Knowledge-Based Systems 209 (2020) 106419

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

A convolutional neural network with sparse representation

Guoan Yang *, Junjie Yang, Zhengzhi Lu, Deyang Liu

School of Automation Science and Engineering, Faculty of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an, Shaanxi 710049, China

ARTICLE INFO

Article history:
Received 21 January 2020
Received in revised form 3 July 2020
Accepted 3 September 2020
Available online 24 September 2020

Keywords:
Image classification
Sparse representation
Convolutional neural network
Feature extraction
Multilayer convolutional sparse coding

ABSTRACT

This paper proposes a sparse representation layer in the feature extraction stage of a convolutional neural network (CNN). Our goal is to add sparse transforms to a target network to improve its performance without introducing an extra calculation burden. First, the proposed method was achieved by inserting the sparse representation layers into a target network's shallow layers, and the network was trained end-to-end using a supervised learning algorithm. Second, In the forward pass the network captured the features through the convolutional layers and sparse representation layers accomplished with wavelet and shearlet transforms. Thirdly, in the backward pass the weights of the learned kernels of the network were updated through a back-propagated error, while the sparse representation layers were fixed and did not require updating. The proposed method was verified on five datasets with the task of image classification: FOOD-101, CIFAR10/100, DTD, Brodatz and ImageNet. The experimental results show that the proposed method leads to higher recognition accuracy in image classification, and the additional computational cost is relatively small compared to the baseline CNN model.

1. Introduction

Recently, deep learning-based approaches have achieved tremendous success in various visual tasks, such as image classification, image segmentation, image denoising, object recognition and image super-resolution [1–6]. With multiple layers to capture features on various scales, a deep network is often regarded as a system similar to the human brain with the ability to learn high-level abstractions of data from images, videos and speech. The convolutional neural network (CNN), as a typical deep learning model, is an efficient end-to-end hierarchical learning system. In the CNN paradigm, a convolutional block (usually composed of a convolutional layer, a pooling layer and an activation function) iteratively extracts features represented by the learned kernels at different scales, while the classifier at the end (usually a fully connected layer) maps the extracted features to the expected labels.

The core of a standard CNN is the stack of layers that possess learnable kernels. A CNN achieves the data-driven task through these learnable kernels: during the learning procedure, an optimization algorithm is used to repeatedly adjust the weights of the learnable kernels in the convolutional layers and the fully connected layers until the result converges. However, this necessary procedure can result in significant computational costs and memory consumption [7,8].

Although this deep structure has been empirically proven to be useful, it has always been argued that this model lacks theoretical support. Therefore, along with various experimental proofs and applications, the theoretical origin of its successful application has also been studied in recent years. Mallat [9] proposed a scattering network based on the wavelet transform. This model replaces the learned filters in the CNN with predefined wavelet functions, and the features extracted by the model show features such as translation invariance and rotation invariance. Wiatowski and Bolcskei [10] developed a theory that encompasses a general convolutional transform, involving semidiscrete frames, general Lipschitz-continuous nonlinearities and general Lipschitz-continuous pooling operator emulation. They also demonstrated a translation invariance result of a vertical nature, in the sense that the features became progressively more invariant of translation as the network depth increased. Ye et al. [11] showed that the missing link between deep learning and classical signal processing approaches was the convolutional framelets for representing a signal by convolving local and nonlocal bases. Furthermore, they demonstrated that the success of deep learning came from a novel signal representation using a nonlocal basis combined with a data-driven local basis, which is indeed a natural extension of classical signal processing theory. Papyan et al. [12] and [13] presented theoretical support for the convolutional sparse coding (CSC) model, including the guarantee of the uniqueness of the

* Corresponding author.
E-mail addresses: gayang@mail.xjtu.edu.cn (G. Yang), nappoo@stu.xjtu.edu.cn (J. Yang), lu947867114@stu.xjtu.edu.cn (Z. Lu), yohn08@stu.xjtu.edu.cn (D. Liu).

https://doi.org/10.1016/j.knosys.2020.106419
0950-7051/

sparsest solution and the stability of the sparsest solution. In [14], a classical CNN with multiple layers was shown to be equivalent to the multilayer convolutional sparse coding (ML-CSC) model, which was solved by a hierarchical threshold algorithm. Inspired by [15], Papyan et al. recommended that the ML-CSC model is equal to the forward pass of the CNN. In the model, the weights of the convolutional operator provided self-learned atoms to ensure the ability to adaptively capture different features. To some extent, this model reveals the principle of CNN.

In recent years, many researchers have mainly focused on changing the topological structure of deep neural networks to solve specific problems or improve performance. However, the trade-off between a time-consuming training process and satisfactory performance has been a problem. Consequently, due to this problem, some studies have resorted to the strategy that combines a CNN having a simple structure with other traditional mathematical, signal processing or machine learning methods to enhance the feature extraction ability. Liu et al. [16] presented a multilevel wavelet CNN (MWCNN) model to achieve a better trade-off between the receptive field size and the computational efficiency. They combined a modified U-Net with a wavelet transform to reduce the size of feature maps in the contracting subnetwork. Zhou et al. [17] proposed active rotating filters (ARFs) that could actively rotate and generate feature maps with location and orientation information. In [18], the authors introduced a new learnable module called the spatial transformer, which allows the spatial operation of data within the network. This module can be inserted into existing convolutional architectures and offers networks the ability to spatially transform feature maps. Luan et al. [19] proposed a Gabor convolutional network, which incorporated Gabor filters into deep convolutional neural networks for enhancing network's adaptation to orientation and scale information. Sun et al. [20] proposed a deep sparse coding network, in which the sparse coding layers offer the network more generalization ability for feature representation. To summarize, the idea behind these methods is to broaden the feature channels of the network through a spatial or frequency transform. The networks can learn more intrinsic feature representations through these transforms; thus, the outputs are more robust.

Therefore, based on these theoretical analyses and the inspiration of pioneering ideas, we intend to propose a method combining a deep convolutional neural network (DCNN) with a sparse representation to achieve a more powerful feature extraction ability. In this proposed model, the sparse representation layers are inserted into the existing CNNs to generate more feature maps with intrinsic characteristics, so that the CNNs can achieve better performance. Based on ML-CSC, we offer some theoretical support for the effectiveness of sparse representation layers and the convolutional layer in the network. Then, we verify our method through the task of image classification and study the sparseness of self-learned kernels of the network. In summary, this paper makes the following contributions:

(1) This paper proposes a hybrid of CNN and sparse representation, providing a method that not only can improve the performance of the CNN model but also can ensure that the fixed parameters from the wavelet and shearlet do not incur additional trainable parameters, namely, keeping the additional computational complexity at a relatively small amount.

(2) We have experimentally verified our method on five datasets FOOD-101, CIFAR10/100, DTD, Brodatz and ImageNet, and compared to the baseline CNN model, our method shows better performance.

(3) This paper theoretically explains the work procedure of the proposed network model based on a novel mathematical model named the multilayer convolutional sparse coding (ML-CSC) model.

The rest of this paper is organized as follows. Section 2 introduces the multilayer convolutional sparse coding model and two concrete sparse representation methods: the wavelet and shearlet transforms. Section 3 provides details of the proposed CNN with sparse representation by explaining the building blocks and network. Section 4 presents the experimental results of the proposed method on various benchmark datasets used for image classification and the measurement of sparseness. Finally, Section 5 concludes this paper and proposes some promising future research directions.

2. Theoretical definition

2.1. Multilayer convolutional sparse coding model

(1) In the sparse representation model, the classical sparsity problem was defined [21] by the l_0 optimization. For a measurement matrix $\boldsymbol{A}$ and an observation sample $\boldsymbol{y}$, the task of the sparest representation of a given signal $\boldsymbol{x}$ can be expressed [21] as:

$$\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_0 \qquad \text{s.t. } \boldsymbol{A}\boldsymbol{x} = y \tag{1}$$

where we have denoted $\boldsymbol{x}$ by l_0 the number of nonzeros in, and given that the l_0 optimization problem is an NP-hard problem, the above formulation has a convex relaxation in the form of solving the l_0 optimization problem [22], which is formally defined as:

$$\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_1 \qquad \text{s.t. } \boldsymbol{A}\boldsymbol{x} = y \tag{2}$$

(2) In Fig. 1, for a fixed dictionary $\boldsymbol{D}$, reconstructing the given a signal $\boldsymbol{X}$ by using the sparsest representation is called sparse coding. The sparse coding mainly solves the following problem [21]:

$$\min_{\boldsymbol{S}} \|\boldsymbol{S}\|_0 \qquad \text{s.t.} \quad \boldsymbol{D}\boldsymbol{S} = \boldsymbol{X} \tag{3}$$

$$\min_{\boldsymbol{S}} \|\boldsymbol{S}\|_1 \qquad \text{s.t.} \quad \boldsymbol{D}\boldsymbol{S} = \boldsymbol{X} \tag{4}$$

In addition, the fixed dictionaries that are combined with the sparse representation have been analytically predefined by wavelet and multiscale geometric analysis theory. Although the sparse coding problem under these definitions can be efficiently achieved, recently, the sparse coding problem have mostly turned to data-driven methods for training data via a learning procedure for adapting the dictionary.

