Learning Deep Sparse Regularizers With Applications to Multi-View Clustering and Semi-Supervised Classification

Shiping Wang, Zhaoliang Chen, Shide Du, and Zhouchen Lin, *Fellow, IEEE*

Abstract—Sparsity-constrained optimization problems are common in machine learning, such as sparse coding, low-rank minimization and compressive sensing. However, most of previous studies focused on constructing various hand-crafted sparse regularizers, while little work was devoted to learning adaptive sparse regularizers from given input data for specific tasks. In this paper, we propose a deep sparse regularizer learning model that learns data-driven sparse regularizers adaptively. Via the proximal gradient algorithm, we find that the sparse regularizer learning is equivalent to learning a parameterized activation function. This encourages us to learn sparse regularizers in the deep learning framework. Therefore, we build a neural network composed of multiple blocks, each being differentiable and reusable. All blocks contain learnable piecewise linear activation functions which correspond to the sparse regularizer to be learned. Furthermore, the proposed model is trained with back propagation, and all parameters in this model are learned end-to-end. We apply our framework to multi-view clustering and semi-supervised classification tasks to learn a latent compact representation. Experimental results demonstrate the superiority of the proposed framework over state-of-the-art multi-view learning models.

Index Terms—Deep learning, sparse regularizer, parameterized activation function, proximal operator, multi-view learning

1 Introduction

A considerable amount of research has indicated the importance of sparse representation in boosting the performance of various machine learning tasks [1], [2], [3]. For example, low-rank minimization generally enforces sparse constraints on singular values. Sparse coding models deal with intractable nonconvex ℓ_0-norm minimization problems by replacing ℓ_0 with its surrogate functions, such as ℓ_1-norm, which leads to more tractable computations [4], [5]. However, these previous studies put more emphases on predefined sparse norms, resulting in hand-crafted rather than data-driven sparse regularizers. Traditionally, most of these methods rely on an iterative algorithm that minimizes an objective function. The inherently sequential structure and data-dependent time complexity result in a major limitation on the efficiency of the algorithms. Meanwhile, such optimization problems are generally non-differentiable and thus suffer from difficulties in computing gradients, which suggests limitations in applying existing sparse regularizers to deep learning architectures for performance boosting and computational acceleration.

Several attempts have shown to be encouraging for improving learning performance when embedding some specific sparse norms into deep neural networks. For example, based on an iterative shrinkage and thresholding algorithm (ISTA) for the ℓ_1-norm regularizer [6], learned ISTA (LISTA) [7] was proposed to train sparse codes with neural networks, where each block was differentiable and reusable. ISTA was further transformed into a structured deep neural network dubbed ISTA-Net [8], which optimized an ℓ_1-norm based compressive sensing reconstruction model. An ℓ_0 regularized encoder [9] was also explored for constructing an effective sparse regularization with time-unfolding feed-forward neural networks. Sprechmann *et al.* demonstrated a principled way to construct learnable pursuit process architectures for structured sparse models, which was derived from the iteration of proximal gradient descent algorithms [10]. Tanaka *et al.* proposed sparse recurrent neural networks to conduct efficient energy information processing [11]. Luo *et al.* mapped a temporally-coherent sparse coding to a special type of stacked recurrent neural networks (sRNN) to learn all parameters simultaneously [12]. These latest research results inspire us to apply back propagation and gradient descents in deep learning frameworks to traditional iterative algorithms.

However, from the perspective of model training, deep neural networks (DNNs) are usually limited to conducting back propagation and gradient descent with differentiable regularizers. Therefore, how to build a network that can deal with non-differentiable objective functions using differentiable blocks in the deep learning framework is a pivotal problem.

• Shiping Wang, Zhaoliang Chen, and Shide Du are with the College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China, and also with the Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, Fuzhou 350108, China. E-mail: shipingwangphd@163.com, chenzl23@outlook.com, dushidems@gmail.com.
• Zhouchen Lin is with the Key Laboratory of Machine Perception (MoE), School of EECS, Peking University, Beijing 100871, China, and also with the Pazhou Lab, Guangzhou 510330, China. E-mail: zlin@pku.edu.cn.

Manuscript received 29 July 2020; revised 1 Apr. 2021; accepted 11 May 2021. Date of publication 21 May 2021; date of current version 4 Aug. 2022. (Corresponding author: Zhouchen Lin.) Recommended for acceptance by W. Liu. Digital Object Identifier no. 10.1109/TPAMI.2021.3082632

Differentiable programming solves this problem by reformulating traditional machine learning methods, which transforms the optimization process into differentiable network structures. In this way, the model can be trained with back propagation, and some key hyperparameters become learnable. Substantial studies have concentrated on this technique [13], [14], [15]. For instance, ADMM-Net was derived from the iterative procedures of an alternating direction method of multipliers (ADMM) algorithm for optimizing an MRI model based on compressive sensing [16]. Xie *et al.* proposed a differentiable linearized ADMM (DLADMM) for solving convex problems with linear constraints [17]. Bertinetto *et al.* taught a deep network to use standard machine learning tools like ridge regression to quickly learn parameters [18]. Because proximal operators are commonly utilized in optimization methods, existing studies have also proved the corresponding relationships between proximal operators and activation functions employed in neural networks, so that neural networks can handle some specific optimization problems [19], [20], [21]. Nevertheless, how to learn valid sparse regularizers via activation functions remains unexplored. To our knowledge, very limited research has been devoted to the general learning framework of sparse regularizers.

In this paper, we propose an efficient deep network framework dubbed deep sparse regularizer learning (DSRL), to adaptively learn data-driven sparse regularizers. Bridged by the proximal operator, we exploit the correspondence between regularizers and parameterized activation functions. Accordingly, we may learn piecewise linear activation functions, which is an indirect way to learn sparse regularizers. Because all iterative blocks in DSRL are differentiable, the proposed model can be trained with back propagation. Further, we apply DSRL to the multi-view learning task, where a fused multi-view latent representation is reconstructed using the proposed framework. The data-driven sparse regularizers learned by DSRL are compared with some predefined surrogates of ℓ_0-norm to validate the effectiveness of our method. Besides, we also compare the performance with hand-crafted sparse surrogates, and experimental results indicate that DSRL outperforms other sparse regularizers. The main contributions of this paper can be summarized in the following four aspects:

1) Convert the problem of learning a sparse regularizer into that of learning an activation function by exploiting the correspondence between regularizers and activation functions.
2) Provide the conditions that a learnable activation function should satisfy to yield a valid regularizer. We further propose two-stage projections such that the conditions can be satisfied when learning the activation function.
3) Propose an end-to-end deep data-driven regularizer learning scheme. Via the parameterized activation functions, the outputs are guaranteed to be appropriately sparse for the given specific task at the best.
4) We apply the proposed method to multi-view clustering and semi-supervised classification. It achieves superior performance on eight real-world datasets compared with specific regularizers and other state-of-the-art methods.

TABLE 1
Several Specified Definitions of $g(\cdot)$ for Sparse Surrogates

Penalty	Formula of $g(x)$, $x \geq 0$, $\lambda \geq 0$
ℓ_p-norm [22]	$g(x) = \lambda x^p$, $0 < p < 1$
Logarithm [23]	$g(x) = \frac{\lambda}{\log(\gamma+1)} \log(\gamma x + 1)$
Geman [24]	$g(x) = \frac{\lambda x}{x+\gamma}$
Laplace [25]	$g(x) = \lambda(1 - \exp(-\frac{x}{\gamma}))$
ETP [26]	$g(x) = \lambda \frac{1-\exp(-\gamma x)}{1-\exp(-\gamma)}$

2 Related Work

A large amount of research has recognized the critical role played by sparse representation. However, most previous studies concentrated on hand-crafted sparsity. Several commonly used sparse surrogates are shown in Table 1 [27]. These defined sparse regularizers are non-decreasing and nonconvex on $(0, \infty)$, as illustrated in Fig. 1. Some of these regularizers are lower semicontinuous. Specifically, ℓ_p-norm is widely used in multiple kernel learning to promote sparse kernel combinations so that the constructed model is more interpretable and scalable [28]. Laplace function is leveraged to conduct a homotopic approximation of the ℓ_0 minimization problem in compressive sensing [25]. These sparse surrogates are also applied to rank regularized optimization problems

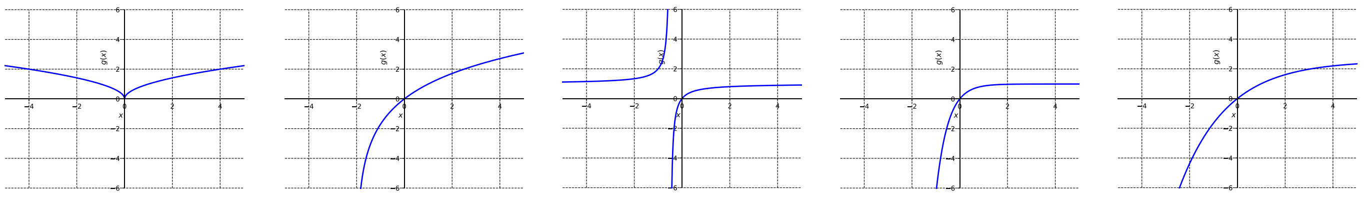
$$\arg\min_{\mathbf{X}} \mathcal{J}(\mathbf{X}) = \mathrm{rank}(\mathbf{X}) + f(\mathbf{X}), \tag{1}$$

where $f(\cdot)$ is generally a differentiable loss function. Because solving the problem with a rank constraint is difficult and even NP-hard, this problem is then transformed into

$$\arg\min_{\mathbf{X}} \mathcal{J}(\mathbf{X}) = \sum_{i=1}^{n} g(\sigma_i(\mathbf{X})) + f(\mathbf{X}), \tag{2}$$

where $\sigma_i(\mathbf{X})$ is the ith singular value of $\mathbf{X} \in \mathbb{R}^{n\times m}$ and $g(\cdot)$ is a surrogate of ℓ_0-norm as listed in Table 1 [27]. On the basis of predefined surrogate functions, Lu *et al.* proposed an iteratively reweighted nuclear norm (IRNN) algorithm to solve nonconvex nonsmooth rank optimizations [29]. Zhang *et al.* further handled nonconvex nonsmooth rank minimization problems with closed-form solutions of ℓ_p-norm when $p = \frac{1}{2}$ and $\frac{2}{3}$ [30]. Dan *et al.* studied low-rank recovery models with the ℓ_p-norm loss and provided a better approximation guarantee [31]. In general, these hand-crafted sparse surrogates tend to approximate specific sparsity and are often sensitive to predefined hyperparameters, which may lead to suboptimal performance. Moreover, due to the particular properties of various surrogates, a specific surrogate function may not be applicable to a wide range of application scenarios, which poses the difficulty in selecting a suitable surrogate.

Many DNNs require sparse weights or outputs, and a number of recent studies [32], [33] also suggested that large-scale DNNs usually contained lots of redundant parameters, which resulted in a waste of computational resources and a high risk of overfitting. There have been several attempts to encourage the sparsity of weights or outputs in DNNs. For instance, sparse autoencoders [34] only allowed a small number of hidden units to be active at once with Kullback-Leibler



(a) l_p-norm (b) Logarithm (c) Geman (d) Laplace (e) ETP

Fig. 1. Illustration of some popular hand-crafted sparse regularizers (For ℓ_p-norm, $p = 0.5$. For all penalties, $\lambda = 1.0$, $\gamma = 0.5$). All these sparse regularizers share some common properties: nonconvex and non-decreasing on $(0, \infty)$.

divergence. Tartaglione *et al.* exploited a simple thresholding approach to promote the sparse property of network parameters [35]. Liu *et al.* pruned redundant connections to generate sparse layers [36]. Bhowmik *et al.* addressed the problem of sparse spike deconvolution from noisy measurements within a Bayesian paradigm, where the sparsity was measured by ℓ_1-norm [37]. Wang *et al.* presented a deep structured model to learn a non-linear function, where the regularization term for the proximal operator was fixed as ℓ_1-norm [38]. Mahapatra *et al.* [39] solved the sparse signal reconstruction problem using a feed-forward deep neural network, which was regularized with ℓ_1-norm sparsity and generalized the ISTA framework. Srinivas *et al.* proposed a new method to control the number of activated neurons, which led to a highly sparse neural network model [40]. Ma *et al.* used an integrated transformed nonconvex ℓ_1 regularizer to promote the sparsity of parameters [41]. Generally, most of these existing works on DNNs promoted sparsity with hand-crafted sparsity penalties or defined thresholding functions. Some of them were based on an unfolded ISTA framework or only handled ℓ_1-norm sparsity. Besides, to the best of our knowledge, very limited research has been done for learning deep sparse regularizers adaptively. In this paper, we address data-driven sparse regularizer learning problems from the viewpoint of activation functions, which is beyond the ISTA learning framework and not limited to certain specific sparse regularizers.

