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. Further, the proposed model is trained with back propagation, and all parameters in this model are learned end-to-end. We apply our framework to the multi-view clustering and semi-supervised classification tasks for learning 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.**

I. 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 the 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. Then ℓ_0 regularized encoder [9] was also explored to construct an effective sparse regularization with time-unfolding feed-forward neural networks. Sprechmann et al. showed a principled way to construct learnable pursuit process architectures for structured sparse models, derived from the iteration of proximal gradient descent algorithms [10]. Tanaka et al. proposed sparse recurrent neural networks to conduct an efficient energy information processing [11]. Luo et al. mapped the 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.

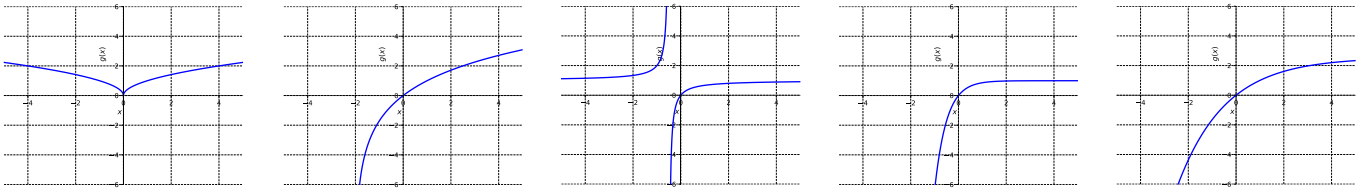
However, from the perspective of model training, deep neural networks (DNNs) are usually limited by conducting back propagation and gradient descent with differentiable regularizers. Therefore, how to build a network that can deal with non-differentiable objective functions using differentiable blocks in the deep learning framework is a pivotal problem. Differentiable programming solves this problem by reformulating the 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 in an alternating direction method of multipliers (ADMM) algorithm for optimizing an MRI model based on compressive sensing [16]. Xie et al. proposed a dif-

This work was partially supported by the National Natural Science Foundation of China (Nos. U1705262 and 61672159), the Technology Innovation Platform Project of Fujian Province under Grant (Nos. 2014H2005 and 2009J1007), the Fujian Collaborative Innovation Center for Big Data Application in Governments, the Fujian Engineering Research Center of Big Data Analysis and Processing.

Shiping Wang, Zhaoliang Chen and Shide Du are with the College of Mathematics and Computer Sciences, Fuzhou University, Fuzhou 350116, China and also with the Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, Fuzhou 350116, China (email: 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, China (email: zlin@pku.edu.cn).

Zhouchen Lin is the corresponding author.



(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 $(0, \infty)$.

ferentiable 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 for quickly learning parameters [18]. Because proximal operators are universally 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 is still unexplored. To our knowledge, very limited research work has been devoted to the general learning framework of sparse regularizers.

In this paper, we propose a new deep network dubbed deep sparse regularizer learning (DSRL) framework, to adaptively learn data-driven sparse regularizers. Bridged by the proximal operator, we exploit a 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 compared sparse regularizers. The main contributions of this paper can be summarized into 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 in eight real-world datasets in comparison with specific regularizers and other state-of-the-art methods.

II. Related Work

TABLE I: Several specified definitions of $g(\cdot)$ for sparse surrogates.

Norm	Definition of $g(x)$, $x \geq 0$ with $\lambda \geq 0$
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]	$g(x) = \lambda \frac{1-\exp(-\gamma x)}{1-\exp(-\gamma)}$

A large amount of research has recognized the critical role played by sparse representation. However, most previous studies concentrated on the hand-crafted sparsity. Several commonly used sparse surrogates are shown in Table I [27]. These defined sparse regularizers are non-decreasing and non-convex on $(0, \infty)$, as is illustrated in Figure 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 [29]. These sparse surrogates are also applied to rank regularized optimization problems:

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

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 i-th 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 I [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 [30]. Zhang et al. further handled nonconvex nonsmooth rank minimization problems with closed-form solutions of ℓ_p-norm when $p = \frac{1}{2}$ and $\frac{2}{3}$ [31]. Dan et al. studied low-rank recovery models with the ℓ_p-norm loss and provided a better approximation guarantee [32]. In general, these hand-crafted sparse surrogates tend to approximate specific sparsity and are often sensitive to predefined hyperparameters, which leads to difficulties in model training and generalization. Moreover, due to 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 [33], [34] 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 [35] only allowed a small number of hidden units to be active at once with Kullback-Leibler divergence. Tartaglione et al. exploited a simple thresholding to promote the sparse property of network parameters [36]. Liu et al. pruned redundant connections to generate sparse layers [37]. 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 [38]. 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 [39]. Mahapatra et al. [40] 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 [41]. Ma et al. used an integrated transformed nonconvex ℓ_1 regularizer to promote the sparsity of parameters [42]. 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 our best 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 some specific sparse regularizers.