Since a dictionary was usually chosen to be redundant, the task-driven dictionary learning strategies [23] for representing a global signal $\boldsymbol{X}$ can be formulated as follows:

$$\min_{\boldsymbol{D},\boldsymbol{S}} \|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{S}\|_2^2 + \beta\|\boldsymbol{S}\|_1 \tag{5}$$

where the first term is a fidelity index of signal $\boldsymbol{X}$, or the index of approximation error minimization, and β is a scalar that adjusts the prior $\beta\|\boldsymbol{S}\|_1$. From this perspective, the dictionary is learned rather than both predefined and designed.

We know that the sparse representation model is traditionally used for modeling local patches extracted from a global signal. Recently, the CSC model was presented [14], and it is dedicated to describing the global signal $\boldsymbol{X} \in R^N$ as a multiplication of a global convolutional dictionary $\boldsymbol{D} \in \boldsymbol{R}^{N\times Nm}$ and a sparse vector $\boldsymbol{S} \in \boldsymbol{R}^{Nm}$ under the condition of m channels as shown in Fig. 1. In other words, the signal $\boldsymbol{X}$ can be described as a linear combination of a few columns (atoms) from the dictionary $\boldsymbol{D}$. The global $\boldsymbol{X} = \boldsymbol{D}\boldsymbol{S}$ and local $\boldsymbol{x}_i = \Phi\boldsymbol{s}_i$ representations of the CSC model are shown in Fig. 1.

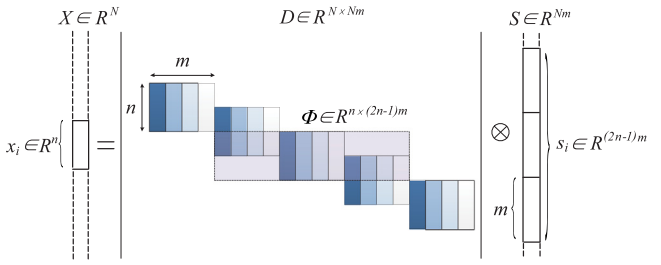

Fig. 1. The ith patch x_i of the global signal $\boldsymbol{X} = \boldsymbol{DS}$ given by $\boldsymbol{x}_i = \boldsymbol{\Phi s}_i$.

However, it was stated that [14], in the ML-CSC model as shown in Fig. 2, both the l_0 and l_1 optimization mentioned above are difficult to use, because the sparseness of $\boldsymbol{S}$ is measured in a localized manner , and the $\boldsymbol{S}$ in ML-CSC is not necessarily sparse in both the l_0 and l_1 sense. Therefore, the $l_{0,\infty}$ norm optimization index is suggested by Papyan in [14].

In [13,14], the suggestion was made to measure the sparsity of $\boldsymbol{S}$ in a localized manner, i.e., the ith n-dimensional patch of the global system $\boldsymbol{X} = \boldsymbol{DS}$, given by $\boldsymbol{x}_i = \Phi \boldsymbol{s}_i$ in a localized manner, as shown in Fig. 1. The stripe-dictionary $\boldsymbol{\Phi}$ of size $n(2n-1)m$ is obtained by extracting the ith patch from the global dictionary $\boldsymbol{D}$ and discarding all the zero columns from it. The stripe vector $\boldsymbol{s}_i$ is the sparse representation of length $(2n-1)m$, containing the coefficients of atoms contributing to $\boldsymbol{x}_i$, as shown in Fig. 1. Obviously, the choice of a convolutional dictionary enables the signal $\boldsymbol{X}$ such that every patch of length n can be sparsely represented using a single shift-invariant local dictionary $\boldsymbol{\Phi}$. Furthermore, in Fig. 1, the sparse vector $\boldsymbol{S}$ of size Nm has a few entries of nonzeros, the sparseness of $\boldsymbol{S}$ in the CSC is measured in a localized manner [12], as follows:

$$\|\boldsymbol{S}\|_{0,\infty} = \max_i \|s_i\|_0 \tag{6}$$

where the $\|\boldsymbol{S}\|_{0,\infty}$ norm is defined as the maximal one for number of the nonzeros in a stripe of length $(2n-1)m$, and the sparseness is computed by sweeping over all stripes. The local signal $\boldsymbol{x}_i$ from $\boldsymbol{X}$ is recovered by $\boldsymbol{\Phi s}_i$ through shifting a local matrix of size nm in all the possible positions. In other words, the ith patch $\boldsymbol{x}_i$ of the global signal $\boldsymbol{X} = \boldsymbol{DS}$, given by $\boldsymbol{x}_i = \Phi \boldsymbol{s}_i$, where $\boldsymbol{\Phi}$ is the convolutional dictionary in stripe manner.

Given a signal $\boldsymbol{X}$, finding its sparest representation $\boldsymbol{S}$ in the $l_{0,\infty}$ sense is equivalent to the following optimization problem:

$$\min \|\boldsymbol{S}\|_{0,\infty} \quad \text{s.t.} \quad \boldsymbol{DS} = \boldsymbol{X} \tag{7}$$

From Fig. 1, we can see that finding a global vector $\boldsymbol{S}$ can describe sparsely every patch in the signal $\boldsymbol{X}$ using the dictionary $\boldsymbol{\Phi}$.

(3) Obviously, the ML-CSC model can be extended to more than two layers in Fig. 2, as elaborated in the following description. Due to the multilayer nature of the ML-CSC model, $\boldsymbol{X} = \boldsymbol{D}_1\boldsymbol{S}_1$, the global signal $\boldsymbol{X} \in R^{Nm}$ and is a linear combination of atoms taken from $\boldsymbol{D}_1 \in R^{N\times Nm}$, where $\boldsymbol{D}_1$ is composed of m local filters of length n_0. Consequently, $\boldsymbol{X} = \boldsymbol{D}_1\boldsymbol{D}_2\boldsymbol{S}_2$ is the linear combination of more entities taken from the dictionary $\boldsymbol{D}_1\boldsymbol{D}_2$, and it can also be viewed as the factorization of the sparse vector $\boldsymbol{S}_1$, $\boldsymbol{S}_1 = \boldsymbol{D}_2\boldsymbol{S}_2$. Thus, it can be seen that $\boldsymbol{S}_1$ is the sparse vector of $\boldsymbol{X} = \boldsymbol{D}_1\boldsymbol{S}_1$, while in $\boldsymbol{S}_1 = \boldsymbol{D}_2\boldsymbol{S}_2$, it is the input signal for the next convolutional operation. Note that $\boldsymbol{S}_1 \in \boldsymbol{R}^{Nm}$ can be regarded as an N-dimensional global signal with m channels; $\boldsymbol{D}_2 \in \boldsymbol{R}^{Nm\times Nl}$ is a stride convolutional dictionary, which possesses l local filters of length $n_1 m$ and skips m entries at a time; $\boldsymbol{S}_2 \in \boldsymbol{R}^{Nl}$ is the sparse vector. A CSC model is proposed [12,22], where a global signal $\boldsymbol{X}$ can be decomposed into the multiplication of a convolutional dictionary [15], i.e., $\boldsymbol{X} = \boldsymbol{D}_1\boldsymbol{D}_2 \cdots \boldsymbol{D}_k\boldsymbol{S}_K$.

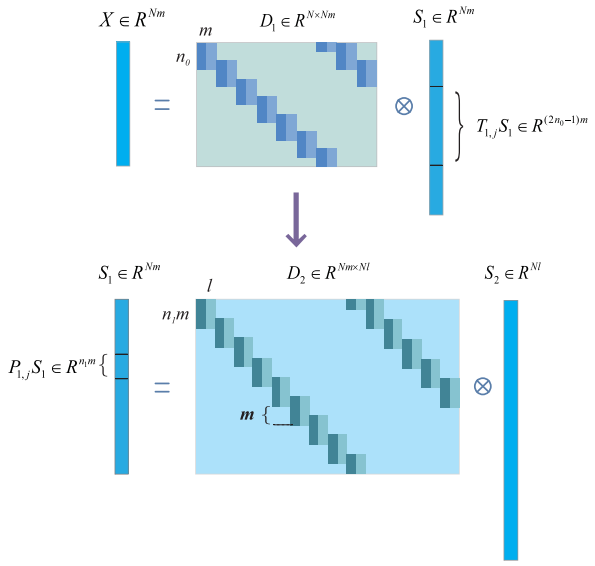

Fig. 2. The signal $\boldsymbol{X}$ decomposes into a multiplication of $\boldsymbol{D}_1$ and $\boldsymbol{S}_1$, and then $\boldsymbol{S}_1$ decomposes further into a multiplication of $\boldsymbol{D}_2$ and $\boldsymbol{S}_2$. That is, $\boldsymbol{X} = \boldsymbol{D}_1\boldsymbol{S}_1 = \boldsymbol{D}_1\boldsymbol{D}_2\boldsymbol{S}_2$.

