Signal Processing 155 (2019) 73–82

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Blur image identification with ensemble convolution neural networks

Rui Wang[a,*], Wei Li[a], Liang Zhang[b]

[a] Key Laboratory of Precision Opto-mechatronics Technology, Ministry of Education, School of Instrumentation Science and Opto-electronics Engineering, Beihang University, No. 37 Xueyuan Road, Beijing 100191, Haidian District, China
[b] Department of Electrical and Computer Engineering, University of Connecticut, 371 Fairfield Way, U-4157, Storrs, CT 06269, United States

ARTICLE INFO

Article history:
Received 21 March 2018
Revised 17 September 2018
Accepted 18 September 2018
Available online 19 September 2018

Keywords:
Blur image classification
Image blur modeling
SFA + SFGN model
Batch normalization
Ensemble deep convolution neural network

ABSTRACT

Blur image classification is a key step to image recovery in image processing. In this article, an ensemble convolution neural network (CNN) is designed to identify and classify four types of blur images: defocus blur, Gaussian blur, haze blur, and motion blur. To achieve this, a two-stage pipeline, comprised of deep compression and ensemble technique, is proposed to enhance model discriminability without incurring additional computing burden. Specifically, our method first prunes the well-known networks, Alexnet and GoogleNet, by an appropriate compression ratio. The pruned networks are denoted as Simplified-Fast-Alexnet (SFA) and Simplified-Fast-GoogleNet (SFGN). Next, we employ an ensemble policy to integrate the SFA with SFGN as SFA+SFGN by assigning their respective weights based on a voting mechanism. In addition, to provide a benchmark set of blur image samples for training and testing blur classification models, we create a new public blur image dataset (available online at http://doip.buaa.edu.cn/info/1092/1073.htm) containing 80,000+ patch-level, naturally blurred photographs, constructed using the improved super-pixel segmentation method, and 200,000+ artificially blurred images. Numerical experiments demonstrate the superior performance of the proposed approach in comparison with the original Alexnet and GoogleNet, as well as other state-of-the-art methods.

1. Introduction

Blur image type classification is essential to blur image recovery. Meanwhile, it is a challenging problem since image blurring may be caused by various factors. For instance, the interference of natural fog can result in the haze blur, optical lens distortion can lead to the defocus blur, the atmospheric turbulence can bring about the Gaussian blur, and the relative motion between the camera and the target during exposure can give rise to the motion blur [1,2]. Such blur images are commonly seen in daily life. However, it is extremely hard to achieve their automatic identification and classification by a computer.

The existing methods for blur image classification can be mainly divided into two groups: those based on *handcrafted* features and those based on *learned* features. The former typically requires reliable prior knowledge about the extraction of the blur features, which can be used to differentiate various types of blur images. In these methods, the features are usually selected manually based on the sample images and then applied to the training of the designated classifiers. On the other hand, the approaches based on learned characteristics use only the original blur images and automatically recognize the difference among the images with different types of blur.

Generally speaking, the handcrafted feature-based methods are suitable for small scale classification tasks. In Liu et al.'s work [3], several handcrafted blur features, for instance, local power spectrum slope and local autocorrelation congruency are utilized to train a Bayes classifier, which realizes the identification of the blur types. A similar method relying on the alpha channel blur feature has been proposed by Su et al. [4], which uses disparate circularity in terms of the extension of blurs. In Gao et al.'s work [5], a method named adaptive frame rate up-conversion is proposed for different motion classification of image sequence, but does not consider other degraded blur types. In article [6], Gaussian blur, defocus blur and motion blur are classified by discrimination functions and decision rules based on the blur features extracted from the twice FFT transform spectrum. In addition, a power spectrum feature-based SVM classifier is proposed in [7] and applied to the assessment of both artificially distorted images and naturally blurred images. In 2014, by means of their own self-created blur detection dataset that contains 1000 images with human labeled ground-truth blur areas, Shi et al. analyze feature discrepancy in gradient, Fourier domain, and data-driven local filters to differentiate between blurred and unblurred image regions [32]. Though the above-mentioned methods can achieve the classification or

* Corresponding author.
E-mail address: wangr@buaa.edu.cn (R. Wang).

https://doi.org/10.1016/j.sigpro.2018.09.027
0165-1684/

evaluation of the blur images to a certain degree, the robustness of these classification methods are not satisfactory for practical applications.

Recently, researchers in this area have shifted their attention from heuristics-based priori methods to the Deep Learning approach, which has been applied to a number of computer vision tasks based on learned features. Based on the model construction method involved, the deep learning approach can be divided into three classes: the generative method, the discriminant method and the mixed method, which combines the generative method with discriminant method. Among them, convolution neural network (CNN) is a typical discriminant model and has been widely used in solving object detection and instance object classification problems. Specifically, Jain and Seung demonstrate the superiority of CNN in de-noising images polluted by Gaussian noise in [8]. Moreover, a simple, single-layered neural network based on multi-valued neurons is proposed by Aizenberg et al. to identify four blur types [9]: defocus blur, rectangular blur, motion blur and Gaussian blur. In 2012, Alexnet [10] was awarded the winner of the classification in ImageNet Large Scale Visual Recognition Competition (ILSVRC-2012). The proposed techniques of weight sharing, Rectified Linear Unit (ReLU), and dropout have shown to have a great impact on the developments of later CNN models. In 2014, VGG [11] and GoogleNet [12] won the crown of detection and classification tasks in ILSVRC-2014, which declared the great progress made by the CNN-based deep learning methods. Specifically, the VGG-19 uses a large number of 3×3 convolution templates, which can not only reduce the model parameters, but also be conducive to deepen the model. On the other hand, GoogleNet, which is also called the Inception model, is connected by a large number of Inception structures. These Inception structures can reduce the model parameters and enrich the diversity of the learned features. Moreover, the three loss layers included in GoogleNet make it an ensemble model, which is resulted from the integration of three weaker CNN models. In 2015, ResNet [13] was developed by He et al. by cascading a large number of residual modules to overcome the overfitting problem. This resulted in a well-trained model with depth at layer 152 and won the first prize of ILSVRC-2015. Besides, the ensemble method as a kind of learning paradigm has been employed to enhance the overall classification performance. In paper [14], a multi-scale CNN method is proposed to improve the recognition of both the scale-invariant representations and the scale-variant representations. Its performance is evaluated based on a challenging image classification task, which involves task-relevant characteristics at multiple scales. The results show that the multi-scale CNN outperforms the single-scale CNN. In article [15], a super-pixel-based multiple local convolution neural network (SML-CNN) model for panchromatic and image classification is proposed and shows clear effectiveness in the experiments. However, to the best of our knowledge, while the CNN model has been used to perform the classification of object or character using the deep(-based) representation, it has not been applied to the classification and identification of image blur patterns.

Most recently, another learning-based method implemented by pre-trained Deep Neural Network (DNN), which is a generative model of deep learning, is proposed by Yan and Shao [16] for blur classification. In their experiments, the DNN model was trained on 36,000 blur images and achieved the classification accuracy of 95.2% based on 6000 testing images. However, the study only considers three types of blur (Gaussian, defocus and motion) and the experiments are only conducted based on simulated blur images.

