Knowledge-Based Systems 205 (2020) 106315

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Learning sparse features with lightweight ScatterNet for small sample training

Zihao Dong [a,*], Ruixun Zhang [b], Xiuli Shao [a], Zengsheng Kuang [a]

[a] College of Computer Science, Nankai University, Tianjin 300350, China
[b] MIT Laboratory for Financial Engineering, Cambridge, MA 02142, USA

ARTICLE INFO

Article history:
Received 20 December 2019
Received in revised form 17 July 2020
Accepted 22 July 2020
Available online 25 July 2020

Keywords:
Lightweight
ScatterNet
Sparse features
Learnable filters
Hybrid architecture

ABSTRACT

Convolutional neural networks (CNNs) have recently achieved impressive performances in image processing tasks such as image classification and object recognition. However, CNNs typically have a large number of parameters, leading to their requirement of a large number of training samples to extract spatial features. To address these limitations, we propose a lightweight ScatterNet with the learnable weight matrix and sparse transformation such as scale transformation and translation to learn sparse filters. This filter based on ScatterNet uses *He* initialization algorithm and learns from input images which are viewed as two-directional sequential data in the initial stage of model training. A Strip-Recurrent module sweeps both horizontally and vertically across the image to compress feature matrices. Then, ScatterNet decomposes the above feature matrices as a learned mixture of different harmonic functions to integrate the spectral analysis into CNNs. Finally, we combine the sequential and spectral features to build our hybrid architectures to complete image classification and segmentation. These architectures can obtain good classification accuracy on both small and large training datasets. Our proposed method is evaluated at both layer and network levels on five widely-used benchmark datasets: MNIST, CIFAR-10, CIFAR-100, Small NORB and Tiny ImageNet. We also study other small sample problems such as medical image segmentation and image classification based on few-shot learning. Experiments show that our proposed layer and hybrid model achieves better accuracy for small sample training.

1. Introduction

Convolutional neural networks (CNNs), as a multilayer learning architecture, are widely used in image classification tasks and particularly effective in extracting spatial features. They have become the preferred model and shown state-of-the-art performance in several applications, and have won the ImageNet Large Scale Visual Recognition Competition (ILSVRC) [1] with AlexNet [2], VGGNet [3] and ResNet [4]. Therefore, CNNs have received immense success in multiple computer vision tasks. Most of recent developments focus on modifying network structures and data augmentation such as now DenseNet [5]. In addition, they relay on the minimization of loss functions such as Cross Entropy, OHEM [6] and Focal loss [7] to improve the accuracy of feature extraction.

With the rapid development of CNNs in the field of image processing, Recurrent Neural Networks (RNNs) have been applied to the analysis of sequential data such as sound and text. More specifically, it can be extended to applications including Natural Language Processing(NLP) [8] and machine translation [9,10]. However, RNN related modules are rarely used in feature extraction of image analysis.

Despite their successful applications, these deep learning models still face several challenges. For example, many architectures based on CNNs always require a large number of training samples, making them prone to overfitting, especially on small datasets. To address the small sample problem, ScatterNet [11] with pre-defined filter bank is proposed to extract invariant features, which is extended to incorporate rotation and scale invariance [12–14] to improve the accuracy of image classification. These enhancements overcome the shortcoming that CNNs only achieve the rotation invariance by data augmentation. However, handcrafted or pre-defined filter banks limit the learning process of models and may not generate the real distribution of the input data. Therefore, the learning filter banks methods [15,16] are proposed to predict meaningful features from small samples by unsupervised approaches. One advantage of these methods is that they do not need to tune the filters on different datasets. Hybrid networks [17,18] combine the advantages of two methods above, and they can learn useful hierarchical features using sizeable labeled training samples.

* Corresponding author.
E-mail address: dongzihao@mail.nankai.edu.cn (Z. Dong).

https://doi.org/10.1016/j.knosys.2020.106315
0950-7051/

The main problem of CNN models based on pre-trained filters is the requirement of a large number of parameters for training and feature learning, leading to an increase of the computational complexity of deep learning frameworks. Therefore, lightweight models are needed to reduce these parameters for small sample training. In addition, insufficient training data may also lead to overfitting given a large number of parameters. Motivated by the aforementioned methods, we propose three novel techniques to develop ScatterNet-based feature extraction algorithms for small dataset problem: (i) utilizing the strip recurrent module with two different strip LSTM layers instead of the convolution operation to obtain compressed sequence features, and sweeping horizontally and vertically in two directions across image; (ii) replacing the traditional convolution filters with a learnable scattering transform filter bank which can be trained to learn locally invariant scattering features from the ScatterNet, and adding a learnable weighted matrix in the proposed architecture; (iii) adopting two sparse operations to make the model more lightweight on the limited small sample datasets. As a result, the proposed architecture is inspired from ScatterNet but combining the wavelet bases with the learned filters. Furthermore, we first use *He* initialization algorithm [19] to obtain the initial value of weight matrices, and then adopt scaling and translation operations to turn the parameters of weights and offsets. These techniques can effectively avoid the overfitting problem with limited training. We also use this layer as a skip layer to build a simple CNN-RNN hybrid architecture, which improves the accuracy of image classification tasks. Experiments are performed on the typical small sample databases, including image classification (CIFAR10, CIFAR100, TinyImageNet, NORB and MNIST), medical image segmentation (ssTEM and DRIVE) and classification based on few shot learning (miniImageNet). Our method of learning sparse features with lightweight ScatterNet achieves the competitive results compared with convolutional frameworks in small size image classification and segmentation, it is better suited to training model on both small and large datasets to reduce the number of parameters.

We summarize the contribution of this paper as follows:

1. We develop a novel lightweight ScatterNet to learn sparse sequence and spectral features. Experiments show that this method is suitable for classification tasks with small sample.
2. The proposed lightweight model only has a small number of parameters, and converges rapidly in training, leading to its robustness and effectiveness.
3. The classification accuracy of the hybrid model based on our proposed lightweight ScatterNet is higher than that of the state-of-the-art convolutional frameworks in several datasets. These architectures provide a new solution for computer vision tasks.

The rest of the paper is organized as follows. Section 2 introduces the related works on small sample training and lightweight models. Section 3 presents the proposed lightweight ScatterNet based on learning sparse features in detail. In Section 4, we conduct extensive experiments to confirm the effectiveness of the proposed method. Section 5 concludes and describes several future research directions.

2. Related works

In this section, we briefly review the literature on image understanding, including image classification and semantic segmentation, especially small sample training based on lightweight models. The related techniques are divided into two primary categories: (1) filter learned methods that update the weights and learn the feature parameters; (2) hybrid networks that combine the features from different stages of learning.

Filter learned methods. Hand-encoded filters have been proposed to address the small sample problem. For example, S. Mallat et al. [20] proposed the Scattering network which is a handcrafted CNN where pre-defined wavelets are utilized to provide sparse image representations. For multiple feature learning, Chan et al. [15] utilized Principal Component Analysis (PCA) to learn the filter banks and build PCANet architecture; Cotter et al. [21] proposed a learned ScatterNet to add the learning operation between scattering orders, which is taken as a locally invariant convolutional layer. Keshari et al. [22] proposed a CNN-based framework to learn structure and strength of filters with pre-trained weights, which is a novel method to deal with small sample size problem. In order to learn key features for few shot learning tasks, Sun et al. [23] achieved meta-transfer learning (MTL) method by learning scaling and shifting function of CNN weights for each dataset. Recently, circular harmonics is used for rotational transformation augmentation onto CNN's weights, yielding significant improvements in performance metrics such as classification accuracy. For example, Worrall et al. [24] presented a new Harmonic Network which is a CNN-like architecture with equivariance to patch-level translation and rotation; Rohan et al. [25] replaced the kernel in the locally scale invariant CNN with scale-steerable kernels, which can be denoted as a log-radial harmonic; Maurice et al. [26] developed Steerable Filter CNNs (SFCNNs) to generate the orientation dependent responses by employing filters that are steerable and learned. In addition, light weighted models with a small number of parameters are quite necessary for image processing tasks such as image classification and object recognition. For instance, the spectral approaches [27–30] utilize a set of spatial filters to transform feature learning into the frequency domain. Fujieda et al. [31] and Huang et al. [29] proposed wavelet CNNs to combine the multiresolution analysis via wavelet transform and convolutional networks into one model; Khan et al. [30] operated on the spectral decomposition using wavelets deconvolution to learn filter widths. Some applications such as texture classification [31] and medical image segmentation [28] based on wavelet CNNs are also proposed and achieved better performance.

Hybrid Networks. This approach is used to modify the multi-layer network structures that combine the low-level features using hand-crafted filters with more complex features learned from subsequent layers. The hybrid ScatterNet [32,18,17] is an upgraded version that uses ScatterNet as a front end and CNN as a learned back end. In more complex hybrid ScatterNet structures, D-SHDL [33] composes of a hand-crafted front-end, an unsupervised learning module and a supervised learning based back-end to evaluate the object classification accuracy; G-SHDL [34] with structural priors uses Restricted Boltzmann Machine instead of PCA layers of D-SHDL to learn an invariant features for semantic segmentation. In other research directions, Katzmann et al. [35] proposed hybrid rotation invariant networks to develop invariant features from limited training data. To analyze medical images, Bekkers et al. [36] built a novel framework with lifting layers, SE(2) group convolution layer and final projection layer for rotation and translation covariant deep learning. Unberath et al. [37] proposed DeepDRR which is a hybrid easy-to-use framework for fast simulation of fluoroscopy from CT scans.

