IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON THEORY, ALGORITHMS, AND APPLICATIONS OF SPARSE RECOVERY

Received December 19, 2018, accepted January 2, 2019, date of publication January 23, 2019, date of current version February 20, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2894841

Procedural Learning With Robust Visual Features via Low Rank Prior

HAIFENG LI[1,2], (Member, IEEE), LI CHEN[1], HAILUN DING[3], QI LI[4], BINGYU SUN[5], AND GUOHUA WU[6]

[1]School of Geosciences and Info-Physics, Central South University, Changsha, China
[2]Henan Laboratory of Spatial Information Application on Ecological Environment Protection, Zhengzhou, China
[3]School of Software, Central South University, Changsha, China
[4]School of Information Science and Engineering, Central South University, Changsha, China
[5]Institute of Intelligent Machine, Chinese Academy of Sciences, Hefei, China
[6]School of Traffic and Transportation Engineering, Central South University, Changsha, China

Corresponding author: Guohua Wu (guohuawu.nudt@gmail.com)

This work was supported in part by the National Science Foundation of China under Grant 41571397, Grant 41501442, Grant 51778242, Grant 61773360, and Grant 41871364, and in part by the Natural Science Foundation of Hunan Province under Grant 2016JJ3144 and Grant 2016JJ2006.

ABSTRACT In order to apply a convolutional neural network (CNN) to unseen datasets, a common way is to train a CNN using a pre-trained model on a big dataset by fine-tuning it instead of starting from scratch. How to control the fine-tuning progress to get the desired properties is still a challenging problem. Our key observation is that the visual features of the pre-trained model have rich information and can be explored during the training process. A natural thought is to employ these features and design a control strategy to improve the performance of the transfer learning process. In this paper, a procedural learning framework using the learned low-rank component of the visual features both in the pre-trained model and the training process is proposed to improve the accuracy and generalizability of the CNN. In this framework, we presented an approach to yield independent visualization features (IVFs). We found via robust independent component analysis that the low-rank components of IVFs provided robust features for our framework. Then, we design a Wasserstein regularization to control the transportation of the distribution of IVFs from a pre-trained model to a final model via the Wasserstein distance. The experiments on the Cifar-10 and Cifar-100 datasets via a VGG-style CNN model showed that our method effectively improves the classification results and convergence speed. The basic idea is that exploring visual features can also potentially inspire other topics, such as image detection and reinforcement learning.

INDEX TERMS Low-rank approximation, procedural learning, knowledge transfer, robustness visual feature, sparse.

I. INTRODUCTION

Convolutional neural network (CNN) has achieved remarkable successes in computer vision, remote sensing image recognition tasks [1], [2]. The training process of the CNN is essentially the extraction of the image features layer by layer. The obtained hierarchical features can be adapted to the underlying distribution of the input data so that the CNN can correctly classify images [1], [3]. However, given that it is limited by the quality and quantity of samples, it is difficult to learn better features in each layer, which results in the decline of the network's generalizability. Therefore, fine-tuning, which is a method that uses pre-trained network weights as the initialization weights for new tasks, allows the model to well learn the image features [4], [5]. Although the fine-tuning provides a better direction for the initial weight information, it cannot play a continuous auxiliary role in training. Recent studies have shown that the rational use of the features acquired by the pre-trained model can also improve the performance in transfer learning [6], [7]. To effectively improve the transfer features, we need to consider the following two basic questions. (1) What are the robust features of the CNN? (2) How do we to transfer the features appropriately?

The activation maximization, which achieves the learned features using the CNN, is proposed [8]. It is assumed that when the input data have high activation after the convolution

2169-3536 © 2019 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

kernel, that the features from the convolution kernel have strong correlations with the input data. We can get the visual features with the maximum activation after the convolution kernels by fixing the weights of the trained neural networks and using the gradient ascent method. However, the visual features obtained using the naive activation maximization technique are not a completely independent representation of the convolution kernel (the details are in the 2nd section). Therefore, we suppress the activation of the other convolution kernels in this layer and achieve the independent representation of the convolution kernel by modifying the objective function of the generated image. Nevertheless, the visual features still have a lot of redundant information that affects the robust features, and so we obtain independent visual features via Robust Principal Component Analysis (RPCA). RPCA, which was proposed by Wright *et al.* [9], Candès *et al.* [10], and Xu *et al.* [11], seeks the best low-rank estimation of a matrix. The decomposed visual features contain low-rank components with robust visual features, sparse errors and redundant information [12]. RVFs, which are the low-rank components of IVFs, belong to the robust representation of a network, which can help new tasks improve without being affected by datasets.

Further, we need to transfer the robust visual features (RVFs) to other learning tasks. However, the learned features of the CNN always change. This means that when using the same model and the same dataset, in spite of the similar performances, they may have different weights. Since the optimization of the CNN is a non-convex problem and the model is over-parameterized, it cannot get a unique solution. The CNN weights are therefore very different, and the visual features also have big differences at the corresponding convolution kernel. It is hard to transfer the RVFs from convolution kernels. Therefore, we treat all the RVFs in each layer as a whole and use their distribution as the base property. As we know, the Wasserstein distance that is sensitive to the differences between distributions can be applied to measure the RVFs distribution [13]. It can effectively help us to control the transfer of RVFs.

Generally speaking, we propose that procedural learning allows the pretrained model to provide a good initial value and, through the use of the learned features, the training performance is improved. We use the distribution of the RVFs to yield the IFVs via RPCA and design a new Wasserstein regularization to realize the feature transfer in each layer. Furthermore, recent works demonstrate that the features in lower layers tend to converge faster than the learning in higher layers [14]. The low-layer features are simple and stable, and the high-layer features are complex and volatile. Therefore, we use different hyper parameters in each layer. The experiments show that this method has a good effect on small datasets. The main contributions of this paper are as follows.

(1) The robust visual features achieved using the robust principal component analyses are the robust features of the CNN. These robust features help to improve the convergence speed and accuracy of the fine-tuning or transfer process from one dataset to another one.

(2) The Wasserstein regularization is designed to control the transfer degree of the robust visual features. We also found that the difficulty of transferring features is correlated with the layers, which provided evidence to support the designed variant strategy for the different layers.

(3) To the best of our knowledge, this is the first time that robust visual features have been used to supervise the training phase, which could be a potential requirement for self-supervised learning.