III. 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 block-wise neural network is designed to learn a data-driven sparse regularizer. Figure 2 illustrates the structure of the proposed framework.

A. Correspondence between Sparse Regularizers and Activation Functions

Proximal operator is 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 some 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 is 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 Figure 2 of [27]). Namely, $0 \in \xi^{-1}(0)$. Therefore, $g(x) = 0$, which is also non-decreasing and nonnegative on $(0, \infty)$. Based on these conditions, $g(x)$ is similar to those hand-crafted sparse regularizers shown in Figure 1 when $x \in (0, \infty)$, which makes the learned sparse regularizer close to the optimal one with minimum constraints.

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

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

where $\theta = \{b\}$ 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 $\xi(x)$ which is non-decreasing and maps a neighborhood of 0 to 0. Because it is tough for the activation function of only one bias to learn a suitable sparse regularizer with given data, we employ piecewise linear functions to approximate the learnable activation function, consisting of more learnable coefficients and biases. Particularly, an activation function of two sets of learnable parameters (θ_1, θ_2) is defined as

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

where $x \in \mathbb{R}$, $0 \le b_1 \le b_2$ and $w_1, w_2 > 0$ are learnable parameters with $\theta_1 = (w_1, b_1)$ and $\theta_2 = (w_2, b_2)$. 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 applications. With

this definition, the inverse function $\xi^{-1}_{(\theta_1,\theta_2)}(y)$ is computed by

$$\xi^{-1}_{(\theta_1,\theta_2)}(y)=\begin{cases}\frac{y-w_1(b_2-b_1)}{w_2}+b_2, & w_1(b_2-b_1)\le y,\\ \frac{y}{w_1}+b_1, & 0\le y<w_1(b_2-b_1),\\ [-b,b], & y=0,\\ \frac{y}{w_1}-b_1, & -w_1(b_2-b_1)\le y<0,\\ \frac{y-w_1(b_2-b_1)}{w_2}-b_2 & y<-w_1(b_2-b_1).\end{cases} \tag{7}$$

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

$$g(x)=\begin{cases}(\frac{1}{2w_2}-\frac{1}{2})x^2+(b_2-\frac{w_1(b_2-b_1)}{w_2})x & \\ +\frac{w_1(w_1-w_2)}{2w_2}(b_2-b_1)^2, & x\ge w_1(b_2-b_1),\\ (\frac{1}{2w_1}-\frac{1}{2})x^2+b_1x, & 0\le x<w_1(b_2-b_1),\\ g(-x), & x<0.\end{cases} \tag{8}$$

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 the theoretic strictness and better interpretation, 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]:

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

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

$$\mathbf{Proj}(b_1,b_2)=\begin{cases}(b_1,b_2), & b_2\ge 0,\frac{w_1-1}{w_1}b_2\le b_1\le b_2,\\ (\rho_1b_1+\rho_2b_2,\rho_2b_1+\rho_3b_2), & \frac{w_1}{1-w_1}b_2<b_1<\frac{w_1-1}{w_1}b_2,\\ (0,0), & b_2\ge 0,b_1\le\frac{w_1}{1-w_1}b_2,\\ (0,0), & b_2\le\min\{0,-b_1\},\\ (\frac{b_1+b_2}{2},\frac{b_1+b_2}{2}), & b_1\ge|b_2|,\end{cases} \tag{11}$$

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 $\rho_3=\frac{w_1^2}{w_1^2+(w_1-1)^2}$.