Inspired by these earlier successful cases of blur type identification [8,9] and the remarkable performance of Alexnet and GoogleNet in image classification tasks [10,12], as well as the multi-scale feature ensemble methods in [14,15], a supervised architecture that integrates the SFA (simple-fast-Alexnet) and SFGN (simple-fast-GoogleNet) is proposed in this paper. To achieve the classification of four blur types (haze blur, Gaussian blur, defocus blur and motion blur) accurately and effectively, we first create a benchmark set of blur image samples for training and testing, which consists of a natural blur image dataset derived from real images by generating super-pixel with an improved simple linear iterative cluster (SLIC) algorithm, and a simulated blur image dataset. Then, individual classifiers, i.e., SFA and SFGN, are designed to construct good models from these datasets. Finally, the weight-based voting methods are employed to form a meta-classifier, i.e., SFA + SFGN, which can be used in online applications for blur classification.

The remainder of the paper is organized as follows: Section 2 overviews the motivation and methodology used to construct the meta-classifier in this work, including the pruning of Alexnet and GoogleNet, the structures of SFA and SFGN, and the ensemble mechanism used in SFA + SFGN. The modified SLIC algorithm is introduced in Section 3 for extracting the blur regions to construct our benchmark blur image datasets and to conduct the pre-processing of the images during online blur classification. Section 4 presents the numerical experiments and the results to verify the performance of the proposed ensemble CNN model. The conclusions are summarized in Section 5.

2. Ensemble architectural details

The ensemble classifier approach has attracted great attention over the last decade due to its empirical success over the single classifier approach in various applications. A key characteristic of an ensemble classifier is that it is constructed by combining the individual decisions of a set of classifiers in a certain manner. In other words, an ensemble is generally constructed by generating and then combining a number of base learners. It is discovered that ensembles are frequently more accurate than the individual classifiers that make them up. Nevertheless, having the base learners as accurate and diverse as possible can usually lead to a good ensemble. With the amazing progress lately in machine learning, especially with various state-of-the-art deep models that are capable of extracting robust features, one may expect to take the existing neural network models and deploy them to the base learners setting. However, those state-of-the-art neural networks typically have up to millions of parameters. These models are generally both computationally and memory intensive, making them difficult to be directly used as base learners for ensemble. To overcome this challenge, our work focuses on the development of ensemble technique and the base learner setting without significantly increasing computational complexity. Inspired by the achievements in network model compression, such as designing compact layers [17], quantizing parameters [18] and network pruning [19], we propose a simple and effective ensemble convolution neural network, in which each base learner is designed by decreasing the deep model size while ensuring classification accuracy at the same time. Specifically, to achieve the overarching goal of creating a blur classification system, we first apply the pre-pruning strategy to compress the Alexnet and GoogleNet as SFA and SFGN, respectively, and then ensemble them as SFA+SFGN through a voting mechanism. The details of these two steps will be elaborated in the following part of this section.

2.1. Deep model pruning

The number of neurons in each convolution layer of a deep model equals to that of the feature maps in the convolution layer. One key property of a network architecture is its ability to produce a good “representation of the data” rather than more feature

maps. Redundant features usually take up plenty of computing resources, and sometimes cause the network to be interfered with trivial details. Therefore, we first prune the deep model of Alexnet and GoogleNet both in length and width in order to remove the reductant representations.

2.1.1. Model length pruning

For the Alexnet architecture, which consists of eight layers including three fully connected layers (FCs) [10], it is known that the FCs are the joint and transfixion to bridge the convolutional layers with neural network classifiers. However, the FCs have up to millions of parameters, which account for 80% of the entire network. In this paper, we propose to remove the first two FCs and preserve only the final one to alleviate overfitting caused by redundant parameters and keep the bridge function at the same time. Note that the dropout method also disappears when removing the first two FCs, which will cause overfitting problems. To address this problem, the batch normalization (BN) [20] layer is employed not only to play the role of dropout, but also to achieve the function of the original normalization method, i.e., local response normalization (LRN) [10].

According to reference [20], the basic principle of batch normalization is illustrated as follows:

$$X_{norm}^{(k)} = \frac{X^{(k)} - E[X^{(k)}]}{\sqrt{Var[X^{(k)}] + \varepsilon}} \tag{1}$$

where $X_{norm}^{(k)}$ is the kth normalized output of the convolution layers, $E[X^{(k)}]$ is the expectation over the batch input samples, $Var[X^{(k)}]$ is the variance of the batch input samples, and ε is a micro-constant. Note that simply normalizing the activations in such way will change the distribution of the original data. To address this, the output of the batch normalization layer is modified as follows:

$$y^{(k)} = \gamma^{(k)} \cdot \overline{X}_{norm}^{(k)} + \beta^{(k)} \tag{2}$$

where $\gamma^{(k)}$, $\beta^{(k)}$ are the pair parameters for scaling and shifting the normalized value $\overline{X}_{norm}^{(k)}$. They are learned along with the original model parameters during the entire training stage. On the other hand, LRN can be expressed as

$$y^{(i)} = \frac{x^{(i)}}{\left(K + \rho \sum_j (x^{(j)})^2\right)^{\eta}} \tag{3}$$

where the activity of neuron is denoted by $x^{(i)}$, after applying the ith kernel, the response-normalized activity is designated by $y^{(i)}$. The constants K, ρ, and η are hyper-parameters whose values are given manually. Notably, j represents the order of channels.

From formulas (1)–(3), it can be seen that the output of the LRN is only related to an individual sample itself, while the output of the BN is determined by the distribution of all training samples in the mini-batch. Therefore, replacing LRN with BN can improve the generalization capability of the network.

As for the GoogleNet architecture, a block called the inception and the application of the "network in network" are its core policy [12]. In GoogleNet, various convolution kernels with the sizes of 1×1, 3×3 and 5×5 are employed in the inception module to learn the feature map with various scale. Then, the scale-variant feature map will be merged in the next layer called filter concatenation. The 1×1 kernels are used before the 3×3 and 5×5 kernels and the computation burden can be reduced by adjusting the number of 1×1 kernels involved. In addition, the operation of the three loss layers in the original GoogleNet can be regarded as an ensemble model that integrates three individual models using a weighted voting method.

Considering that our 4-class classification task is relatively straightforward and does not need as many model parameters as in ImageNet, which involves thousands of objects, we trim the length of the original GoogleNet by simply removing all of the corresponding model architectures in the last two loss layers and keeping just the first individual model. Thus, a shortened version of GoogleNet, referred to as *GoogleNet-2loss*, is formed.

2.1.2. Model width pruning

In addition to model pruning in length, model width pruning is also conducted in our work to compress the deep network even further. Note that, in the existing studies on model pruning [21,22], it is usually carried out by first learning the connectivity through pre-training and then removing the small-weight connections. In our model, however, the width pruning can be viewed as a kind of pre-pruning, which is carried out by directly reducing the number of neurons in the networks without pre-training. The advantage of this is that it does not need to traverse each connection weight to determine pruning, and thus can save a significant amount of pre-training time. On the other hand, when conducting the neuron reduction using our pre-pruning method, one key question needs to be answered first: what is the optimal reduction ratio of the networks in order to achieve desired accuracy in the blur classification task? Unfortunately, no analytical solution is available. Therefore, we turn to numerical experiments as an alternative. Specifically, we trained the modified Alexnet networks as well as the modified GoogleNet networks in our simulated training dataset, and pruned the neurons on each layer in these networks at different ratios, including 100%, 80% 50% and 40% respectively. The resulting classification accuracy under these compression ratios are plotted in Fig. 1.