II. PROCEDURAL LEARNING FRAMEWORK

Due to the limited quality and quantity of samples, although the fine-tuning method using pre-trained model weights as the initialization weights of the new tasks allows the model to well learn the features, the pre-trained model cannot help in the process of network training. Procedural learning is a pre-trained model that can help another model with the same structure or part of the same layer structure to provide rich visual features in the training of new tasks. This means that the current task acquires better generalization performance through the knowledge of the pre-trained model during the training process. Unlike transfer learning [15], procedural learning does not have any differences in data distributions between the source and target domains or any differences in the predictive functions. The idea of visual features in a pre-trained model needs be explored. We start with a good pre-trained model that was trained using large datasets. The ImageNet datasets contain millions of images and 1000 classes [16]. Generally, models trained using the ImageNet datasets have high generalization and contain rich visual features in each layer. With the activation maximization algorithm, we can simply visualize the learned features in the convolution kernel. This visual feature can be transferred between the pre-trained model and the new model in the new task.

We extend the native activation maximization algorithm [8], which only analyzes the visual features (VFs) of one convolution kernel without considering other convolution kernels. We propose the independent activation maximization algorithm that can retain the visual features of the convolution kernel while suppressing other convolution kernels' activations. However, this is not sufficient because the large datasets are complex. For the pre-trained model to achieve good results, the VFs of each convolution kernel will retain a lot of meaningless noise. We apply robust principal component analysis (RPCA) to obtain the low-dimensional subspaces of the independent visual features (IVFs). These robust visual features (RVFs) better represent the learned features of the convolution kernel. When each convolution kernel's visual feature is disentangled from the features of other convolution kernels, the overall features in each convolutional layer are abundant. Considering the hierarchical features of the CNN, the features in the same layer have relatively similar representations. The distribution of the RVFs in each layer in a

pre-trained model can serve as the target distribution for the approximation of each layer in the new network's training phase. To reduce the computational complexity, the VFs of the model require generalizability under new tasks. It is only necessary to obtain the IVFs of the convolution kernel. In this case, we design a Wasserstein regularization that could update the weights of the new network based on the differences in the feature distributions in each layer between the training model and the pre-trained model. Fig.1 gives an illustration of the procedural learning framework.

III. THE IFV VIA ROBUST PRINCIPAL COMPONENT ANALYSIS METHOD

A. INDEPENDENT ACTIVATION MAXIMIZATION

In the image classification task, it is assumed that for a training set $D = \{X^n, c^n\}_{n=1}^N$. X^n represents the input data of the n-th sample, and y^n represents the corresponding label. $\Theta = \{W^l, b^l\}_{l=1}^L$ represents the weights that the CNN needs to train. W^l represents the weight on the l-th layer, and b^l is the bias. The last layer of the *softmax* output is defined as $f(X) = e^X / \sum_{j=1}^{|X|} e^{X^j}$. The empirical risk of the CNN is minimized as

$$R(f) = \min_{\Theta} \frac{1}{N} \sum_{n=1}^{N} J(f(X^n), c^n). \tag{1}$$

The CNN mainly uses multiple hidden layers to obtain the hierarchical features of the input data. The internal learning mechanism is called the black box, which is still an open question. To improve the interpretability of CNNs, Erhan *et al.* [8] proposed a visualization method for the learned features using the convolution kernel.

Features of the input data can be captured using the convolution kernel, as well as some visual features of the convolution kernel. We believe that when the image is input into the CNN and the output value of the convolution kernel is large, the convolution kernel is considered to have high activation of certain features of the image. In fact, the images cannot cover all styles in the dataset, and some of them are low quality. High activations of the convolution kernel in this dataset are obtained; meanwhile, it is also difficult to define what features the convolution kernel learned. The learned features of that convolution kernel cannot be determined. Therefore, in order to obtain the common features, the acquisition of the visual features can eliminate the data limitations and use the weights from the CNN. Erhan *et al.* [8] propose the method of activation maximization. It initializes the input image with random noise, propagates the image passing through the CNN to compute the activation of the target convolution kernel, and then propagates the activation backwards through the CNN to compute the update direction of the input image.

When the CNN converges using the back propagation algorithm, we can get a set of weights Θ^*. $h_{ij}(X, \Theta)$ is defined as the value of the j-th convolution kernel in the i-th layer. By fixing the weights Θ^*, we can solve the following objective function:

$$X^* = \arg\max h_{ij}(X, \Theta^*). \tag{2}$$

The final picture X^* can be used as a feature learned from the convolution kernel, and it is called the maximum activation image. In the following studies [17]–[20], regularization is added to the original objective function to make the generated maximum activation image more intuitive. Recently, Olah *et al.* [21] obtained a high-quality maximum activation image by adjusting the gradient. In general, the main purpose of these methods is to reduce the high-frequency noise in the maximum activation image, and to focus on the interpretability of the generated maximum activation image. However, these methods do not consider the effect of the maximum activation image on other convolution kernels.

We use the maximum activation image for one of the convolution kernels in the layer, and then forward it through the CNN to obtain the output value of all convolution kernels in that layer. We find that the activation value of the convolution kernel may not be the maximum of this layer since there are other kernels on which the image can obtain a larger activation value. Although this maximum activation image is the maximum activation of the convolution kernel, it may not be an independent visual feature of the convolution kernel. This phenomenon gradually decreases with deeper layers, which coincide with the observation that it is independent of the high-layer semantic features [22]. Therefore, the visual feature of the maximum activation image is entangled between the convolution kernels.

In procedural learning, we need to extract the IVFs of the convolution kernels. While the maximum activation image of the target convolution kernel can be disentangled, the activations of other convolution kernels are lower. This means that we need to get the maximum $(h_{ij}(X, \Theta^*) - h_{i,-j}(X, \Theta^*))$. $h_{i,-j}(X, \Theta^*))$ represents all the maximum activation images after removing kernel j in layer i. There are many convolution kernels in each convolution layer, and the outputs of a convolution kernel needs to suppress the outputs of all the other convolution kernels. The average activation value of each convolution kernel is subtracted to achieve the IVFs as follows:

$$X^* = \arg\max(h_{ij}(X, \Theta^*) - \frac{1}{J} \sum_{j=1}^{J} h_{i,-j}(X, \Theta^*)). \tag{3}$$

The IVFs reflect the independent learned features of the convolution kernels, and these features alleviate the entanglement of different convolution kernels.

B. THE ROBUST VISUAL FEATURES

For the pre-trained model, it is the provider of the transfer features in training. The pre-trained models have rich IVFs based on large datasets. Nevertheless, the IVFs still have problems. Each visual feature is not so stable relative to the small dataset of the new task because the richness of the representations is already greater than the IVFs of the small dataset.