B. 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 by the theory in Section III-A, $g(\mathbf{X}_{ij})=\int_0^{\mathbf{X}_{ij}}(\xi^{-1}_{\mathbf{\Theta}}(y)-y)dy$ for any $i\in\{1,\cdots,n\}$ and $j\in\{1,\cdots,m\}$ with $\mathbf{\Theta}$ to be learned. 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:

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

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

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

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

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

Based on the above analysis, we propose an end-to-end deep learning framework which learns data-driven sparse regularizers. Each unit of the proposed framework is structured as a single differentiable block, as demonstrated in Figure 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 i-th 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 $\mathbf{\Theta}$. And DSRL is comprised of t differentiable blocks of learnable parameters $\mathbf{\Theta}$ and L, which can be learned end-to-end by back propagation [refer to Section B of Appendix]. 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 adaptive optimization of learnable parameters, and learns potential sparse regularizers in a data-driven way, which largely improves the performance of given tasks.

The time complexity of the proposed method is $O(ktnm^2)$ for forward propagation, where k denotes the number of

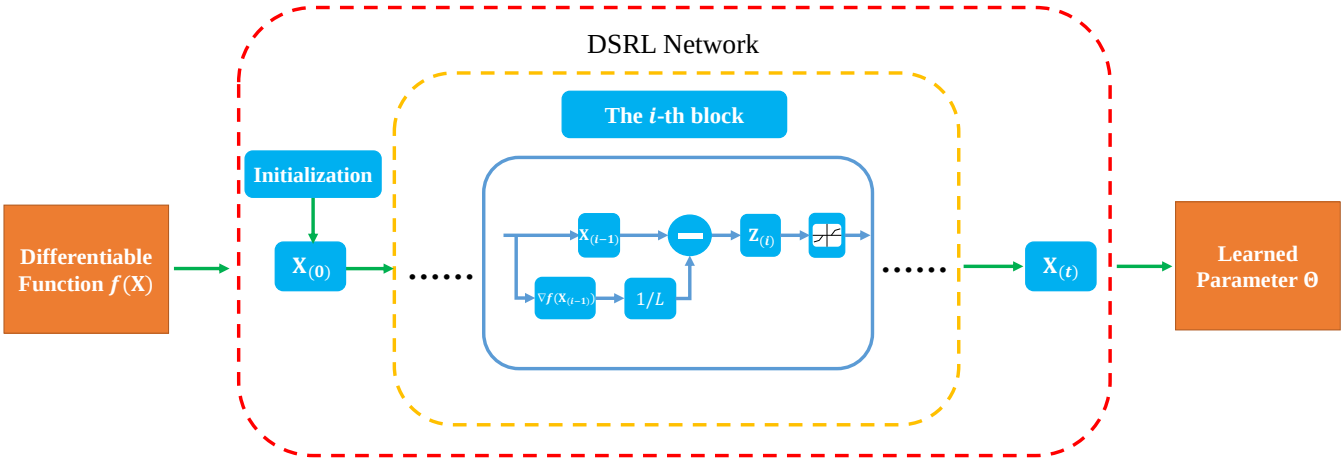


Fig. 2: Framework of the proposed method DSRL. It consists of multiple unfolded blocks, where a basic block is made up of several differentiable units as demonstrated in the blue grids.

Algorithm 1 Deep Sparse Regularizer Learning (DSRL)

Input: A differentiable function $f(\mathbf{X})$ and the number of blocks t.

Output: Learned parameter set $\mathbf{\Theta}$.

Initialize the data matrix $\tilde{\mathbf{X}} \in \mathbb{R}^{n\times m}$;
Initialize learnable parameter set $\mathbf{\Theta}^{(0)}$ and $L^{(0)}$;
Initialize counter k = 0;
while not convergent **do**
  The parameter $\mathbf{\Theta}^{(k)}$ is projected onto the convex set $\mathcal{S}$, denoted by $\bar{\mathbf{\Theta}}^{(k)}$;
  Update $\mathbf{Z}_{(0)} = \tilde{\mathbf{X}} - \frac{1}{L^{(k)}}\nabla f(\tilde{\mathbf{X}})$;
  Update $\mathbf{X}_{(0)} = \xi_{\bar{\mathbf{\Theta}}^{(k)}}(\mathbf{Z}_{(0)})$;
  for $i = 1, \cdots, t$ **do**
    Update $\mathbf{Z}_{(i)} = \mathbf{X}_{(i-1)} - \frac{1}{L^{(k)}}\nabla f(\mathbf{X}_{(i-1)})$;
    Update $\mathbf{X}_{(i)} = \xi_{\bar{\mathbf{\Theta}}^{(k)}}(\mathbf{Z}_{(i)})$;
  end for
  Update $\mathbf{\Theta}^{(k+1)}$ and $L^{(k+1)}$ with back propagation and loss function $\mathcal{J}(\tilde{\mathbf{X}}, \mathbf{X}_{(t)}) = \frac{1}{2}\|\tilde{\mathbf{X}} - \mathbf{X}_{(t)}\|_F^2$;
  Update counter $k = k + 1$;
end while
return The learned parameter set $\mathbf{\Theta}$.

epochs and t is the number of blocks. Because a small t often achieves acceptable performance (as described in Section IV-G), the speed of the proposed DSRL is relatively fast. After training, the parameters of activation functions are learned, and we obtain a reconstructed sparse output by the one-time forward propagation.

