. 677–689, 2016.

[28] J. Rother, L. Cai, G. Luo et al., "Lithium-ion battery state of health estimation with short-term current pulse test and support vector machine," *Microelectronics Reliability*, pp. 1216–1220, 2018.

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

[30] Y. Sun, X. Wang, and X. Tang, "Deeply learned face representations are sparse, selective, and robust," in *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, June 2015.

[31] L. Cai, J. Meng, D.-I. Stroe, G. Luo, and R Teodorescu, "An evolutionary framework for lithium-ion battery state of health estimation," *Journal of Power Sources*, vol. 412, pp. 615–622, 2019.

[32] L. Luo, W. Ouyang, X. Wang et al., "Deep learning for generic object detection: a survey," *International Journal of Computer Vision*, vol. 128, no. 2, pp. 261–318, 2020.

[33] R. Fieguth, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, USA, June 2014, https://arxiv.org/abs/1311.2524.

[34] A. Krizhevsky, L. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Neural Information Processing Systems*, vol. 25, no. 2, 2012.

[35] Z. Yang, M. Mourshed, K. Liu, X. Xu, and S. Feng, "A novel competitive swarm optimized rbf neural network model for short-term solar power generation forecasting," *Neurocomputing*, vol. 397, pp. 415–421, 2020.

[36] L. Cai, J. Meng, S. Daniel-Ioan, J. Peng, G. Luo, and R. Teodorescu, "Multi-objective optimization of data-driven model for lithium-ion battery SOH estimation with short-term feature," *IEEE Transactions on Power Electronics*, vol. 35, no. 11, pp. 11855–11864, 2020.

[37] J. Shao, C. C. Loy, and X. Wang, "Deeply learned attributes for crowded scene understanding," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, Boston, MA, USA, June 2015.

[38] Y. Tai, J. Yang, X. Liu et al., "A persistent memory network for image restoration," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4539–4547, Venice, Italy, October 2017.

[39] J. Donahue, L. A. Hendricks, S. Guadarrama et al., "Long-term recurrent convolutional networks for visual recognition and description," 2014, https://arxiv.org/abs/1411.4389.

[40] Y. Sun, X. Wang, and X. Tang, "Hybrid deep learning for computing face similarities," in *Proceedings of the IEEE International Conference on Computer Vision*, Madurai, India, May 2013.

[41] M. Liu and Y. Ding, "A unified selective transfer network for arbitrary image attribute editing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, April 2019.

[42] D. Ren, K. Zhang, Q. Wang, Q. Hu, and W. Zuo, "Neural blind deconvolution using deep priors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3341–3350, Seattle, WA, USA, November 2020.

[43] K. Kang and X. Wang, "Fully convolutional neural networks for crowd segmentation," 2014, https://arxiv.org/abs/1411.4464.

[44] F. Feng, S. Teng, K. Liu et al., "Co-estimation of lithium-ion battery state of charge and state of temperature based on a hybrid electrochemical-thermal-neural-network model," *Journal of Power Sources*, vol. 455, p. 227935, 2020.

[45] Q. Ouyang, Z. Wang, K. Liu, G. Xu, and Y. Li, "Optimal charging control for lithium-ion battery packs: a distributed average tracking approach," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 5, pp. 3430–3438, 2019.

[46] F. Feng, X. Hu, K. Liu et al., "A practical and comprehensive evaluation method for series-connected battery pack models," *IEEE Transactions on Transportation Electrification*, vol. 6, no. 2, pp. 391–416, 2020.