3. The proposed method

In this section, we introduce the lightweight ScatterNet with sparse-feature learning, which is divided into three parts: (1) learn sequential and sparse features of filters for small sample training, (2) build hybrid architectures to perform image classification and segmentation, and (3) analyze the implementation details and computational complexity of our algorithm.

3.1. Preliminary

We first introduce the problem definition and notations for filter learning such as convolutional layers, wavelets and scattering transforms, following related work [20,21,30].

Conventional Convolutional Layers. We use $I^{(i)}(c, \mathbf{u}(x, y))$ to denote the input image or the feature map from the ith convolutional layer, where the number of channels is $c \in \{0, \ldots, C_i - 1\}$, and $\mathbf{u}(x, y)$ is a vector of pixel coordinates in the plane dimensions. The convolution operation can be regarded as a filtering process. If $h_f^{(i)}(c, \mathbf{u}(x, y))$ is defined as fth filter of ith layer, the output feature map of a standard convolutional layer in the conventional CNN is:

$$Y^{(i+1)}(c, \mathbf{u}(x, y)) = \sum_{c=0}^{C_i - 1} I^{(i)}(c, \mathbf{u}(x, y)) * h_f^{(i)}(c, \mathbf{u}(x, y)). \tag{1}$$

The activation function $\sigma(.)$ and batch normalization $b(.)$ are generally added behind a convolutional layer, so the final output of the typical convolutional layer is:

$$O^{(i+1)}(c, \mathbf{u}(x, y)) = b, \sigma(Y^{(i+1)}(c, \mathbf{u}(x, y))). \tag{2}$$

Wavelets Transforms. The wavelet transform is completed by replacing the filter $h_f^{(i)}(c, \mathbf{u}(x, y))$ of Eq. (1) with the fixed wavelet basis function, which consists of a mother wavelet filter $\psi_{j,\theta}(\mathbf{u}(x, y))$ and a low pass filter $\phi_j(\mathbf{u}(x, y))$. Therefore, the filter based on wavelet transformation $h_f^{(i)}(c, \mathbf{u}(x, y))$ can be converted to:

$$h_f^{(i)}(c, \mathbf{u}(x, y)) \longrightarrow \{\psi_{j,\theta}(\mathbf{u}(x, y)), \phi_j(\mathbf{u}(x, y))\} \tag{3}$$

where $\{j, \theta\}$ is the dilatation and rotation of features, j is the scale and θ is the rotation angle between 0 and π. A mother wavelet filter $\psi_{(j,\theta)}$ with 2^j dilatation and θ rotation can be decomposed to:

$$\psi_{j,\theta}(\mathbf{u}(x, y)) = 2^{-j}\psi(2^{-j}R_{-\theta}\mathbf{u}(x, y)) \tag{4}$$

where R is the transformation matrix, the range of the dilatation j is $1 \le j \le J$.

A scaled low pass is used to give the invariant scattering coefficient, which is defined with only the scale term:

$$\phi_{j,\theta}(\mathbf{u}(x, y)) = 2^{-j}\phi(2^{-J}\mathbf{u}(x, y)). \tag{5}$$

We integrate the results from Eqs. (4) and (5), and convolve them with the input image or feature map $I(c, \mathbf{u}(x, y))$. Therefore, the 2D wavelet transform operation $WT(.)$ is defined as:

$$\begin{aligned} WT(I(c, \mathbf{u}(x, y))) = \{&I(c, \mathbf{u}(x, y)) * \psi_{j,\theta}(\mathbf{u}(x, y)), \\ &|I(c, \mathbf{u}(x, y)) * \phi_j(\mathbf{u}(x, y))|\}. \end{aligned} \tag{6}$$

Original Scattering Transforms. In Eq. (6), the modulus terms $U = |I(c, \mathbf{u}(x, y)) * \psi_{j,\theta}(\mathbf{u}(x, y))|$ represent locally invariant features. Suppose a transformation path is $p = (\lambda_1, \lambda_2, \ldots, \lambda_m)$, the modulus propagator on a path p is defined as the form of nested convolution:

$$U_p I = ||\ldots|I * \psi_{\lambda_1}| * \psi_{\lambda_1}|\ldots * \psi_{\lambda_m}|. \tag{7}$$

The invariant Scattering coefficient can be computed by a scaled lowpass filter $\phi_{(j,\theta)}$:

$$S_p I = U_p I(c, \mathbf{u}(x, y)) * \phi_J(\mathbf{u}(x, y)). \tag{8}$$

If the value of path p is incremented iteratively, the scattering coefficient on the next path $p + \lambda$ is $\{S_p I, U_{p+\lambda} I\}_\lambda$, where $(p + \lambda) \in (\lambda_1, \lambda_2, \ldots, \lambda_m, \lambda)$. Therefore, the final scattering transform process is defined as:

$$WT(I(c, \mathbf{u}(x, y))) * U_p = \{S_p I, U_{p+\lambda} I\}_\lambda. \tag{9}$$

3.2. The lightweight scattering network

In this paper, we propose to use the lightweight network based on Scattering Transform for filter learning. Fig. 1 shows that the method can be divided into three steps: (1) a novel Strip-Recurrent module is adopted to compress the input images or feature maps and extract the sequence features, (2) input $I^{(i)}(c, \mathbf{u}(x, y))$ is filtered by ScatterNet with fixed parameters, and (3) a learned weight matrix A is defined as a learning term of the filter based on scattering transform which uses sparse transformation to reduce its dimensionality.

Strip-Recurrent module. Inspired by Visin et al. [38], we propose the Strip-Recurrent module with two LSTM layers to get sequence features of the input image by strip sweeping the feature map, which is different from the feature learning method of single bi-direction RNN layer. Suppose $I(c, \mathbf{u}(x, y))$ is the input image or feature map with the width x, height y and number of channels c, we split the input into a set of patches $P = \{p_{l,m}\}$, where l and m are the horizontal index and the vertical index, and their total number is $L{*}M$. Suppose a patch size is $x_p \times y_p$, (l, m)-th patch will be defined as $p_{l,m} \in \mathbb{R}^{x_p \times y_p \times c_p}$. We first sweep every image patch $p_{l,m}(x_p \times y_p \times c_p)$ horizontally with a LSTM layer, where $x_p \times y_p$ is set to 1×1, followed by the same operation for the above results vertically. The whole process can be defined as:

$$\begin{aligned} &\text{Input}: \quad I^{(i)}(c, \mathbf{u}(x, y)) \xrightarrow{split} \{p_{l,m}\}, \\ &\quad \text{for } l = 1, 2, \ldots, L \text{ and } m = 1, 2, \ldots, M \\ &\text{Horizontal sweep}: \quad H = LSTM(I^{(i)}(c, \mathbf{u}(x, y)), p_{l,m}), \\ &\quad \text{for } l = 1, 2, \ldots, L \\ &\text{Vertical sweep}: \quad V = LSTM(H, p_{l,m}), \ \text{for } l = 1, 2, \ldots, M \end{aligned} \tag{10}$$

where H is the result by a *LSTM* layer after horizontal sweep, and V is the result by a *LSTM* layer after vertical sweep. After these two sweeps of Strip-Recurrent module and permuting, the size of the input feature map is changed from $H \times W \times C_{in}$ to $H \times W \times C_{out}/2$. Fig. 2 provides a visual illustration.

Learnable Scattering Transforms. Fig. 1 shows that the scattering transform is formulated as the fixed filter. The proposed scattering-based method mainly completes the 2D-wavelet transform, combines the invariant terms with low pass terms, and learns the weights of different terms in the mixing process. In the ScatterNet, the grayish modules represent transformation matrices with fixed parameters, the dark gray modules are denoted as the weighted matrices with learned parameters. Firstly a novel Strip-Recurrent module is adopted to extract the sequence features and compress the input image or feature map $I^{(i)} \in \mathbb{R}^{H \times W \times C_i}$ to $I'^{(i)} \in \mathbb{R}^{H \times W \times C_i/2}$; then $I'^{(i)}$ is filtered by the real and imaginary orient scattering transforms (the top branch) and the scaled low pass terms (the bottom branch), which increase the channel dimension from $C_i/2$ to $(2K + 1)C_i/2$, where K is the number of orientations. The down sampled results $I_1'^{(i)} \in \mathbb{R}^{H/2 \times W/2 \times 6C_i}$ of orient scattering are combined with the component $I_2'^{(i)} \in \mathbb{R}^{H/2 \times W/2 \times C_i/2}$ of low pass part to give the result $Y^{(i)}$ according to Eq. (9). Finally, we convolve $Y^{(i)}$ with learned weight matrix A to give the proposed output $O^{(i)}$:

$$O^{(i)} = \{S_p I, U_{p+\lambda} I\}_\lambda = \sum_{C=0}^{C-1} (I'^{(i)} * \phi_j + |I'^{(i)} * (\psi_{j,\theta})_\lambda|) * A \tag{11}$$

where DT$\mathbb{C}$WT [39] is chosen as our wavelet filter $\psi_{j,\theta}$ due to its fast implementation. However a disadvantage of this method is that K, the number of orientations of wavelets, is restricted to 6; the path sequence λ is treated as a diagonal matrix; the learned weight matrix A consists of two weight parameters $\{a(c), k(c)\}$

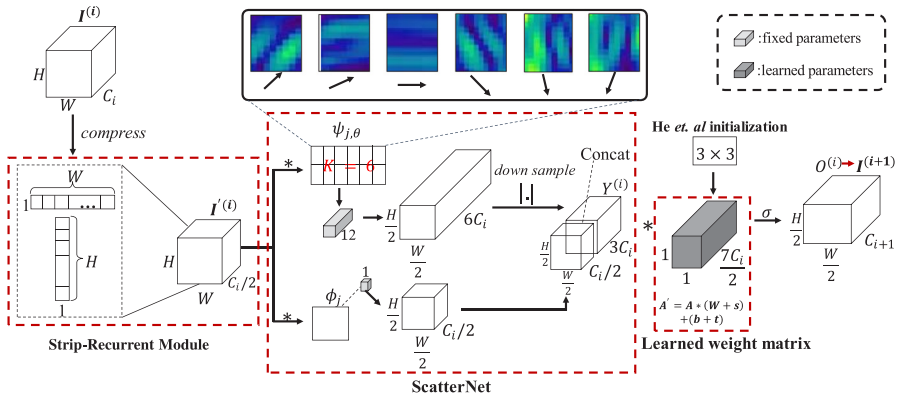


Fig. 1. The block diagram of Lightweight Scattering transform layer with learned weight matrix.