3 Proposed Method

To learn sparse regularizers adaptively in a data-driven manner, we first construct a connection between sparse regularizers and activation functions via proximal operators. By the connection, learning a sparse regularizer is equivalently transformed into learning a parameterized activation function in a deep neural network. Accordingly, a blockwise neural network is designed to learn a data-driven sparse regularizer. Fig. 2 illustrates the structure of the proposed framework.

3.1 Correspondence Between Sparse Regularizers and Activation Functions

Proximal operators are widely used in various machine learning optimization problems. We start with a univariate proximal operator, i.e.,

$$\mathbf{Prox}_g(y) = \arg\min_x \mathcal{J}(x) = \frac{1}{2}(x-y)^2 + g(x), \tag{3}$$

where $g(\cdot)$ can be a sparse regularizer. It was proved in [27] that $\xi(y) \equiv \mathbf{Prox}_g(y)$ is a non-decreasing function of y. Thus it can serve as an activation function of a neural network. On the other hand, given a non-decreasing function $\xi(x)$, we can define

$$\begin{aligned} g(x) &= \int_0^x (\xi^{-1}(y) - y)dy \\ &= \int_0^x \xi^{-1}(y)dy - \frac{1}{2}x^2, \end{aligned} \tag{4}$$

where $\xi(y): \mathbb{R} \to \mathbb{R}$ is a univariate function and $\xi^{-1}(y)$ is the inverse function of $\xi(y)$. Note that if $\xi^{-1}(\cdot)$ is not single-valued, $g(x)$ is still well defined. It was proved in [21] that the proximal operator of such a $g(x)$, defined in (3), is exactly $\xi(x)$, because the optimality condition of (3) is $0 \in (\xi^{-1}(x) - x) + (x - y)$. Thus we have shown the correspondence between $\xi(x)$ and $g(x)$ via the proximal operator. If $g(x)$ is a sparse regularizer, then $\xi(x)$ has to map a neighborhood of 0 to 0 (see Fig. 2 of [27]). Namely, $0 \in \xi^{-1}(0)$. Therefore, $g(x) = 0$, and is nonnegative on $(0, \infty)$. These conditions will be used for learning valid sparse regularizers.

As an example, by considering a commonly used ℓ_1 regularizer $b|x|$, we can check that $\arg\min_x \frac{1}{2}(x-y)^2 + b|x| = \xi_{\boldsymbol{\theta}}(y)$, in which

$$\xi_{\boldsymbol{\theta}}(x) = \begin{cases} x - b, & b \le x, \\ 0, & -b \le x < b, \\ x + b, & x < -b, \end{cases} \tag{5}$$

where $\boldsymbol{\theta} = \{b\}$ can be a learnable parameter set with $b \ge 0$.

With the above analysis, learning a sparse regularizer $g(x)$ is transformed into learning an activation function $\xi(x)$ that is non-decreasing and maps a neighborhood of 0 to 0. Because it is difficult for the activation function of only one parameter to learn a suitable sparse regularizer with the given data, we employ piecewise linear functions to approximate the learnable activation function, consisting of more learnable parameters. Particularly, an activation function of two sets of learnable parameters $(\boldsymbol{\theta_1}, \boldsymbol{\theta_2})$ is defined as

$$\xi_{(\boldsymbol{\theta_1},\boldsymbol{\theta_2})}(x) = \begin{cases} w_2(x - b_2) + w_1(b_2 - b_1), & b_2 \le x, \\ w_1(x - b_1), & b_1 \le x < b_2, \\ 0, & -b_1 \le x < b_1, \\ w_1(x + b_1), & -b_2 \le x < -b_1, \\ w_2(x + b_2) + w_1(b_1 - b_2), & x < -b_2, \end{cases} \tag{6}$$

where $x \in \mathbb{R}$, $0 \le b_1 \le b_2$ and $w_1, w_2 > 0$ are learnable parameters with $\boldsymbol{\theta_1} = (w_1, b_1)$ and $\boldsymbol{\theta_2} = (w_2, b_2)$. Noting that the form of the activation function is not limited to two parameter sets, we consider (6) as an example, which is a trade-off between computational complexity and learning accuracy to achieve desired performance in practical

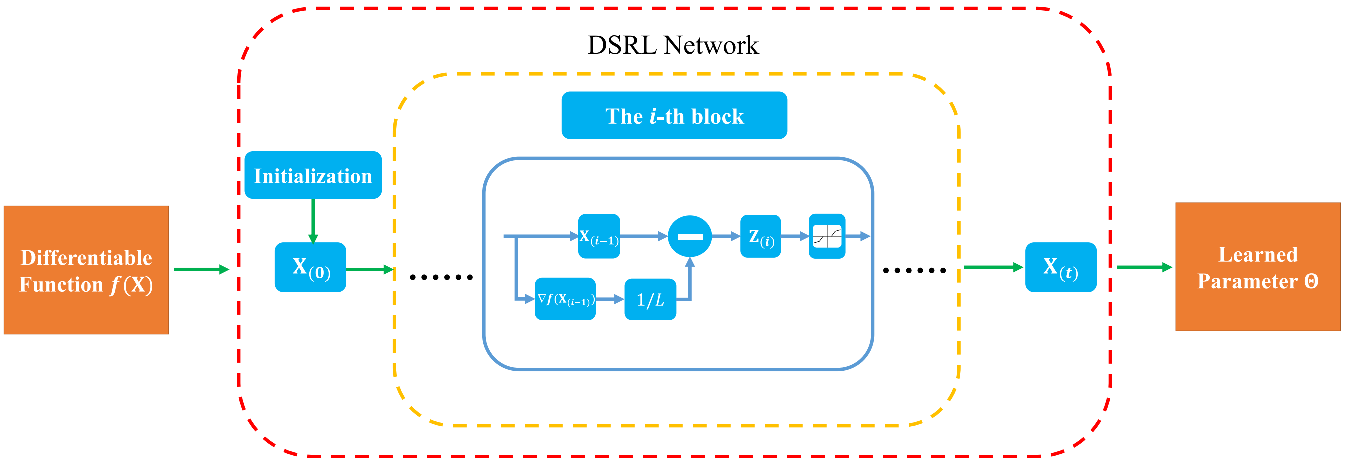


Fig. 2. Framework of the proposed DSRL method. It consists of multiple unfolded blocks in which a basic block is made up of several differentiable units, as demonstrated in the blue shapes.

applications. With this definition, the inverse function $\xi^{-1}_{(\boldsymbol{\theta}_1,\boldsymbol{\theta}_2)}(y)$ is computed by

$$\xi^{-1}_{(\boldsymbol{\theta}_1,\boldsymbol{\theta}_2)}(y) = \begin{cases} \frac{y-w_1(b_2-b_1)}{w_2} + b_2, & w_1(b_2-b_1) \le y, \\ \frac{y}{w_1} + b_1, & 0 \le y < w_1(b_2-b_1), \\ [-b, b], & y = 0, \\ \frac{y}{w_1} - b_1, & -w_1(b_2-b_1) \le y < 0, \\ \frac{y-w_1(b_2-b_1)}{w_2} - b_2 & y < -w_1(b_2-b_1). \end{cases} \tag{7}$$

Therefore, the sparse regularizer learned by a parameterized activation function is derived on the basis of (4):

$$g(x) = \begin{cases} (\frac{1}{2w_2} - \frac{1}{2})x^2 + (b_2 - \frac{w_1(b_2-b_1)}{w_2})x & \\ + \frac{w_1(w_1-w_2)}{2w_2}(b_2-b_1)^2, & x \ge w_1(b_2-b_1), \\ (\frac{1}{2w_1} - \frac{1}{2})x^2 + b_1 x, & 0 \le x < w_1(b_2-b_1), \\ g(-x), & x < 0. \end{cases} \tag{8}$$

It is observed from the formula that the learned sparse regularizer $g(x)$ is symmetric about the y-axis. When $x = 0$, $g(x)$ is exactly equal to 0.

For the sake of theoretic strictness and better interpretability, it is required that $w_1, w_2 > 0$, $0 \le b_1 \le b_2$, $g(x) \ge 0$ and $g(x)$ is non-decreasing when $x \ge w_1(b_2-b_1)$. Then the conditions become [refer to Section A of Appendix, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2021.3082632]:

$$\begin{aligned} & w_1 > 0, 1 \ge w_2 > 0, \\ & b_2 \ge b_1 \ge \max\left\{0, \frac{w_1 - 1}{w_1} b_2\right\}. \end{aligned} \tag{9}$$

Directly projecting parameter set $\mathbf{\Theta} = (w_1, w_2, b_1, b_2)$ onto (9) is also difficult. We may first project (w_1, w_2) and then project (b_1, b_2) after fixing (w_1, w_2). The projection of (w_1, w_2) is formulated as $w_1 = \max\{w_1, \epsilon\}$ and $w_2 = \min\{\max\{w_1, \epsilon\}, 1\}$, where ϵ is a small positive value. After fixing (w_1, w_2), we project (b_1, b_2) onto $\mathcal{S}_b = \{(b_1, b_2) | b_2 \ge b_1 \ge \max\{0, \frac{w_1-1}{w_1} b_2\}\}$. To be exact, when $0 < w_1 \le 1$, the projection $\mathbf{Proj}(b_1, b_2)$ of (b_1, b_2) onto $\mathcal{S}_b$ is

$$\mathbf{Proj}(b_1, b_2) = \begin{cases} (b_1, b_2), & b_1 \ge 0, b_2 \ge 0, b_1 \le b_2, \\ (0, b_2), & b_1 < 0, b_2 > 0, \\ (0, 0), & b_2 \le \min\{0, -b_1\}, \\ (\frac{b_1+b_2}{2}, \frac{b_1+b_2}{2}), & b_1 \ge |b_2|. \end{cases} \tag{10}$$

When $w_1 > 1$, the projection of (b_1, b_2) onto $\mathcal{S}_b$ becomes

$$\mathbf{Proj}(b_1, b_2) = \begin{cases} (b_1, b_2), & b_2 \ge 0, \frac{w_1-1}{w_1} b_2 \le b_1 \le b_2, \\ (\rho_1 b_1 + \rho_2 b_2, \rho_2 b_1 + \rho_3 b_2), & \frac{w_1}{1-w_1} b_2 < b_1 < \frac{w_1-1}{w_1} b_2, \\ (0, 0), & b_2 \ge 0, b_1 \le \frac{w_1}{1-w_1} b_2, \\ (0, 0), & b_2 \le \min\{0, -b_1\}, \\ (\frac{b_1+b_2}{2}, \frac{b_1+b_2}{2}), & b_1 \ge |b_2|, \end{cases} \tag{11}$$

where the parameter set $\{\rho_1, \rho_2, \rho_3\}$ is given as $\rho_1 = \frac{(w_1-1)^2}{w_1^2+(w_1-1)^2}$, $\rho_2 = \frac{w_1(w_1-1)}{w_1^2+(w_1-1)^2}$ and $\rho_3 = \frac{w_1^2}{w_1^2+(w_1-1)^2}$.