IV. 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.

Given 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 i-th 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 certain criterion $\text{loss}(\{\mathbf{X}_i\}_{i=1}^v; \mathbf{y})$. Considering that different views may come with varying dimensions, 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 as the following simple form

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

where $\boldsymbol{\alpha} = [\alpha_1; \cdots; \alpha_v] \in \mathbb{R}^v$ 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 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 $\boldsymbol{\alpha}$. And adaptive weights can be optimized by the ADMM algorithm. Particularly, suppose that $vec(\cdot)$ is the matrix vectorization operator, then the optimization subproblem with respect to $\boldsymbol{\alpha}$ is written as

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

While keeping the weighted vector $\boldsymbol{\alpha}$, we compute the optimal solution $\mathbf{W} = \mathbf{Prox}_g^\sigma(\sum_{j=1}\alpha_j\mathbf{W}_j)$. Because $f(\mathbf{W}) = \frac{1}{2}||\mathbf{W} - \sum_{j=1}^v \alpha_j\mathbf{W}_j||_F^2$ 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,

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

TABLE II: A brief description of the test datasets.

with a fixed $\boldsymbol{\alpha}$, we formulate the problem as

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

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 of given labeled data points. Specifically, $\mathbf{L}_{ij}=1$ if the i-th data point belongs to the j-th 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 more attention to the learned sparse regularizer and thus adopt a straightforward closed-form solution. Because $f(\mathbf{W})$ in (21) is differentiable, DSRL is also applicable to solving this problem.

A. Datasets

In this subsection, eight publicly available datasets are used to validate the effectiveness of the proposed method DSRL. 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 Figure 3. It is suggested from the figure that the input images may be captured with varied viewing angles, illumination colors and resolution variations, which motivates us to extract multi-view low-level features using feature descriptors for varying datasets. Here we provide more details for the feature extractors of these test datasets.

ALOI: A collection of object images which were taken under varied light conditions and rotation angles[1]. Four commonly used features include 64-D RGB color histograms, 64-D HSV color histograms, 77-D color similarities and 13-D Haralick features.

[1]https://elki-project.github.io/datasets/multi_view

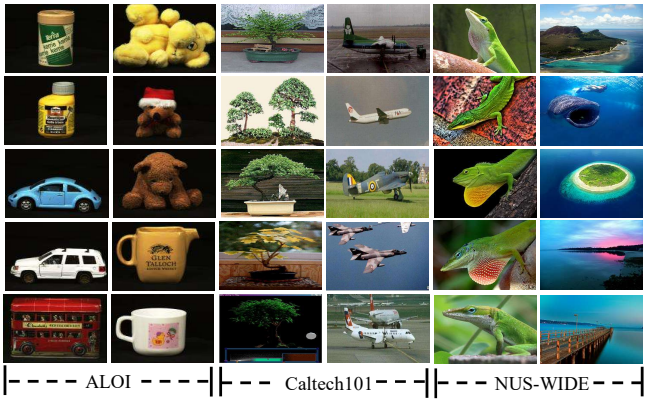


Fig. 3: Several sample images from the test image datasets.

Caltech101-7/Caltech101-20: Caltech101 is a popular object recognition dataset with 101 classes of images[2]. We follow the previous work [43] and select 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: It 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 (LDA) and 9-D Neighborhood Preserving Embedding (NPE) 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 8 object classes and each class has 30 images[5]. Following [44], we select 7 classes composed of tree, building, airplane, cow, face, car and bicycle. 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

[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

of 40 subjects, which were taken at various times, differing in the lighting and facial expressions[6].

Youtube: It is a video dataset with 2,000 instances in 10 topics, along with six views from both visual and audio features, including 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, including the numbers of samples, features, views and data types, is presented in Table II.