From Fig. 1, we can see that compared to the uncompressed neural network (i.e., reduction ratio = 1), the networks compressed at ratios 80% and 50% have minimal drop in classification accuracy. However, when we compress the neural network with ratio of 40%, a larger decline in classification accuracy appears. The results suggest that preserving 50% of the neurons in each layer of Alexnet and GoogleNet can maintain a good balance between computation efficiency and classification accuracy.

Finally, note that the function of nonlinear mapping in the network model relies on the activation functions. The Rectified Linear Units (ReLU) and Leaky Rectified Linear Unit (LReLU) [23] are the most commonly used activation functions at present. Both of them are unsaturated activation functions, which have the potential to solve the so-called vanishing gradient problem and accelerate convergence. The activation function in Alexnet and GoogleNet is ReLU [23] as shown in Fig. 2(a), whose output is 0 when the input eigenvalue is less than or equal to 0. In other words, the ReLU units may "die" during training. This phenomenon is typically referred to as "dying ReLU". In this case, a large number of neurons in a network can become stuck in dead states, effectively decreasing the model capacity. However, to ensure the stability of the network architecture after the pre-pruning process above, we hope that the neurons in the modified model should be fully utilized. Fortunately, LReLU [23], illustrated in Fig. 2(b), will instead have a small negative slope (here is 0.01). Instead of having the function take value zero when the input $x < 0$ in ReLU, an LReLU takes small negative values in this input range. With this property, LReLU can ensure all of the neuron units in the active state to achieve the mapping and screening of the features. Therefore, LReLUs can be employed to fix the "dying ReLU" problem and is adopted to replace ReLU in our proposed pruned model.

In the subsequent discussions, we referred to the two network models under compression ratio 0.5, with batch normalization and LReLU activation functions as *Simplified-Fast-Alexnet* (SFA) and *Simplified-Fast-GoogleNet* (SFGN), respectively.

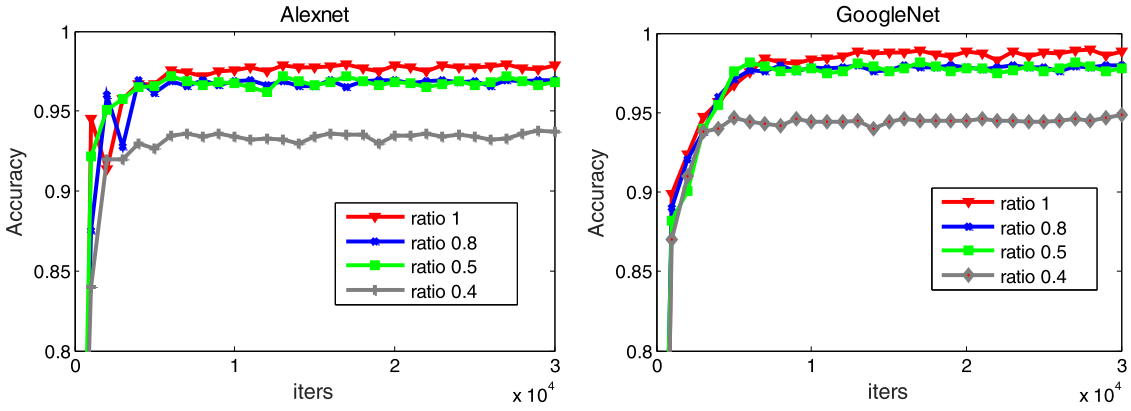


Fig. 1. Classification accuracy of Alexnet and GoogleNet under different compression ratios.

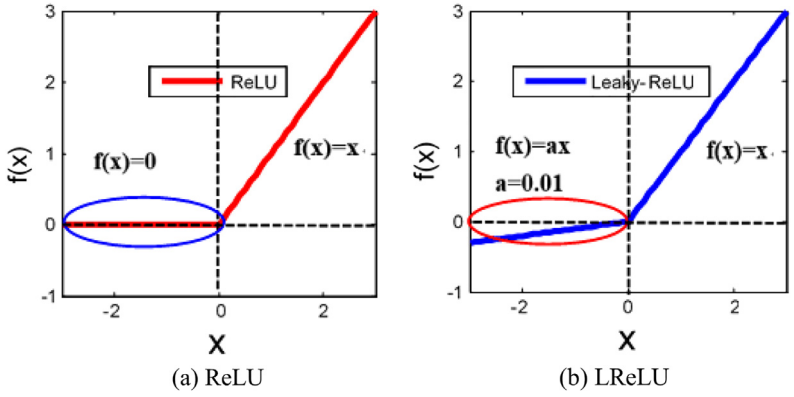


Fig. 2. The schematic diagrams of nonlinear mapping functions.

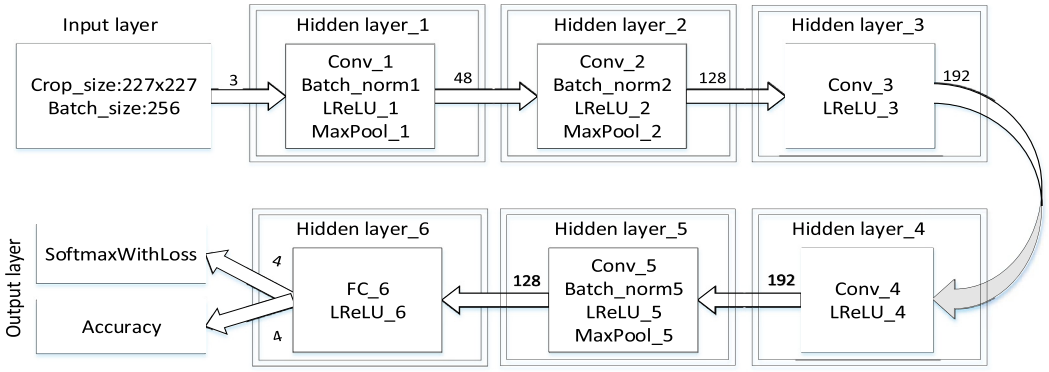


Fig. 3. Architecture of the SFA model.

2.2. SFA and SFGN model architectures

Following the above-mentioned pruning strategy, SFA and SFGN can be generated. Their respective detailed architectures are described below.

The SFA architecture, as shown in Fig. 3, has been developed and described in details in our prior work [24]. The modification made to the original Alexnet can be summarized as the following steps: first, the number of neurons in each convolution layer of Alexnet is proportionally compressed by a ratio of 0.5. Secondly, the first two FC layers are removed from the original Alexnet to enhance the computational performance. Thirdly, to address the overfitting problem, batch normalization is used in SFA to replace the local response normalization to normalize the learned features. The last is that, the LReLU is utilized to replace the ReLU to improve the model ability of feature learning and feature representation.

The SFGN architecture is illustrated in Fig. 4. In addition to the compressing the number of neurons by a ratio of 50% and the application of batch normalization and LReLU activation function, only the first loss layer is retained when pruning the GoogleNet model, while the rest of the loss layers are discarded.

As shown in Fig. 4, seven hidden layers are embedded in the SFGN architecture. Hidden layers 1 and 2 include the operations of convolution, normalization, nonlinear mapping and pooling. Hidden layer 6 contains the average pooling, convolution and nonlinear operation. The loss and accuracy computations are embedded in the output layer. The hidden layers 3–5, in which the kernel size of MaxPooling is 3×3, are the inception modules.