3.2 Implicitly Learnable Deep Sparse Regularizer

Generic optimization problems with learnable sparse regularizers $g(\cdot)$ can be written as

$$\min_{\mathbf{X}} \mathcal{J}(\mathbf{X}) = f(\mathbf{X}) + g(\mathbf{X}), \tag{12}$$

where $g(\mathbf{X}_{ij}) = \int_0^{\mathbf{X}_{ij}} (\xi^{-1}_{\mathbf{\Theta}}(y) - y)dy$ for any $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$ with $\mathbf{\Theta}$ to be learned by the theory in Section 3.1. The function $f(\mathbf{X})$ is differentiable, and its gradient is Lipschitz continuous. The iteration rule of the proximal gradient method for solving Problem (12) is as follows:

$$\begin{aligned} \mathbf{X}^{(k+1)} &= \arg\min_{\mathbf{X}} f(\mathbf{X}^{(k)}) + \langle \nabla f(\mathbf{X}^{(k)}), \mathbf{X} - \mathbf{X}^{(k)} \rangle \\ &\quad + \frac{L}{2} ||\mathbf{X} - \mathbf{X}^{(k)}||_F^2 + g(\mathbf{X}) \\ &= \arg\min_{\mathbf{X}} \frac{L}{2} \left|\left| \mathbf{X} - \mathbf{X}^{(k)} + \frac{1}{L} \nabla f(\mathbf{X}^{(k)}) \right|\right|_F^2 + g(\mathbf{X}), \end{aligned} \tag{13}$$

where L is the Lipschitz constant of $\nabla f(\cdot)$, i.e.,

$$||\nabla f(\mathbf{X}) - \nabla f(\mathbf{Y})||_F \le L||\mathbf{X} - \mathbf{Y}||_F, \tag{14}$$

for any $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{n \times m}$. Denoting $\mathbf{W} = \mathbf{X}^{(k)} - \frac{1}{L}\nabla f(\mathbf{X}^{(k)})$, the optimization problem above is exactly

$$\mathbf{Prox}_{\frac{g}{L}}(\mathbf{W}) = \arg\min_{\mathbf{X}} \frac{1}{2}||\mathbf{X} - \mathbf{W}||_F^2 + \frac{1}{L}g(\mathbf{X}), \tag{15}$$

where $\mathbf{Prox}_{\frac{g}{L}}(\cdot)$ is the proximal operator that is related to the regularizer $\frac{1}{L}g(\cdot)$. With these notations, the updating rule of $\mathbf{X}^{(k+1)}$ can be

$$\mathbf{X}^{(k+1)} = \mathbf{Prox}_{\frac{g}{L}}\left(\mathbf{X}^{(k)} - \frac{1}{L}\nabla f(\mathbf{X}^{(k)})\right). \tag{16}$$

Based on the above analysis, we propose an end-to-end deep learning framework that learns data-driven sparse regularizers. Each unit of the proposed framework is structured as a single differentiable block, as demonstrated in Fig. 2. Each block accepts the output from the previous block as an input, and feeds the calculated value to the next block. Consequently, the proposed method can be implemented with a block-wise neural network architecture. Specifically, the ith block computes the output with

$$\mathbf{Z}_{(i)} = \mathbf{X}_{(i-1)} - \frac{1}{L}\nabla f(\mathbf{X}_{(i-1)}), \tag{17}$$

$$\mathbf{X}_{(i)} = \xi_{\mathbf{\Theta}}(\mathbf{Z}_{(i)}), \tag{18}$$

where $\xi_{\mathbf{\Theta}}(\cdot)$ is the activation function parameterized by the set $\mathbf{\Theta}$. DSRL is comprised of t differentiable blocks of the learnable parameters $\mathbf{\Theta}$ and L, which can be learned end-to-end by back propagation [refer to Section B of Appendix, available in the online supplemental material]. Algorithm 1 illustrates the proposed DSRL in detail, which can theoretically approximate any sparse regularizer $g(x)$ in (12). Compared with hand-crafted regularizers that are challenging and time-consuming for parameter selection, the proposed DSRL is more flexible to varying data because of its adaptive optimization of learnable parameters. It learns potential sparse regularizers in a data-driven way, which may steadily improve the performance of given tasks.

Algorithm 1. Deep Sparse Regularizer Learning (DSRL)

```
Input: A differentiable function $f(\mathbf{X})$ and the number of blocks $t$.
Output: Learned parameter set $\mathbf{\Theta}$.
 1: Initialize the data matrix $\tilde{\mathbf{X}} \in \mathbb{R}^{n\times m}$;
 2: Initialize learnable parameter set $\mathbf{\Theta}^{(0)}$ and $L^{(0)}$;
 3: Initialize counter $k = 0$;
 4: while not convergent do
 5:     The parameter $\mathbf{\Theta}^{(k)}$ is projected onto the convex set $\mathcal{S}$,
        denoted by $\mathbf{\Theta}^{(k)}$;
 6:     Update $\mathbf{Z}_{(0)} = \tilde{\mathbf{X}} - \frac{1}{L^{(k)}}\nabla f(\tilde{\mathbf{X}})$;
 7:     Update $\mathbf{X}_{(0)} = \xi_{\mathbf{\Theta}^{(k)}}(\mathbf{Z}_{(0)})$;
 8:     for $i = 1, \dots, t$ do
 9:         Update $\mathbf{Z}_{(i)} = \mathbf{X}_{(i-1)} - \frac{1}{L^{(k)}}\nabla f(\mathbf{X}_{(i-1)})$;
10:         Update $\mathbf{X}_{(i)} = \xi_{\mathbf{\Theta}^{(k)}}(\mathbf{Z}_{(i)})$;
11:     end for
12:     Update $\mathbf{\Theta}^{(k+1)}$ and $L^{(k+1)}$ with back propagation and
        loss function $\mathcal{J}(\tilde{\mathbf{X}}, \mathbf{X}_{(t)}) = \frac{1}{2}\|\tilde{\mathbf{X}} - \mathbf{X}_{(t)}\|_F^2$;
13:     Update counter $k = k + 1$;
14: end while
15: return The learned parameter set $\mathbf{\Theta}$.
```

The computational complexity of the proposed method is linearly related to the block number t for forward propagation. Because a small t value often achieves acceptable performance (as described in Section 4.7), the speed of the proposed DSRL method is relatively fast. After training, the parameters of the activation functions are learned, and we obtain a reconstructed sparse output by one-time forward propagation.

4 Experimental Analysis

In this section, comprehensive experiments on publicly available real-world datasets are conducted to validate the superiority of the learned sparse regularizer by DSRL in terms of multi-view clustering and semi-supervised classification.

Consider multi-view data $\mathcal{X} = \{\mathbf{X}_i\}_{i=1}^v$ with $\mathbf{X}_i \in \mathbb{R}^{n\times d_i}$, where n and v are the sample and view numbers, and d_i is the feature number of the ith view data. Consequently, the multi-view clustering task is to learn a cluster indicator $\mathbf{y} \in \{0,1\}^n$ from the given multi-view data with a certain criterion $\mathrm{loss}(\{\mathbf{X}_i\}_{i=1}^v; \mathbf{y})$. Due to varying dimensions of different view data, we attempt to learn an optimal affinity matrix from the evaluated multi-view similarity matrices $\mathcal{W} = \{\mathbf{W}_i\}_{i=1}^v$ of $\mathcal{X} = \{\mathbf{X}_i\}_{i=1}^v$. In order to verify the superiority of the proposed learnable sparse regularizer method, we formulate the multi-view clustering task in the following simple form:

$$\begin{gathered}\arg\min_{\boldsymbol{\alpha},\mathbf{W}} \frac{1}{2}\left\|\mathbf{W} - \sum_{j=1}^v \boldsymbol{\alpha}_j\mathbf{W}_j\right\|_F^2 + g(\mathbf{W}), \\ \text{subject to } \mathbf{0} \le \boldsymbol{\alpha} \le \mathbf{1}, \boldsymbol{\alpha}^T\mathbf{1} = 1,\end{gathered} \tag{19}$$

where $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1; \dots; \boldsymbol{\alpha}_v] \in \mathbb{R}^v$ is a v-dimensional column vector representing the weights of all views, and $g(\cdot)$ is a sparse regularizer yet to be learned. The fused affinity matrix of the multi-view data is represented as a convex hull of all views, and the representation coefficients are learned adaptively from the optimization objective. Since the view number v tends to be small, a separate algorithm can be developed to compute the optimal value of $\boldsymbol{\alpha}$. Adaptive weights can be optimized by the ADMM algorithm. In particular, suppose that $vec(\cdot)$ is the matrix vectorization operator, then the optimization subproblem with respect to $\boldsymbol{\alpha}$ is written as

$$\begin{gathered}\min_{\boldsymbol{\alpha}} \frac{1}{2}\|[vec(\mathbf{W}_1), \dots, vec(\mathbf{W}_v)]\boldsymbol{\alpha} - vec(\mathbf{W})\|_F^2, \\ \text{subject to } \mathbf{0} \le \boldsymbol{\alpha} \le \mathbf{1}, \boldsymbol{\alpha}^T\mathbf{1} = 1.\end{gathered} \tag{20}$$

While keeping the weighted vector $\boldsymbol{\alpha}$, we compute the optimal solution $\mathbf{W} = \mathbf{Prox}_g(\sum_{j=1}^v \boldsymbol{\alpha}_j\mathbf{W}_j)$. Because $f(\mathbf{W}) = \frac{1}{2}||\mathbf{W} - \sum_{j=1}^v \boldsymbol{\alpha}_j\mathbf{W}_j||_F^2$ is differentiable, we can apply the proposed DSRL framework to learning an optimal data-driven sparse regularizer $g(\mathbf{W})$.

As to the multi-view semi-supervised classification task, with a fixed value of $\boldsymbol{\alpha}$, we formulate the problem as

$$\begin{aligned}\arg\min_{\mathbf{W}} \frac{1}{2}\left\|\mathbf{W} - \sum_{j=1}^v \boldsymbol{\alpha}_j\mathbf{W}_j\right\|_F^2 &+ \frac{\mu}{2}\mathrm{Tr}\big(\mathbf{Y}^T(\mathbf{D} - \mathbf{W})\mathbf{Y}\big) \\ &+ \frac{\nu}{2}\|\mathbf{Y} - \mathbf{L}\|_F^2 + g(\mathbf{W}),\end{aligned} \tag{21}$$

where $\mathbf{D} = [\mathbf{D}_{ij}]_{n\times n}$ is a diagonal matrix with $\mathbf{D}_{ii} = \sum_{j=1}^n \mathbf{W}_{ij}$, $\mu > 0$ and $\nu > 0$ are fixed regularization parameters, and $\mathbf{L} = [\mathbf{L}_{ij}]_{n\times c}$ is the matrix to indicate the limited number

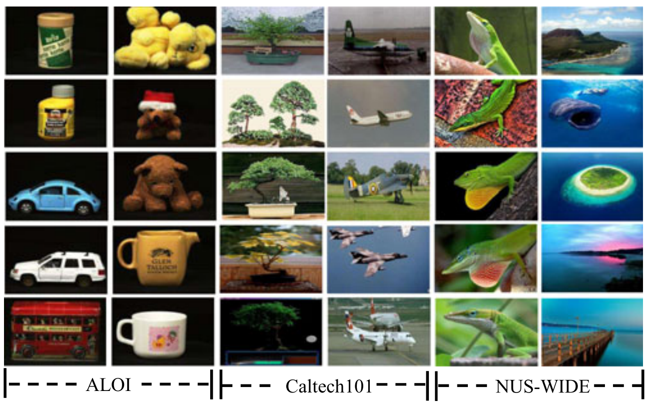


Fig. 3. Several sample images from the test image datasets.

of labeled data points. Specifically, $\mathbf{L}_{ij}=1$ if the ith data point belongs to the jth class, and $\mathbf{L}_{ij}=0$ otherwise. Consequently, $\mathbf{Y}$ is the predictive class assignment indicator matrix that can be computed by

$$\mathbf{Y}=\left(\mathbf{I}+\frac{\mu}{\nu}(\mathbf{D}-\mathbf{W})\right)^{-1}\mathbf{L}. \tag{22}$$

Actually, the optimal $\mathbf{Y}$ can also be solved with some learnable methods for computational acceleration, but we pay greater attention to the learned sparse regularizer and thus adopt a straightforward closed-form solution. Because $f(\mathbf{W})=\frac{1}{2}\left\|\mathbf{W}-\sum_{j=1}^{v}\boldsymbol{\alpha}_j\mathbf{W}_j\right\|_F^2+\frac{\mu}{2}\mathrm{Tr}(\mathbf{Y}^T(\mathbf{D}-\mathbf{W})\mathbf{Y})+\frac{\nu}{2}\|\mathbf{Y}-\mathbf{L}\|_F^2$ in (21) is differentiable, DSRL is also applicable to solving this problem.

4.1 Datasets

In this subsection, eight publicly available datasets are used to validate the effectiveness of the proposed DSRL method. These datasets are derived from real-world image applications, ranging from images to videos. Several sample images are randomly collected from the test datasets, as demonstrated in Fig. 3. It is suggested that the input images are captured at varied viewing angles, illumination and resolution variations, which motivates us to extract multi-view low-level features using feature descriptors for each dataset. Here we provide more details for the feature extractors of these test datasets.