B. Performance Evaluation

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

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

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

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

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

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

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

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

C. Parameter Setup

So as to validate the effectiveness and efficiency of the proposed method DSRL, several popular state-of-the-art compared methods are used for multi-view clustering (K-Means, MLAN [45], SwMC [43], MSC-IAS [46], MCGC [47] and BMVC [48]) and semi-supervised classification (KNN, SVM, AdaBoost, MVAR [49], MLAN [45] and HLR-M^2VS [50]).

There are some parameter settings for the compared methods to be clarified in advance. All methods are tuned as 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, and the dimension of the intact space is fixed as 500. For MCGC, the regularization parameter β is set as 0.1. The number of adaptive neighbors ranges in $[1, 10]$ for MLAN. For BMVC, we randomly generate 10% multi-view training data for the non-linear anchor embedding. For MVAR, the trade-off weight for each view is fixed as λ = 1000, and the redistribution parameter over views is set as r = 2. As to HLR-M^2VS, two weighted factors are tuned as $\lambda_1 = 0.2$ and $\lambda_2 = 0.4$.

As for the proposed method DSRL, 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. An 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 8G RAM.

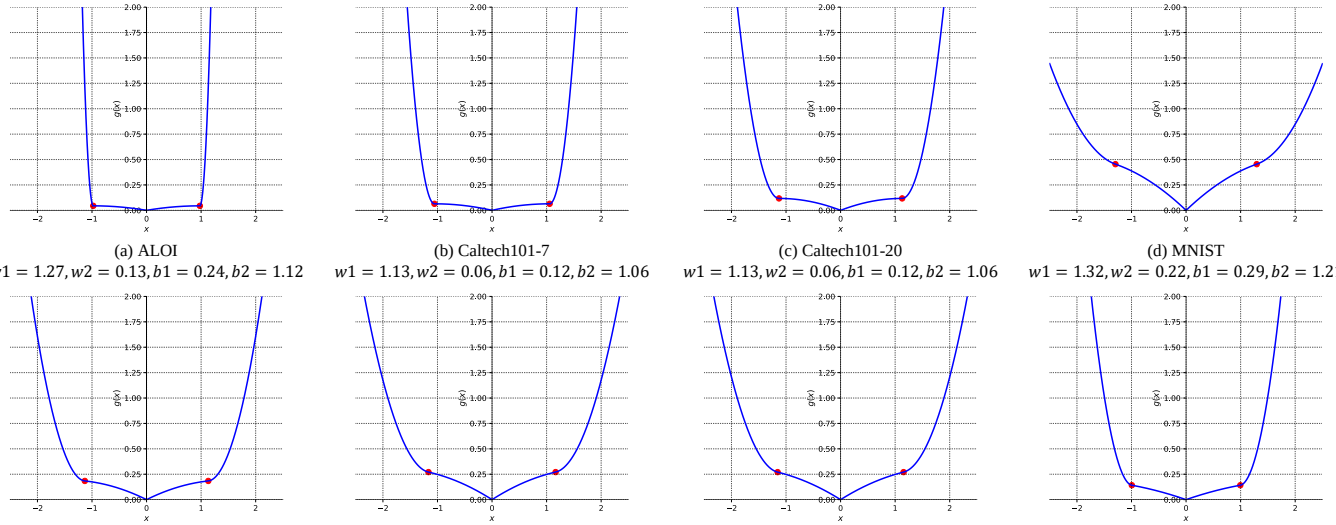
D. Multi-View Clustering

Figure 4 shows the learned sparse regularizer $g(x) = \int_0^x (\xi_{(\theta_1,\theta_2)}^{-1}(y) - y)dy$ by the activation function $\xi_{(\theta_1,\theta_2)}(x)$ 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 learned parameters of activation functions obey (9). We observe that all figures of learned regularizers are symmetric about the y-axis, and $g(x) = 0$ when $x = 0$. In all test datasets, the learned sparse regularizer functions are nonnegative and monotonically 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 $x = 0$. Notice 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 III reports the clustering accuracy on all test datasets with various surrogate functions $g(x)$ for ACC, NMI and ARI. The baseline method records the performance of directly solving $f(\mathbf{W})$ without sparse regularizers. In order to provide a fair comparison for all defined sparse regularizers, both λ and γ range in $(0, 1.0]$. As for the ℓ_p-norm, the value p 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 achieves better performance than all compared manually designed $g(x)$. Table III also provides the sparsity of the learned $g(x)$, where sparsity is defined by 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 an affinity matrix evaluation. All methods still yield sparser outputs successfully in most datasets, and promote the sparsity significantly in all datasets except ALOI and Caltech101-7. Nonetheless, it can

[6]http://cam-orl.co.uk/facedatabase.html

[7]http://archive.ics.uci.edu/ml/datasets



(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 $g(x) = \int_0^x (\xi^{-1}_{(\theta_1,\theta_2)}(y) - y)dy$ in all test datasets for multi-view clustering. All learned parameters $\xi_{(\theta_1,\theta_2)}(x)$ in activation functions are listed under each subfigure, where the points $x = \pm w_1(b_2 - b_1)$ are marked in red.