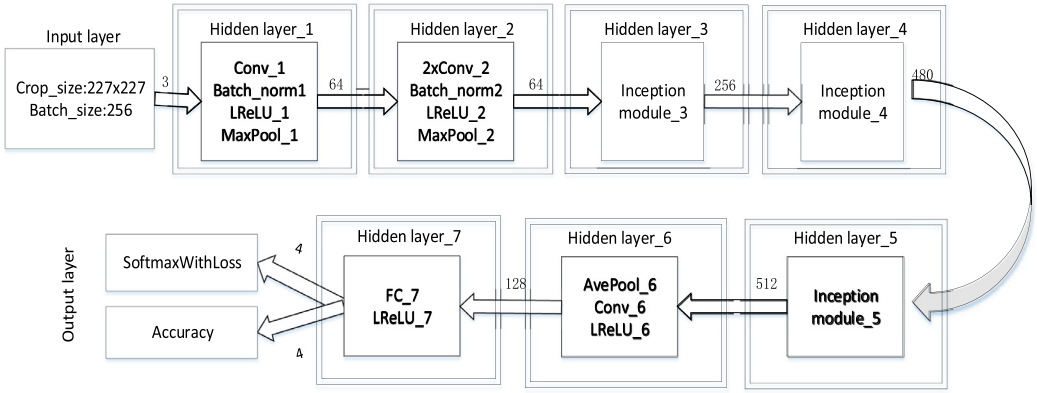


Fig. 4. Architecture of the SFGN model.

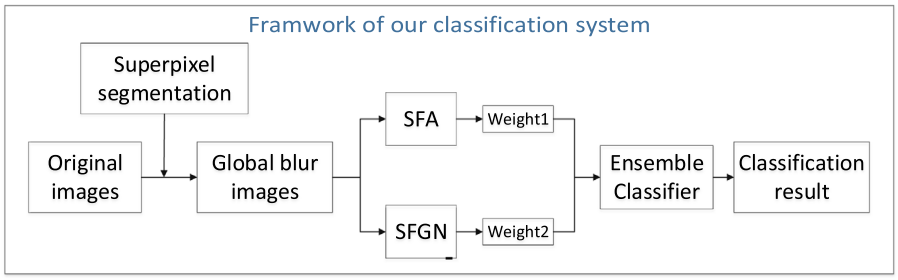


Fig. 5. Architecture of the proposed classification system.

2.3. Ensemble model of SFA + SFGN design

As stated above, the simplified network models SFA and SFGN are designed to inherit the powerful classification capabilities and high accuracy performance of the original Alexnet and GoogleNet without incurring more computing burden. On the other hand, the architectures of SFA and SFGN are different. If we take SFA and SFGN as base learners, in light of the statement at the beginning of Section 2, an ensemble model can be constructed to further improve the classification performance of these two individual classifiers.

To construct an ensemble classifier, bagging and boosting are two of the most well-known learning methods [25,26] and both can integrate a set of classifiers through voting. Among the two, bagging is based on generating replicated boot strap samples of the data, while boosting is usually performed by adjusting the weights of the training instances. Besides the difference of their sampling schemes, it is preferable for the boosting method to start with an initial choice model that has a slightly lower associated error rate than random guessing. Thus, boosting is more complex than bagging from this point of view, and less convenient to implement in practice. Also, it is generally acknowledged that boosting often leads to higher accuracy, while bagging results in more stability [27].

In this paper, we propose to construct our ensemble classifier using the bagging method. The framework of our ensemble classification system is shown in Fig. 5. Here, the classification accuracies of SFA and SFGN are denoted as C1 and C2, respectively, and the corresponding weights of SFA and SFGN are defined as Weight1 = C1/(C1+C2) and Weight2 = C2/(C1+C2), respectively.

It should be noted that the SFA, SFGN, and the ensemble classifier developed in this paper are for classification of *globally blurred images*. Therefore, as shown in Fig. 5, when a blur image is entered into the classification system, the super-pixel segmentation method is first performed to identify the blurred regions of the original image. For an image that is locally blurred, a number of patches, each being globally blurred, are extracted from the original image and are classified by weighted SFA and SFGN. The overall blur type of the original image is then determined based on the output of the ensemble classifier.

3. Generating blur image training data

In order to acquire a large number of blurred images to train the deep learning model, this paper first constructs a simulated blurred image dataset from a mass of clear images. In addition, we also collect a great number of naturally blurred images from domestic and international websites. However, the set of such real (i.e., non-simulated) blur images may contain abundant ones that are only locally blurred. Thus, as mentioned above, we first use the improved SLIC super-pixel segmentation method to extract blurred area from the blurred images to form a real blurred image dataset containing only global blurred images. The details of this procedure are described next.

3.1. Simulated blurred image data generation

The blurring of an image can be regarded as an image degradation process from high-quality to low-quality [16,28]:

$$F(x) = h(x) * f(x) + n(x) \tag{4}$$

where F denotes the degraded image, f is the lossless image, h represents the blur kernel, i.e., the point spread function (PSF), $*$ denotes the convolution operator, and $n(x)$ indicates the additional noise. Here, $n(x)$ is the Gaussian white noise.

In many practical applications, such as remote sensing and satellite imaging, the Gaussian kernel function is viewed as the kernel function of atmospheric turbulence, which is defined as

$$h(x,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{x_1^2 + x_2^2}{2\sigma^2}\right), x \in R \tag{5}$$

where σ is the kernel radius and R is the support region usually meeting the 3σ-criteria [29].

Motion blur is another blur type considered in this paper, which is caused by the relative linear motion between the target and the camera [30], and its PSF is defined as:

$$h(x) = \begin{cases} \frac{1}{M_d}, (x_1, x_2)\begin{pmatrix}\sin(\omega)\\ \cos(\omega)\end{pmatrix} = 0, x_1^2 + x_2^2 \le \frac{{M_d}^2}{4} \\ 0, \text{ otherwise} \end{cases} \tag{6}$$

where M_d denotes the length of motion in pixels and ωindicates the angle between motion direction and the x axis.

Defocus blur is the most commonly seen type of blur in daily life and it can be modeled by the cylinder function:

$$h(x) = \begin{cases} \frac{1}{\pi r^2}, \sqrt{x_1^2 + x_2^2} \le r^2 \\ 0, \text{otherwise} \end{cases} \tag{7}$$

where r is the blur radius and is proportional to the extent of defocus.

Finally, haze blur is caused by the interference of natural fog. From [2], haze blur images can be simulated by the following equation:

$$I(x) = [J(x)t(x) + A(1 - t(x))] * h_{\text{APSF}} \tag{8}$$

where $t(x) = e^{-\beta d(x)}$ indicates the medium transmission, I and J represent the haze and the inherent haze-free images, respectively, A is the global air light color vector, $x = (x, y)^T$ denotes the pixel position in the image, and h_{APSF} is the convolution matrix that can be obtained from APSF kernel $h_r(x)$ in the image domain.

Thus, to construct a simulated blur image, one can simply select a specific blur kernel function, assign the values to the parameters involved, convolve them with the original image and add certain noises. The blur images in the commonly used Berkeley dataset, Pascal VOC 2007 dataset, etc., are all constructed using this method.