ALOI. This dataset includes a collection of object images taken under varied light conditions and rotation angles.[1] The four commonly used features are 64-D RGB color histograms, 64-D HSV color histograms, 77-D color similarities, and 13-D Haralick features.

Caltech101-7/Caltech101-20. Caltech101 is a popular object recognition dataset with 101 classes of images.[2] We follow a previous work [42] in selecting the widely used subsets Caltech101-7 and Caltech101-20. Six extracted features are available: 48-D Gabor, 40-D wavelet moments (WM), 254-D CENTRIST, 1,984-D histogram of oriented gradients (HOG), 512-D GIST and 928-D LBP features.

MNIST. This is a well-known dataset of handwritten digits.[3] Three types of features are extracted from all test images: 30-D IsoProjection, 9-D Linear Discriminant Analysis, and 9-D Neighborhood Preserving Embedding features.

NUS-WIDE. As a web image dataset for object recognition,[4] we select eight classes of six feature sets: 64-D color histogram, 225-D block-wise color moments, 144-D color correlogram, 73-D edge direction histogram, 128-D wavelet texture and 500-D bag of words from SIFT descriptors.

MSRC-v1. It is an image dataset with eight object classes, each with 30 images.[5] Following [43], we select seven classes composed of trees, buildings, airplanes, cows, faces, cars and bicycles. Five visual feature sources are extracted from each image: 24-D color moment, 576-D HOG, 512-D GIST, 256-D local binary pattern, and 256-D CENTRIST features.

ORL. This database contains ten different face images, each of 40 subjects, which are taken at various times with different lighting and facial expressions.[6]

Youtube. This is a video dataset containing 2,000 instances in 10 topics, along with six views of both visual and audio features, which are the 2,000-D cuboids histogram, 1,024-D hist motion estimate, 64-D HOG features, 512-D MFCC features, 64-D volume streams, and 647-D spectrogram streams.[7]

A summary of these eight test datasets for comparative experiments is presented in Table 2, including the numbers of samples, features, views and data types.

4.2 Performance Evaluation

For clustering tasks, three well-known evaluation metrics are applied to the comparative experiments, including clustering accuracy (ACC), normalized mutual information (NMI), and adjusted rand index (ARI). Given sample x_i for any $i\in\{1,\ldots,n\}$, the predicted clustering label and the real label are denoted as p_i and q_i, respectively. The ACC is defined as

$$\mathrm{ACC}=\frac{\sum_{i=1}^{n}\delta(q_i,\mathrm{map}(p_i))}{n}, \tag{23}$$

where $\delta(a,b)=1$ if $a=b$, and $\delta(a,b)=0$ otherwise. Here, $\mathrm{map}(\cdot)$ is the best permutation mapping that matches the predicted clustering labels to the ground truths. Denote the predictive clustering result as $\tilde{\mathbf{C}}=\{\tilde{\mathbf{C}}_i\}_{i=1}^{\tilde{c}}$ and the ground truth as $\mathbf{C}=\{\mathbf{C}_j\}_{j=1}^{c}$, then NMI is calculated by

$$\mathrm{NMI}=\frac{\sum_{i=1}^{\tilde{c}}\sum_{j=1}^{c}|\tilde{\mathbf{C}}_i\cap\mathbf{C}_j|\log\frac{n|\tilde{\mathbf{C}}_i\cap\mathbf{C}_j|}{|\mathbf{C}_i||\mathbf{C}_j|}}{\sqrt{\left(\sum_{i=1}^{\tilde{c}}|\tilde{\mathbf{C}}_i|\log\frac{|\tilde{\mathbf{C}}_i|}{n}\right)\left(\sum_{j=1}^{c}|\mathbf{C}_j|\log\frac{|\mathbf{C}_j|}{n}\right)}}. \tag{24}$$

ARI characterizes the agreement between two partitions $\mathbf{C}$ and $\tilde{\mathbf{C}}$, defined as

$$\mathrm{ARI}=\frac{\sum_{ij}\binom{n_{ij}}{2}-\left[\sum_i\binom{a_i}{2}\sum_j\binom{b_j}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_i\binom{a_i}{2}+\sum_j\binom{b_j}{2}\right]-\left[\sum_i\binom{a_i}{2}\sum_j\binom{b_j}{2}\right]/\binom{n}{2}}, \tag{25}$$

1. https://elki-project.github.io/datasets/multi_view
2. http://www.vision.caltech.edu/Image_Datasets/Caltech101/
3. http://yann.lecun.com/exdb/mnist/
4. https://lms.comp.nus.edu.sg/wp-content/uploads/2019/research/nuswide/NUS-WIDE.html
5. http://riemenschneider.hayko.at/vision/dataset/task.php?did=35
6. http://cam-orl.co.uk/facedatabase.html
7. http://archive.ics.uci.edu/ml/datasets

TABLE 2
A Brief Description of the Test Datasets

Dataset ID	Datasets	# Samples	# Views	# Total features	# Classes	Data types
1	ALOI	1,079	4	218	10	Object image
2	Caltech101-7	1,474	6	3,766	7	Object image
3	Caltech101-20	2,386	6	3,766	20	Object image
4	MNIST	2,000	3	48	10	Digit image
5	NUS-WIDE	1,600	6	1,134	8	Web image
6	MSRC-v1	210	5	1,622	7	Object image
7	ORL	400	4	1,689	40	Face image
8	Youtube	2,000	6	4,311	10	Video data

where $[n_{ij}] = |\tilde{\mathbf{C}}_i \cap \mathbf{C}_j|$, $a_i = |\tilde{\mathbf{C}}_i|$ and $b_j = |\mathbf{C}_j|$. Higher values of all these metrics indicate better performance.

As for multi-view semi-supervised learning, we compute its classification accuracy for the performance evaluation. All experiments are repeated ten times, with accuracy means and standard deviations taken as the final results.

4.3 Parameter Setup

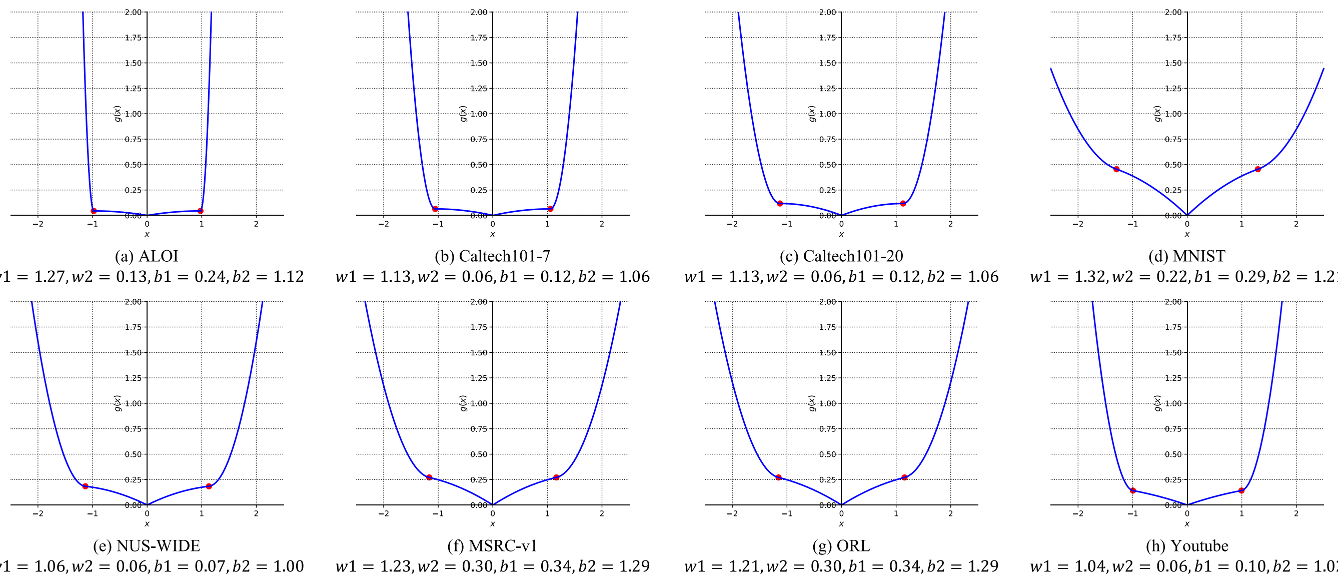
So as to validate the effectiveness and efficiency of the proposed DSRL method, several popular state-of-the-art methods are used for performance comparisons of multi-view clustering (K-Means, MLAN [44], SwMC [42], MSC-IAS [45], MCGC [46] and BMVC [47]) and semi-supervised classification (KNN, SVM, AdaBoost, MVAR [48], MLAN [44] and HLR-M^{2}VS [49]).

Here, we clarify some of the parameter settings used in the compared methods. All methods are tuned using their default settings if feasible. For other open hyperparameters, we adopt the following settings. The number of the nearest neighbors for MSC-IAS is fixed as 3, while the dimension of the intact space is fixed as 500. For MCGC, the regularization parameter β is set to 0.1. For MLAN, the number of adaptive neighbors ranges within [1,10]. For BMVC, we randomly generate 10 percent multi-view training data for non-linear anchor embedding. For MVAR, the trade-off weight for each view is fixed at $\lambda = 1000$, while the redistribution parameter over the views is set at $r = 2$. For HLR-M^{2}VS, two weighted factors are tuned as $\lambda_1 = 0.2$ and $\lambda_2 = 0.4$.

As for the proposed DSRL method, the activation function defined in (6) is employed. We set the block number as $t = 10$. The learning rate is fixed as $lr = 0.02$ for clustering and $lr = 0.05$ for semi-supervised classification. The initialization for the parameterized activation function is tuned as $w_1 = w_2 = 1.0$, $b_1 = 1.0$ and $b_2 = 2.0$. All methods are run on a computer with an i5-9500 CPU and 8GB RAM.

4.4 Multi-View Clustering

Fig. 4 shows the learned sparse regularizer $g(x) = \int_0^x (\xi^{-1}_{(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)}(y) - y)dy$ by the activation function $\xi_{(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)}(x)$ of DSRL on all test datasets for clustering tasks. The learned parameter $\boldsymbol{\Theta}$ differs in varied datasets, as a result of learning sparse regularizers in a data-driven manner. All the learned parameters of the activation functions obey (9). We observe that all the curves of the learned regularizers are symmetric about the y-axes, and $g(x) = 0$ when $x = 0$. In all test datasets, the learned sparse regularizer functions are nonnegative and monotonically



(a) ALOI
$w1 = 1.27, w2 = 0.13, b1 = 0.24, b2 = 1.12$

(b) Caltech101-7
$w1 = 1.13, w2 = 0.06, b1 = 0.12, b2 = 1.06$

(c) Caltech101-20
$w1 = 1.13, w2 = 0.06, b1 = 0.12, b2 = 1.06$

(d) MNIST
$w1 = 1.32, w2 = 0.22, b1 = 0.29, b2 = 1.21$

(e) NUS-WIDE
$w1 = 1.06, w2 = 0.06, b1 = 0.07, b2 = 1.00$

(f) MSRC-v1
$w1 = 1.23, w2 = 0.30, b1 = 0.34, b2 = 1.29$

(g) ORL
$w1 = 1.21, w2 = 0.30, b1 = 0.34, b2 = 1.29$

(h) Youtube
$w1 = 1.04, w2 = 0.06, b1 = 0.10, b2 = 1.02$

Fig. 4. The learned sparse regularizer $g(x) = \int_0^x (\xi^{-1}_{(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)}(y) - y)dy$ on test datasets for multi-view clustering. All learned parameters $\xi_{(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)}(x)$ of the activation functions are listed under each subfigure, with the points $x = \pm w_1(b_2 - b_1)$ marked in red.