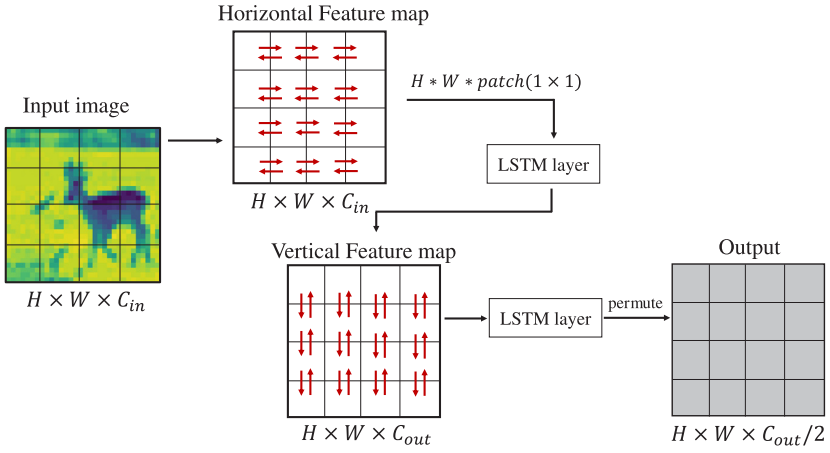


Fig. 2. The structure of Strip-Recurrent module.

corresponding to ϕ_j and $\psi_{j,\theta}$, respectively, and its initial value is set to a random matrix. Therefore, suppose the path γ is an index variable, the concrete form of weight matrix A can be defined as:

$$A = \{a(c), k(c)\}_\gamma = \begin{cases} a(c)[\gamma], & \gamma = 1 \\ k(c)[\gamma], & 1 < \gamma \le \frac{JK+1}{2} \end{cases} \tag{12}$$

As a result, the filter learning process of lightweight ScatterNet is summarized in Algorithm 1. The input image or feature map I^N needs to be extracted into several patches, which is used for horizontal and vertical sweeping of feature matrices.

Learning Sparse Features of Filters. Eq. (11) shows that learnable scattering transform is equivalent to the standard convolutional operation like Eq. (1). The differences include: (1) we have replaced the input I with the result after sequence feature extraction and scattering transformation; (2) the weight W of convolutional layer is replaced by a learning matrix multiplicator A, and this computational process can be viewed as a 1×1 convolution. In order to make this learning approach more suitable for small sample training problems, the sparse transformations such as the scale transformation s and translation t are proposed to reduce the number of learning parameters.

The proposed concept of sparse transformation of learned weighted matrix A and offset b is illustrated in Fig. 3. Fig. 3(a) shows the parameter-tuning of traditional convolutional layer. The base-model (classifier) f based on CNNs updates all parameters $[W, b]$ by iterative training on the dataset $\mathcal{D}$, which is optimized by gradient decent as follows:

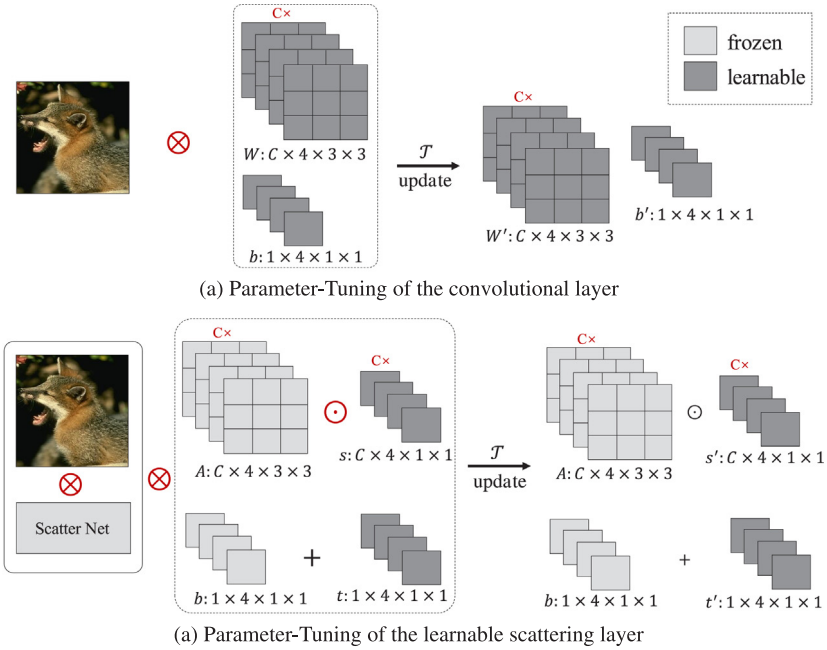
$$\{[W', b']; f'\} \longleftarrow \{[W, b]; f\} - \alpha \nabla \mathcal{L}_{cnn}([W, b]; f) \tag{13}$$

Algorithm 1 The Filter Learning of Lightweight ScatterNet

Notation: N is the size of training set, n is the number of extract patches
Input: Image or feature map I^N
Output: Weight matrix A
for each layer $l := 1$ to *numLayer* **do**
  $[p]^N \leftarrow extractPatch(I^N, Patchsize)$
  H&V Sweep: $Y \overset{H}{\leftarrow} LSTM(I^N, [p]^N)$, $Y \overset{V}{\leftarrow} LSTM(Y, [p]^N)$
  ScatterNet: $O^{(i)} = \sum_{C=0}^{C-1}(Y^{(i)} * \phi_j + |Y^{(i)} * \psi_{j,\theta}|)$
  $A \leftarrow \text{reshape}(random(A))$
  for $j := 1$ to N **do**
    $fmap_j = O^{(i)} * A$
  end for
  $I^N \leftarrow ReLu(fmap)$
end for

where α is the learning rate, and $\mathcal{L}_{cnn}(.)$ is the loss function of model training based on convolutional layers. Suppose cross-entropy function is used to compute the loss value between the input x and the ground truth y, it can be denoted as:

$$\mathcal{L}_{cnn}([W, b]; f) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} l(f(x; [W, b]), y). \tag{14}$$



(a) Parameter-Tuning of the convolutional layer

(a) Parameter-Tuning of the learnable scattering layer

Fig. 3. Illustration of sparse transformation of the learnable the ScatterNet-based filter which significantly reduce the number of parameters in the model.

In the example of Fig. 3(a), given a computer vision task $\mathcal{T}$, the kernel size $k = 3$ and channels C, after parameters $[W, b]$ learning and update, the number of parameters $[W', b']$ obtained by CNNs is calculated as: $C * 4 * 3 * 3 + 1 * 4 * 1 * 1 = 36C + 4$.

Fig. 3(b) shows the parameter tuning of proposed learnable scattering layer. Here, the values of A and b are fixed in a learning process, and only the parameters of scale transformation operator s and translation operator t are learned using stochastic gradient descent method. Suppose the base classifier is f, the whole optimization process with the training loss $\mathcal{L}_{tr}(.)$ can be expressed as:

$$\{[s', t']; f'\} \longleftarrow \{[s, t]; f\} - \gamma \nabla \mathcal{L}_{tr}([s, t]; f). \tag{15}$$

Therefore, we further calculate Eq. (11) through the proposed sparse transformation, which can be defined as:

$$O^{(i)}(X, A, s', t') = X * (s' \cdot A) + (b + t') \tag{16}$$

where $s' \odot A$ represents element-wise multiplication, and X is the output of the scattering transform with a Strip-Recurrent module as shown Fig. 1. Taking Fig. 3(b) as an typical example of filter with 3×3 kernel, the weighted matrix A is scaled by $s(1 \times 1)$ then shifted by $t(1 \times 1)$, so the number of parameters is: $C * 4 * 1 * 1 + 1 * 4 * 1 * 1 = 4C + 4$. Therefore, it is obvious that sparse transformation $\{s, t\}$ can reduce the training parameters effectively due to a column matrix is learned rather than learning a complete weight matrix A.

Given the above derivation, the sparse feature learning steps within a computer vision task $\mathcal{T}$ are summarized in Algorithm 2, which is used for light-weight model training to obtain the optimal sparse weight matrix A.

Learning initial value of weight matrix A. Like the traditional convolutional layer, learning initial value of A is equivalent to the weight initialization, which will affect the convergence rate of gradient descent and model learning efficiency. In the proposed learnable scattering layer, batch normalization (in Eq. (2)) is added to weaken the adverse effect of bad initialization, but we find that it sometimes reduces the classification accuracy on some datasets of image classification such as CIFAR-10. He et al. initialization [19] is used to be an optimal method for weight initialization due to its robustness and considering the rectifier nonlinearities, so the initial value of A is calculated from the following equation:

$$A_{init} = Random(n_{in}, n_{out}) * \sqrt{\frac{2}{n_{in} * \alpha}}, \qquad n_{in} = n_{out} = (7 * (\frac{C}{2}) \times F \times h \times h) \tag{17}$$

where α is the negative half-axis slope of ReLU, n_{in} is the length of the weight matrix A, h is the kernel size, and C and F are the input and output channels, respectively.