3.2. Naturally blurred image data generation

As we all know, there exists a huge amount of naturally generated blur images. However, most of these images are locally blurred, while the blur classification method proposed in this work only applies to globally blurred images. Therefore, as illustrated in Fig. 5, the first step before applying the classifiers is to identify and separate the blur regions of the images. As mentioned above, in this paper we propose a modified SLIC (simple linear iterative cluster) super-pixel segmentation method to achieve the blur region extraction from a blur image. In this subsection, we discuss the detailed implementation of this method.

The SLIC method [15] is widely used in natural scene or object segmentation tasks. This method is based on K-means clustering to generate super-pixels, and the metrics considered are the distances in the LAB color space [l a b] and Euclidean space [x y]. Specifically, the LAB color distance is defined as follows:

$$d_{lab} = \sqrt{(l_i - l_j)^2 + (a_i - a_j)^2 + (b_i - b_j)^2} \tag{9}$$

where l, a, b are the different channels of an image in LAB color space. The Euclidean distance is defined by

$$d_{xy} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{10}$$

where x and y represent the Euclidean coordinates of a pixel. The main clustering metric of SLIC is defined by:

$$d_{metric} = \sqrt{\left(\frac{d_{lab}}{N_{ab}}\right)^2 + k * \left(\frac{d_{xy}}{N_{xy}}\right)^2} \tag{11}$$

where N_{ab} is the maximal value of d_{lab}, N_{xy} is the size of the super-pixel, and k is a scaling factor to control the relative weight between d_{lab} and d_{xy}.

Unfortunately, the original SLIC method is not directly applicable to blur region segmentation due to the lack of blur-related features involved. To overcome this issue and make the SLIC suitable for blur region segmentation, we first extract the blur features of local power spectrum slope, gradient histogram span, maximum saturation of an image and construct a distance metric that incorporate these blur features. Below, we briefly review the extraction of each blur feature and give the definition of blur distance metric d_{blur}.

Local power spectrum slope [3]: The definition of local power spectrum is as follows:

$$S(u, v) = \frac{1}{M \times N}|F(u, v)|^2 \tag{12}$$

where M, N denote the image size, and $F(u, v)$ is the Fourier spectrum. Let $s(f, \theta)$ denote the local power spectrum in polar coordinates. Then, the relationship between the frequency and the local power spectrum is given by

$$S(f) = \sum_{\theta} S(f, \theta) \approx \text{A}/f^{-\alpha} \tag{13}$$

Thus, the local power spectrum slope in logarithmic coordinates can be expressed as:

$$\alpha \approx \frac{\log(A/s(f))}{\log f} \tag{14}$$

where A is the amplitude of spectrum and α is the index of frequency f. Note that, due to the low-pass-filtering characteristic of a blurred region, some high frequency components of the image are lost. As a result, the amplitude spectrum slope of a blurred region tends to be steeper than that of an unblurred region.

Gradient histogram span [3]: The distribution of the gradient magnitude serves as an important clue in blur detection. Blurred regions rarely contain sharp edges and should have small gradient magnitude. Accordingly, the distributions of the gradient magnitude of blurred regions should have shorter tails than that of other regions. Thus, Gaussian modeling can be used to detect the blur region. Moreover, in order to overcome the effect of light, the image contrast is considered, and the gradient histogram span is defined by:

$$Q = \frac{\sigma}{C_m + \varepsilon} \tag{15}$$

where σ controls the span of the gradient histogram, ε is a micro-constant, and C_m is the image contrast of the region m. The definition of C_m is as follows:

$$C_m = \frac{L_{\max} - L_{\min}}{L_{\max} + L_{\min}} \tag{16}$$

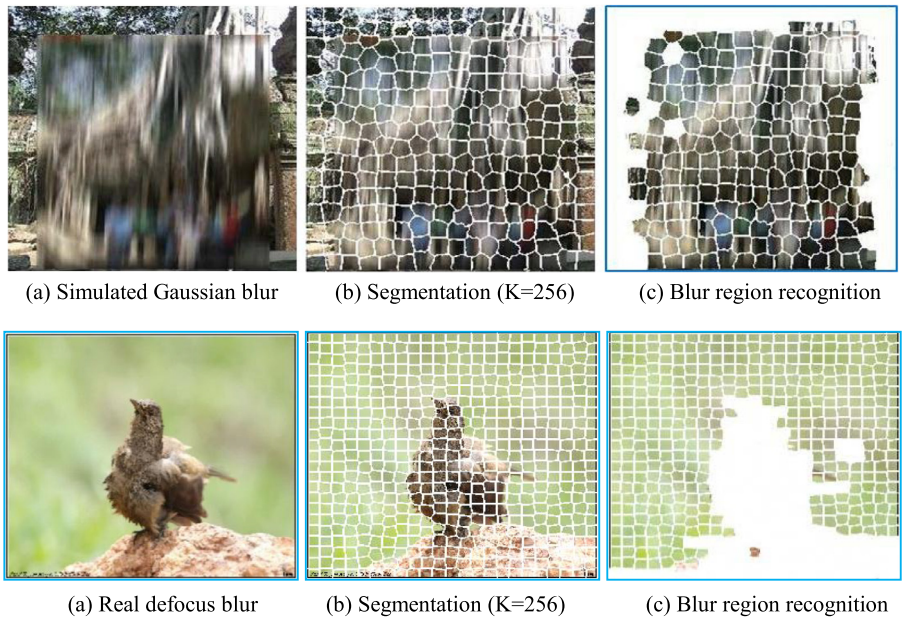
where $L_{\max}$ and $L_{\min}$ are the maximum and minimum pixel values of the region m, respectively

Maximum saturation [3]: The definition of local region saturation is given by:

$$Sa = 1 - \frac{3}{(R + G + B)}[\min(R, G, B)] \tag{17}$$

where the R, G, B represent the three color channels of the input RGB images and the maximum saturation is defined as $Sa_{\max} = \max(Sa)$. Note that unblurred regions are likely to have more vivid colors than blur regions. As a result, the maximum value of saturation in a blurred region is expected to be smaller than those in unblurred regions.

Based on the above analysis, the power spectrum slope of a blur region is generally greater than that of a clear region, while the



(a) Simulated Gaussian blur (b) Segmentation (K=256) (c) Blur region recognition

(a) Real defocus blur (b) Segmentation (K=256) (c) Blur region recognition

Fig. 6. Schematic of blur region recognition based on improved SLIC.

gradient histogram span and maximum saturation of a blur region are generally smaller than those of a clear region. Therefore, we define the blur feature metric d_{blur} as follows:

$$d_{blur} = \frac{1}{\alpha} + Q + Sa_{\max} \tag{18}$$

where α, Q, $Sa_{\max}$ are the local power spectrum slope, gradient histogram span and maximum saturation defined above. With d_{blur}, we further modify the metric of the SLIC for super-pixel segmentation as

$$dist = \sqrt{k_1 * \left(\frac{d_{lab}}{N_{\text{lab}}}\right)^2 + k_2 * \left(\frac{d_{xy}}{N_{xy}}\right)^2 + k_3 * \left(\frac{d_{blur}}{N_{blur}}\right)^2} \tag{19}$$

where k_1, k_2, k_3 are the scale factors satisfying $k_1 + k_2 + k_3 = 1$, and N_{lab}, N_{xy} and N_{blur} are the maxima of d_{lab}, d_{xy}, and d_{blur} in the neighborhood of the seed point, respectively. Clearly, the distance metric (19) of the modified SLIC method considers not only the color and spatial distance, but also the blur feature distance during the super-pixel generation. Examples of super-pixel segmentation of simulated and naturally blurred images based on the improved SLIC method are illustrated in Fig. 6.