TABLE 3
Clustering Accuracy (Mean% and Standard Deviation%) and Sparsity (Proportion of Near Zero Outputs) of the Proposed DSRL Method and Hand-Crafted Sparse Surrogates $g(x)$ Defined in Table 1, Where the Best Performance is Highlighted in Bold

Datasets \ Methods		Baseline	ℓ_p-norm	Logarithm	Geman	Laplace	ETP	DSRL
ALOI	ACC	75.7 (2.1)	76.6 (2.1)	75.3 (2.2)	61.9 (3.1)	61.7 (3.1)	61.3 (3.5)	**78.7 (1.8)**
	NMI	77.1 (2.0)	77.1 (1.5)	75.9 (1.8)	64.6 (1.0)	64.4 (1.1)	64.6 (0.8)	**78.7 (1.7)**
	ARI	58.1 (3.9)	57.5 (3.4)	51.8 (4.5)	33.3 (3.1)	33.2 (3.1)	32.3 (3.6)	**61.7 (3.7)**
	Sparsity	91.79	92.18	92.34	98.12	98.22	98.04	91.83
Caltech101-7	ACC	82.9 (0.1)	82.9 (0.3)	83.0 (0.4)	64.0 (5.8)	65.9 (0.6)	65.5 (5.8)	**83.8 (1.7)**
	NMI	59.9 (1.1)	60.1 (0.7)	60.3 (0.8)	37.6 (5.3)	37.6 (5.6)	37.6 (5.5)	**61.6 (4.1)**
	ARI	59.3 (2.0)	61.8 (4.8)	61.5 (4.2)	17.9 (1.6)	21.0 (4.8)	20.0 (5.3)	**61.9 (2.2)**
	Sparsity	91.68	92.11	97.74	97.90	97.92	99.03	91.73
Caltech101-20	ACC	71.5 (1.3)	71.4 (0.7)	70.9 (1.2)	68.7 (1.2)	66.9 (1.5)	67.0 (1.4)	**72.9 (1.1)**
	NMI	63.0 (2.8)	66.3 (3.0)	63.5 (3.0)	57.9 (2.4)	55.7 (2.3)	56.9 (2.8)	**68.2 (1.4)**
	ARI	71.7 (6.2)	62.2 (6.9)	57.1 (9.3)	38.5 (6.8)	34.8 (5.7)	37.5 (5.9)	**73.8 (2.2)**
	Sparsity	87.59	97.22	97.45	98.40	98.55	98.35	92.18
MNIST	ACC	84.2 (3.2)	85.4 (2.7)	85.4 (2.7)	81.8 (0.5)	79.5 (2.3)	79.0 (1.1)	**85.6 (0.3)**
	NMI	74.6 (1.2)	74.9 (0.8)	74.9 (0.9)	73.7 (0.3)	72.1 (0.7)	71.8 (0.7)	**75.6 (0.2)**
	ARI	74.4 (2.8)	75.1 (2.7)	75.1 (2.4)	64.8 (0.5)	61.1 (2.6)	57.9 (2.3)	**75.4 (0.4)**
	Sparsity	89.92	93.33	93.16	97.12	99.27	99.31	93.34
NUS-WIDE	ACC	39.1 (1.5)	36.9 (0.5)	39.6 (1.2)	36.8 (0.3)	37.4 (0.5)	37.5 (0.4)	**40.3 (0.1)**
	NMI	21.7 (1.9)	22.9 (0.7)	25.6 (1.2)	22.7 (0.5)	26.0 (0.6)	26.2 (0.6)	**26.5 (0.4)**
	ARI	14.5 (0.4)	14.3 (0.9)	15.0 (0.6)	15.0 (0.5)	12.4 (0.2)	12.2 (0.2)	**15.5 (0.2)**
	Sparsity	64.76	89.98	89.44	88.46	98.30	98.21	88.13
MSRC-v1	ACC	77.5 (4.9)	79.6 (0.3)	78.3 (3.4)	79.8 (0.2)	79.8 (1.3)	79.5 (0.3)	**83.4 (4.3)**
	NMI	71.0 (2.4)	72.2 (0.6)	71.2 (1.9)	72.6 (0.6)	71.4 (0.8)	71.5 (0.7)	**77.0 (3.0)**
	ARI	62.9 (3.9)	65.0 (0.6)	63.7 (3.0)	65.1 (0.5)	64.7 (1.5)	64.4 (0.7)	**69.6 (4.4)**
	Sparsity	79.39	79.40	79.61	83.76	85.12	85.34	91.80
ORL	ACC	81.5 (1.4)	77.6 (0.9)	75.3 (0.6)	77.8 (0.6)	79.8 (0.8)	79.8 (0.9)	**83.6 (1.2)**
	NMI	89.9 (0.6)	87.0 (0.3)	86.7 (0.4)	87.2 (0.4)	88.4 (0.5)	88.2 (0.4)	**91.3 (0.4)**
	ARI	73.2 (1.2)	61.6 (2.5)	52.2 (1.4)	62.0 (1.0)	67.2 (1.5)	66.5 (1.0)	**75.7 (1.5)**
	Sparsity	91.24	98.63	98.34	98.64	98.61	98.63	97.13
Youtube	ACC	38.9 (1.8)	40.5 (1.5)	41.2 (1.1)	35.8 (1.1)	38.6 (0.7)	38.1 (0.6)	**42.1 (0.7)**
	NMI	24.5 (1.7)	25.2 (1.3)	24.8 (1.0)	20.0 (1.3)	24.5 (0.7)	23.9 (0.7)	**27.0 (0.7)**
	ARI	15.1 (0.8)	16.1 (2.2)	16.3 (1.5)	12.4 (1.0)	12.9 (0.3)	12.1 (0.8)	**18.3 (0.7)**
	Sparsity	91.72	91.73	91.88	95.89	99.16	99.18	92.91

increasing on $(w_1(b_2 - b_1), \infty)$, but they may not be monotonically increasing on $(0, w_1(b_2 - b_1))$, which is drastically different from hand-crafted sparse regularizers. It can be seen from these figures that all learned sparse regularizers are differentiable except at $x = 0$. Note that DSRL does not need to approximate any hand-crafted sparse regularizer. Instead, it aims to learn task-specific sparse regularizers for given data. Therefore, the curves of learned $g(x)$ differ from those of hand-crafted sparse regularizers. However, these regularizers share some common characteristics: nonconvex, nondecreasing on $(0, \infty)$, and $g(0) = 0$.

Table 3 presents the clustering accuracy on all test datasets with various surrogate functions $g(x)$ in terms of ACC, NMI and ARI. The baseline method records the performance of directly solving $f(\mathbf{W})$ without sparse regularizers. To provide a fair comparison for all the defined sparse regularizers, both λ and γ range in $(0, 1.0]$. As for the ℓ_p-norm, the p value ranges in $(0, 1)$. From the experimental results, we have the observation that the performance of these specific regularizers is comparable in some datasets. At the same time, the proposed DSRL method achieves better performance than manually designed $g(x)$. Table 3 also provides the sparsity of the learned $g(x)$, where the sparsity is defined as the proportion of near zero outputs ($x \leq 0.01$). Notice that the input data are sparse for relatively large-scale datasets since the Gaussian kernel and nearest neighbors are used for affinity matrix evaluation. All methods still yield sparser outputs successfully in most datasets, and significantly promote the sparsity in all datasets except ALOI and Caltech101-7. Nonetheless, it can be seen that excessive sparseness may lead to decreased clustering performance, and DSRL improves the clustering performance with suitable sparse outputs. These observations indicate that the parameterized activation functions succeed in learning a data-driven sparse representation of the similarity matrices, and thus such learned sparse regularizers are more robust when applied to various datasets. Distinct from traditional hand-crafted sparse regularizers, DSRL can learn a more suitable sparse regularizer that is tailored for a given dataset. In other words, DSRL can learn a data-driven sparse regularizer, which intuitively provides strong generalization capability in practical applications.

Table 4 compares the clustering performance of DSRL and several state-of-the-art methods in terms of ACC, NMI and ARI. An example of visualization for clustering results in the MNIST dataset is demonstrated in Fig. 5. It can be observed that most of the multi-view clustering methods achieve better performance than single-view K-Means clustering. The experimental results also suggest that the proposed approach gains high accuracy and is effective on all test datasets. DSRL performs the best according to all metrics on seven of the eight test datasets. For the MNIST dataset, the proposed model also achieves the second best performance by all evaluation

TABLE 4
Clustering Accuracy (Mean% and Standard Deviation%) of All Compared Multi-View Clustering Methods, Where the Best Performance is Highlighted in Bold and the Second Best is Underlined

Datasets \ Methods		K-Means	MLAN	SwMC	MSC-IAS	MCGC	BMVC	DSRL
ALOI	ACC	47.5 (3.3)	59.0 (5.2)	45.7 (0.0)	59.4 (4.3)	52.4 (0.0)	54.8 (0.0)	**78.7 (1.8)**
	NMI	47.3 (2.1)	59.4 (4.3)	45.7 (0.0)	70.1 (1.8)	52.5 (0.0)	43.8 (0.0)	**78.7 (1.7)**
	ARI	33.0 (2.9)	34.5 (5.6)	17.8 (0.0)	53.2 (3.5)	25.9 (0.0)	32.8 (0.0)	**61.7 (3.7)**
Caltech101-7	ACC	49.6 (5.8)	78.0 (0.0)	66.5 (0.0)	71.3 (4.3)	64.3 (0.0)	57.9 (0.0)	**83.8 (1.7)**
	NMI	32.7 (1.9)	**63.0 (0.0)**	57.0 (0.0)	49.5 (3.8)	53.6 (0.0)	47.0 (0.0)	61.6 (4.1)
	ARI	30.2 (4.1)	57.2 (0.0)	42.7 (0.0)	52.1 (6.7)	49.8 (0.0)	41.8 (0.0)	**61.9 (2.2)**
Caltech101-20	ACC	31.3 (2.5)	52.6 (0.8)	54.1 (0.0)	41.9 (2.7)	54.6 (0.0)	47.4 (0.7)	**72.9 (1.1)**
	NMI	34.5 (1.1)	47.4 (0.3)	45.2 (0.0)	36.8 (2.5)	57.5 (0.0)	57.0 (0.3)	**68.2 (1.4)**
	ARI	18.9 (1.8)	19.8 (0.7)	19.8 (0.0)	16.9 (3.0)	38.8 (0.0)	35.0 (0.7)	**73.8 (2.2)**
MNIST	ACC	73.9 (7.2)	77.1 (0.5)	77.9 (0.0)	74.8 (0.3)	**88.7 (0.0)**	62.6 (2.2)	85.6 (0.3)
	NMI	68.1 (2.3)	75.5 (0.7)	70.9 (0.0)	74.5 (0.9)	**77.4 (0.0)**	56.2 (0.9)	75.6 (0.2)
	ARI	63.9 (4.3)	68.9 (1.0)	66.2 (0.0)	67.3 (0.8)	**78.8 (0.0)**	47.8 (2.1)	75.4 (0.4)
NUS-WIDE	ACC	32.0 (0.7)	34.7 (3.2)	22.9 (0.0)	32.4 (1.8)	34.3 (0.0)	36.6 (1.3)	**40.3 (0.1)**
	NMI	17.5 (0.7)	22.8 (2.0)	13.8 (0.0)	21.1 (0.6)	21.8 (0.0)	19.0 (0.4)	**26.5 (0.4)**
	ARI	9.00 (0.7)	13.8 (3.5)	3.60 (0.0)	11.4 (0.7)	13.3 (0.0)	13.5 (0.4)	**15.5 (0.2)**
MSRC-v1	ACC	46.3 (1.7)	68.1 (0.0)	78.6 (0.0)	47.5 (2.0)	75.2 (0.0)	63.8 (0.0)	**83.4 (4.3)**
	NMI	40.2 (1.5)	63.0 (0.0)	73.0 (0.0)	50.0 (1.7)	72.4 (0.0)	57.4 (0.0)	**77.0 (3.0)**
	ARI	26.9 (1.7)	50.4 (0.0)	65.2 (0.0)	31.0 (2.0)	64.3 (0.0)	48.8 (0.0)	**69.6 (4.4)**
ORL	ACC	59.0 (2.4)	77.8 (0.0)	74.8 (0.0)	73.3 (2.2)	81.0 (0.0)	56.7 (0.0)	**83.6 (1.2)**
	NMI	77.9 (1.4)	88.5 (0.0)	88.5 (0.0)	86.8 (1.4)	90.3 (0.0)	74.6 (0.0)	**91.3 (0.4)**
	ARI	46.3 (2.8)	66.9 (0.0)	56.4 (0.0)	62.7 (3.3)	70.0 (0.0)	60.0 (0.0)	**75.7 (1.5)**
Youtube	ACC	24.2 (1.6)	16.3 (1.0)	19.1 (0.0)	28.5 (0.8)	30.0 (0.0)	41.5 (0.7)	**42.1 (0.7)**
	NMI	15.1 (0.6)	6.14 (1.1)	11.1 (0.0)	15.7 (0.5)	17.4 (0.0)	25.7 (0.7)	**27.0 (0.7)**
	ARI	7.91 (0.9)	1.98 (0.6)	3.61 (0.0)	9.50 (9.3)	8.80 (0.0)	17.8 (0.4)	**18.3 (0.7)**

metrics. Overall, these results validate the feasibility and superiority of the proposed DSRL method.