For a global signal $\boldsymbol{X}$, a set of convolutional layer dictionaries $\{\boldsymbol{D}_i\}_{i=1}^K$ and a vector as $\boldsymbol{\lambda}$ define the deep convolutional coding problem (DCP_λ) as:

$$find \quad \{\boldsymbol{S}_i\}_{i=1}^K \quad \text{s.t.} \quad \boldsymbol{S}_{i-1} = \boldsymbol{D}_i\boldsymbol{S}_i, \quad \|\boldsymbol{S}_i\|_{0,\infty} \le \lambda_i, \forall 1 \le i \le K \tag{8}$$

where the scalar λ_i is the ith entry of $\boldsymbol{\lambda}$. For a global signal $\boldsymbol{X}$, the above problem consists of solving for a set of sparse representations, $\{\boldsymbol{S}_i\}_{i=1}^K$, where every solution of it is locally sparse and can be easily solved.

Finally, we elaborate the learning parameters for the ML-CSC model. From Eq. (8) we can obtain the sparse vector $\boldsymbol{S}_K$ through solving the *DCP* problem, and it is expressed as $\mathrm{DCP}_\lambda^*\left(\mathrm{X}, \{\mathrm{D_i}\}_{i=1}^K\right)$. Then, we extend the task-driven dictionary learning problem to the multilayer convolutional sparse representational setting problem, as follows: For a set of global signals $\{\boldsymbol{X}_j\}$, their corresponding labels $\{h(\boldsymbol{X}_j)\}$, a loss function L, and a vector $\boldsymbol{\lambda}$, we can define the deep learning problem (DLP_λ) as:

$$\min_{\{D_i\}_{i=1}^K, U} \sum_j L\left(h\left(\mathbf{X}_j\right), U, DCP_\lambda^*\left(\mathbf{X}_j \{D_i\}_{i=1}^K\right)\right) \tag{9}$$

The solution for the above equation results in an end-to-end mapping from a set of input signals to their corresponding labels. Moreover, $\boldsymbol{X}_j$ holds for the jth input signal, and U is the parameter set that determines the classifier. In this paper, in classification task, the sparse coding results are fed into the classifier to produce an output. In our paper, it is computationally challenging to find the coding results, and a nonnegative thresholding algorithm is adopted to efficiently solve the coding problem. This algorithm is similar to the hard-and soft-thresholding algorithm used for classical dictionary learning [24,25], which is equal to the ReLU activation function in a CNN.

2.2. Sparse representation methods

Now that the deep learning model has been analyzed by ML-CSC, we would like to adopt predefined dictionaries or filters, such as with wavelets and shearlets, to enhance the ability of feature extraction for a standard CNN. Therefore, in our method,

sparse representation layers are deployed to capture features such as edges and contours, which can be regarded as common primary features. Deep convolutional layers can then focus on representing the class-specific features.

Here, we use two representative methods of sparse representation: the wavelet transform and shearlet transform.

(1) Wavelet transform: Since the wavelet transform was proposed [26], it has attracted the long-term attention of researchers in the fields of image processing and computer vision. The wavelet transform is formulated as:

$$Wf(a,b) = \int_{-\infty}^{\infty} f(x)\psi_{a,b}(x)dx = \langle f, \psi_{a,b} \rangle \tag{10}$$

where $f(x)$ denotes the signal, and $\psi_{a,b}(x)$ denotes the wavelet function. Wavelets are generated from a mother wavelet function ψ, and then $\psi_{a,b}$ can be obtained as:

$$\psi_{a,b}(x) = \frac{1}{\sqrt{a}}\psi\left(\frac{x-b}{a}\right) \tag{11}$$

where a indicates the shift of scale and b indicates translation. Due to its superior characteristics in local analysis of a signal, wavelet have been applied in signal analysis tasks, such as filtering, denoising, compression, and transmission, and have also been applied in image processing tasks such as image compression, classification, recognition and diagnosis, and decontamination. Recently, a method based on the combination of deep learning and the wavelet transform has also been proposed [27–29], revealing its strong ability to represent signals. In our model, the Haar wavelet is mainly used and the effectiveness of other wavelets is verified.

(2) Shearlet transform: Although wavelets have shown excellent performance and have been applied in various visual tasks, they lack the ability to represent flexible directional information for 2D images. Based on a simple and rigorous mathematical framework, the Shearlet [30] has been shown to provide a more flexible theoretical tool for multiscale geometric analysis, as shown in Fig. 3.

For $\forall \xi = (\xi_1, \xi_2) \in \boldsymbol{R}^2$, $\xi_1 \neq 0$, let the Fourier transform $\hat{\psi}(\xi) = \hat{\psi}(\xi_1, \xi_2) = \hat{\psi}(\xi_1)\hat{\psi}(\xi_2/\xi_1)$; then, the shearlet transform of image $f \in L^2(\boldsymbol{R}^2)$ is defined as:

$$SH_{\psi}f(a,s,t) = \langle f, \psi_{a,s,t} \rangle \tag{12}$$

where $\psi_{a,s,t}(x) = \left|\det \boldsymbol{M}_{a,s}\right|^{\frac{1}{2}} \psi\left(\boldsymbol{M}_{a,s}^{-1}x - t\right)$, $\boldsymbol{M}_{a,s} = [a, \sqrt{a}s; 0, \sqrt{a}]$, and then we can obtain:

$$\left\{\psi_{a,s,t}(x) : a > 0, s \in \boldsymbol{R}, t \in \boldsymbol{R}^2\right\} \tag{13}$$

which is called the shearlet system. Each $\boldsymbol{M}_{a,s}$ can be decomposed into a shear matrix $\boldsymbol{B}_s = [1, s; 0, 1]$ and an anisotropic dilation matrix $\boldsymbol{A}_a = [a, 0; 0, \sqrt{a}]$. Thus, each matrix $\boldsymbol{M}_{a,s}$ contains two kinds of operations: anisotropic dilation performed by $\boldsymbol{A}_a$ and directional shearing performed by $\boldsymbol{B}_s$. As shown in Fig. 3, the support areas in the frequency domain of shearlets at different scales are trapeziums that lie along the line with a slope of k, and the trapeziums are symmetrical about the origin. Therefore, determined by the parameters a, s and t, shearlets have powerful local representation abilities. With the decrease of the scaling parameter a, the shearlet transform of the image can not only describe the positions of the internal edges but also indicate the direction of the edges. A discrete shearlet transform can be realized by sampling three parameters $(a, s, t) \in \boldsymbol{R}^+ \times \boldsymbol{R} \times \boldsymbol{R}^2$, which represent scaling, direction and translation, respectively.

3. Proposed method

In this section, we first describe the overall structure of the proposed network. Then, the sparse representation layer and the our backpropagation algorithm are illustrated.

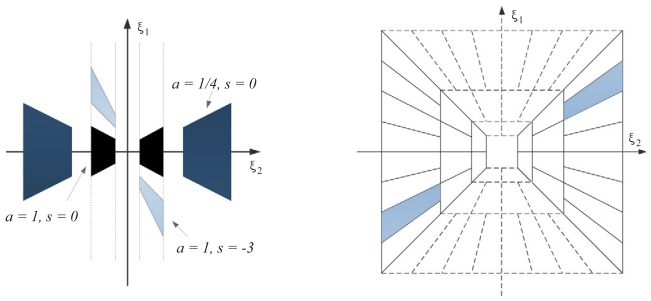

Fig. 3. Shearlet transform.

3.1. Proposed structure

The proposed structure is shown in Fig. 4, where several sparse representation layers are added directly in front of the shallow convolution layers of a standard CNN with multiple layers. The shallow representation of an image is obtained through both the convolutional layers and the sparse transforms. Then, there is a series of blocks consisting of a convolutional layer, a pooling layer and an activation function. The pooling layer that follows behind each convolutional layer is used to reduce the computation of the deeper layers and then provides a form of translation invariance of the activation function in the convolutional layer through subsampling. Moreover, each activation function is used to achieve a nonlinear mapping. The features are iteratively extracted through the connected layers next, and finally the features are reshaped and transmitted to the fully connected layers to make a prediction.