Algorithm 2 Sparse feature Learning steps within a task $\mathcal{T}$

1: **Input:** vision task $\mathcal{T}$, learning rate γ, base classifier f, sparse learning parameters $[s, t]$, feature X is $O^{(i)}$ of Algorithm 1
2: **Output:** Updated f' and $[s, t]$, Updated feature map $O^{(i)}$
3: **for** samples in $\mathcal{T}$ **do**
4: Evaluate (loss function) $\mathcal{L}_{tr}(.)$
5: Optimize $[s, t]$ by Eq. (14)
6: **end for**
7: **while** *not done* **do**
8: Update $O^{(i)}$ by Eq. (15)
9: Compute *ClassAcc* for $\mathcal{T}$
10: **end while**

3.3. Implementation

In our lightweight scattering transform layer, we use DT$\mathbb{C}$WT as the base filter to extract the feature representations that are dense over the scale. This layer extracts scattering coefficients by the dual-tree wavelets at 1 scales (J) and 6 orientations (θ), which means that the lowpass and bandpass coefficients can be mixed at the same resolution. Therefore the implementation process can

be expressed as:

$$((I * \psi_{J,\theta}) \downarrow 2 * \phi_j) \uparrow 2 \Rightarrow ((I * \psi_{1,6}) \downarrow 2 * \phi_1) \uparrow 2 \tag{18}$$

including downsampling ($\downarrow$) and upsampling ($\uparrow$).

Eq. (18) shows one order of scattering transform at a single scale. It can be expanded into a second order ScatterNet by stacking two of these layers, which is equivalent to one order of scattering transform at four scales. Therefore, it can be flexibly converted as:

$$(((I * \psi_{J,\theta}) \downarrow 2) * \psi_{J,\theta}) \downarrow 2 * \phi_j \uparrow 4 \Leftrightarrow ((I * \psi_{2,\theta}) \downarrow 4 * \phi_1) \uparrow 4. \tag{19}$$

The memory consumption and computational complexity of proposed algorithm will be analyzed in the following sections.

3.3.1. Memory cost

A standard convolutional layer has C_i input channels, C_{i+1} output channels and kernel size S. By the convolution calculation, it will generate $S^2C_iC_{i+1}$ training parameters.

The input feature map is firstly compressed and converted to the corresponding sequence features by the proposed Strip-Recurrent module, and its channels change from C_i to $C_i/2$. However, two LSTMs in Fig. 2 have increased the number of extra parameters which is $4 * ((X_d + Y_d) * Y_d + Y_d)$, where X_d is the input dimensions, Y_d is the output dimensions, they are used to denote the size of the input and output feature map.

As a result, the total number of learnable parameters in each of proposed layers with scale $J = 1$ and orientation $K = 6$ is:

$$\begin{aligned} \#params &= \frac{(JK+1)C_iC_{i+1}}{2} + LSTMs \\ &= \frac{7}{2}C_iC_{i+1} + 4 * ((X_d + Y_d) * Y_d + Y_d). \end{aligned} \tag{20}$$

3.3.2. Computational cost

A standard convolutional layer with kernel size S has S^2C_{i+1} multiplies per input pixels.

Our proposed layers exploit the DT$\mathbb{C}$WT as the base filter for each input channel. A regular discrete wavelet transform has $2K(1-2^{-2J})$ multiplies for J scale. Because a DT$\mathbb{C}$WT has 4 DWTs for an input, its computational cost will be $8K(1-2^{-2J})$, which is equivalent to FPS (Floating Point operations per Seconds). Suppose orientation K is set to 6 and scale $J = 1$ for each filter, the cost of DT$\mathbb{C}$WT will be 36. In Fig. 1, we can see that the computational cost of our learnable scattering layers is composed of DT$\mathbb{C}$WT and the feature combining process. Because using a decimated wavelet decomposition will make a problem that the combining process only works on one quarter the spatial size after one first scale, the cost of the combination process becomes $\frac{4}{4}C_{i+1}$. In a Strip-Recurrent module, its calculation is equivalent to a standard convolutional operation with (1×1) kernel size. Therefore, the computational cost is computed as:

$$(\frac{7}{2}C_{i+1} * \frac{1}{4} + 36) + 1^2C_{i+1} \approx 2C_{i+1} + 36 < S^2C_{i+1}. \tag{21}$$

Suppose output channels C_{i+1} of most convolutional layers is higher than 6, so the final value of Eq. (21) is significantly lower than S^2C_{i+1} with 3 kernel size. However, if the kernel size S is 1 or $C_{i+1} < 6$, the computational cost of our proposed layers will become higher than the standard convolutional layer.

3.4. The hybrid architectures

Based on our proposed learnable scattering layer, the hybrid networks are developed which utilize VGG-like structure in image classification and full convolutional networks such as U-net [40] in image segmentation as the base architectures, respectively. This hybrid frameworks are able to learn hierarchical features in the small size sample learning.

3.4.1. Application in image classification

In the image classification application, we use a VGG-like network as a base architecture. In Fig. 4, the proposed hybrid architecture comprises of a single-stream deep network with two learnable lightweight scattering layers. These inserted proposed layers replace the last two convolutional layers (denoted by dashed boxes), which generate feature responses from different levels of the primary network stream. Finally, these responses are combined in a shared output layer to predict classification probabilities through two Fully-Connected (FC) layers. Fig. 4 shows an example of the hybrid model based on VGG-6 base architecture, where the pooling and batch-normalization layers are inserted between each two convolutional layers.

In the original VGG-like networks, the final prediction may lose certain feature details due to the down sampling of multi-scale feature maps by the pooling layers. To address this problem, we add a skip structure that combines the final prediction layer with lower scale layers with finer strides. The architecture of skips is illustrated in the dotted box at the bottom of Fig. 4. The proposed layer with batch normalization acts as a skip layer that connecting the output features of low-scale and high-scale convolutional layers.

3.4.2. Application in image segmentation

Based on the standard U-net [40], we propose the hybrid U-net which uses the proposed lightweight scattering transform layer as a symmetric encoder–decoder network with the skip connections between different layers with the same feature map size. This skip structure extracts features from a large of receptive field but containing accurate spatial features. Therefore, it can be utilized to predict the probability map for small size image segmentation. We mainly adopt two modifications on the original U-net: (1) we use the proposed lightweight scattering transform layer with Batch Normalization and ReLU instead of the original convolutional layer; (2) OHEM algorithm [6] is applied for training the hybrid U-net model instead of the cross-entropy loss function. In the lightweight scattering network, scale J is set to 1, orientation θ is set to 6, kernel size is equal to 3×3 pixels, and a Strip-Recurrent module is removed from the proposed layer to prevent model from training overfitting.

In order to make this network more suitable for addressing the small sample size learning problem, a small modification is made to simplify the structure of U-net. All input images are padded to $320 \times 320 \times 1$ pixel matrixes by reflecting a region of 32 pixels around the borders to alleviate the effect of boundary artifacts; In the feature extraction stage, all convolutional layers are replaced by the proposed learning scattering layers. Finally, two 1×1 convolution layers transform these pixel-wise features to the predicted probability matrixes.

This network is optimized by the OHEM algorithm [6], which online selects the top k hard samples and computes the corresponding binary cross-entropy loss between predictions and annotated segmentation masks. OHEM can be regarded as a sparse method (only the top k) to calculate the loss value to address the imbalance of positive and negative samples. Through repeated experiments, we see that it achieves much better performance than CE loss in the case of small size databases. Suppose M_i is the ith prediction mask, M_i' is the ith ground truth and $\mathcal{L}_{CE}^{(1,k)}$ is a set of k cross-entropy loss values, the OHEM loss $\mathcal{L}_{ohem}$ in the model to address the problem of image segmentation can be defined as:

$$\begin{aligned} \mathcal{L}_{CE}^{(i)} &= M_i \log(M_i') \\ \mathcal{L}_{CE}^{(1,k)} &= top\_k\{\mathcal{L}_{CE}^{(1)}, \mathcal{L}_{CE}^{(2)}, \ldots, \mathcal{L}_{CE}^{(n)}\} \\ \mathcal{L}_{ohem} &= \frac{1}{k}\sum_{k} \mathcal{L}_{CE}^{(1,k)} \end{aligned} \tag{22}$$

where $top\_k(.)$ is used to compute the result sets including top k values in an input set of n data.