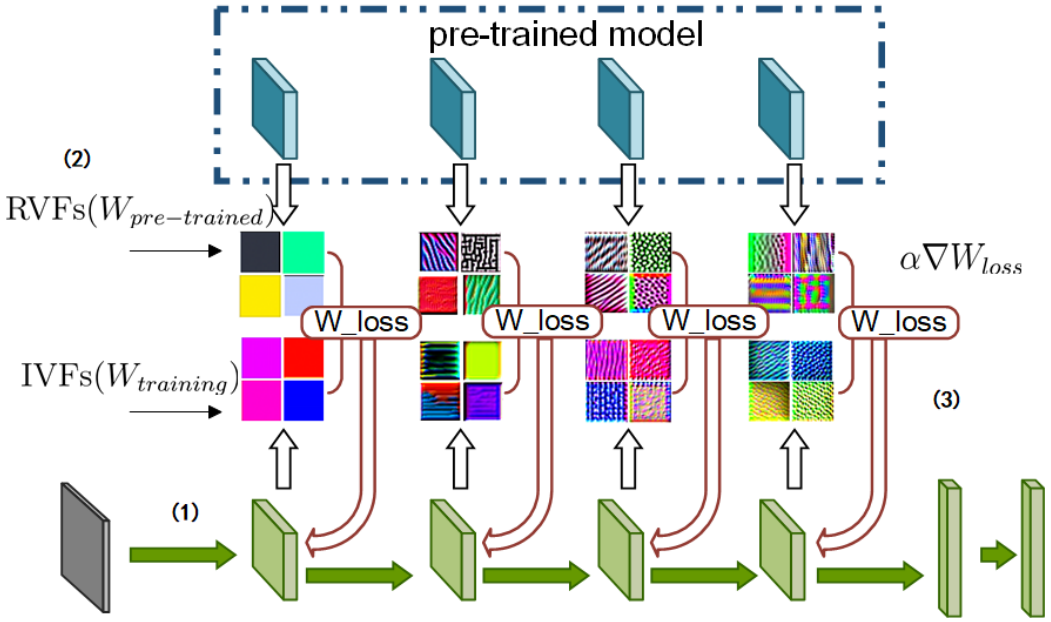


FIGURE 1. The framework for procedural learning. The method can be divided into three steps. (1) The RVFs of the pre-trained model are achieved by the IFV via the RPCA method. (2) The network is trained using fine-tuning, and we obtain all of the IVFs of each layer of the training model at the same time. (3) The difference in the distributions of the features is measured using the designed Wasserstein regularization and the network weights are updated.

The IVFs of the pre-trained model contain representations in other directions in the dataset. These directions have little effect on the original dataset, but for new tasks, the direction of feature optimization is greatly affected. Therefore, when the features of the pre-trained model are used to guide the learning features of the new task model, it is important to use the main direction of the IVFs, and additional feature directions make the training results unsatisfactory. We need to get the robust features of the pre-trained model that preserve the main structures of the features and are not disturbed.

The structure of the IVFs is decomposed and the noise is removed. The decomposition form is suitable for the application of the RPCA. It is a matrix decomposition method. To solve the low-rank matrix recovery problem, RPCA decomposes the matrix into two matrix components. One of them is a low-rank matrix, which mainly contains the structure of the matrix. The other is a sparse matrix, which is mainly redundant noise. The given independent visual feature is $X^* \in \mathbb{R}^{m\times n}$. Its decomposition process is $X^* = S+N$. The matrix S is the low-rank matrix of the IVFs, which retains the structure of the features, and it includes the main direction of the features. The matrix E is sparse redundant noise. The optimization problem can be shown as follows:

$$\min_{S,N}(\|S\|_*, \|N\|_0) \quad s.t.\ X^* = S + N. \tag{4}$$

Let $\|S\|_* = \sum_i \sigma_i(S)$ denote the nuclear norm of the matrix S, and let $\|N\|_0$ denote the ℓ_0-norm of N, which is the number of non-zero elements of the matrix N. The recovery of the low-rank matrix S and the sparse N is a two-objective optimization problem. By a factor $\lambda > 0$, the optimization problem in Eq.4 can be converted into a single-objective optimization problem as follows:

$$\min_{S,N}(\|S\|_* + \lambda\|N\|_0) \quad s.t.\ X^* = S + N. \tag{5}$$

In practice, let $\lambda = \frac{1}{\sqrt{\max(m,n)}}$ [11]. The optimization problem in Eq.5 is an NP-hard problem, so the objective function of this problem needs to be relaxed. Since the nuclear norm of the matrix is the envelope of the matrix rank, the ℓ_1 norm of the matrix is a convex hull of the ℓ_0 norm. Thus, the optimization problem in Eq.5 is relaxed to a convex optimization problem as follows,

$$\min_{S,N}(\|S\|_* + \lambda\|N\|_1) \quad s.t.\ X^* = S + N. \tag{6}$$

Let $\|N\|_1 = \sum_{ij}|E_{ij}|$ denote the ℓ_1-norm of E, which is a long vector in $\mathbb{R}^{m\times n}$. We use the dual approach to solve this optimization problem [23]–[25]. Since the dual norm of the nuclear norm is the spectral norm, the dual problem of the optimization problem in Eq.6 is shown as follows:

$$\max_{Y}\langle X^*, Y\rangle \quad s.t.\ J(Y) \le 1, \tag{7}$$

where

$$\langle X^*, Y\rangle = \mathrm{tr}(X^{*T}Y) \quad J(Y) = \max(\|Y\|_2, \lambda^{-1}|Y|_\infty), \tag{8}$$

and $|Y|_\infty$ is the maximum absolute value of the matrix Y. When the optimization problem in Eq.7 obtains the optimal value Y^*, $J(Y^*) = 1$ must be satisfied. It is obvious that the problem is non-linear, non-smooth and can be solved using the constrained steepest ascent. First, we find the projection Y_k of Y onto the normal cone $N(Y)$. We define $N(Y_k) = \{aD :$

$a \leq 0, D \in \partial J(Y_k)\}$ and the steepest ascent direction W_k as $W_k = X^* - x_k^*$. Then, we do a line search along the direction using

$$\delta_k = \arg\max_{\delta \geq 0} \langle X^*, \frac{Y_k + \delta W_k}{J(Y_k + \delta W_k)} \rangle, \tag{9}$$

and the estimate of Y is updated as

$$Y_{k+1} = \frac{Y_k + \delta W_k}{J(Y_k + \delta W_k)}. \tag{10}$$

The matrixes S and N can be easily obtained using the estimated value $\hat{Y}$. The matrix S is the RVF of the pre-trained model. It effectively helps learn new tasks. The RVFs in each layer are achieved and stored before the new task begins. It also needs an algorithm in order to properly use the RVFs to train new model during the training process.