Zhang et al. [31] experimentally drew a conclusion that not all layers in the network contribute to the output equally, since CNN having structures similar to VGG are more sensitive to the weights of the deeper convolution layers that are close to the output. We also experimentally found that there is influence on classification performance, regarding whether the sparse representation layers are placed in the deeper convolutional stages or not. We initially compared the three schemes for inserting the sparse representation layers as follows: (1). Inserting the sparse representation layers in front of each convolutional layer of the entire network. (2). Inserting the sparse representation layers in front of each convolutional layer that is in the front part of the network. (3). Inserting the sparse representation layers in front of each convolutional layer that is in the back part of the network. Finally, we can see that scheme (2) leads to the best performance. This is also consistent with the result of Zhang in [31]. Based on the above results, we mainly insert the sparse representation layers into the shallow layers rather than the entire network to reduce the computational burden. With the filter bank of the sparse representation layer, more feature maps can be produced. For the wavelet transform, one map can produce 4 submaps on every scale, while the map size will reduce by half. However, we only conduct the filtering process and cancel the downsampling process so that the map size will not change. In this paper, we only use the wavelet transform with one scale. Similar to the wavelet transform, for each feature map, we use the shearlet transform to produce multiple submaps that have the same size as the original map.

From the view of the ML-CSC model, in standard CNN for a given image $\boldsymbol{X}$, it can be obtained by the first convolutional layer of a standard CNN model, as follows:

$$\boldsymbol{X} = \boldsymbol{D}_1\boldsymbol{S}_1 \tag{14}$$

where $\boldsymbol{S}_1$ denotes the sparse output as feature map, and the learned kernel of the first convolutional layer is adjusted to form a

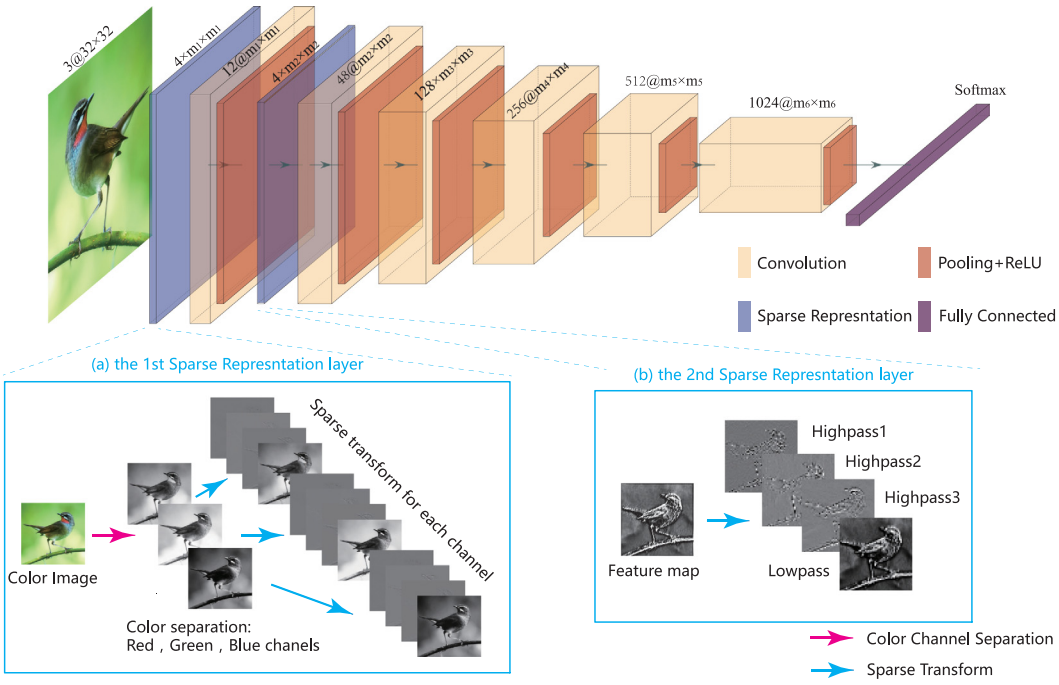

Fig. 4. The proposed structure. The whole network is illustrated at the top, where blue represents the sparse representation layers, light yellow represents the convolutional layers, deep yellow represents the pooling and activation functions, and purple represents the fully connected layers. The sparse transforms are illustrated at the bottom. Square (a) shows the first sparse representation layer in the RGB color channels and conducting the sparse transform on each channel. Square (b) shows a sparse representation layer in the deeper part conducting the sparse transform on a feature map so that more feature maps are produced. Here, we use the same filter (wavelet or shearlet) for RGB channels.

proper dictionary $\boldsymbol{D}_1$. For the second convolutional layer it learns the representation as:

$$\boldsymbol{S}_1 = \boldsymbol{D}_2\boldsymbol{S}_2 \tag{15}$$

where $\boldsymbol{S}_1$, $\boldsymbol{D}_2$, and $\boldsymbol{S}_2$ are the input as feature map, convolutional dictionary and the sparse output as feature map, respectively. Here the learned kernel in the two convolutional layer is adjusted to form a proper dictionary $\boldsymbol{D}_2$. For multi-layer convolutional network there can be described below:

$$\boldsymbol{X} = \boldsymbol{D}_1\boldsymbol{D}_2\cdots\boldsymbol{D}_K\boldsymbol{S}_K \tag{16}$$

However, once a sparse representation layer is inserted in front of the first convolutional layer in a standard CNN model, the above problems will occur as:

$$\boldsymbol{X} = \boldsymbol{D}_1^S\boldsymbol{D}_1\boldsymbol{S}_1' \tag{17}$$

where $\boldsymbol{X}$ is still the input image, $\boldsymbol{S}_1'$ is the sparse output from the first convolutional layer, and $\boldsymbol{D}_1^S$ is the predefined filter namely a sparse transform in the sparse representation layer. Because the original sparse output $\boldsymbol{S}_1$ is factorized by $\boldsymbol{D}_1^S$, then the learned kernels in the second convolutional layer will learn a representation $\boldsymbol{S}_1'$ below:

$$\boldsymbol{S}_1' = \boldsymbol{D}_2^S\boldsymbol{S}_2' \tag{18}$$

where $\boldsymbol{S}_2'$ is the sparse output from the second convolutional layer, $\boldsymbol{D}_2^S$ denotes the predefined filter in the second sparse representation layer that is inserted in front of the second convolutional layer of a standard CNN model.

Then combining the Eq. (17) and (18) we can obtain as follows:

$$\boldsymbol{X} = \boldsymbol{D}_1^S\boldsymbol{D}_1\boldsymbol{D}_2^S\boldsymbol{S}_2' \tag{19}$$

The above discussion indicates that the learning process of the convolutional layer is affected by the previous sparse transform results [15]. Therefore, the representation task of the our network is partly undertaken by the predefined filters such as wavelet or shearlet. In comparison with the data-driven schemes, the proposed framework is actually a combination of data-driven learned kernels and analytically designed kernels.

Finally, the proposed network in this paper can be expressed as:

$$\boldsymbol{X} = \boldsymbol{D}_1^S\boldsymbol{D}_1\boldsymbol{D}_2^S\boldsymbol{D}_2\cdots\boldsymbol{D}_K\boldsymbol{S}_K' \tag{20}$$

It also has been proven [14] that there is local sparseness of the sparse representation at the ith stripe, and $\boldsymbol{S}_i$ satisfies:

$$\|\boldsymbol{S}_i\|_{0,\infty} \le \|\boldsymbol{S}_K\|_{0,\infty} \prod_{j=i+1}^{K} \|\boldsymbol{D}_j\|_0 \tag{21}$$

where $l_{0,\infty}$ is the mentioned measure of local sparseness, l_0 is the l_0 norm for the global sparseness measurement and $\|\boldsymbol{D}_j\|_0$ is equal to the number of nonzeros in any atom from $\boldsymbol{D}_j$. It has been known that more abstraction representations can be obtained at a higher depth, so the $l_{0,\infty}$ norm is used to limit the ability of the depth of the representation for the deep CNN.

3.2. Learning procedure

Let the set of $P(P \ge 1)$ dictionaries $\left\{\boldsymbol{D}_p\right\}_{p=1}^{P}$ denote the learned layers, the set of $Q(Q \ge 1)$ dictionaries $\left\{\boldsymbol{D}_q^S\right\}_{q=1}^{Q}$ denote the predefined sparse representation layers, where the two kinds of layers are constructed by shifting kernels to all possible positions. The aforementioned learning task for the signal set $\left\{\boldsymbol{X}_j\right\}$ in our model can be formulated as:

$$\min_{\{\boldsymbol{D}_p\}_{p=1}^{P},U} \sum_j L\left(h\left(\boldsymbol{X}_j\right), U, DCP_\lambda^*\left(\boldsymbol{X}_j, \left\{\boldsymbol{D}_p, \boldsymbol{D}_q^S\right\}_{p+q}^{P+Q}\right)\right) \tag{22}$$

where the learned dictionaries and predefined dictionaries both contribute to the convolutional sparse coding, while only the learned dictionaries are searched during the training process.