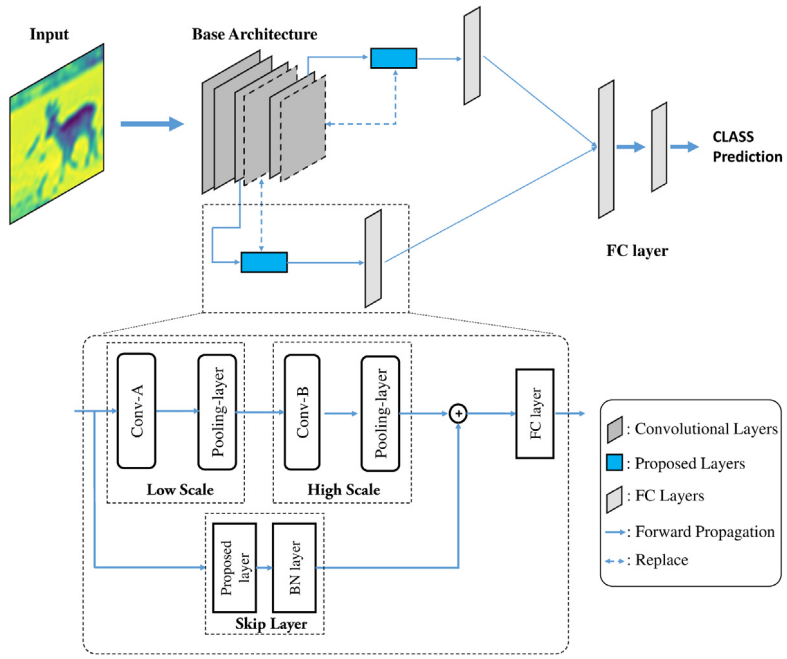


Fig. 4. A hybrid network with proposed learnable scattering layer applied to image classification, where the input image size is $H \times W \times 3$.

Table 1
Experimental protocol for layer-level evaluation on CIFAR-10, CIFAR-100 and Tiny ImageNet databases.

Datasets	Layer combination	Training data	Testing data
CIFAR-10	A-F; {BC,CD,DE,BD}	50k	10k
CIFAR-100	A-F; {BC,CD,DE,BD}	40k	10k
Tiny ImageNet	A-F; {BC,CD,DE,BD}	10k	1k

Table 2
Experimental protocol for layer-level performance on small training data from MNIST, CIFAR-10, and NORB databases.

Datasets	Small training data	Standard training	Standard testing
MNIST	100:100:1k; 1k:1k:5k	50k	10k
CIFAR-10	100:100:1k; 1k:1k:5k	40k	10k
NORB	100:100:1k; 1k:1k:5k	20k	24.3k

4. Experimental results

To evaluate the performance of the proposed algorithm, we conduct experiments in two main scenarios including both layer-level and hybrid network comparisons. In the layer-level comparison, we perform an ablation study among different layers and verify the classification accuracy on small size training data by varying the size of the training set. In the hybrid network comparison, the proposed hybrid models are applied to the small sample classification, medical image segmentation and image classification in the few-shot learning fields. Our proposed methods are evaluated on multiple datasets and we compare them with CNN-based architectures such as VGG [3], ResNet [4] and U-Net [40]. Experimental details of the different conditions are described in the following sections.

4.1. Databases and experimental protocol

To evaluate the classification efficiency of the novel proposed layer, the experiments are performed with varying combinations of layers on three databases: CIFAR-10 [41], CIFAR-100 [41] and Tiny ImageNet [42]. As shown in Table 1, we swap out convolutional layers for proposed lightweight scattering layers in 10 ways: $A, B, \ldots, F$ (single layer replacement) and BC, CD, DE, BD (two layers replacement). Following Keshari et al. [22], the proposed method is also evaluated with 14 training data sizes (Table 2), 100, 200, $\ldots$, $1k$, $2k$, $\ldots$, $5k$, on MNIST [43], CIFAR-10 [41] and NORB [44] databases. In addition, experiments are performed on CIFAR-10 and CIFAR-100 to compare the hybrid networks including the architecture in Fig. 4 and ResNet-based model with the state-of-the-art CNN-like frameworks. A small sample dataset miniImageNet [45] is used for few-shot learning evaluation, which comprises of 100 classes with 800 samples of 84×84 color images per class. Finally, we perform experiments on the ssTEM ISBI [46] and DRIVE databases [47] for medical image segmentation tasks, where ssTEM ISBI only contains 30 images of pixel size 512×512 (27 for train, 3 for test); DRIVE has 40 fundus images with 20 training and 20 testing samples.

4.2. Implementation details

Lightweight scattering layer. We use a simple but powerful VGG-like architecture with our lightweight scattering layer as shown in Table 3. In this network, He et al. method [19] is used to initialize weight matrix A of the proposed layer with the scale $J = 1$ and the angles $K = 6$. It is optimized with stochastic gradient descent with momentum 0.9, and initial learning rate 0.1 which is scaled by a factor 0.2 after 60,80 and 100 epochs. In addition, the batch size and total epochs are set to 128 and 120, respectively. Finally, all the parameters of the proposed layer are regularized with batch normalization.

Hybrid architectures. In the VGG-like hybrid network with skip layer for image classification, we use VGG-13 as the base architecture and add dropout after these convolutional layers with drop probability $p = 0.3$. In the resnet-like hybrid network, the learning rate is set to 0.001, batch size is set to 10 and the training epochs are set to 20, we remove the Strip-Recurrent module in the lightweight scattering layer to reduce the complexity of deep models and avoid the training problem of overfitting. The hybrid U-net model is trained for 200 epochs using the Adam optimizer, the initial learning rate is set to $2e-4$ and decayed by 0.1 per epoch starting from 20th epoch. To further improve the

Table 3
VGG-like networks used for layer level comparison experiment as the base architecture. C is set to 96 in our experiment.

Name of layers	Output size
Conv-A	$C \times H \times W$
Conv-B	$C \times H \times W$
Conv-C	$2C \times H/2 \times W/2$
Conv-D	$2C \times H/2 \times W/2$
Conv-E	$4C \times H/4 \times W/4$
Conv-F	$4C \times H/4 \times W/4$
Conv-G[a]	$8C \times H/8 \times W/8$
Conv-H[a]	$8C \times H/8 \times W/8$
FC	num classes $\times 1$

[a]indicates this layer is only added in Tiny ImageNet experiments.

prediction accuracy of the proposed method on a small number of samples, we perform data augmentation such as flip, crop and brightness preprocessing on the original medical images.

4.3. Layer-level comparison

To evaluate the accuracy of the proposed lightweight ScatterNet with sparsification on small samples of different databases, a VGG-like network with 6 convolutional layers is used for CIFAR-10 and CIFAR-100, and 8 convolutional layers are only utilized in Tiny ImageNet experiments. This common architecture performs well in evaluating classification accuracy at the layer level compared with the same models based on Convolutional layers.

Ablation among different layers. The ablation study is performed in a way that convolutional layers are replaced by other novel filter-based layers responsibly. Due to the base architecture in Table 3 with 6 or 8 layers, there will be a lot of different permutations and combinations to form a new framework for the ablation experiments. Therefore, we only study the performance comparison of swapping 1 or 2 layers, where two directly connected lightweight scattering layers are stacked by the implementation of Eq. (19).

From Table 4, we can see that variant networks with proposed lightweight ScatterNet always achieves best performance than the other competitors on three datasets. These experimental results validate that it can improve the accuracy when one or two lightweight scattering layers are used in the VGG-like network, especially replacing the convolutional layers at the back end. Compared with invariant layers [21] and SSF [22], a Strip-Recurrent operation in our proposed layer can extract more fine sequence features instead of convolutional + pooling layer, for example, Conv→ $\{B, D\}$ based on proposed layers achieves the significant highest accuracy of all combined architectures: 94.1% in CIFAR-10, 75.0% in CIFAR-100 and 64.0% in Tiny ImageNet.

Table 5
Training duration (ms) comparison among different VGG-like methods (only Conv→ $\{B, D\}$). The number of training iterations is 120.

Datasets	CIFAR-10		CIFAR-100		Tiny ImageNet	
	Time	Acc.	Time	Acc.	Time	Acc.
Cotter et al. [21]	6.91e4	92.7	7.24e4	71.3	5.30e5	59.3
Proposed (no sparse)	7.40e4	93.2	7.43e4	73.4	5.84e5	61.4
Proposed (sparse)	**6.84e4**	**94.1**	**6.90e4**	**75.0**	**5.29e5**	**64.0**

However, the Strip-Recurrent module leads to longer training time. Table 5 shows that: (i) learning sparse features of filters can greatly reduce the training time compared with the non-sparse methods, this strategy makes model training faster and improves classification accuracy by 1 to 2 percent under the same experimental conditions; (ii) compared with the invariant layers [21], our proposed non-sparse methods add a Strip-Recurrent module in lightweight ScatterNet, but they achieve 93.2%, 73.4% and 61.4% accuracy on CIFAR-10, CIFAR-100 and Tiny ImageNet datasets, which are higher than 92.7%, 71.3% and 59.3% of the corresponding structure based on invariant layers. These results demonstrate the efficiency of the proposed Strip-Recurrent module.

Performance of the proposed method and Comparison with Existing Algorithms. The performance of Conv→ $\{B, D\}$ in Table 4 based on the proposed layer is evaluated on three classification datasets by varying the training data size. Fig. 5 shows that: (i) the proposed algorithm generally yields higher performance compared with the original ScatterNet [11], and the sparse weight matrix A and Strip-Recurrent module play an important role in the improvement of classification accuracy; (ii) on the smaller size MNIST dataset, our method outperforms another existing algorithm such as PCANet [15], Deep Hybrid Network [17] and ScatterNet [11] in almost all sizes of training data, which illustrates that learned features can be adapted with smaller training samples using lightweight learning ScatterNet; (iii) SSF based on dictionary initialization [22] is more suitable for training the CNN model with smaller training data (0.1k–0.5k), but its performance become lower than other methods such as PCANet as the size of datasets get larger. Our proposed method can address this problem, for example, it outperforms all the comparison algorithms in the training data size of 2k-5k. Therefore, the performance of proposed method is robust in both small and large data training process; (iv) for improving the classification accuracy, the weight matrix A with sparse transformation (s, t) is used to extract the key features of small size samples and its effectiveness is verified its valid by the experiments.