4.5 Multi-View Semi-Supervised Classification

As an application to multi-view semi-supervised learning, we conduct experiments with 10 percent randomly generated labeled data. The learned sparse regularizers $g(x)$ used for semi-supervised classification are illustrated in Fig. 6. It can be seen from this figure that although the learned parameters differ from those in clustering scenarios, they share some common properties. We also compare the performance of DSRL and various defined sparse regularizers, as shown in Table 5. Unlike clustering tasks, it is noticed that the original affinity matrix without sparse constraints leads to poor performance for semi-supervised classification. All methods yield sparser outputs than those in clustering tasks, and obtain more significant improvements in classification accuracy. Although DSRL does not always generate the sparsest outputs, it achieves the

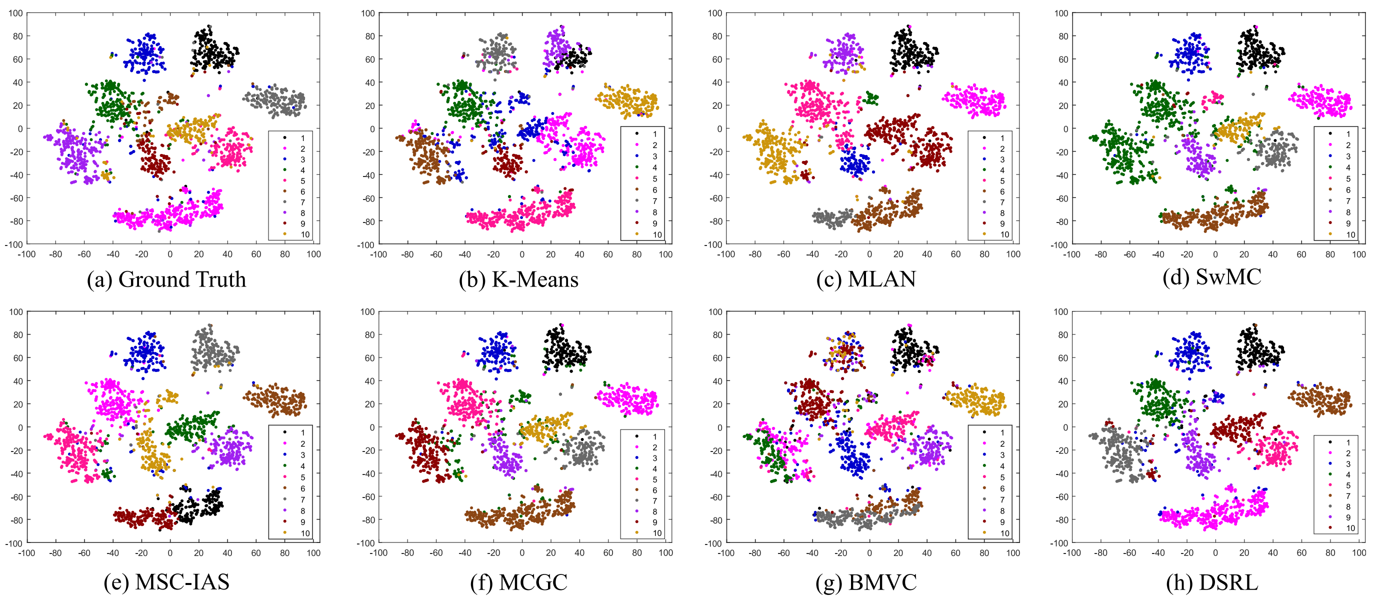
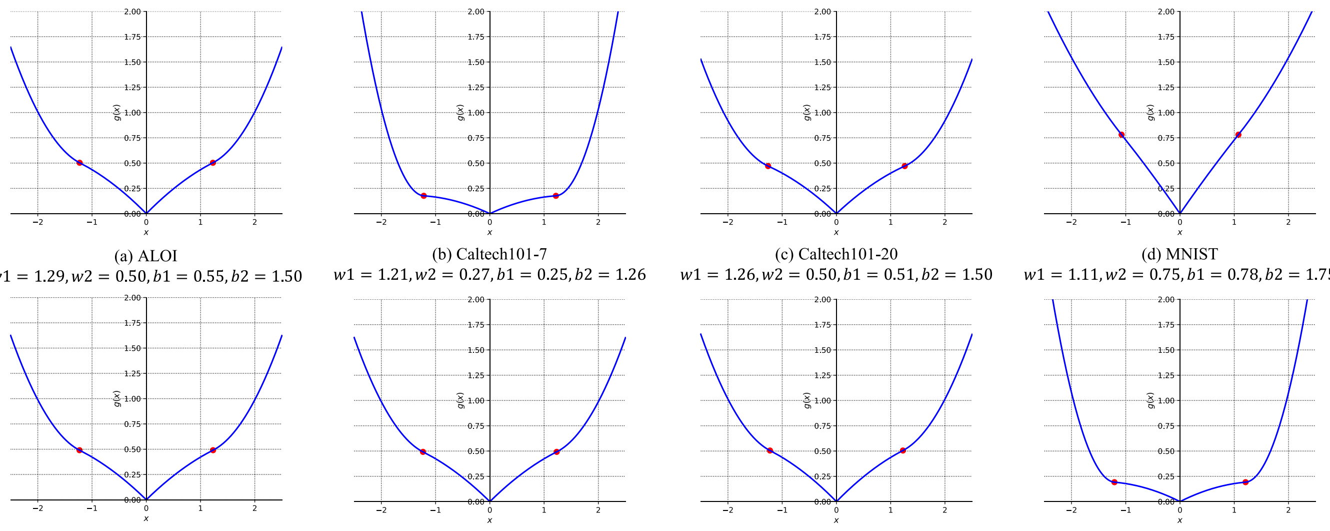


Fig. 5. Visualization for multi-view clustering results in dataset MNIST. Here, the high-dimensional input data are projected onto a two-dimensional subspace using t-SNE, then the corresponding data points are marked in varying colors according to their predictive labels.



(a) ALOI $w1 = 1.29, w2 = 0.50, b1 = 0.55, b2 = 1.50$

(b) Caltech101-7 $w1 = 1.21, w2 = 0.27, b1 = 0.25, b2 = 1.26$

(c) Caltech101-20 $w1 = 1.26, w2 = 0.50, b1 = 0.51, b2 = 1.50$

(d) MNIST $w1 = 1.11, w2 = 0.75, b1 = 0.78, b2 = 1.75$

(e) NUS-WIDE $w1 = 1.26, w2 = 0.50, b1 = 0.53, b2 = 1.50$

(f) MSRC-v1 $w1 = 1.27, w2 = 0.50, b1 = 0.53, b2 = 1.50$

(g) ORL $w1 = 1.29, w2 = 0.50, b1 = 0.55, b2 = 1.50$

(h) Youtube $w1 = 1.23, w2 = 0.27, b1 = 0.27, b2 = 1.26$

Fig. 6. The learned sparse regularizer $g(x) = \int_0^x (\xi^{-1}_{(\boldsymbol{\theta}_1,\boldsymbol{\theta}_2)}(y) - y)dy$ on test datasets for multi-view semi-supervised classification. All learned parameters $\xi_{(\boldsymbol{\theta}_1,\boldsymbol{\theta}_2)}(x)$ of the activation functions are listed under each subfigure, with the points $x = \pm w_1(b_2 - b_1)$ marked in red.

TABLE 5
Classification Accuracy (Mean% and Standard Deviation%) and Sparsity (Proportion of Near Zero Outputs) of the Proposed Method DSRL and Compared Hand-Crafted Sparse Surrogates $g(x)$ Defined in Table 1, Where the Best Performance is Highlighted in Bold

Datasets \ Methods		Baseline	ℓ_p-norm	Logarithm	Geman	Laplace	ETP	DSRL
ALOI	ACC	59.9 (9.7)	90.6 (1.2)	90.9 (1.1)	84.8 (5.6)	86.7 (3.4)	77.0 (7.5)	**91.6 (1.1)**
	Sparsity	91.79	97.95	98.17	98.79	98.71	92.78	97.95
Caltech101-7	ACC	80.4 (5.1)	92.1 (0.5)	92.0 (0.9)	89.6 (1.8)	89.8 (1.4)	87.6 (2.0)	**93.7 (0.9)**
	Sparsity	91.68	98.67	98.42	98.45	98.19	97.96	97.32
Caltech101-20	ACC	36.3 (1.9)	78.4 (1.7)	82.5 (1.8)	82.2 (1.2)	80.4 (1.6)	81.4 (1.2)	**84.2 (1.3)**
	Sparsity	87.59	98.14	98.28	98.76	98.86	98.83	98.00
MNIST	ACC	55.5 (9.7)	72.7 (5.0)	64.7 (5.7)	62.2 (3.3)	62.6 (5.8)	66.8 (6.2)	**87.6 (1.2)**
	Sparsity	89.92	93.26	93.19	93.14	93.10	93.12	95.00
NUS-WIDE	ACC	23.5 (6.7)	36.6 (2.1)	28.9 (4.6)	24.2 (7.6)	26.4 (6.0)	23.9 (6.8)	**44.3 (2.4)**
	Sparsity	64.76	96.02	95.90	98.47	98.80	99.03	96.01
MSRC-v1	ACC	52.3 (9.5)	72.9 (7.9)	74.7 (6.6)	72.6 (6.4)	72.1 (6.8)	72.9 (7.2)	**82.2 (4.3)**
	Sparsity	79.39	93.27	93.12	96.24	97.39	97.88	93.32
ORL	ACC	43.1 (4.3)	40.3 (3.9)	36.8 (4.3)	38.6 (5.5)	41.3 (4.0)	41.2 (5.9)	**54.9 (3.8)**
	Sparsity	91.24	91.31	91.39	91.37	92.40	92.38	97.10
Youtube	ACC	32.3 (5.9)	36.9 (0.8)	44.6 (1.5)	29.4 (1.4)	34.7 (2.1)	43.5 (1.7)	**48.0 (1.1)**
	Sparsity	91.72	99.63	99.53	99.69	99.60	99.50	99.06

TABLE 6
Classification Accuracy (Mean% and Standard Deviation%) of All Compared Semi-Supervised Classification Methods, Where the Best Performance is Highlighted in Bold and the Second Best is Underlined

Datasets \ Methods	KNN	SVM	AdaBoost	MVAR	MLAN	HLR-M^{2}VS	DSRL
ALOI	45.8 (3.0)	37.0 (6.6)	69.9 (9.5)	67.1 (6.0)	82.7 (2.5)	$\underline{87.7\ (1.7)}$	**91.6 (1.1)**
Caltech101-7	85.4 (0.8)	$\underline{87.2\ (1.9)}$	85.5 (1.2)	83.6 (0.6)	57.1 (1.2)	84.6 (1.0)	**93.7 (0.9)**
Caltech101-20	67.0 (0.7)	$\underline{71.0\ (2.3)}$	61.7 (2.5)	68.9 (4.6)	47.0 (1.7)	65.6 (2.1)	**84.2 (1.3)**
MNIST	86.5 (1.1)	$\underline{87.2\ (1.3)}$	63.3 (6.0)	84.1 (1.4)	68.8 (2.3)	87.1 (0.4)	**87.6 (1.2)**
NUS-WIDE	31.7 (2.1)	43.8 (1.9)	33.7 (3.0)	33.0 (2.2)	**47.9 (1.2)**	28.7 (1.1)	$\underline{44.3\ (2.4)}$
MSRC-v1	55.7 (5.9)	58.9 (5.9)	33.5 (5.2)	49.3 (5.0)	$\underline{81.8\ (1.9)}$	53.3 (5.3)	**82.2 (4.3)**
ORL	47.0 (3.4)	46.9 (2.4)	10.0 (0.2)	48.6 (4.2)	51.0 (2.3)	$\underline{52.0\ (4.4)}$	**54.9 (3.8)**
Youtube	35.9 (1.5)	$\underline{42.8\ (1.0)}$	27.4 (4.7)	37.6 (1.8)	36.4 (1.0)	32.7 (1.1)	**48.0 (1.1)**