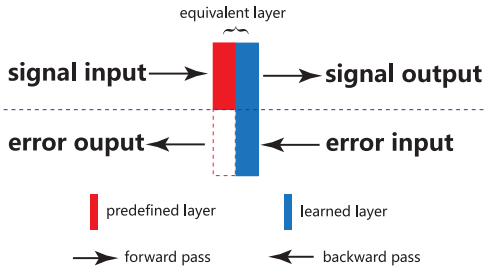

Fig. 5. For the forward pass, both the predefined filters and the learned filters have an effect on the output, while only the learned filters are updated in the backward pass.

With this formulation, we can perceive that our proposal is to add some fixed dictionaries to broaden the search scope for the optimal solution.

Before being modified by sparse representation layers, the weights of the convolutional layers in standard CNNs are updated using the backpropagation (BP) algorithm [32]. Unlike the convolutional layers, sparse representation layers in our method are predefined and remain fixed during the backpropagation process. During training, kernels of sparse representation layers skip the update when the backward error back propagates, which can be easily implemented on popular deep learning platforms. In other words, sparse representation layers are only effective in the forward pass.

The sparse transform is equivalent to a preprocessing operation in the shallow layer of the network, where sparse representation can extract high-frequency features such as edges, contours, shapes and textures; thus, the subsequent CNN can directly convolve the features, thereby improving the performance of the CNN. In our model, a sparse representation layer is directly connected with a convolutional layer, i.e., the predefined filter and learned filters are combined by Hadamard product operator, and considering the two stages as a whole. For the connection of a sparse representation layer and a learned layer, as shown in Fig. 5, the equivalent convolutional filter kernels of this connection can be formulated as:

$$f_{eq} = f_s \circ f_l \tag{23}$$

where "$\circ$" refers to the Hadamard product, f_{eq} is an equivalent convolutional filter, f_l is the learned filter and f_s is the sparse representation filter or filter banks. In particular, the produced channel numbers by using this combination vary with the specific sparse transform. Thereby, the modified update can be formulated as:

$$\delta = \frac{\partial L}{\partial f_{eq}} = f_s \circ \frac{\partial L}{\partial f_l} \tag{24}$$

$$f_{eq}^{k+1} = f_{eq}^{k} - \eta\delta \tag{25}$$

where δ is the error, L is the loss function, and η is the learning rate. Equipped with sparse filters, a more efficient and flexible representation can be obtained by extending feature channels.

4. Experimental verification

In this section, the sparse representation enhancement will be elaborated in detail based on the standard CNN. We evaluated our method on the Brodatz, CIFAR10/100 [33], Food-101 [34], DTD [35] and ImageNet. In our experiments, a platform equipped with an i7-8700k CPU, 16 G memory, and an NVIDIA GeForce GTX 2060 SUPER are used.

Fig. 6. Five categories chosen from the dataset with 101 kinds of food in total.

4.1. General configurations

Our method is implemented on a 7-layer AlexNet-like network, which is used as a reference baseline for the CNN model. As shown in Table 1, the proposed network consists of 6 feature extraction blocks (convolutional layers) and 1 fully connected layer. In front of the first two convolutional layers there were two sparse representation layers, which were constructed by a 1-level Haar wavelet transform or a 1-level shearlet transform with 4 decomposition directions to realize the sparse transform. After each convolutional layer is a max pooling layer and an ReLU function. During the training stage, we applied data augmentation and preprocessing for all datasets with random horizontal flipping, resizing and mean subtraction. To investigate directly the effects of the sparse representation layer, we did not use dropout or pruning [36,37] in our method. We used a batchsize of 100 for CIFAR10, CIFAR100 and ImageNet. For Food-101, DTD and Brodatz, due to the small amount of the images, we used a batchsize of 15 to have more iterations per epoch on a small training set. For all training, adaptive moment estimation (ADAM) [38] optimization algorithm was adopted, and the initial learning rate was set to 0.001. On each dataset, different networks were trained with 200 epochs, and the best performance of each network was recorded. To evaluate the proposed method, our approach was compared with both the baseline CNN and some other representative networks.

Note that a 7-layer AlexNet-like network is selected as the baseline network that the sparse transforms of the first two convolutional layers were removed from Table 1.

4.2. Validation on different datasets

1. FOOD-101 Dataset

In the FOOD-101 dataset, there are 101 categories in total that contain visually and semantically similar food images, with 1000 images in each category. In this section, we built a subset of the dataset using 5 kinds of food: pie, salad, churros, donuts and macaron, as shown in Fig. 6. In each category, 900 images were used for training and 100 images were used for testing.

Table 2 shows the performance of different networks. As can be seen from the table, our method achieved satisfactory results. The dataset is quite simple, so ResNet 18 and VGG 16 may tend to overfit on this dataset, resulting in poor classification accuracy. The baseline network achieved an accuracy of 94.6%, while the baseline enhanced by the Haar wavelet transform achieved an accuracy of 96.0%, and the baseline enhanced by the shearlet transform reached 97.8% accuracy, resulting in an improvement in accuracy of 1.4% and 3.2%, respectively.

2. CIFAR10 Dataset and CIFAR100 Dataset

Table 1
Proposed structure.

Input	Image input	Images of 3 × 32 × 32, normalized	Input
Sparse transform1	Wavelet/Shearlet transform	Images of 12 × 32 × 32, channels increased	layer1
conv1	Convolution	32 × 3 × 3 convolutions with stride = 1, padding = 1	
relu1	ReLU	ReLU activation function	
Sparse transform2	Wavelet/Shearlet transform	Images of 48 × 32 × 32, channels increased	layer2
conv2	Convolution	64 × 3 × 3 convolutions with stride = 1, padding = 1	
relu2	ReLU	ReLU activation function	
pool1	Max pooling	2 × 2 maxpooling	
conv3	Convolution	128 × 3 × 3 convolutions with stride = 1, padding = 1	layer3
relu3	ReLU	ReLU activation function	
conv4	Convolution	256 × 3 × 3 convolutions with stride = 1, padding = 1	layer4
relu4	ReLU	ReLU activation function	
pool2	Max pooling	2 × 2 maxpooling	
conv5	Convolution	512 × 3 × 3 convolutions with stride = 1, padding = 1	layer5
relu5	ReLU	ReLU activation function	
conv6	Convolution	1024 × 3 × 3 convolutions with stride = 1, padding = 1	layer6
relu6	ReLU	ReLU activation function	
pool3	Global maxpooling	8 × 8 global maxpooling	
fc1	Fully Connected	Fully connected layer	layer7

Table 2
Performances on FOOD-101.

Method	Baseline	ResNet18 [39]	VGG16 [3]	Baseline+Haar Wavelet	Baseline+Shearlet
Top-1Acc	94.6%	89.9%	89.4%	96.0%	97.8%

(1) CIFAR10: We mainly tested different networks on CIFAR10 and compared three different networks on CIFAR100. The CIFAR10 dataset consists of 60,000 color images in 10 classes, with the size of 32 × 32, including 50,000 training images and 10,000 testing images. This dataset contains various categories with object orientation and scale variations. Experiments were conducted to compare our method with the baseline networks.

As shown in Table 3, the best performance now belongs to DenseNet121, which has achieved a top-1 accuracy of 95.04% on our experimental platform. Then, ResNet110 followed, which reached an accuracy of 93.87%. VGG 16 also showed promising results at 91.23%. The accuracy of NIN was 89.64%. The baseline network achieved 88.08% accuracy, while the baseline equipped with sparse representation layers showed obvious improvements. When equipped with the Haar wavelet, the classification accuracy is 89.54%, which is a 1.46% improvement; when equipped with the shearlet transform, the classification accuracy is 89.40%, which is a 1.32% improvement. Moreover, we also added the two kinds of sparse transforms to the shallow layers of VGG 16, after which VGG 16 gained improvements of 0.32% and 0.33%. Thus, the effect of the sparse transform in VGG 16 is not as obvious as that of AlexNet.

More specifically, the top-1 accuracy of the classification for each class is shown in Fig. 7. It is clear that for most classes, our method has a higher accuracy than the baseline network. In addition, for classes such as truck, frog, bird and car, our method performs significantly better than the baseline network. There is a 6.89% improvement in truck recognition, an 11.31% improvement in frog recognition, a 7.74% improvement in bird recognition and a 6.25% improvement in car recognition, which validates the effectiveness of the feature extraction ability of sparse representation layers.