Weight Visualization. We analyze the weight matrix A learned from the proposed method and W learned from the CNN model.

Table 4
Classification accuracy (%) for testing layer level performance on several databases including CIFAR-10, CIFAR-100 and Tiny ImageNet. The VGG-like network in Table 1 is used as the base model, 'Conv→ $\{X\}$' denotes 'Conv-X' is replaced by new layers correspondingly.

Datasets	CIFAR-10 (50k)			CIFAR-100 (50k)			Tiny ImageNet (100k)		
Base model	91.9			70.3			59.1		
Layers	Inv [21]	SSF [22]	Ours	Inv [21]	SSF [22]	Ours	Inv [21]	SSF [22]	Ours
Conv→ $\{A\}$	91.3	91.5	92.2	69.5	70.1	70.4	57.7	58.2	60.3
Conv→ $\{B\}$	91.8	91.7	93.3	70.7	71.2	73.9	59.5	61.0	63.0
Conv→ $\{C\}$	92.3	92.9	93.5	71.2	72.7	74.6	59.8	61.6	62.9
Conv→ $\{D\}$	91.2	91.8	93.5	70.1	72.1	73.7	59.3	60.2	63.0
Conv→ $\{E\}$	91.6	92.2	93.7	70.0	71.3	73.8	59.4	60.4	63.9
Conv→ $\{F\}$	90.5	92.1	93.6	68.9	70.5	73.7	57.8	59.1	63.2
Conv→ $\{B, C\}$	91.2	91.8	93.0	69.1	70.8	72.6	57.7	59.2	62.0
Conv→ $\{C, D\}$	92.1	92.0	93.6	70.1	72.7	74.0	59.5	60.7	62.6
Conv→ $\{D, E\}$	89.1	91.1	93.2	67.3	70.4	71.8	59.8	60.2	61.8
Conv→ $\{B, D\}$	92.7	93.1	**94.1**	71.3	73.2	**75.0**	59.3	61.7	**64.0**

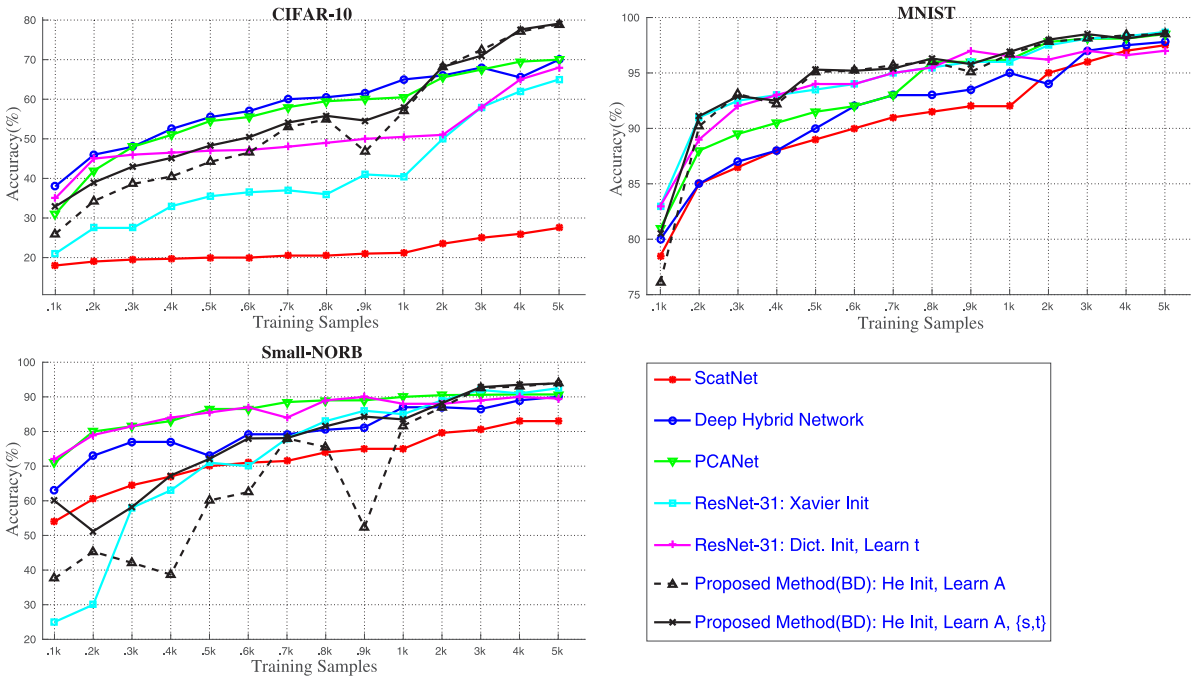


Fig. 5. Classification accuracy (%) predicted by different models for vary training samples from CIFAR-10, CIFAR-100 and NORB datasets.

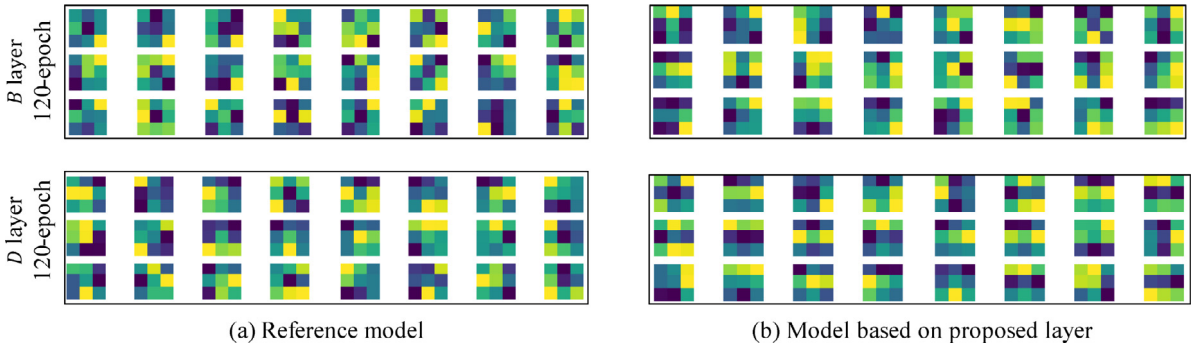


Fig. 6. Weight visualization of B layer and D layer of the (a) reference model based on convolutional layer and (b) novel model based on proposed layer.

Fig. 6 shows the B and D layer weights from the base model in Table 3 trained on CIFAR-10 database: the weight of B layer trained at 120 epoch, and the weight of D layer trained at 120 epoch. It is straightforward to see that lightweight ScatterNet trained weights are sparser and have less noise compared to CNN trained weights on CIFAR-10 dataset. The weight matrix A has better structural and sparse features, which illustrates that our proposed model utilizes good filters.

Discussion of Scale J. To analyze the effect of different parameter settings of scale J, we have also performed the related experiment of Conv$\rightarrow$ $\{B, D\}$ at three different scales ($J = 1, 2, 3$), which is calculated by Eq. (19). Fig. 7 shows that the classification accuracy of the proposed model decreases greatly with the increase of scales. This result indicates setting J to 1 dramatically improve the performance of lightweight ScatterNet.

4.4. Hybrid network comparison

In the previous section, based on the lightweight scattering layer, we propose the hybrid model which uses it as a skip layer for image classification and the hybrid U-net with this proposed layer for image segmentation. To evaluate the performance of the proposed hybrid architecture, we perform experiments in two scenarios: image classification on the entire training samples of CIFAR-10 and CIFAR-100, and segmentation with a small number of medical images. At the same time, we also evaluate the performance of this proposed layer with different number of small samples in the deep hybrid frameworks such as ResNet.

For the classification performance comparison, in Table 6, we report the optimal obtain results of our proposed hybrid network on the CIFAR-10 and CIFAR-100. VGG-13 with two lightweight scattering layers as skip layers achieves the competitive performance with the All conv [48], VGG-16 [3], FitNet [49] and ResNet-1001 [50]. The Wide ResNets [51] achieves the best accuracy, but with a large number of layers (28), 5900M multiplies and 36.5M parameters.

To evaluate the computation complexity of different models, we compute the number of parameters and multiples in convolutional neural networks, but not including batch normalization and other types of layers. The number of multiples represents FLOPs, which is a metric of the computational cost of models. In Table 6, we can clearly see that Conv$\rightarrow$ $\{B, D\}$ based on the proposed layer achieves competitive classification accuracy with a minimal amount of multiples (213M) and 11.2M parameters. The results of Hybrid model based on VGG-13 show that the skip layers are suitable for enhancing the accuracy of large networks with a relatively large number of parameters and multiples.

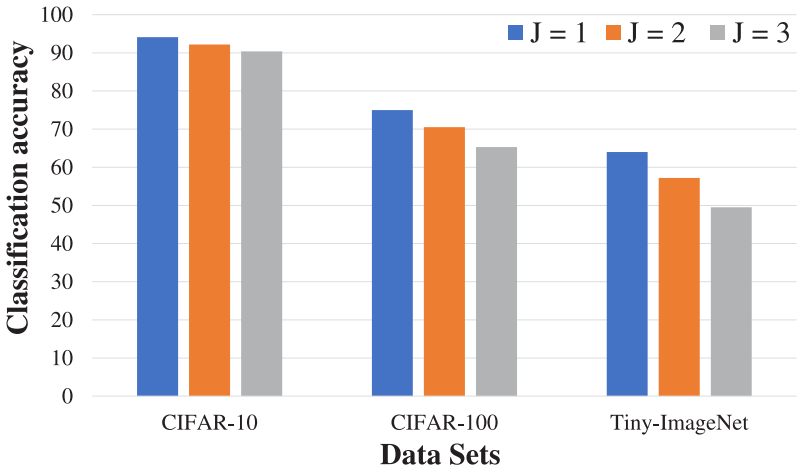


Fig. 7. The classification accuracy of Conv→ $\{B, D\}$ at three different scales J.