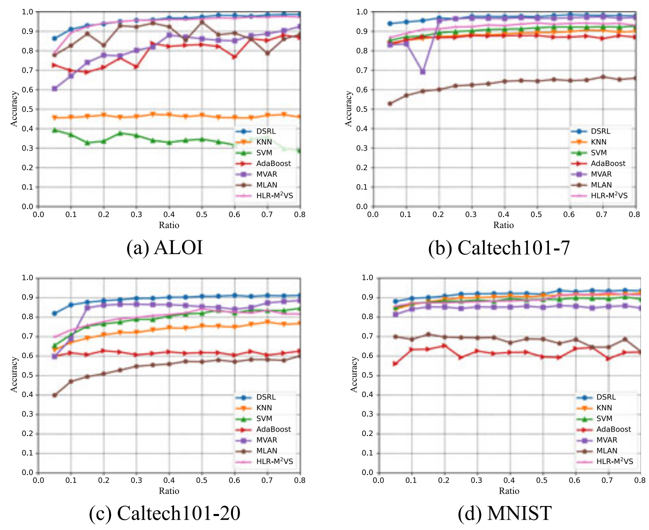


Fig. 7. Performance of all compared methods in semi-supervised classification tasks as the ratio of labeled data ranges in $\{0.05, 0.10, \ldots, 0.80\}$.

best performance on all datasets with considerable sparseness, which partially suggests the importance and necessity of data-driven learning for sparse regularizers. Moreover, it was difficult and time-consuming to select suitable hyperparameters for these defined sparse regularizers in the experiments, and DSRL solves this problem by learning sparse regularizers adaptively. Table 6 compares DSRL with other state-of-the-art semi-supervised classification methods, indicating that DSRL outperforms the others in seven of the eight test datasets. Further, we compare the performance of all methods as the ratio of labeled data ranges in $\{0.05, 0.10, \ldots, 0.80\}$ in Fig. 7. Overall, DSRL performs best on all test datasets, and gains higher accuracy with very limited numbers of labeled data points. The desired performance in semi-supervised classification further validates the effectiveness of the proposed DSRL method.

4.6 Runtime Analyses

In this subsection, we compare the runtimes of all defined sparse surrogates and the proposed DSRL method in Fig. 8. The runtimes of all methods are related to the input data dimensions, therefore these methods run faster on the MSRC-v1 and ORL datasets. As to other datasets with more samples, these methods require more time to yield sparse outputs. The time costs of hand-crafted sparse surrogates are comparable, and the experimental results indicate that DSRL outperforms other methods in terms of computational cost. Especially when used for semi-supervised classification, DSRL only takes less than half of the time required by other sparse surrogates. This can account for why we adopt a higher learning rate in semi-supervised classification than in clustering tasks, suggesting that DSRL converges faster to produce optimal sparse outputs in this sense.

4.7 Parameter Sensitivity

In this subsection, we examine the convergence and robustness of the proposed DSRL method. Its loss values with various numbers of iterations on different test datasets are demonstrated in Fig. 9. We observe that the loss objective value of DSRL gradually decreases as the number of iterations increases. Eventually, when the iteration number is large enough, it becomes stable with slight fluctuations, which suggests that it approaches convergence.

The performance of DSRL for clustering tasks with various numbers of blocks is reported in Fig. 10 in terms of ACC, NMI and ARI. In all figures, the block number ranges in $\{2, 4, \ldots, 24\}$, and the learning rate lr is fixed as 0.02. Several significant observations can be made. First, a small block number t leads to an acceptable result, which indicates that a smaller block number can be set to speed up computation. Second, the three performance evaluation metrics increase with block number for most datasets, becoming stable with slight fluctuations at $t > 10$. This is an empirical explanation for why we fix $t = 10$ in previous experiments to obtain acceptable results. Third, the influence of block number is not significant on some datasets such as ALOI, Caltech101-7, NUS-WIDE and Youtube, however, overall the experimental results follow our previous observations. We also demonstrate the influence of block numbers for multi-view semi-supervised classification in Fig. 11. In this case, the classification performance is more robust to the block number, and the experiments further validate our previous analyses.

5 Conclusion and Future Work

In this paper, we proposed an effective neural network model dubbed DSRL for learning data-driven sparse regularizers adaptively, which was a block-wise deep neural network with learnable activation functions. In this model, we

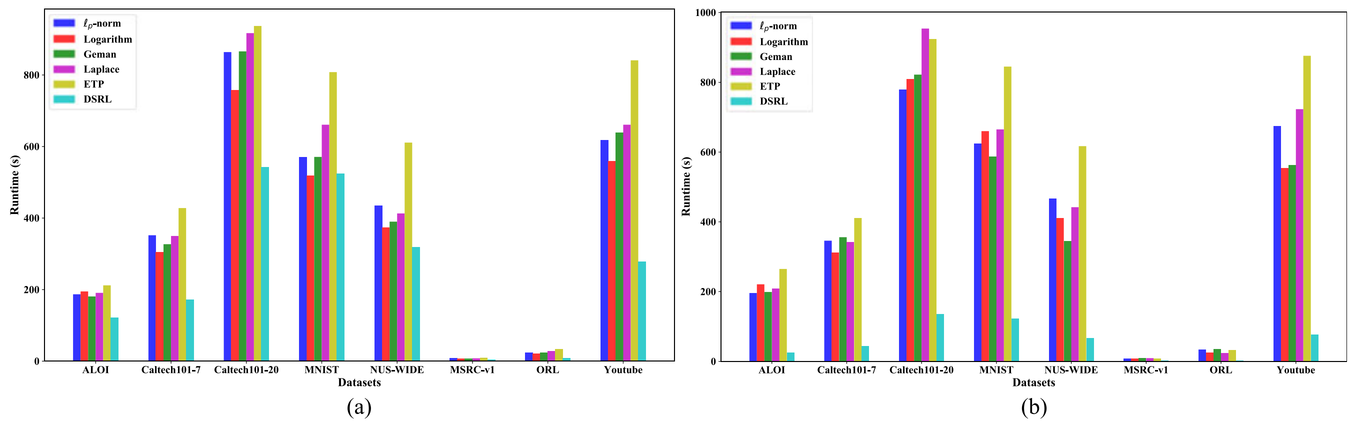
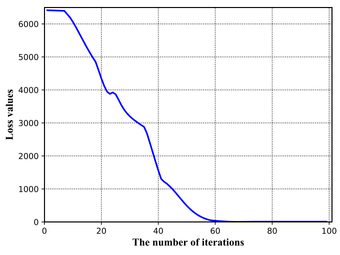
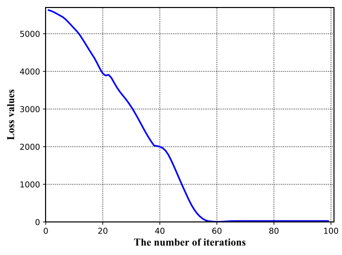


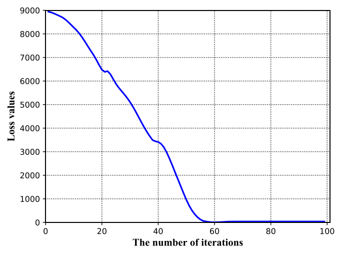
Fig. 8. Runtime comparison for all sparse surrogates and DSRL in (a) clustering and (b) semi-supervised classification.



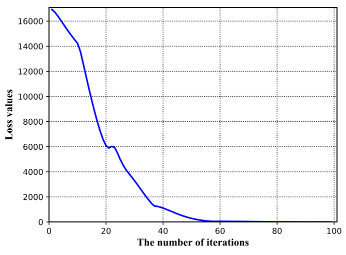
(a) ALOI



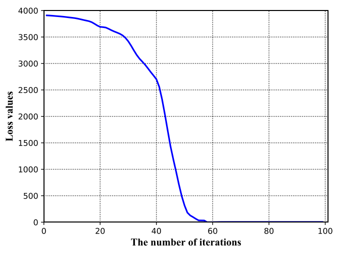
(b) Caltech101-7



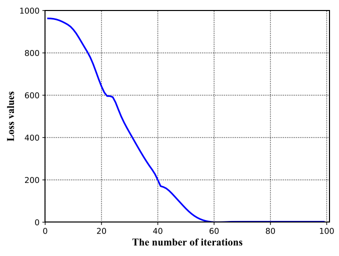
(c) Caltech101-20



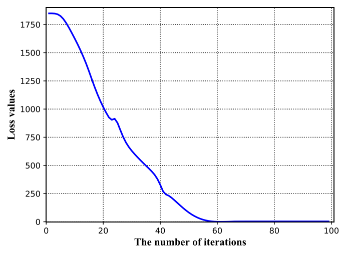
(d) MNIST



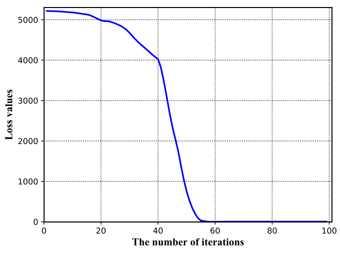
(e) NUS-WIDE



(f) MSRC-v1

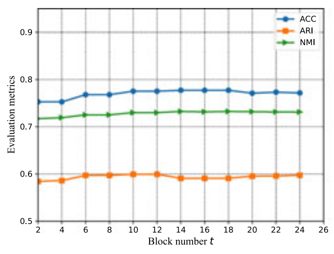


(g) ORL

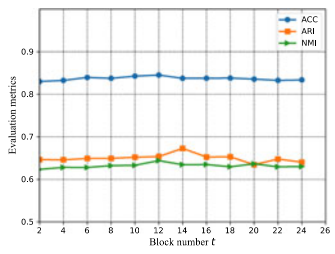


(h) Youtube

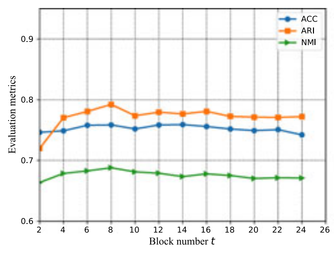
Fig. 9. The convergence curves of the proposed DSRL method on all test datasets.



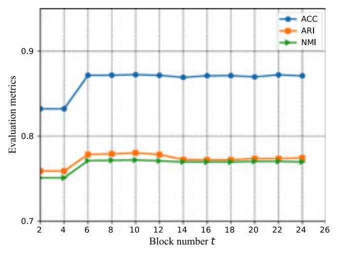
(a) ALOI



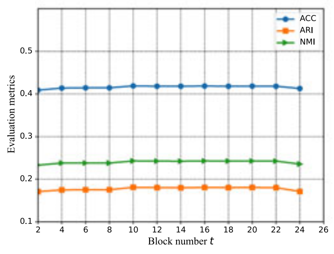
(b) Caltech101-7



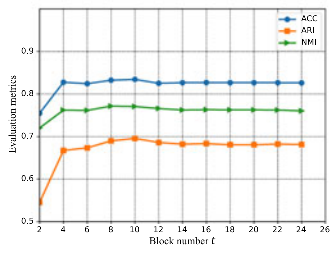
(c) Caltech101-20



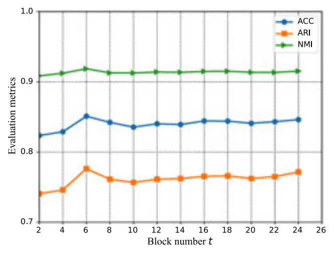
(d) MNIST



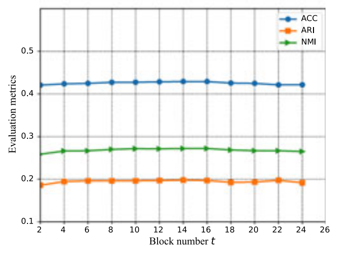
(e) NUS-WIDE



(f) MSRC-v1



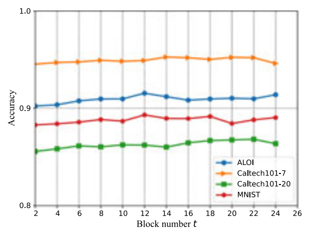
(g) ORL



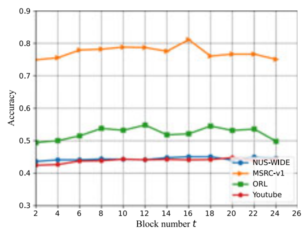
(h) Youtube

Fig. 10. Relationship between clustering performance (ACC, ARI, and NMI) and block number in $\{2, 4, \ldots, 24\}$ of the proposed DSRL method.

exploited the correspondence between sparse regularizers and parameterized activation functions via proximal operators, where sparse regularizers could be obtained from the integration of activation functions. This provided a solid theoretical justification for DSRL. The proposed DSRL method was demonstrated to be capable of solving optimization problems with adaptive sparse regularizers, which were not limited to hand-crafted sparse weights or outputs. Finally, we compared the performance of DSRL with those of hand-crafted sparse regularizers on eight real-world multi-view datasets. DSRL achieved superior performance in terms of multi-view clustering and semi-supervised classification. This approach is expected to provide some insights on learning adaptive sparse regularizers for various machine learning tasks. Currently, the learned sparse regularizers are only for entrywise sparsity. In future work, we will explore learnable multivariate sparse regularizers.