(2) CIFAR100: The CIFAR100 dataset has 60,000 color images in 100 classes, with the size of 32 × 32. We compared three different networks on CIFAR100: the baseline network, the baseline combined with the wavelet and the baseline with the shearlet. Table 4 shows the results, here the wavelet transform still improves the performance of the baseline network from 61.41% to 64.59%. However, the shearlet worsens the performance from 61.41% to 59.87%.

Table 3
Performances on each class of CIFAR10.

Method	Layers	#param	top-1 Acc
Baseline	7	6.5M	88.08%
Baseline+Haar Wavelet	7	6.5M	89.54%
Baseline+Shearlet	7	6.5M	89.40%
VGG16	16	134.3M	91.23%
VGG16+Haar Wavelet	16	134.3M	91.54%
VGG16+Shearlet	16	134.3M	91.56%
ResNet110	110	1.7M	93.87%
NIN [40]	–	–	89.64%
DenseNet121 [41]	121	7.9M	95.04%

Table 4
Performances on CIFAR100.

Network	Baseline	Baseline + wavelet	Baseline + shearlet
Accuracy	61.41%	64.59%	59.87%

3. Describable Textures Dataset

The Describable Textures dataset is a texture database, consisting of 5640 images that are organized according to a list of 47 terms. There are 120 images per category, and in these images, at least 90% of their surfaces represents the attributes of the category. To further validate the texture capturing ability of sparse representation layers, we constructed a subset of this set by using 10 categories and conducted experiments on the dataset to compare our method and other methods, as shown in Fig. 8. In each category, 100 images were used for training and 20 images were used for testing.

As shown in Table 5, on this dataset, our proposed methods perform better than other networks, although the general performance is not so satisfactory. This dataset is challenging since the images in this dataset contain various directional information. Directional information can be described through the sparse representation layer, which leads to a higher classification accuracy. The baseline network achieves 60.0% accuracy. MobileNet and ShuffleNet are two networks with light structures, achieving accuracies of 22.0% and 22.5%, respectively. The baseline network

Table 5
Performances on DTD.

Method	Baseline	MobileNet [42]	ShuffleNet [43]	Baseline+Haar Wavelet	Baseline+Shearlet
Top-1 Acc	60.0%	22.0%	22.5%	60.5%	61.0%

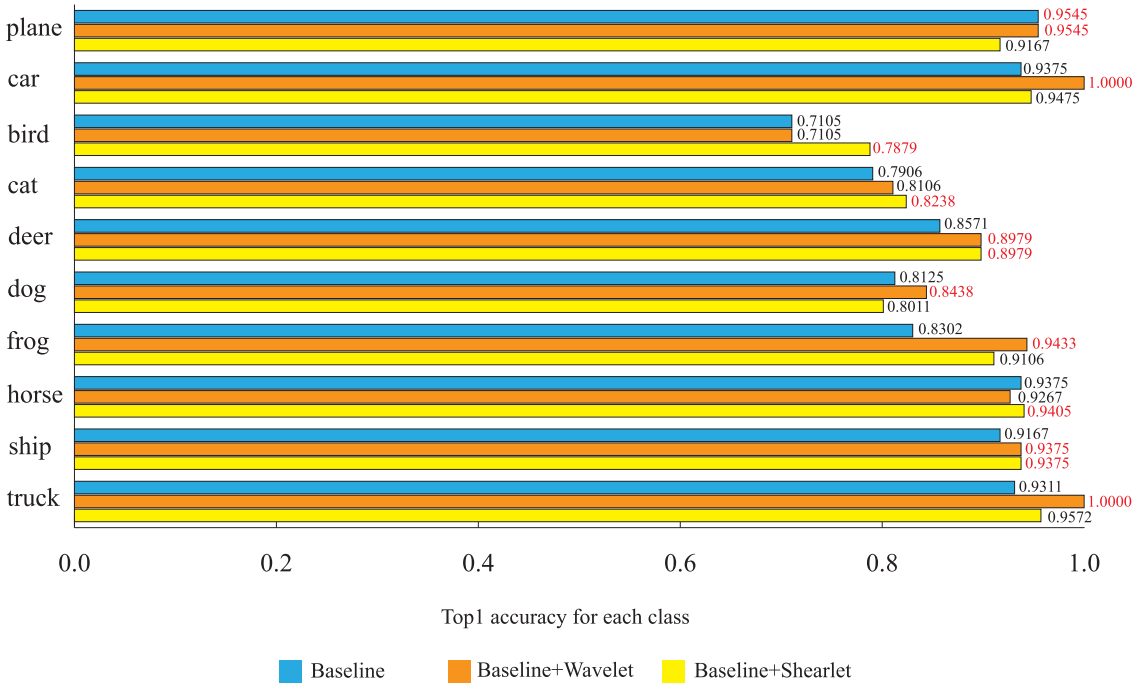

Fig. 7. Top-1 accuracy for each class classification.

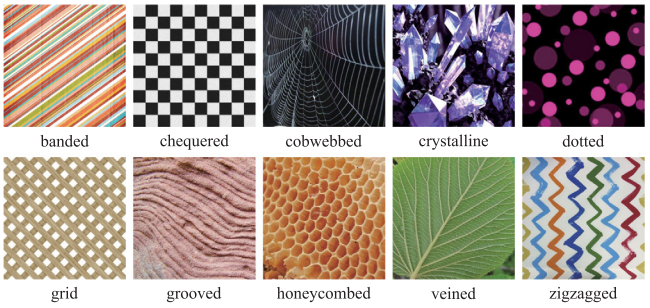

Fig. 8. 10 categories of texture chosen from the original DTD.

equipped with the Haar wavelet can improve the accuracy by 0.5%, and the baseline equipped with the shearlet can improve the accuracy by 1.0%. It can be seen in Fig. 9 that there are obvious differences in the feature maps of shallow layers between the baseline network alone and baseline network with the shearlet transform.

4. Brodatz Dataset

We also conducted another experiment on the famous texture dataset: Brodatz. The Brodatz dataset contains 112 categories of grayscale texture images of size 640 × 640, with only one image in each category. Nevertheless, we manually constructed a subset with 10 types of selected textures, as shown in Fig. 10. To enlarge the dataset, each original 640 × 640 image was randomly cropped into 500 patches with a size of 64 × 64, of which 450 patches were randomly chosen as training data and the rest were used as testing data. Moreover, we randomly rotated some of the training and testing data to create a rotated version because most of the original patches share the same direction. The networks were trained and tested on both datasets, and the results are shown in Table 6

It can be observed that all methods perform well on this dataset. In the original version, VGG 11 and ShuffleNet achieve 99.8% accuracy, which performs better than our proposed method. For the rotated version, the recognition accuracy of ShuffleNet dropped by 0.6%, and VGG 11 dropped by 0.4%. Both of our methods have dropped by 0.2%, indicating that our method, compared to VGG 11 and ShuffleNet, is more robust to the rotation transform.

5. ImageNet dataset

Ultimately, we conducted an experiment on the ImageNet dataset. Due to the excessively large amount of images and a limited computation ability, we used the tiny ImageNet dataset (Stanford CS231N) that contains 100,000 color images that are categorized into 200 classes. Table 7 shows the result. This result is similar to the result on CIFAR100. The wavelet plays a positive role for the baseline, while the shearlet plays a negative role.

4.3. Performances with different transforms

To evaluate the performance of different transform settings, experiments were conducted to verify the effectiveness of different wavelets and different decomposition levels. In this case, we still use the same structure of the network that was previously used in sharing with baseline network, only changing the sparse representation layer. For the wavelet transform, we train the networks on CIFAR10 with 200 epochs and employ ADAM optimization with a learning rate of 0.001. For the shearlet transform, a subset of Food-101 is used. The network has undergone 100 epochs of training, and the optimization algorithm is the same as before.

(1) As shown in Table 8, the Haar wavelet, Daubechies wavelets, Coiflet wavelets and Biorthogonal wavelets were tested.

Table 6
Performances on Brodatz.

Method	Baseline	VGG11	ShuffleNet	Baseline+Haar Wavelet	Baseline+Shearlet
Acc on Brodatz	99.4%	99.8%	99.8%	99.6%	99.6%
Acc on rotated version	99.0%	99.4%	99.2%	99.4%	99.4%

Table 7
Performances on ImageNet.

Network	Baseline	Baseline + wavelet	Baseline + shearlet
Accuracy	43.1%	43.9%	41.1%

Table 8
Performances with different wavelets.

Wavelet	Haar	db1	db2	db3	db4
Accuracy	89.54%	89.48%	89.94%	89.03%	88.82%
Wavelet	bior1.1	bior2.2	bior3.3	coif1	coif2
Accuracy	89.60%	89.53%	89.03%	88.93%	89.63%

Table 9
Performances with different shearlet decomposition levels.