IV. WASSERSTEIN REGULARIZATION

A. WASSERSTEIN DISTANCE

The optimization problem of the CNN is a non-convex optimization. The local optimal solutions obtained for each model are different. Intriguingly, the performances of the generalization abilities are not significantly different [26]. From the perspective of the maximum activation image, for example, we assume that the learned features of the second convolution kernel in the first layer imply a rightward direction. In another local optimal, the learned features using the same convolution kernel may imply a leftward direction. This does not mean that there is no convolution kernel for representing a leftward direction for another local optimum solution. The models get different local optimums, and therefore, the model's generalization have little difference. In addition, these also show that the RVFs of the corresponding convolution kernel to be used as the transfer features are inappropriate. We use the RVFs' distribution in the convolutional layer to promote learning in new tasks.

Recently, the Wasserstein distance has made a lot of progress in the field of machine learning [27]–[29]. Compared with the traditional KL algorithm, the Wasserstein distance is more in line with the mathematical definition of the distance. The differences between the distributions are also linearly related to the Wasserstein result. The Wasserstein distance formula is shown as follows:

$$w(P, Q) = \inf_{\gamma \in \Pi(P,Q)} \mathbb{E}_{(x,y)\gamma}[\|x - y\|]. \tag{11}$$

P and Q denote two distributions that need to be measured. $\Pi(P, Q)$ indicates the joint distribution of P and Q. (x, y) represents the sampling data for some joint distribution γ. We can see that the Wasserstein distance is defined as the lower bound of the sample's expected distance $\mathbb{E}_{(x,y)\gamma}[\|x - y\|]$ under all joint distributions.

In procedural learning, P represents the distribution of the RVFs in the convolutional layers in the pre-trained model. For new tasks, we choose the IVFs with low computational complexity that better represent the new small dataset. Q represents the distribution of the IVFs on the same layer in the new task model. The distribution of features in each layer can be estimated using an empirical distribution. From [8], the learned features of each convolution kernel usually appear repeatedly. This repetitive feature is meaningless, and it increases the computational complexity. The VFs can be appropriately reprocessed like with mean pooling. Its empirical distribution can be written as

$$P_i = \sum_{j=1}^{J} x_{p,j}^* \delta \quad Q_i = \sum_{j=1}^{J} x_{q,j}^* \delta. \tag{12}$$

δ represents the Dirac function for each pre-processed VF. P_i and Q_i represent the i-th convolution layer. x^* represents the estimation of each visual feature. By setting $\sum x_{p,j}^* = \sum x_{q,j}^* = 1$, Eq.11 can be solved using the Kantorovich formula. Let B denote the probability coupling sets between two empirical distributions as follows:

$$B = \{\gamma \in \mathbb{R}^{\kappa \times \kappa} | \gamma 1_\kappa = P_i, \gamma^T 1_\kappa = Q_i\}. \tag{13}$$

1_κ represents an all-ones vector with the dimension of κ. The solution of the Eq.11 is as follows,

$$\gamma_0 = \arg\min_{\gamma \in B} \langle \gamma, C \rangle_F. \tag{14}$$

$\langle ., . \rangle_F$ represents the Frobenius dot product. $C = c(P_i, Q_i)$ represents the difference between distributions P_i and Q_i, which mainly reflects cost differences between the VFs of the convolutional kernels from one network to another network. We define the L_1 distance as the difference,

$$C(a, b) = \|P_a - Q_b\|_1. \tag{15}$$

After the definition is complete, we need to solve the problem.

B. OPTIMIZATION VIA ENTROPIC REGULARIZATION

γ_0 is a linear problem; however, the time complexity of a direct solution is $n^3 \log n$, which is time consuming when the number of VFs is large. We add a regularization term so that the accuracy is relaxed while the computations are significantly accelerated. The loss of accuracy does not affect the final calculation.

We turn the primal problem into the dual problem [30]–[32], as shown below

$$^d W_p^p(P, Q) = \sup_{\alpha, \beta \in C_M} \alpha^\top P + \beta^\top Q,$$
$$M_C = \{(\alpha, \beta) \in \mathbb{R}^{\kappa \times \kappa} : \alpha_\kappa + \beta_{\kappa'} \leq C_{\kappa,\kappa'}\}. \tag{16}$$

Eq.11 is a linear problem, and so its dual form is equivalent to the primal problem. It is easy to see that the Lagrange variable α is the sub-gradient of the first parameter. However, the computational complexity of α is still very high. Its time complexity is $O(K^2)$, where K represents the dimension of the output space. Cuturi [33] proposed a method of adding

regular terms that turned the primal problem into a strictly convex function. The form is as follows:

$$
\begin{aligned}
{}^{\lambda}W_p^p(P,Q) &= \inf_{T\in\Pi(P,Q)} \langle \gamma, C\rangle - \frac{1}{\lambda}H(\gamma),\\
H(\gamma) &= -\sum_{\kappa,\kappa'} \gamma_{\kappa,\kappa'} \log \gamma_{\kappa,\kappa'}.
\end{aligned} \tag{17}
$$

The solution of the transport matrix depends on the *diagonal scaling* of the matrix $\mathbf{K} = e^{-\lambda C-1}$. The solution is as follows:

$$
\gamma^* = \mathrm{diag}(u)\mathbf{K}\mathrm{diag}(v). \tag{18}
$$

for $u = e^{\lambda\alpha}$ and $u = e^{\lambda\beta}$, where α and β are the Largrange variables for Eq.17. The form of the problem, which is also equivalent to the *matrix balance* [34], has been studied using numerical linear algebra and can be solved using the Sinkhron-Knopp algorithm [35]. In this case, the gradient a of the objective function can be determined by the scaling vector u as

$$
\alpha = \frac{\log u}{\lambda} - \frac{\log u^{\top}\mathbf{1}}{\lambda K}\mathbf{1}. \tag{19}
$$

Therefore, we can use a more efficient iterative algorithm to compute the gradient, and we use the Wasserstein distance to train the network weights.

C. PROCEDURAL LEARNING ALGORITHM

When the differences in the distributions of VFs are obtained, we use the hyper parameter flexibly in each layer that is a consequence of the diverse complexity of the low-layer features and high-layer features.

$$
L(\theta) = L_B(\theta) + \sum_{i=1}^{I} \alpha_i W(p_i, q_i). \tag{20}
$$

$L_B(\theta)$ represents the normal loss function in the training process, and I represents the different layers. The IFV-RPCA algorithm applies RPCA to obtain the RVFs of the IVFs. The IFVs algorithm is a gradient ascent algorithm, which requires significant iterations to achieve better convergence. In fact, the differences between two VFs do not require an exact solution for procedural learning. Just like with teachers' guidance to students, it is sufficient to correct the learned features at some critical moments. Therefore, the regularization term can be added after a few epochs. We choose one epoch. The process learning algorithm can be summed up as Alg.1.