Finally, in order to determine whether the generated super-pixel is a blur or clear region, information entropy and singular value decomposition features of the super-pixel are extracted. The information entropy reflects the amount of information contained in the pictures and is defined as:

$$H = \sum_{i=0}^{255} p_{ij} \log_2^{p_{ij}} \tag{20}$$

where p_{ij} is the probability of pair (i, j) and can be calculated by

$$p_{ij} = \frac{f(i, j)}{N \times M} \tag{21}$$

where i represents the gray value of the pixel, j represents the mean of the neighbor of the pixel, $f(i, j)$ is the frequency of the pair (i, j) in the picture, and $N \times M$is the size of the image. Since blur images can be regarded as experiencing disappearance of the high frequency bandwidth, the information entropy of a blur image should be lower than a clear image.

From article [4], the singular value decomposition decomposes an image into a weighted sum of a number of eigen-images, where the weights are exactly equal to the singular values themselves. Larger singular values correspond to the larger-scale eigen-images and small singular values correspond to smaller-scale eigen-images. Since the blur images can be regarded as the loss of high-frequency details, the weights, i.e., the significant singular values, of blur regions are greater than those of the clear regions. Therefore, we define the metric of the singular value feature as follows:

$$SVD_{metric} = \frac{\sum_{i=0}^{n} \lambda_i}{\sum_{j=0}^{L} \lambda_j} \tag{22}$$

where λ is the singular of the super-pixel, constant n is set to 6 and L is the size of the super-pixel. Finally, a super-pixel is identified as purely blurred when both $H \geq 1.3$ and $SVD_{metric} \leq 0.9$ hold simultaneously.

4. Experiments results and analysis

4.1. Datasets for model training and testing

Training dataset: The Oxford building dataset and Caltech 101 dataset are selected as our training set. A total of 10,000 images are chosen from the two datasets randomly, among which a quarter is degraded by the Gaussian blur PSF with the kernel size of R randomly selected in the range of [3,11] and σ in the range of [1,10]; another quarter is degraded by the motion blur PSF with the blur parameter M randomly selected in the range of [9,17] and ω in the range of [0°, 180°]; the third quarter is degraded by the defocus blur PSF with the blur parameter r randomly selected in the range of [5,25]; and the remaining ones are treated with the simulated haze blur PSF defined in formula (8) with parameter A randomly selected in the ranges of [200, 255]. The Gaussian white noise n is generated with mean selected from [−2, 2] and variance from [1,10]. Among these artificially blurred images, half of them are partially blurred, while the others are blurred over the entire image. In addition, 3300 real/naturally blurred pictures are

Fig. 7. Sample images in blur datasets. (a)–(d) are from the simulated blur image dataset and (e)–(f) are from the real blur image dataset.

collected from popular domestic and international websites such as Baidu.com, Flicker.com, and Pabse.com. Using the improved SLIC method described in Section 3.2, the final training sample patches are cropped from the obtained blur images above with the crop size of $128 \times 128 \times 3$ and the stride of 64 pixels. The blur types are labeled as 0-defocus, 1-Gaussian, 2-haze, 3-motion. Finally, the training dataset consisting of 200,000 simulated blur patches and about 62,000 real/natural blur patches are obtained to train the designate classifiers.

Testing dataset1: The images in the Berkeley dataset and Pascal VOC 2007 dataset are selected to form one testing dataset (i.e., with simulated blur). A total of 21,000 test sample patches are generated using the same procedure of the training patches. Among these patches, 5560 haze blur image patches possess the same sources with training samples and the rest are evenly allocated to the other three classes.

Testing dataset2: In order to testify the performance of the proposed classifier in practical applications, a dataset consisting of 13,810 real/naturally blurred image patches is constructed. The samples are collected from the same internet sources as the blur samples in the training dataset.

All the samples in both the training dataset of simulated blur images and the one with real/naturally blurred images are uniformly distributed among the four blur types to enhance the generalization of the suggested classifier. The samples in the two testing datasets are random distributed. Several sample images with artificial blur and real blur are shown in Fig. 7.

4.2. Performance of single SFA and SFGN model

The classification models of SFA and SFGN are trained on a PC with NVIDIA-GTX-1080 8GB GPU under the Caffe framework. The resulting train loss and classification accuracy curves are shown in Fig. 8.

As one can see from Fig. 8(a), while both curves show similar trend, the training loss of SFA appears to be more volatile than the SFGN model. This is due to the imparity of the batch size of the two models (SFA-64 and SFGN-256). In Fig. 8(b), SFA and SFGN both start with almost identical learning progress. Then, after about 1000 iterations, the SFGN model starts to show a bit higher accuracy than the SFA model. Note that while the figures show the performance of the two models for iterations 0–16,000, the model training processes are terminated only when both the losses and the accuracy curves reach their relative stable states.

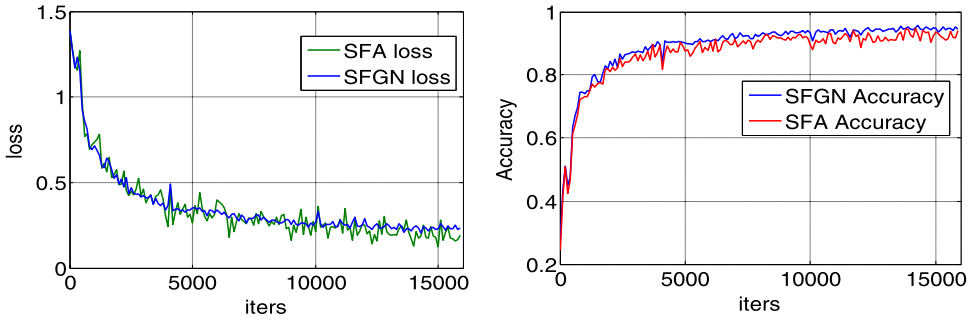
When the individual classifiers are obtained, the filter parameters and the learned feature maps of each layer can be acquired. Visualization results of the filter parameters and the learned feature maps of some layers are illustrated in Fig. 9. Specifically, Fig. 9 (a) is the original sample RGB image, and Fig. 9.(a) shows the weights initialization of the filter kernels of the first convolution layer, whose parameters satisfy the Gaussian distribution. The weight map samples are illustrated in Fig. 9[(b)–(e)]. As one can see, after a long training time, a smooth filter with no noise contamination, no important correlation and no structural mess can be obtained. This indicates that the model parameters are well learned. The feature maps samples are shown in Fig. 9[(b)–(e)]. Among them, (b) and (c) are feature maps learned from the shallow layers of conv_1 and conv_2, where they reflect the global features such as the shape features and texture features. On the other hand, (d) and (e) are the feature maps learned from the deep layers of conv_4 and conv_5, which mirror the local characteristics that have low readability and are difficult for understanding.