Decomposition level	1	2	3	4
High-frequency subbands	4	12	28	60
Top-1 accuracy	97.8%	99.6%	99.8%	99.9%
Training time for 1 epoch	72 s	89 s	162 s	299 s

Since all images in CIFAR10 have small sizes, the largest kernel size of the wavelet transform is limited to 8 × 8, which is achieved by the db4 wavelet. As can be seen from the table, the best accuracy of the db2 wavelet is 89.94%.

(2) In Table 9, the performances of the shearlet with different decomposition levels are presented. Here, we use 4 different decomposition levels, including 1, 2, 3 and 4. The more decomposition levels we use, the more directional information we obtain. According to the experimental results, it can be observed that more decomposition levels result in higher classification accuracy, while the processing is also more time consuming. There is great improvement between 1-level decomposition and 2-level decomposition in terms of accuracy (1.8%), while the running time does not increases much (17 s). However, when using 3-level and 4-level decompositions, the running time will be greatly increased (73 s and 137 s, respectively), but the performance will not be improved as much (1%). As shown in Fig. 11, after 2-level decomposition, the improvement of classification slows down, while the increase of running time accelerates.

4.4. Sparseness and FLOPs

In this section, we measure the sparseness of the convolutional kernels in different networks and record the training time to see how the sparse representation layer influences the network. Here, we mainly compare these three networks: the baseline network, the baseline with the wavelet and the baseline with the shearlet.

(1) Sparseness: We adopted the l_1 norm to measure the sparseness of the convolutional kernels from these networks. We first trained the networks on Food-101 and then loaded the network data to calculate the sparseness. Since the sparse transforms are inserted in the shallow layers of the network, we mainly studied the self-learned kernels of the first convolutional layers behind the first sparse representation layer. There is no sparse representation layer in the baseline network, so we used its first convolutional layer. There was a great number of filters, and we randomly selected 10 filters from each kind and calculated the l_1 norm. Table 10 shows the result.

From Table 10, we can see that with the sparse transforms, the self-learned kernels of the convolutional layer in the network become more sparser, i.e., the parameters of the convolutional kernels became small. The average l_1 norm of the kernels in the baseline network is 1.1095, and that of the baseline network with the wavelet is 0.5490, while the baseline network with the shearlet is 0.3613. The sparse transform helps the self-learned kernels of the convolutional layer improve the sparseness by approximately half. Although the baseline with the shearlet achieves better average sparseness, the shearlet can sometimes lead to worse results according to the experiments above. Here, we suppose that to achieve better performance, there are likely other conditions for kernels to satisfy rather than only sparsity, which will be investigated in our future research.

Table 10
Sparseness of the kernels of the 1st convolutional layer.

sparseness / filter / nets	filter1	filter2	filter3	filter4	filter5
baseline	0.9702	1.2331	1.2627	1.3338	0.7488
baseline+wavelet	0.7691	0.4243	0.6051	1.2278	0.2451
baseline+shearlet	0.3376	0.4119	0.3459	0.2236	0.3786

sparseness / filter / nets	filter6	filter7	filter8	filter9	filter10	average
baseline	1.0863	1.2721	1.4769	0.6653	1.0453	1.1095
baseline+wavelet	0.7423	0.5838	0.1813	0.5769	0.1338	0.5490
baseline+shearlet	0.4028	0.3428	0.3926	0.4431	0.3341	0.3613

Table 11
Sparseness of the kernels of the 5th convolutional layer.

sparseness / filter / nets	filter1	filter2	filter3	filter4	filter5
baseline	0.9125	0.6269	0.8197	0.5862	0.7115
baseline+wavelet	0.7787	0.8686	0.6753	1.7190	0.9209
baseline+shearlet	0.8601	0.3865	1.5417	0.5461	0.8461

sparseness / filter / nets	filter6	filter7	filter8	filter9	filter10	average
baseline	0.4333	0.6585	0.5608	0.8533	1.0076	0.7170
baseline+wavelet	1.2081	1.4437	1.3752	1.7825	1.1515	1.1924
baseline+shearlet	0.7658	1.0957	2.0293	0.5022	0.7717	0.9525

Moreover, we also calculated the sparseness of learned kernels in the 5th convolutional layer where no sparse representation layers are inserted. Table 11 shows that the sparseness of the baseline network achieves 0.7170 on average. The baseline network with the wavelet achieves 1.1924 on average, and the baseline network with the shearlet achieves 0.9525 on average. At the 5th convolutional layer, the baseline network shows the highest sparseness. To be more specific, we calculated the variance of the results. In the baseline network, the variance of the sparseness of the 10 kernels is 0.0317. The variance of the baseline network with the wavelet is 0.1490. The variance of the baseline network with the shearlet is 0.2555. Here the baseline network can also be more concentrated.

(2) FLOPs: MACs were calculated to evaluate the complexity (calculation burden) of the networks and the storage space occupied by the training parameters; Table 12 shows the result:

It can be seen from Table 12 that the trainable parameters are almost the same and there is slight increase of MACs in our proposed method.

According to the results in this section, a sparse representation layer will force the self-learned kernels of the next convolutional

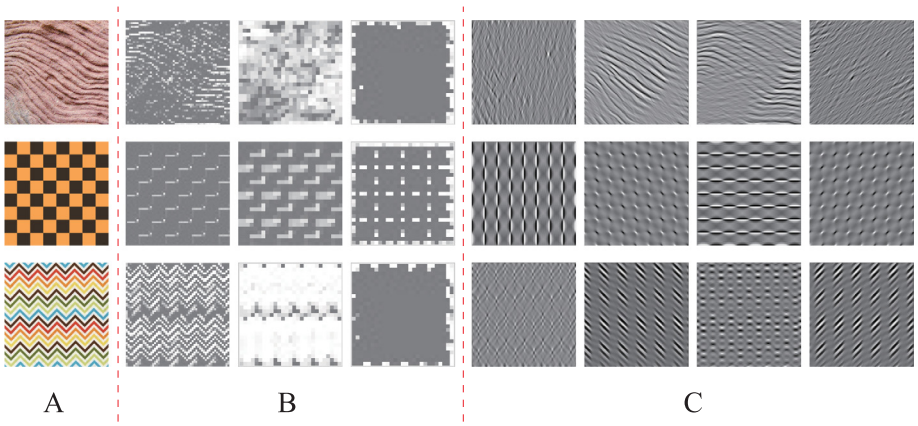

Fig. 9. Features extracted by kernels of the shallow layers of the network: (A) the original texture image, (B) the corresponding features extracted by the shallow layers of the baseline network, (C) the corresponding features extracted by the shallow layers of the baseline network with shearlet transform. By this comparison, it can be observed that the shearlet-enhanced baseline network preserves the direction characteristics better than does baseline network.

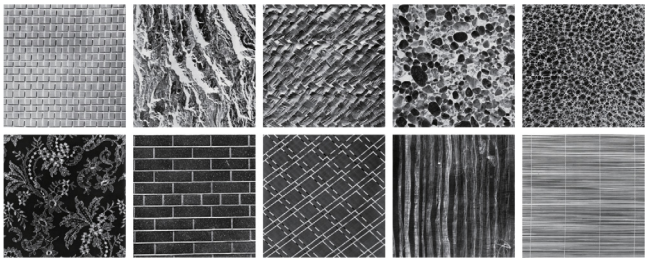

Fig. 10. Subset of Brodatz.

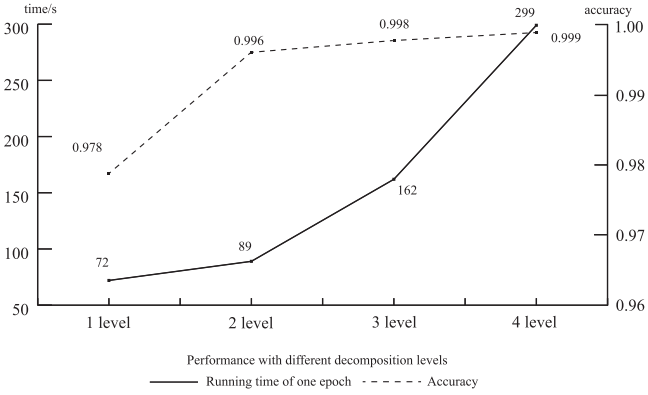

Fig. 11. Tendency of accuracy and running time with the increase of shearlet decomposition levels.

layer to become sparser. Moreover, with the sparse representation layers, the sparseness of learned kernels in the deeper convolutional layers of the network are likely changed. Such a change of sparseness may lead to the difference of the performances. However, based on the classification experiments results, the change of sparseness does not always guarantee the improvement of performance. There are occasions in which the recognition accuracy of the baseline network is superior to the baseline with the shearlet, while the wavelet seems to improve the performance of original networks on the various datasets.

4.5. Summary

In this section, we mainly validate the effectiveness of the proposed method, that is, the sparse representation layer in the CNN.