Algorithm 1 Procedural Learning Algorithm

Data: D, $W_{pre\_trained}$
Result: W_{new_task}
$\mathrm{RVFs}_{pre\_trained} \leftarrow$ IFV by RPCA($W_{pre\_trained}$) ;
Fine tune a pre-trained model;
Initialize $W_{new\_task} \sim W_{pre\_trained}$;
Usual loss function ;
while $D_{batch} \in D$ **do**
  $W_{new\_task} \leftarrow W_{new\_task} - \alpha\nabla W_{new\_task}$
end
$\mathrm{IVFs}_{new\_task} \leftarrow \mathrm{IVF}(W_{pre\_task})$;
Wasserstein regularization ;
$W\_loss = Wasserstein(\mathrm{IVFs}_{new\_task}, \mathrm{RVFs}_{pre\_trained})$;
$W_{new\_task} \leftarrow W_{new\_task} - \alpha\nabla W\_loss$

V. EXPERIMENTS AND ANALYSIS

A. EXPERIMENTAL SETUP

We design a CNN with a 4-layer VGG-style architecture. The number of convolution kernels of each layer is the same as the first four layers of the VGG16 network [36]. The layer sizes are as follows (stride, filter sizes, output channels): (1×1, 3×3, 64), (1×1, 3×3, 64), (2×2, 3×3, 128), and (1×1, 3×3, 128). The model easily uses the weights of the pre-trained models in ImageNet and receives guidance from the RVFs in training. Then, we use the VGG16 network. The datasets are Cifar-10 and Cifar-100 [37], both of which contain 60,000 images. The difference between them is that Cifair10 contains 10 classes of images, while Cifar-100 contains 100 classes.

To explore the superiority of visual features in the direct use of the pre-trained model's weights, we build another model where we replace the Wasserstein regularization with the L2 regularization and compare it with procedural learning. According to our preliminary analysis, independent maximum activation images are more valuable. To prove this, we compare the RVF algorithm with the IVF algorithm. Moreover, different initialization methods also have different impacts on procedural learning. Therefore, we use different initialization methods. In summary, 8 sets of experiments are designed for each dataset. They are learning from scratch (Scratch), learning from fine-tuning (Tune), procedural learning from scratch (Scratch+RVFs), procedural learning from fine-tuning and (Tune+RVFs), learning from scratch with L2 regularization (Scratch+L2), learning from fine-tuning with L2 regularization (Tune+L2), learning from scratch with the IVF algorithm (Scratch+IVFs), and learning from fine-tuning with the IVF algorithm (Tune+IVFs). In the experiments, the batch size is 32. In the fine-tuning series, the learning rate is 0.0006. In addition, the learning rate is 0.001 in the training from scratch series. In procedural learning, the learning rate of the first two layers is 0.0002, and the learning rate of the last tow layers is 0.0005.

B. RESULTS AND DISCUSSION

First, we obtain the IVFs of the model and apply the RPCA algorithm to further obtain the RFVs. Some experimental results are shown in Fig.2. The RVFs retain their robust representations for the convolution kernel. Although the differences between IVFs and RFVs are not significant, the experimental results are different. According to the experimental settings, the final results of all the experiments on different datasets are shown in Table.1. The procedural learning with the initialization of the pre-trained model's weights

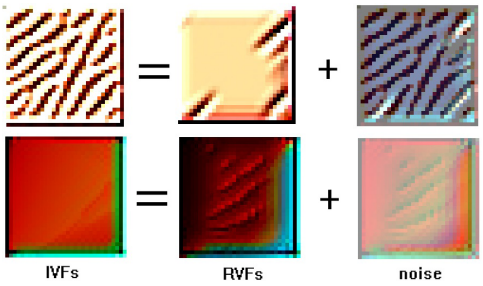


FIGURE 2. The relationship between the RVFs and IVFs. The RVF is a low-rank structure of the IVF.

obtains the best performance on both datasets. The accuracy and loss of all methods in the training process are shown in the Fig.3. The methods using the fine-tuning series achieve better results than the training from scratch series. In both series, the methods combined with procedural learning have the best performance. Among all methods, fine-tuning with procedural learning converges quickly and achieves good results.

The L2 regularization keeps the new network's weights close to the original model. The original weights are used directly. It is like freezing the corresponding convolutional layer. Only a small update is made to the weights in the training process, and the generalization ability is completely limited by the quality of the pre-trained model. In the fine-tuning series experimental results, L2 regularization has a negative impact. The basic fine-tuning model achieves a better result than combining it with L2 regularization. In the training from scratch series, the final effect of the L2 regularization is similar to the benchmark method. However, it has a faster convergence speed and less loss than the basic training from scratch method. This also shows that the direct use of weights for the transfer is not effective.

By comparing the RVFs algorithm with the IVFs algorithm, we find that in the training from scratch series, the result of using the RVFs algorithm is better than that obtained by the IVFs algorithm. In the Cifar-10 experiment, the decrease in the loss of the IVFs algorithm takes a very rugged route and even has a large jitter. The final classification accuracy of the model is also very poor. In the fine-tuning series of the methods, the method using the RVFs algorithm on the Cifar-10 dataset achieves the best results. It converges faster than the baseline fine-tuning method, and the loss decreases faster. It also can be observed that at the 15th round, this method exceeds the results of the IVFs algorithm. However, the accuracy of the procedural learning method is improved only marginally on the Cifar-100 dataset. Its effectiveness is not obvious.

We use a deeper model, the VGG16, on Cifar-10, and the accuracy of the test set during the training process is shown in Fig.4. The method of fine-tuning series is still better than training from scratch. The results of each method are shown