Table 6
Classification result (%), the number of parameters and calculate amount comparison between proposed hybrid network in Fig. 4 and other state-of-art methods. #param is the number of parameters, #multi is the number of multiples per 32 × 32 × 3 image.

Methods	CIFAR-10	CIFAR-100	layer	#param	#multi
All conv [48]	92.8	66.3	8	**1.4M**	281M
VGG-16 [3]	91.6	–	16	138M	313M
FitNet [49]	91.6	65.0	19	2.5M	382M
ResNet-1001 [50]	95.1	77.3	1000	10.2M	4453M
WRN-28-10 [51]	**96.1**	**91.2**	28	36.5M	5900M
Conv→ $\{B, D\}$	94.1	75.0	6	11.2M	**213M**
Proposed (VGG-13+skips)	95.0	76.1	10	14.7M	428M

Table 7
The classification result (Mean Accuracy %) for CIFAR-100 database with different small training samples, which is reported for fine-tuned ResNet-like model and learning the scattering filters by our proposed method.

Base models	Fine-tuning (sample size)			
	10k	20k	30k	40k
ResNet-34	22.59 ± 1.68	32.84 ± 2.12	41.25 ± 1.35	47.87 ± 1.23
ResNet-50	15.04 ± 3.96	24.09 ± 2.94	30.88 ± 1.22	39.24 ± 1.07
ResNet-101	15.08 ± 3.30	22.56 ± 2.87	32.07 ± 1.94	36.32 ± 1.52
ResNet-152	13.99 ± 2.01	21.52 ± 1.73	30.62 ± 1.48	37.37 ± 0.82
Base models	Proposed learning (sample size)			
	10k	20k	30k	40k
ResNet-34	**32.24 ± 2.23**	**44.62 ± 3.04**	**52.39 ± 2.75**	**55.69 ± 4.32**
ResNet-50	27.29 ± 2.64	40.97 ± 1.27	49.29 ± 0.89	54.49 ± 0.68
ResNet-101	24.41 ± 4.31	38.03 ± 3.06	47.36 ± 2.63	50.01 ± 0.08
ResNet-152	21.62 ± 1.73	38.69 ± 1.42	42.04 ± 1.17	49.91 ± 0.10

To further demonstrate the effectiveness of the hybrid model with proposed layer on small sample datasets, we report two different studies on (i) deep ResNet-based architecture and (ii) relation framework for few-shot learning. These overall results are presented on CIFAR-100 and miniImageNet databases, respectively.

Classification Results on Limited Training Data. To evaluate the performance of our method in the deeper networks, we use four variants of ResNet as the base training architectures. In our hybrid ResNet models evaluation experiment, the basic Resnet block and BottleNeck still consist of standard convolutional layers, but the first layer and convolutional layers used for feature down-sampling with stride 2 are replaced with proposed lightweight scattering layers. Table 7 shows that our proposed learning methods effectively improve the performance of ResNet-like models such as ResNet-34, ResNet-50, ResNet-101 and ResNet-152, compared with conventional fine-tuning approaches. With the ResNet-34 training at 40k data size, the best accuracy 55.69% is at least 7% better than the conventional fine-tuning based method. These prediction results show that learning sparse features with lightweight learnable scattering layer can be used to achieve improved performance in limited training samples.

Table 8
Classification results (%) based on few-shot learning on the miniImageNet dataset.

Algorithm	1-shot, 5-way	5-shot, 5-way
Proto Nets [52]	49.4	**68.2**
Proto Nets (+Proposed)	**51.4**	67.4
Matching Nets [45]	44.2	**57.0**
Matching Nets (+Proposed)	**46.0**	56.0
ML [53]	48.1	**63.2**
ML (+Proposed)	**50.5**	63.1

Few-shot learning. On the miniImageNet database, we use three well-known few-shot learning frameworks including prototypical networks [52], matching network [45] and model-agnostic meta-learning [53] as the baseline model. The convolutional layers in each framework are replaced by the lightweight scattering transform layers. Table 8 shows that our proposed methods yield about 2% more classification accuracy than original models for 1-shot, 5-way (5 different classes, 1 sample for each category), and the accuracy of our methods are close to that of original models when the sample size of each category is increase to 5.

Segmentation Results on Small Size datasets. In order to extend this proposed layer to the segmentation task, we compare the performance of the hybrid U-net with the original U-net on the ssTEM ISBI and DRIVE databases. Fig. 8 shows the segmentation accuracy and error of these two models training 100–150 epochs on ssTEM ISBI and DRIVE datasets. Our proposed hybrid model achieves higher accuracy and lower loss value with fewer training epochs. For qualitative analysis of segmentation results, some examples with pixel-level classification are shown in Fig. 9. The first two rows are the segmentation results of DRIVE and the last two rows are the segmentation results of ssTEM. With the same iterative training, the hybrid U-net (Fig. 9(d)) can segment more accurate blood vessels and cell shapes with less noise. These above results indicate that our lightweight model can be applied to segmentation tasks such as medical image segmentation with a small amount of training data through fast iterative training and better convergence.

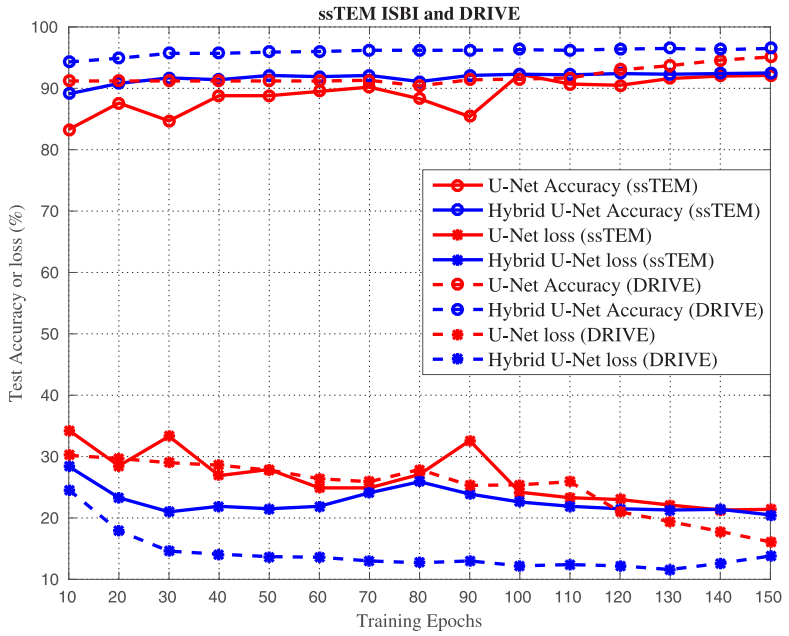


Fig. 8. The prediction accuracy and error curves of two models including U-net and Hybrid U-net trained 150 epochs on ssTEM ISBI and DRIVE datasets.

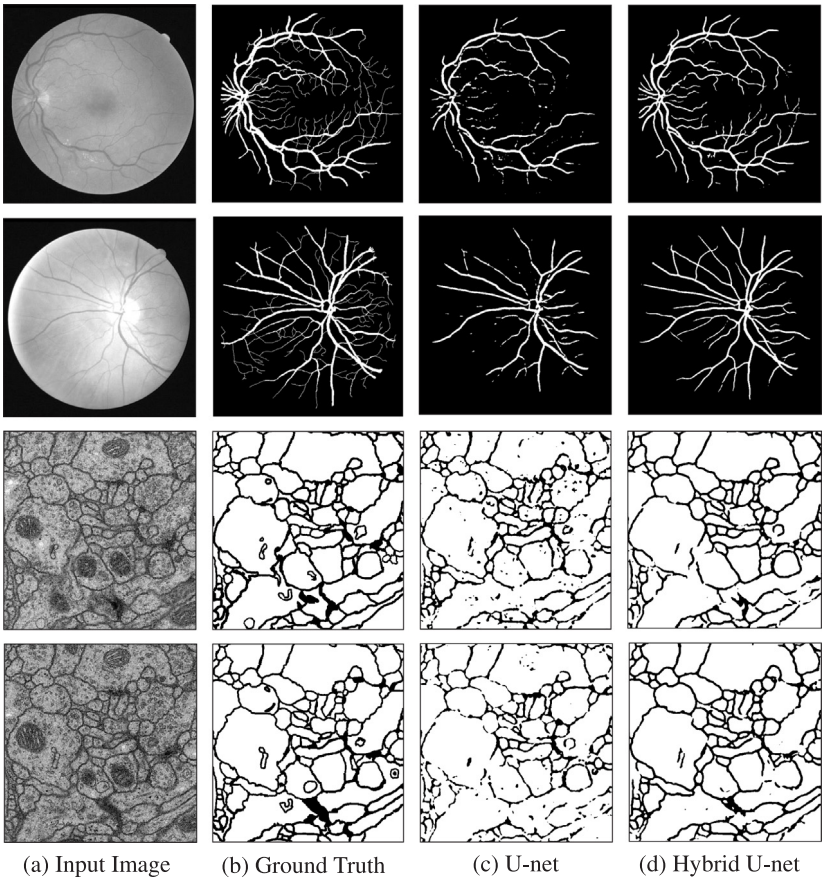


Fig. 9. Comparison of segmentation results between U-net and Hybrid U-net trained at 100 epochs.