4.3. The integrated CNN performance

The comparisons among the original Alexnet, GoogleNet, SFA, SFGN, and the proposed ensemble CNN are carried out under several criteria. The results are summarized in Table 1. Here, P_N is the number of model parameter, L_N is the model depth; F_T is the forward propagation time, B_T is the error backward propagation time, CLF_T is the average time required to identify a single image, Tr_T is the model training time. In addition, Error denotes the classification error rate over the testing dataset1 and voting weight represents the voting weights of the SFA and SFGN models to form the ensemble classifier.

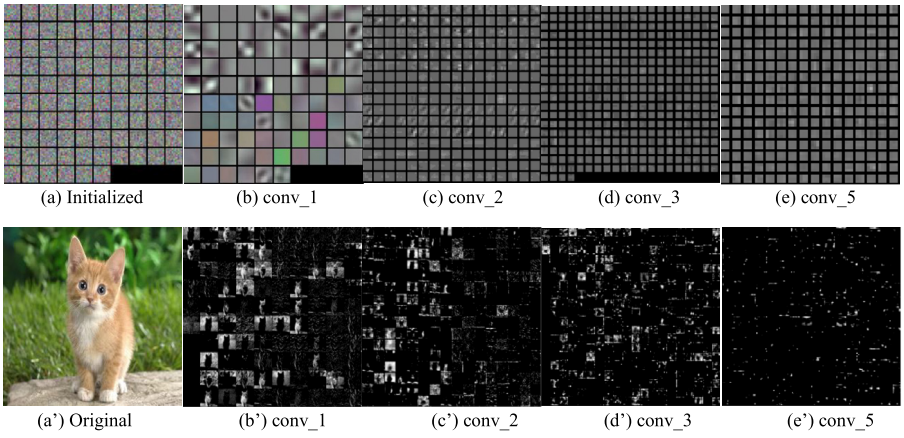
As one can see from the Table 1, the P_N of Alexnet is over 1000 times of SFA and GoogleNet is almost 7 times of SFGN. While F_T of different models are of the same order of magnitude, B_T is dramatically different due to the great disparity in the numbers of model parameters to be learned in the models compared. In addition, the CLF_T of SFA is only about 13.5% of Alexnet's CLF_T, and SFGN is about 6 times faster than GoogleNet. This implies that SFA and SFGN are more suitable in practical real-time applications. Moreover, the total training times of SFA and SFGN are both less than one day, while the Alexnet and GoogleNet require about two days each. Finally, the classification error rate of SFA suffers a 1.05% drop compared to the original Alexnet, while the drop is 0.11% from GoogleNet to SFGN. However, the short training time cost and the fast classification speed of SFA and SFGN allow for the construction of the ensemble classifier that easily outperforms both Alexnet and GoogleNet in terms of training time and classification accuracy.

In addition to Alexnet and GoogleNet, we also compare the proposed method with the state-of-the-art. Specifically, the original



(a) Train loss curves (b) Accuracy curves

Fig. 8. Loss and accuracy curves of the SFA model and SFGN model.



(a) Initialized (b) conv_1 (c) conv_2 (d) conv_3 (e) conv_5

(a') Original (b') conv_1 (c') conv_2 (d') conv_3 (e') conv_5

Fig. 9. Weights maps of Conv_1 layer and Conv_2 layer. (a) The 48 filters kernels of size $11 \times 11 \times 3$ learned by the Conv_1 layer on the $227 \times 227 \times 3$ input images; (b) The 128 convolution kernel of size $5 \times 5 \times 1$ learned by the Conv_2 layer on the $27 \times 27 \times 3$ feature of the Maxpool_1 layer.

Table 1
Comparison of different models under several criteria.

Name	P_N	F_T/ms	B_T/ms	F_B_T/ms	CLF_T/s	Tr_T/h	Error (%)	Voting weight
Alexnet [10]	58,649,189	69	138.131	207.84	0.578	43	2.26	–
SFA [24]	50,489	13.485	8.223	20.965	0.078	22.47	3.21	0.48
GoogleNet [12]	6,797,700	40.733	85.935	127.012	0.496	42.42	1.79	–
GoogleNet-2loss	3,497,700	33.285	70.397	103.801	0.356	40.24	1.57	–
SFGN	1,017,700	21.665	46.430	68.203	0.081	21.17	1.88	0.52
Ensemble-classifier	1,068,189	35.150	54.653	89.168	0.159	43.64	1.11	–

architectures of Bayes classifier [3] and two-step way [4] are selected in the comparison. These methods need to detect the blur region first, before classifying the obtained blur areas. In our algorithm, however, the blur detection was accomplished in the preprocessing stage and only the whole blurred patches are sent to the classifiers for identification. Another classification method selected in the Gaussian radial basis function-based support vector machine (SVM) classifier, which has been successfully applied to classifying the ovarian cancer images in our preliminary work [31]. In this paper, we will focus on its blur image classification performance. Other commonly used classifiers such as Softmax and Random Forest are also chosen for comparison. In our implementation, Bayes [3], SVM [31], Softmax and Random Forest are all designed with 35 handcrafted blur features including statistic features, texture features and spectrum features, and then are evaluated based on our testing datasets. In addition, the single-layered NN [9], DNN framework [16] are selected for comparison as well. The classification accuracy rate is employed to determine the classification performance and is defined by

$$\text{Accuracy} = \frac{N_{correct}}{N_{total}} \times 100\% \tag{23}$$

Table 2
Comparison of the ensemble classifier and the state-of-the-art.

Methods	Features	Accuracy1	Accuracy2
Two-step way [4]		88.78%	
***Bayes* [3]**		70.07%	54.16%
***SVM* [31]**	Handcrafted	*82.73%*	*80.22%*
Softmax		*75.68%*	*72.64%*
Random Forest		*83.46%*	*75.41%*
Single-layered NN [9]		94%–97%	
DNN [16]		95.2%	
***Alexnet* [10]**		*97.74%*	*94.10%*
***GoogleNet* [12]**	Learned	*98.21%*	*95.86%*
GoogleNet-2losses		*98.33%*	*95.91%*
***SFA* [24]**		*96.99%*	*93.75%*
SFGN		*98.12%*	*95.81%*
Ensemble classifier		*98.89%*	*96.72%*

where $N_{correct}$ denotes the number of correctly classified samples, and N_{total} indicates the total number of test samples.

The comparison results are summarized in Table 2. Note that the classification accuracies of two-step way [4], single-layered NN [9] and DNN [16] included in the table are the ones reported in their respective references, while the accuracy data of the other methods are obtained by testing on our datasets. Note that Accu-

racy1 and Accuracy2 stand for the accuracy results based the testing dataset1 and testing dataset2, respectively. Note also that reference [9] only provides the accuracy of single-layered NN for single class classification task, while the other methods demonstrate the classification accuracy for all four blur types considered.

It can be observed from Table 2 that the prediction accuracy (>90%) of learned feature-based methods is generally superior to the ones (<90%) whose features are handcrafted. The classification accuracy of SFA on the simulated testing dataset is 96.99%, which is slightly lower than Alexnet's 97.74%. Nevertheless, it is still better than the DNN model of 95.2%. The classification accuracy of SFGN is 98.12%, which outperforms the SFA model but less than the classification performance of the ensemble classifier of 98.89%. In addition, the classification performance of SFA, SFGN and the ensemble classifier on the real/natural blur datasets are 93.75%, 95.81% and 96.72%, respectively. Clearly, the ensemble classifier has the best classification accuracy on the real/natural blur image dataset. Consider also that the experiment data in Table 1 have shown the outstanding classification accuracy, computational efficiency and real-time performance of the ensemble classifier compared to Alexnet and GoogleNet. Therefore, we claim that the ensemble classifier of SFA and SFGN is a highly efficient and accurate tool for blur image classification.