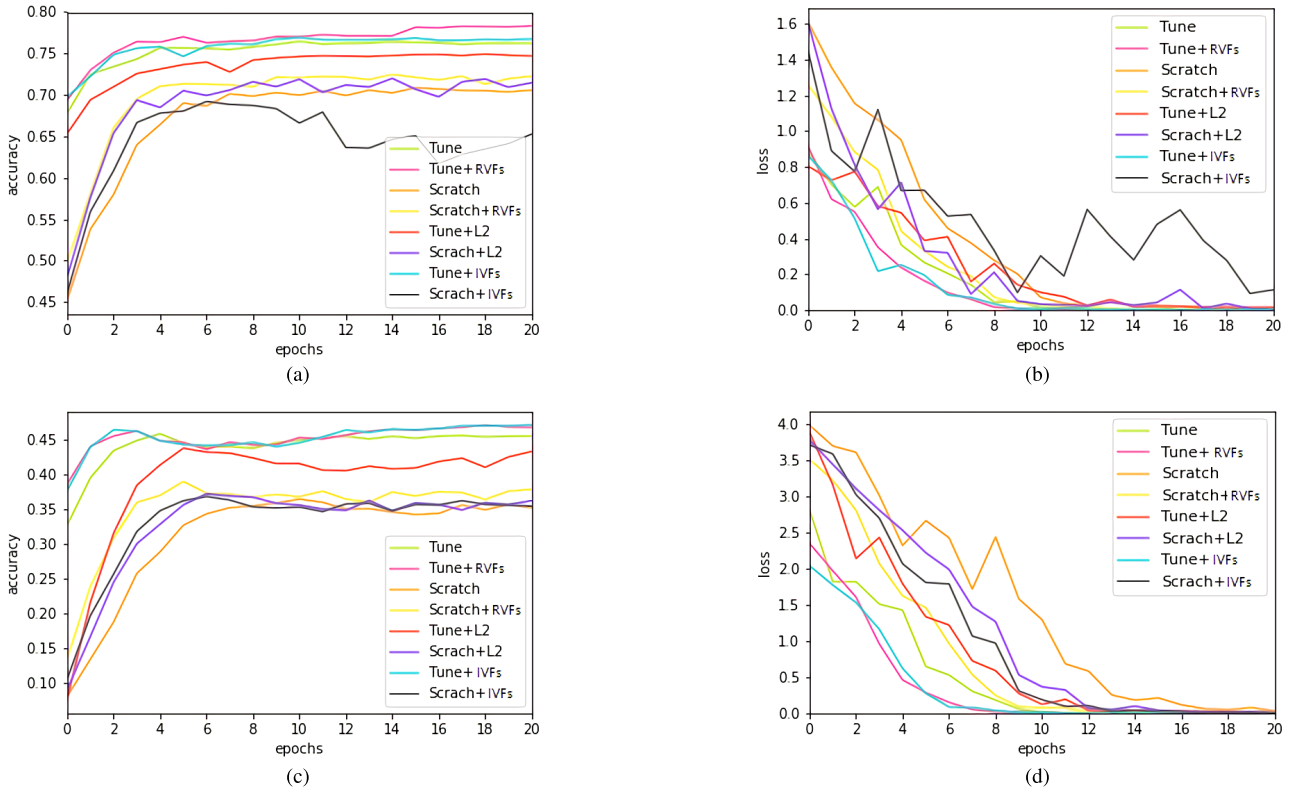


FIGURE 3. Training on the Cifar-10 and Cifar-100 datasets. In this plot, we compare procedural learning with other methods. (a) Accuracy on Cifar-10. (b) Training error on Cifar-10. (c) Accuracy on Cifar-100. (d) Training error on Cifar-100.

TABLE 1. Accuracy on the Cifar-10 + Cifar-100 datasets with various methods on the shallow network.

Dataset	Scratch	Scratch+L2	Scratch+IVFs	Scratch+RVFs	Tune	Tune+L2	Tune+IVFs	Tune+RVFs
Cifar-10	70.55 ± 0.2	71.45 ± 0.1	65.25 ± 0.1	72.24 ± 0.2	76.21 ± 0.0	74.69± 0.1	76.73± 0.2	**78.32**± 0.1
Cifar-100	35.18 ± 0.1	36.19± 0.2	35.43± 0.0	37.81± 0.3	45.47± 0.0	43.26 ± 0.1	46.72 ± 0.2	**47.07**± 0.2

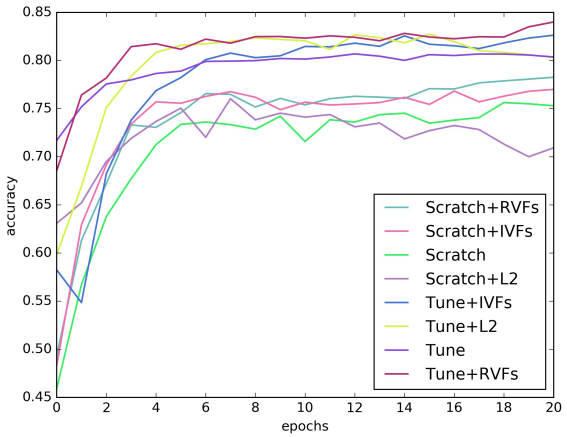


FIGURE 4. Accuracy on Cifar-10. We compare the procedural learning with other methods using VGG16.

TABLE 2. Accuracy on the Cifar-10 datasets with various methods using VGG16.

Method	Original	with L2	with IVFs	with RVFs
Scratch	75.49± 0.2	70.01± 0.3	76.79± 0.1	**78.05**± 0.2
Tune	80.56± 0.1	80.60± 0.2	82.32± 0.2	**83.49**± 0.1

in Table.2. In addition to the L2 regularization method, among the various methods, the performance of the deep network is much better than that of the shallow network. The method of using L2 regularization to utilize the knowledge of the pre-trained model during the training process damages the performance of the network, and its accuracy declines in the later stage of training. The fine-tuning with procedural learning achieves the best results. To analyze these phenomena, we compare the differences between two VFs algorithms.

We select the 22nd convolution kernel of the third convolutional layer in the pre-trained model. Two algorithms are used to obtain the corresponding visual features. Then, we propagate the images forward in the network and sum up the activation values of each convolution kernel in this layer. The level of this value reflects the response of the convolution kernel to the features. The experimental results are shown as Fig.5.

As shown in Fig.5, both visual feature images have large outputs for the 22nd convolution kernel, but neither of them is the largest value for this layer. The number of convolution kernels with larger activation values than the 22nd kernel with the visual feature image using the RVFs is less than when using the IVFs. Therefore, the feature of each convolution kernel can be better represented by the RVFs algorithm. When the representations of each convolution kernel are relatively