Table 12
Parameters and FLOPs.

Model	Param(M)	MACs(G)
Baseline	6.50	0.49
Baseline+wavelet	6.49	0.51
Baseline+shearlet	6.50	0.50

Although it does not always achieve the best performance compared with other methods, the sparse transform actually shows its effectiveness in the classification task of various datasets. In combination with the predefined sparse transforms, the network usually obtains a better and robust representation ability and a more powerful feature extraction capability. This can also achieve higher recognition accuracy without increasing an excessive computational burden.

5. Conclusion

In this paper, we first introduced the state-of-the-art ML-CSC model, which perceives the standard CNN as a set of data-driven learnable dictionaries. Along with this model, we then added sparse representation layers to a target network to enhance the ability of feature extraction. The sparse representation layers have been realized by a predefined wavelet and shearlet transform and can efficiently extract direction information such as edges and contours, with which the target network can learn a more robust mapping. Experiments show that our method can improve the performance of image classification on different datasets. In our future work, we will further explore the use of mathematical or signal processing methods to analyze CNN theory and try to find ways to improve the network with more theoretical support.

CRediT authorship contribution statement

Guoan Yang: Conceptualization, Methodology, Writing - review & editing, Supervision. **Junjie Yang:** Conceptualization, Methodology, Software, Data curation, Writing - original draft. **Zhengzhi Lu:** Visualization, Investigation, Data curation. **Deyang Liu:** Software, Validation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was partially funded by the National Natural Science Foundation of China under grant numbers 61673314, 61573273 and the National Key Research and Development Program of China under grant 2018YFB1700104.

References

[1] Y. Lecun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, et al., Backpropagation applied to handwritten zip code recognition, Neural Comput. 1 (4) (1989) 541–551.
[2] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ImageNet classification with deep convolutional neural networks, Communnications of the ACM 60 (6) (2017) 84–90.
[3] K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, Technical Report, Visual Geometry Group, Department of Engineering Science, University of Oxford, University of Oxford and Google Deep Mind, 2014.
[4] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015, pp. 3431-3440.
[5] J. Donahue, L.A. Hendricks, M. Rohrbach, et al., Long-term recurrent convolutional networks for visual recognition and description, IEEE Trans. Pattern Anal. Mach. Intell. 39 (4) (2017) 677–691.
[6] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, IEEE Trans. Pattern Anal. Mach. Intell. 39 (6) (2017) 1137–1149.
[7] S.S. Sarwar, P. Panda, K. Roy, Gabor filter assisted energy efficient fast learning convolutional neural networks, in: IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED), Taipei, 2017, pp. 1-6.
[8] T. Wiatowski, H. Bölcskei, Deep convolutional neural networks based on semi-discrete frames, in: IEEE International Symposium on Information Theory (ISIT), Hong Kong, 2015, pp. 1212-1216.
[9] J. Bruna, S. Mallat, Invariant scattering convolution networks, IEEE Trans. Pattern Anal. Mach. Intell. 35 (8) (2013) 1872–1886.
[10] Thomas Wiatowski, Helmut Bolcskei, A mathematical theory of deep convolutional neural networks for feature extraction, IEEE Trans. Inform. Theory 64 (3) (2017) 1845–1866.
[11] J.-C. Ye, Y. Han, E. Cha, Deep convolutional framelets: A general deep learning framework for inverse problems, SIAM J. Imaging Sci. 11 (2) (2018) 991–1048.
[12] V. Papyan, J. Sulam, M. Elad, Working locally thinking globally: Theoretical guarantees for convolutional sparse coding, IEEE Trans. Signal Process. 65 (21) (2017) 5687–5701.
[13] V. Papyan, J. Sulam, M. Elad, Working locally thinking globally - part II: Stability and algorithms for convolutional sparse coding, 2016, arXiv:1607.02009.
[14] V. Papyan, Y. Romano, M. Elad, Convolutional neural networks analyzed via convolutional sparse coding, J. Mach. Learn. Res. 18 (83) (2017) 1–52.
[15] R. Rubinstein, M. Zibulevsky, M. Elad, Double sparsity: Learning sparse dictionaries for sparse signal approximation, IEEE Trans. Signal Process. 58 (3) (2010) 1553–1564.
[16] P. Liu, H. Zhang, K. Zhang, L. Lin, W. Zuo, Multi-level wavelet-CNN for image restoration, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Salt Lake, 2018, pp. 773-782.
[17] Y. Zhou, Q. Ye, Q. Qiu, J. Jiao, Oriented response networks, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, 2017, pp. 4961-4970.
[18] B. Cîrstea, L. Likforman-Sulem, Tied spatial transformer networks for digit recognition, in: International Conference on Frontiers in Handwriting Recognition (ICFHR), Shenzhen, 2016, pp. 524-529.
[19] S. Luan, C. Chen, B. Zhang, J. Han, J. Liu, Gabor convolutional networks, IEEE Trans. Image Process. 27 (9) (2018) 4357–4366.
[20] X. Sun, N.M. Nasrabadi, T.D. Tran, Supervised deep sparse coding networks for image classification, IEEE Trans. Image Process. 29 (2020) 405–418.
[21] D.L. Donoho, M. Elad, Optimally sparse representation in general (nonorthogonal) dictionaries via l_1 minimization, Proc. Natl. Acad. Sci. 100 (5) (2003) 2197–2202.
[22] D. Glasner, S. Bagon, M. Irani, Super-resolution from a single image, in: IEEE International Conference on Computer Vision (ICCV), Kyoto, 2009, pp. 349-356.
[23] J. Mairal, F. Bach, J. Ponce, Task-driven dictionary learning, IEEE Trans. Pattern Anal. Mach. Intell. 34 (4) (2012) 791–804.
[24] S.S.B. Chen, D.L. Donoho, M.A. Saunders, Atomic decomposition by basis pursuit, SIAM J. Sci. Comput. 20 (1) (1998) 33–61.
[25] J.A. Tropp, A.C. Gilbert, Signal recovery from random measurements via orthogonal matching pursuit, IEEE Trans. Inform. Theory 53 (12) (2007) 4655–4666.
[26] S.G. Mallat, A theory for multiresolution signal decomposition: The wavelet representation, IEEE Trans. Pattern Anal. Mach. Intell. 11 (7) (1989) 674–693.
[27] E. Kang, J. Min, J.C. Ye, A deep convolutional neural network using directional wavelets for low-dose X-ray CT reconstruction, Med. Phys. 44 (10) (2017) E360–E375.
[28] T. Williams, R. Li, Advanced image classification using wavelets and convolutional neural networks, in: IEEE International Conference on Machine Learning and Applications, Anaheim, CA, 2016, pp. 233-239.
[29] L. Fang, H. Zhang, J. Zhou, et al., Image classification with an RGB-channel nonsubsampled contourlet transform and a convolutional neural network, Neurocomputing 396 (2020) 266–277.
[30] W.-Q. Lim, The discrete shearlet transform: A new directional transform and compactly supported shearlet frames, IEEE Trans. Image Process. 19 (5) (2010) 1166–1180.
[31] Chiyuan Zhang, Samy Bengio, Yoram Singer, Are all layers created equal?, 2019, arXiv:1902.01996.
[32] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back propagating errors, Nature 323 (6088) (1986) 533–536.
[33] A. Krizhevsky, G. Hinton, Learning multiple layers of features from tiny images, in: Handbook of Systemic Autoimmune Diseases, 1, (4) 2009.
[34] L. Bossard, M. Guillaumin, L.V. Gool, Food-101-mining discriminative components with random forests, in: European Conference on Computer Vision (ECCV), in: Lecture Notes in Computer Science, vol. 8694, Zurich, 2014, pp. 446–461.
[35] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, A. Vedaldi, Describing textures in the wild, in: IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 3606-3613.
[36] Long Chen, Lei Fan, Jianda Chen, et al., A full density stereo matching system based on the combination of CNNs and slanted-planes, IEEE Trans. Syst. Man Cybern. 50 (2) (2020) 397–408.
[37] Linghua Zeng, Xinmei Tian, Accelerating convolutional neural networks by removing interspatial and interkernel redundancies, IEEE Trans. Cybern. 50 (2) (2020) 452–464.
[38] D. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv:1412.6980.
[39] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770-778.
[40] Lin Min, Q. Chen, S. Yan, Network in network, 2013, arXiv:1312.4400.
[41] G. Huang, Z. Liu, L.v.d. Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 2261-2269.
[42] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, et al., MobileNets: Efficient convolutional neural networks for mobile vision applications, 2017, arXiv:1704.04861.
[43] X. Zhang, X. Zhou, M. Lin, J. Sun, ShuffleNet: An extremely efficient convolutional neural network for mobile devices, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, 2018, pp. 6848-6856.