5. Conclusion and future work

In this paper, we propose a new method to learn sparse features with lightweight ScatterNet for small sample size training. Although the fixed filters such as scattering transform can reduce the number of parameters for DNNs, they may not generate real distributions of features. Therefore, we add a Strip-Recurrent module as the front end and a learnable weight matrix to learn different types of features such as sequences and spectrums. However, these improved modules may increase the number of parameters and multipliers of the DNN model. Therefore, we perform sparse transformation such as scale transformation and

translation, which can reduce the dimension of the learnable weight matrix. Utilizing different hybrid architectures and experiments on multiple image tasks, we demonstrate the effectiveness of our proposed approach in both large and small datasets. Therefore, the proposed models have a certain degree of robustness in terms of lightweight and accuracy.

In terms of future work, it is worth further exploring the modification of our proposed filter, to completely replace the traditional convolutional layer instead of alternative replaced or inserted by some ablation experiments. It is also interesting to use other trained filters such as unsupervised filters to initialize the proposed network, which can adopt the filters to accomplish different types of computer vision tasks. In addition, we need to consider how to further reduce the time complexity caused by the patch-based Strip-Recurrent modules.

CRediT authorship contribution statement

Zihao Dong: Research concept and design, Collection and assembly of data, Data analysis and interpretation, Experiments, Writing the article, Critical revision of the article. **Ruixun Zhang:** Revision of the article. **Xiuli Shao:** Final approval of article. **Zengsheng Kuang:** Revision of the article.

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 supported in part by the Tianjin Intelligent Manufacturing Fund, China Project under Grant 201907206 and Grant 201907210, and Tianjin advanced Internet Manufacturing special Fund, China Project under Grant 18ZXRHGX00110.

References

[1] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., Imagenet large scale visual recognition challenge, Int. J. Comput. Vis. 115 (3) (2015) 211–252.
[2] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
[3] S. Liu, W. Deng, Very deep convolutional neural network based image classification using small training sample size, in: 2015 3rd IAPR Asian Conference on Pattern Recognition, ACPR, IEEE, 2015, pp. 730–734.
[4] 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.
[5] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708.
[6] A. Shrivastava, A. Gupta, R. Girshick, Training region-based object detectors with online hard example mining, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 761–769.
[7] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2980–2988.
[8] T. Mikolov, Statistical language models based on neural networks, in: Presentation at Google, Mountain View, 2nd April 80, 2012.
[9] I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, in: Advances in Neural Information Processing Systems, 2014, pp. 3104–3112.
[10] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, in: International Conference on Learning Representations, ICLR 2015, 2015.
[11] J. Andén, S. Mallat, Multiscale scattering for audio classification, in: International Society for Music Information Restrieval Conference, Miami, FL, 2011, pp. 657–662.
[12] L. Sifre, S. Mallat, Rigid-Motion Scattering for Image Classification (Ph.D. dissertation), Citeseer, 2014.
[13] L. Sifre, S. Mallat, Combined scattering for rotation invariant texture analysis, in: ESANN, vol. 44, 2012, pp. 68–81.
[14] L. Sifre, S. Mallat, Rotation, scaling and deformation invariant scattering for texture discrimination, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 1233–1240.
[15] T.-H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, Y. Ma, PCANet: A simple deep learning baseline for image classification? IEEE Trans. Image Process. 24 (12) (2015) 5017–5032.
[16] Y. Gan, J. Liu, J. Dong, G. Zhong, A PCA-based convolutional network, 2015, arXiv preprint arXiv:1505.03703.
[17] E. Oyallon, E. Belilovsky, S. Zagoruyko, Scaling the scattering transform: Deep hybrid networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5618–5627.
[18] E. Oyallon, A Hybrid Network: Scattering and Convnet, 2016.
[19] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1026–1034.
[20] J. Bruna, S. Mallat, Invariant scattering convolution networks, IEEE Trans. Pattern Anal. Mach. Intell. 35 (8) (2013) 1872–1886.
[21] F. Cotter, N. Kingsbury, A learnable ScatterNet: Locally invariant convolutional layers, 2019, arXiv preprint arXiv:1903.03137.
[22] R. Keshari, M. Vatsa, R. Singh, A. Noore, Learning structure and strength of CNN filters for small sample size training, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 9349–9358.
[23] Q. Sun, Y. Liu, T.-S. Chua, B. Schiele, Meta-transfer learning for few-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 403–412.
[24] D.E. Worrall, S.J. Garbin, D. Turmukhambetov, G.J. Brostow, Harmonic networks: Deep translation and rotation equivariance, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5028–5037.
[25] R. Ghosh, A.K. Gupta, Scale steerable filters for locally scale-invariant convolutional neural networks, 2019, arXiv preprint arXiv:1906.03861.
[26] M. Weiler, F.A. Hamprecht, M. Storath, Learning steerable filters for rotation equivariant CNNs, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 849–858.
[27] S. Fujieda, K. Takayama, T. Hachisuka, Wavelet convolutional neural networks, 2018, arXiv preprint arXiv:1805.08620.
[28] L. Liu, J. Wu, D. Li, L. Senhadji, H. Shu, Fractional wavelet scattering network and applications, IEEE Trans. Biomed. Eng. 66 (2) (2018) 553–563.
[29] H. Huang, R. He, Z. Sun, T. Tan, Wavelet-srnet: A wavelet-based CNN for multi-scale face super resolution, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 1689–1697.
[30] H. Khan, B. Yener, Learning filter widths of spectral decompositions with wavelets, in: Advances in Neural Information Processing Systems, 2018, pp. 4601–4612.
[31] S. Fujieda, K. Takayama, T. Hachisuka, Wavelet convolutional neural networks for texture classification, 2017, arXiv preprint arXiv:1707.07394.
[32] M. Yang, Z.-H. Liu, Z.-D. Cheng, J.-S. Xu, C.-F. Li, G.-C. Guo, Deep hybrid scattering image learning, J. Phys. D: Appl. Phys. (2018).
[33] A. Singh, ScatterNet Hybrid Frameworks for Deep Learning (Ph.D. thesis), University of Cambridge, 2019.
[34] A. Singh, N. Kingsbury, Generative scatternet hybrid deep learning (G-SHDL) network with structural priors for semantic image segmentation, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, IEEE, 2018, pp. 2991–2995.
[35] A. Katzmann, M.-S. Seibel, A. Mühlberg, M. Sühling, D. Nörenberg, S. Maurus, T. Huber, H.-M. Groß, Hybrid Rotation Invariant Networks for Small Sample Size Deep Learning, 2018.
[36] E.J. Bekkers, M.W. Lafarge, M. Veta, K.A. Eppenhof, J.P. Pluim, R. Duits, Roto-translation covariant convolutional networks for medical image analysis, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2018, pp. 440–448.
[37] M. Unberath, J.-N. Zaech, S.C. Lee, B. Bier, J. Fotouhi, M. Armand, N. Navab, DeepDRR–a catalyst for machine learning in fluoroscopy-guided procedures, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2018, pp. 98–106.
[38] F. Visin, K. Kastner, K. Cho, M. Matteucci, A. Courville, Y. Bengio, Renet: A recurrent neural network based alternative to convolutional networks, 2015, arXiv preprint arXiv:1505.00393.
[39] I.W. Selesnick, R.G. Baraniuk, N.G. Kingsbury, The dual-tree complex wavelet transform, IEEE Signal Process. Mag. 22 (6) (2005) 123–151.
[40] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015, pp. 234–241.
[41] A. Krizhevsky, G. Hinton, et al., Learning Multiple Layers of Features from Tiny Images, Tech. rep., Citeseer, 2009.

[42] Y. Le, X. Yang, Tiny imagenet visual recognition challenge, 2015, CS 231N.
[43] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al., Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[44] Y. LeCun, F.J. Huang, L. Bottou, et al., Learning methods for generic object recognition with invariance to pose and lighting, in: CVPR (2), Citeseer, 2004, pp. 97–104.
[45] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al., Matching networks for one shot learning, in: Advances in Neural Information Processing Systems, 2016, pp. 3630–3638.
[46] S. Gerhard, J. Funke, J. Martel, A. Cardona, R. Fetter, Segmented Anisotropic ssTEM Dataset of Neural Tissue, figshare, 2013.
[47] J. Staal, M.D. Abràmoff, M. Niemeijer, M.A. Viergever, B. Van Ginneken, Ridge-based vessel segmentation in color images of the retina, IEEE Trans. Med. Imaging 23 (4) (2004) 501–509.
[48] J.T. Springenberg, A. Dosovitskiy, T. Brox, M. Riedmiller, Striving for simplicity: The all convolutional net, 2014, arXiv preprint arXiv:1412.6806.
[49] A. Romero, N. Ballas, S.E. Kahou, A. Chassang, C. Gatta, Y. Bengio, Fitnets: Hints for thin deep nets, 2014, arXiv preprint arXiv:1412.6550.
[50] K. He, X. Zhang, S. Ren, J. Sun, Identity mappings in deep residual networks, in: European Conference on Computer Vision, Springer, 2016, pp. 630–645.
[51] S. Zagoruyko, N. Komodakis, Wide residual networks, 2016, arXiv preprint arXiv:1605.07146.
[52] J. Snell, K. Swersky, R. Zemel, Prototypical networks for few-shot learning, in: Advances in Neural Information Processing Systems, 2017, pp. 4077–4087.
[53] C. Finn, P. Abbeel, S. Levine, Model-agnostic meta-learning for fast adaptation of deep networks, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org, 2017, pp. 1126–1135.