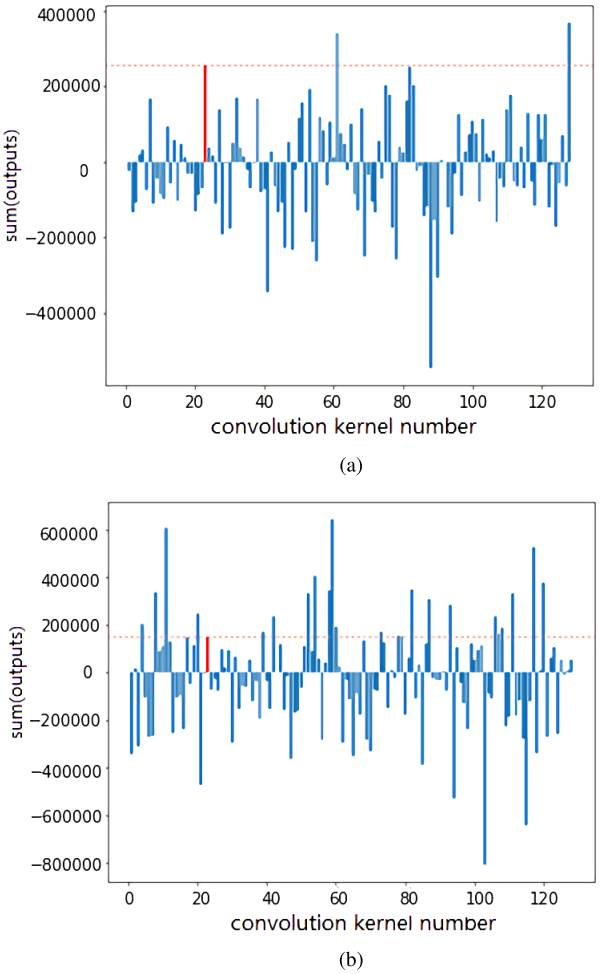


FIGURE 5. The activation value of the maximum activation image of the other convolution kernel on the same layer. (a) The summed output value of each channel of the image using the RVFs algorithm. (b) The summed output value of each channel of the image using the IVFs algorithm.

independent and robust, the feature distribution obtained from the convolutional layer will be more abundant. Then, we select some convolution kernels in this convolution layer and generate images using the IVFs and RVFs algorithms. The images obtained by the two algorithms are compared in Fig.6.

The images obtained by two generation algorithms are very similar from the perspective of feature representation. By the comparisons in Fig.5, we can see that the RVFs retain the main contents of the features and suppress the redundant responses of the other convolution kernels, which improves the richness of the feature representations. On the other hand, the visual feature images obtained by filter3 in layer2 are

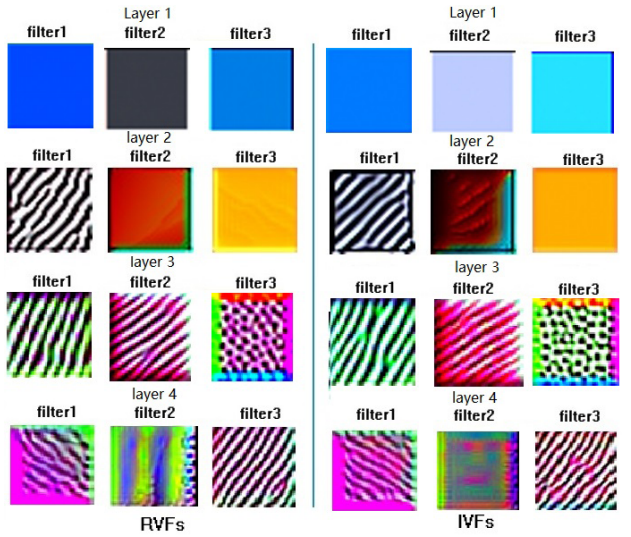


FIGURE 6. Comparison of the maximum activation images of the RVFs and IVFs algorithms.

basically the same for the two algorithms, which means that the learned features from this convolution kernel are highly robust. The visual differences between the RVFs and IVFs are small, but this difference has a great influence on the final results from Table.1 and Table.2. Procedural learning achieves better results using RVFs. It also shows that the differences between the RVFs and IVFs become smaller as the number of layers increases. Because it is close to the classifier, the convolution kernels have richer semantic information of the visual features.

Through the analysis of these two algorithms, the RVFs allow the convolution layer to have richer feature representation. In the Cifar-10 experiment, the training from scratch with the IVFs algorithm has poor performance, and the curve of the loss decrease has large variations . Compared to the loss curve of procedural learning, the latter is very stable and has achieved a better result. In the fine-tuning series, a good initialization value guarantees the learning foundation. These two algorithms do not have large jitters in their loss curves. However, the RVFs algorithm guarantees robust feature representations and gradually takes the lead during training. In the Cifar-100 experiment, the results of the two algorithms do not have very large differences. Even the best result achieved using this dataset for all the models is still not good. This shows that the learned features of the model are not rich enough. The generalization ability of this model is low.

VI. CONCLUSIONS AND FUTURE WORKS

The procedural learning method can make use of the good initial values of the pre-trained model, but it also acquires useful RVFs via the low rank priors during the training process. It can help the CNN to learn using small datasets. Procedural learning uses the rich feature distribution to improve the generalization ability of the pre-trained model on new tasks. This method of using prior knowledge is different from existing studies. In the experiment, although the procedural learning algorithm has better final results than the other methods, the improvement is not very large. In procedural learning, we use the empirical distribution method to estimate the distribution of the maximum activation images. By using this method, it is difficult to precisely estimate the true distribution of the features. As a result, the visual features provided during the training process are relatively limited. Second, the RVFs algorithm only considers suppressing other convolution kernels, but it does not consider how to achieve a better representation of a certain convolution kernel. This leaves room for future improvements in procedural learning.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[2] Z. Shao, J. Cai, and Z. Wang, "Smart monitoring cameras driven intelligent processing to big surveillance video data," *IEEE Trans. Big Data*, vol. 4, no. 1, pp. 105–116, Mar. 2018.

[3] W. Zhou, S. Newsam, C. Li, and Z. Shao, "PatternNet: A benchmark dataset for performance evaluation of remote sensing image retrieval," *ISPRS J. Photogramm. Remote Sens.*, vol. 145, pp. 197–209, Nov. 2018.

[4] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. (2014). "How transferable are features in deep neural networks?" [Online]. Available: https://arxiv.org/abs/1411.1792

[5] W. Zhou, S. Newsam, C. Li, and Z. Shao, "Learning low dimensional convolutional neural networks for high-resolution remote sensing image retrieval," *Remote Sens.*, vol. 9, no. 5, pp. 1–20, May 2017.

[6] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. (2014). "Deep domain confusion: Maximizing for domain invariance." [Online]. Available: https://arxiv.org/abs/1412.3474

[7] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. 32nd Int. Conf. Mach. Learn.*, pp. 97–105, Jul. 2015.

[8] D. Erhan, Y. Bengio, A. Courville, and P. Vincent, "Visualizing higher-layer features of a deep network," Tech. Rep., 2009.

[9] J. Wright, A. Ganesh, S. Rao, Y. Peng, and Y. Ma, "Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 2080–2088.

[10] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *J. ACM*, vol. 58, no. 3, May 2011, Art. no. 11.

[11] H. Xu, C. Caramanis, and S. Sanghavi, "Robust PCA via outlier pursuit," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 2496–2504.

[12] X. Chen, Y. Di, J. Duan, and D. Li, "Linearized compact ADI schemes for nonlinear time-fractional SchrÃűdinger equations," *Appl. Math. Lett.*, vol. 84, pp. 160–167, Oct. 2018.