(a)



(b)

Fig. 11. Relationship between classification performance and block number in $\{2, 4, \ldots, 24\}$ of the proposed DSRL method on (a) ALOI, Caltech101-7, Caltech101-20 and MNIST, (b) NUS-WIDE, MSRC-v1, ORL and Youtube.

ACKNOWLEDGMENTS

The work of Shiping Wang was supported in part by the NSF of China under Grant U1705262, and in part by the NSF of

Fujian Province under Grants 2020J01130193 and 2018J07005. The work of Zhouchen Lin was supported in part by the NSF China under Grants 61625301 and 61731018, in part by the Major Scientific Research Project of Zhejiang Lab under Grants 2019KB0AC01 and 2019KB0AB02, in part by the Beijing Academy of Artificial Intelligence, and in part by Qualcomm.

REFERENCES

[1] Y. Li, C. Chen, F. Yang, and J. Huang, "Hierarchical sparse representation for robust image registration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 9, pp. 2151–2164, Sep. 2018.

[2] Y. Zhang, H. Zhang, M. Yu, S. Kwong, and Y. Ho, "Sparse representation-based video quality assessment for synthesized 3D videos," *IEEE Trans. Image. Process.*, vol. 29, pp. 509–524, Jul. 2020.

[3] D. Zou, X. Chen, G. Cao, and X. Wang, "Unsupervised video matting via sparse and low-rank representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 6, pp. 1501–1514, Jun. 2020.

[4] H. Gao and H. Huang, "Stochastic second-order method for large-scale nonconvex sparse learning models," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 2128–2134.

[5] F. Sun, J. Guo, Y. Lan, J. Xu, and X. Cheng, "Sparse word embeddings using l1 regularized online learning," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 2915–2921.

[6] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imag. Sci.*, vol. 2, no. 1, pp. 183–202, 2009.

[7] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 399–406.

[8] J. Zhang and B. Ghanem, "ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1828–1837.

[9] Z. Wang, Q. Ling, and T. S. Huang, "Learning deep ℓ_0 encoders," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 2194–2200.

[10] P. Sprechmann, A. M. Bronstein, and G. Sapiro, "Learning efficient sparse and low rank models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1821–1833, Sep. 2015.

[11] G. Tanaka *et al.*, "Spatially arranged sparse recurrent neural networks for energy efficient associative memory," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 1, pp. 24–38, Jan. 2020.

[12] W. Luo, W. Liu, and S. Gao, "A revisit of sparse coding based anomaly detection in stacked rnn framework," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 341–349.

[13] A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, and J. Z. Kolter, "Differentiable convex optimization layers," in *Proc. Neural Inf. Process. Syst.*, 2019, pp. 9558–9570.

[14] H. Yang, W. Wen, and H. Li, "Deephoyer: Learning sparser neural network with differentiable scale-invariant sparsity measures," in *Proc. 8th Int. Conf. Learn. Representations*, 2020.

[15] P. Tian, Z. Wu, L. Qi, L. Wang, Y. Shi, and Y. Gao, "Differentiable meta-learning model for few-shot semantic segmentation," in *Proc. 34th AAAI Conf. Artif. Intell.*, 2020, pp. 12087–12094.

[16] J. Sun *et al.*, "Deep ADMM-Net for compressive sensing MRI," in *Proc. Neural Inf. Process. Syst.*, 2016, pp. 10–18.

[17] X. Xie, J. Wu, G. Liu, Z. Zhong, and Z. Lin, "Differentiable linearized ADMM," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 6902–6911.

[18] L. Bertinetto, J. F. Henriques, P. H. S. Torr, and A. Vedaldi, "Meta-learning with differentiable closed-form solvers," in *Proc. 7th Int. Conf. Learn. Representations*, 2019.

[19] A. Bibi, B. Ghanem, V. Koltun, and R. Ranftl, "Deep layers as stochastic solvers," in *Proc. 7th Int. Conf. Learn. Representations*, 2019.

[20] P. L. Combettes and J.-C. Pesquet, "Deep neural network structures solving variational inequalities," *Set-Valued Variational Anal.*, vol. 28, pp. 491–518, 2020.

[21] J. Li, C. Fang, and Z. Lin, "Lifted proximal operator machines," in *Proc. 33rd AAAI Conf. Artif. Intell.*, 2019, pp. 4181–4188.

[22] L. E. Frank and J. H. Friedman, "A statistical view of some chemometrics regression tools," *Technometrics*, vol. 35, no. 2, pp. 109–135, 1993.

[23] J. H. Friedman, "Fast sparse regression and classification," *Int. J. Forecasting*, vol. 28, no. 3, pp. 722–738, 2012.

[24] D. Geman and C. Yang, "Nonlinear image recovery with half-quadratic regularization," *IEEE Trans. Image. Process.*, vol. 4, no. 7, pp. 932–946, Jul. 1995.

[25] J. D. Trzasko and A. Manduca, "Highly undersampled magnetic resonance image reconstruction via homotopic ℓ_0-minimization," *IEEE Trans. Med. Imag.*, vol. 28, no. 1, pp. 106–121, Jan. 2009.

[26] C. Gao, N. Wang, Q. R. Yu, and Z. Zhang, "A feasible nonconvex relaxation approach to feature selection," in *Proc. 25th AAAI Conf. Artif. Intell.*, 2011, pp. 356–361.

[27] C. Lu, C. Zhu, C. Xu, S. Yan, and Z. Lin, "Generalized singular value thresholding," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 1805–1811.

[28] M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien, "ℓ_p-norm multiple kernel learning," *J. Mach. Learn. Res.*, vol. 12, pp. 953–997, 2011.

[29] C. Lu, J. Tang, S. Yan, and Z. Lin, "Generalized nonconvex nonsmooth low-rank minimization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 4130–4137.

[30] H. Zhang, C. Gong, J. Qian, B. Zhang, C. Xu, and J. Yang, "Efficient recovery of low-rank matrix via double nonconvex nonsmooth rank minimization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 10, pp. 2916–2925, Oct. 2019.

[31] C. Dan, H. Wang, H. Zhang, Y. Zhou, and P. Ravikumar, "Optimal analysis of subset-selection based l_p low-rank approximation," in *Proc. 33rd Conf. Neural Inf. Process. Syst.*, 2019, pp. 2537–2548.

[32] V. Papyan, Y. Romano, and M. Elad, "Convolutional neural networks analyzed via convolutional sparse coding," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 2887–2938, 2017.

[33] S. Liu, "Learning sparse neural networks for better generalization," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, 2020, pp. 5190–5191.

[34] W. Luo, J. Li, J. Yang, W. Xu, and J. Zhang, "Convolutional sparse autoencoders for image classification," *IEEE Trans. Neural Netw. and Learn. Syst.*, vol. 29, no. 7, pp. 3289–3294, Jul. 2018.

[35] E. Tartaglione, S. Lepsøy, A. Fiandrotti, and G. Francini, "Learning sparse neural networks via sensitivity-driven regularization," in *Proc. Neural Inf. Process. Syst.*, 2018, pp. 3878–3888.

[36] X. Liu, W. Li, J. Huo, L. Yao, and Y. Gao, "Layerwise sparse coding for pruned deep neural networks with extreme compression ratio," in *Proc. 34th AAAI Conf. Artif. Intell.*, 2020, pp. 4900–4907.

[37] A. Bhowmik, A. Adiga, C. Seelamantula, F. Hauser, J. Jacak, and B. Heise, "Bayesian deep deconvolutional neural networks," in *Proc. 2nd Neural Inf. Process. Syst. Workshop Bayesian Deep Learn.*, 2017.

[38] S. Wang, S. Fidler, and R. Urtasun, "Proximal deep structured models," in *Proc. Neural Inf. Process. Syst.*, 2016, pp. 865–873.

[39] D. Mahapatra, S. Mukherjee, and C. S. Seelamantula, "Deep sparse coding using optimized linear expansion of thresholds," 2017, *arXiv: 1705.07290*.

[40] S. Srinivas, A. Subramanya, and R. Venkatesh Babu, "Training sparse neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2017, pp. 138–145.

[41] R. Ma, J. Miao, L. Niu, and P. Zhang, "Transformed ℓ_1 regularization for learning sparse deep neural networks," *Neural Netw.*, vol. 119, pp. 286–298, 2019.

[42] F. Nie, J. Li, and X. Li, "Self-weighted multiview clustering with multiple graphs," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2017, pp. 2564–2570.

[43] F. Nie, J. Li, and X. Li, "Parameter-free auto-weighted multiple graph learning: A framework for multiview clustering and semi-supervised classification," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 1881–1887.

[44] F. Nie, G. Cai, and X. Li, "Multi-view clustering and semi-supervised classification with adaptive neighbours," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 2408–2414.

[45] X. Wang, Z. Lei, X. Guo, C. Zhang, H. Shi, and S. Z. Li, "Multi-view subspace clustering with intactness-aware similarity," *Pattern Recognit.*, vol. 88, pp. 50–63, 2019.

[46] K. Zhan, F. Nie, J. Wang, and Y. Yang, "Multiview consensus graph clustering," *IEEE Trans. Image. Process.*, vol. 28, no. 3, pp. 1261–1270, Mar. 2019.

[47] Z. Zhang, L. Liu, F. Shen, H. T. Shen, and L. Shao, "Binary multi-view clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 7, pp. 1774–1782, Jul. 2019.

[48] H. Tao, C. Hou, F. Nie, J. Zhu, and D. Yi, "Scalable multi-view semi-supervised classification via adaptive regression," *IEEE Trans. Image. Process.*, vol. 26, no. 9, pp. 4283–4296, Sep. 2017.

[49] Y. Xie, W. Zhang, Y. Qu, L. Dai, and D. Tao, "Hyper-Laplacian regularized multilinear multiview self-representations for clustering and semisupervised learning," *IEEE Trans. Cybern.*, vol. 50, no. 2, pp. 572–586, Feb. 2020.

Shiping Wang received the PhD degree from the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China, in 2014. From August 2015 to August 2016, he was a research fellow with Nanyang Technological University. He is currently a professor at the College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China. His research interests include machine learning, computer vision, and granular computing.

Zhaoliang Chen received the BS degree in 2019 from the College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China, where he is currently pursuing the PhD degree. His current research interests include machine learning, deep learning, and recommender systems.

Shide Du received the BS degree in 2019 from the College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China, where he is currently working toward the MS degree. His current research interests include machine learning, differentiable programming, and deep learning.

Zhouchen Lin (M'00-SM'08-F'18) received the PhD degree from Peking University in 2000. He is currently a professor at the Key Laboratory of Machine Perception, School of Electrical Engineering and Computer Science, Peking University. His research interests include computer vision, image processing, machine learning, pattern recognition, and numerical optimization. He was the area chair of CVPR, ICCV, NIPS/NeurIPS, AAAI, IJCAI, ICLR and ICML for many times and is the program co-chair of ICPR 2022. He was an associate editor for the *IEEE Transactions on Pattern Analysis and Machine Intelligence* and currently is an associate editor for the *International Journal of Computer Vision*. He is a fellow of the IAPR.