5. Conclusion

In this paper, an highly accurate and efficient ensemble classifier denoted as SFA+SFGN is developed for handling the classification of defocus blur, Gaussian blur, haze blur and motion blur in digital images. The novel base learners in the proposed ensemble model, Simplified-Fast-Alexnet (SFA) and Simplified-Fast-GoogleNet (SFGN), are created by pre-pruning the original Alexlet and GoogleNet, respectively. In addition, to provide a benchmark dataset for blur image classification task, a new public blur image dataset - BHFID (Beihang University Fuzzy Image Database) containing naturally blur photographs and artificially blurred images is created and can be accessed at http://doip.buaa.edu.cn/info/1092/1073.htm. To accomplish this, we propose and apply an improve the simple linear iterative clustering (SLIC) method to generate super-pixels and segment the actual blurred regions in an image. This ensures the applicability of the samples in the dataset for online blur classification. Using this labeled dataset, we design and train our SFA+SFGN classifier to perform the task of identifying and classifying four types of blur images. To investigate the performance of the proposed ensemble classifier, we test it, along with Alexnet, GoogleNet and other blur classification methods, based on the simulated blur image dataset and naturally blurred image dataset in BHFID. The experiment results demonstrate the superior performance of the proposed ensemble classifier in classification accuracy. As of computational efficiency, the model training time of our ensemble classifier (44.84 h) is comparable to Alexnet and GoogleNet, while the average single image classification time of 0.159 s by the ensemble classifier dramatically outperforms Alexnet and GoogleNet. Therefore, the success of the proposed ensemble classifier makes us believe that this work provides an effective compression and ensemble method, which can facilitate the use of the notable complex neural networks in variety of applications.

Acknowledgment

This work is supported in part by a grant from National Natural Science Foundation of China (61673039).

References

[1] J.H. Elder, Zucker, S.W. Local, Scale control for edge detection and blur estimation, IEEE Trans. Pattern Anal. Mach. Intell. 20 (7) (1998) 699–716.
[2] R. Wang, R. Li, H. Sun, Haze removal based on multiple scattering model with super-pixel algorithm, Signal Process. 127 (C) (2016) 24–36.
[3] R. Liu, Z. Li, J. Jia, Image partial blur detection and classification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.
[4] B. Su, S. Lu, C.L. Tan, Blurred image region detection and classification, in: Proceedings of the International Conference on Multimedia, Scottsdale, Az, Usa, 2011, pp. 1397–1400. November 28- December. 2011.
[5] X. Gao, Y. Yang, B. Xiao, Adaptive frame rate up-conversion based on motion classification, Signal Process. 88 (12) (2008) 2979–2988.
[6] X.J. Bi, T. Wang, Adaptive blind image restoration algorithm of degraded image, in: Proceedings of the Congress on Image and Signal Processing, 2008, pp. 536–540. CISP '08. 2008.
[7] E. Mavridaki, V. Mezaris, No-reference blur assessment in natural images using Fourier transform and spatial pyramids. 2015, pp. 566–570
[8] V. Jain, H.S. Seung, Natural image denoising with convolutional networks, in: Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, 2008, pp. 769–776. December.
[9] I. Aizenberg, C. Butakoff, V. Karnaukhov, et al., Blurred image restoration using the type of blur and blur parameter identification on the neural network, in: Proceedings of SPIE-The International Society for Optical Engineering, 4667, 2002, pp. 460–471.
[10] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Proceedings of the International Conference on Neural Information Processing Systems, 2012, pp. 1097–1105.
[11] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556 [cs.CV], 2014.
[12] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.
[13] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[14] N. Van, E. Postma, Learning scale-variant and scale-invariant features for deep image classification, Pattern Recognit. 61 (2017) 583–592.
[15] W. Zhao, L. Jiao, W. Ma, et al., Super-pixel-based multiple local CNN for panchromatic and multispectral image classification, IEEE Trans. Geosci. Remote Sens. 55 (7) (2017) 4141–4156.
[16] R. Yan, L. Shao, Blind image blur estimation via deep learning, IEEE Trans. Image Process. 25 (4) (2016) 1910–1921.
[17] E.L. Denton, W. Zaremba, J. Bruna, Y. LeCun, R. Fergus, Exploiting linear structure within convolutional networks for efficient evaluation, in: Proceedings of the Advances in neural information processing systems, 2014, pp. 1269–1277.
[18] M. Rastegari, V. Ordonez, J. Redmon, A. Farhadi, Xnor-net: imagenet classification using binary convolutional neural networks, in: Proceedings of the European Conference on Computer Vision, 2016, pp. 525–542.
[19] S. Han, J. Pool, J. Tran, W. Dally, Learning both weights and connections for efficient neural network, in: Proceedings of the Advances in Neural Information Processing Systems, 2015, pp. 1135–1143.
[20] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, in: Proceedings of the International Conference on Machine Learning., 2015, June2015, pp. 448–456. June.
[21] H. Van Nguyen, K. Zhou, R. Vemulapalli, Cross-domain synthesis of medical images using efficient location-sensitive deep network, in: Proceedings of the International Conference on Medical Image Computing and Computer Assisted Intervention, 2015, pp. 677–684. October.
[22] S. Han, H. Mao, W.J. Dally, Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv:1510.00149 [cs.CV], 2015.
[23] V. Nair, G.E. Hinton, Rectified linear units improve restricted Boltzmann machines, in: Proceedings of the International Conference on Machine Learning, 2010, pp. 807–814.
[24] R. Wang, W. Li, R. Qin, J. Wu, Blur image classification based on deep learning, in: Proceedings of the IEEE International Conference on Imaging Systems and Techniques (IST), 2017, pp. 1–6.
[25] J.R. Quinlan, Bagging, boosting, and C4. 5, in: Proceedings of the AAAI/IAAI, 1, 1996, pp. 725–730. August.
[26] A. Lemmens, C. Croux, Bagging and boosting classification trees to predict churn, J. Mark. Res. 43 (2) (2006) 276–286.
[27] ZH. Zhou, Ensemble Methods: Foundations and Algorithms, Taylor & Francis, 2012.
[28] R. Molina, J. Mateos, A.K. Katsaggelos, Blind deconvolution using a variational approach to parameter, image, and blur estimation, IEEE Trans. Image Process. 15 (12) (2006) 3715–3727.
[29] F. Chen, J. Ma, An empirical identification method of gaussian blur parameter for image deblurring, IEEE Trans. Signal Process. 57 (7) (2009) 2467–2478.
[30] D. Kundur, D. Hatzinakos, Blind image deconvolution, IEEE Signal Process. Mag. 13 (3) (2002) 43–64.
[31] R. Wang, R. Li, Y. Lei, Q. Zhu, Tuning to optimize svm approach for assisting ovarian cancer diagnosis with photoacoustic imaging, Bio-medical Mater. Eng. 26 (s1) (2015) S975–S981.
[32] J. Shi, L. Xu, J. Jia, Discriminative blur detection features, in: Proceedings of the CVPR, 2014.