[13] L. Rüschendorf, "The Wasserstein distance and approximation theorems," *Probab. Theory Rel. Fields*, vol. 70, no. 1, pp. 117–129, Mar. 1985.

[14] R. Shwartz-Ziv and N. Tishby. (2017). "Opening the black box of deep neural networks via information." [Online]. Available: https://arxiv.org/abs/1703.00810

[15] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

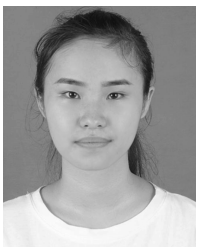
[17] A. Nguyen, J. Clune, Y. Bengio, A. Dosovitskiy, and J. Yosinski, "Plug & play generative networks: Conditional iterative generation of images in latent space," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3510–3520.

[18] A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune. (2016). "Synthesizing the preferred inputs for neurons in neural networks via deep generator networks." [Online]. Available: https://arxiv.org/abs/1605.09304

[19] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 5188–5196.
[20] D. Li, J. Wang, and J. Zhang, "Unconditionally convergent $L1$-Galerkin fems for nonlinear time-fractional Schrödinger equations," *SIAM J. Sci. Comput.*, vol. 39, no. 6, pp. A3067–A3088, Dec. 2017.
[21] C. Olah, A. Mordvintsev, and L. Schubert. (2017). Feature visualization. Distill. [Online]. Available: https://distill.pub/2017/feature-visualization
[22] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.
[23] Z. Lin, A. Ganesh, J. Wright, L. Wu, M. Chen, and Y. Ma, "Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix," Coordinated Sci. Lab., Urbana, IL, USA, Tech. Rep. UILU-ENG-09-2214, 2009.
[24] J. Wen, Z. Zhou, Z. Liu, M.-J. Lai, and X. Tang, "Sharp sufficient conditions for stable recovery of block sparse signals by block orthogonal matching pursuit," *Appl. Comput. Harmon. Anal.*, to be published.
[25] J. Wen, H. Chen, and Z. Zhou, "An optimal condition for the block orthogonal matching pursuit algorithm," *IEEE Access*, vol. 6, pp. 38179–38185, 2018.
[26] D. J. Im, M. Tao, and K. Branson. (2016). "An empirical analysis of deep network loss surfaces." [Online]. Available: https://arxiv.org/abs/1612.04010
[27] M. Arjovsky, S. Chintala, and L. Bottou. (2017). "Wasserstein GAN." [Online]. Available: https://arxiv.org/abs/1701.07875
[28] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5769–5779.
[29] J. Wen, Z. Zhou, J. Wang, X. Tang, and Q. Mo, "A sharp condition for exact support recovery with orthogonal matching pursuit," *IEEE Trans. Signal Process.*, vol. 65, no. 6, pp. 1370–1382, Mar. 2017.
[30] S. T. Rachev and R. M. Shortt, *Duality Theorems for Kantorovich-Rubinstein and Wasserstein Functionals*. 1990.
[31] C. Frogner, C. Zhang, H. Mobahi, M. Araya, and T. A. Poggio, "Learning with a Wasserstein loss," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2053–2061.
[32] J. Wen, J. Wang, and Q. Zhang, "Nearly optimal bounds for orthogonal least squares," *IEEE Trans. Signal Process.*, vol. 65, no. 20, pp. 5347–5356, Oct. 2017.
[33] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2292–2300.
[34] P. A. Knight and D. Ruiz, "A fast algorithm for matrix balancing," *IMA J. Numer. Anal.*, vol. 33, no. 3, pp. 1029–1047, Jul. 2013.
[35] P. A. Knight, "The Sinkhorn–Knopp algorithm: Convergence and applications," *SIAM J. Matrix Anal. Appl.*, vol. 30, no. 1, pp. 261–275, Mar. 2008.
[36] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556
[37] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

HAIFENG LI (M'15) received the master's degree in transportation engineering from the South China University of Technology, Guangzhou, China, in 2005, and the Ph.D. degree in photogrammetry and remote sensing from Wuhan University, Wuhan, China, in 2009. He was a Research Associate with the Department of Land Surveying and Geo-Informatics, The Hong Kong Polytechnic University, Hong Kong, in 2011, and a Visiting Scholar with the University of Illinois at Urbana–Champaign, Urbana, IL, USA, from 2013 to 2014. He is currently a Professor with the School of Geosciences and Info-Physics, Central South University, Changsha, China. He has authored over 30 journal papers. His current research interests include geo/remote sensing big data, machine/deep learning, and artificial/brain-inspired intelligence. He is also a Reviewer for many journals.

LI CHEN received the B.Sc. and M.Sc. degrees from the School of Software, Central South University, China, in 2015 and 2018, respectively. He is currently pursuing the Ph.D. degree with the School of Geosciences and Info-Physics, Central South University. His research interests include transfer learning, deep learning, and visualization.



HAILUN DING is currently pursuing the degree with the School of Software Engineering, Central South University. Her research interests include deep learning and network interpretability for deep learning.

QI LI is currently pursuing the degree with the School of Information Science and Engineering, Central South University. His research interests include deep learning and visualization.

BINGYU SUN received the Ph.D. degree from the University of Science and Technology of China, in 2004. In 2005, he was a Research Assistant with The Chinese University of Hong Kong. He is currently a Research Fellow with the Hefei Institute of Intelligence Machines, Chinese Academy of Sciences. He engages mainly in machine learning and intelligent decision. He is in charge of the projects, including the Tianjin agriculture Internet of Things, the Integration and Demonstration of Agricultural E-commerce, and so on. He has published more than 20 papers in several international journals, including the IEEE TRANSACTIONS series. His research interests include data mining, knowledge discovery, big data, and agriculture Internet of Things.

GUOHUA WU received the B.S. degree in information systems and the Ph.D. degree in operations research from the National University of Defense Technology, China, in 2008 and 2014, respectively. From 2012 to 2014, he was a Visiting Researcher with the University of Alberta, Edmonton, AB, Canada. He is currently a Professor with the School of Traffic and Transportation Engineering, Central South University, Changsha, China. His research interests include intelligent planning and scheduling, evolutionary computation, and data mining. He has authored over 50 referred papers, including those published in the IEEE TRANSACTIONS ON SYSTEM MAN CYBERNETICS: SYSTEMS, *Information Sciences*, and *Computers & Operations Research*. He serves as an Editorial Board Member of the *International Journal of Bio-Inspired Computation*, as a Guest Editor for *Information Sciences* and *Memetic Computing*, and as an Associate Editor for *Swarm and Evolutionary Computation*.