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

TABLE III: Clustering 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 I, where the best performance is highlighted in bold.

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)	3.6 (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)**

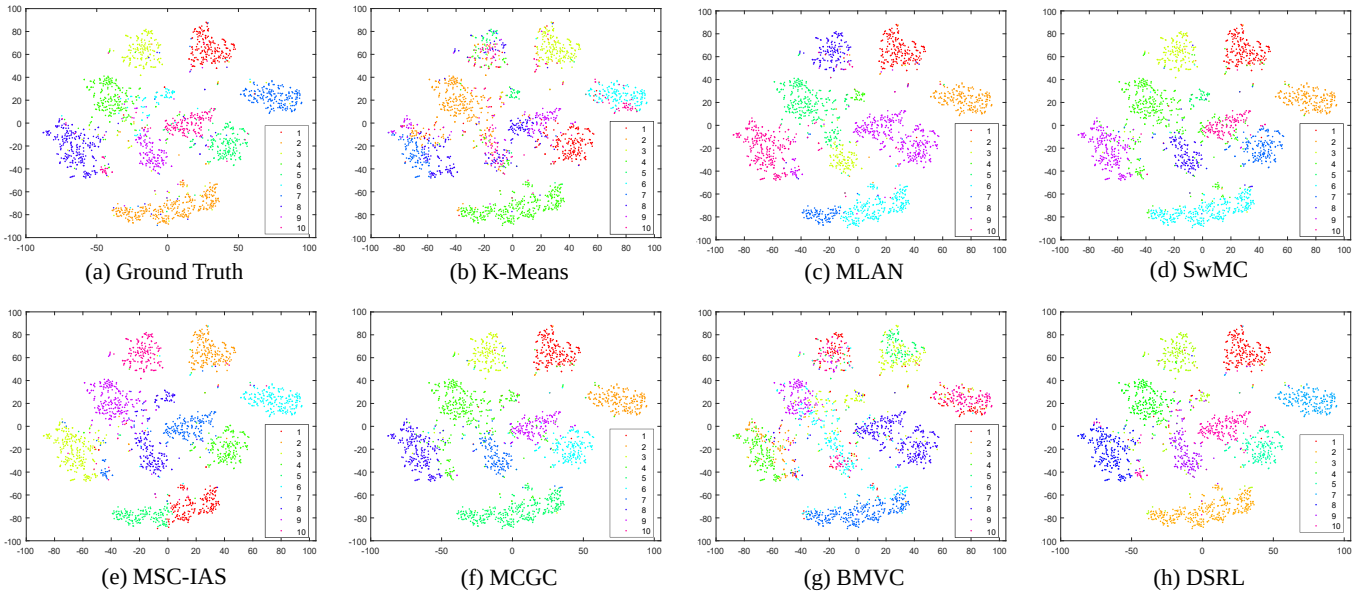
TABLE IV: Clustering accuracy (mean% and standard deviation%) of all compared multi-view clustering methods, where the best performance is highlighted in bold and the second best result is underlined.

be seen that excessive sparseness may lead to the performance decrease in clustering tasks, 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 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 which is tailored for given datasets. In other words, DSRL can learn a data-driven sparse regularizer, which intuitively exhibits strong generalization capability in practical applications.

Table IV compares the clustering performance of DSRL with several existing state-of-the-art methods in terms of ACC, NMI and ARI. An example of visualization for clustering results in dataset MNIST is demonstrated in Figure 5. It is observed that most of multi-view clustering methods achieve superior 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 by all metrics when tested on seven of eight test datasets. For dataset MNIST, the proposed model also achieves the second best performance by all evaluation metrics. Overall, these compared results validate the feasibility and superiority of the proposed method DSRL.

E. Multi-View Semi-Supervised Classification

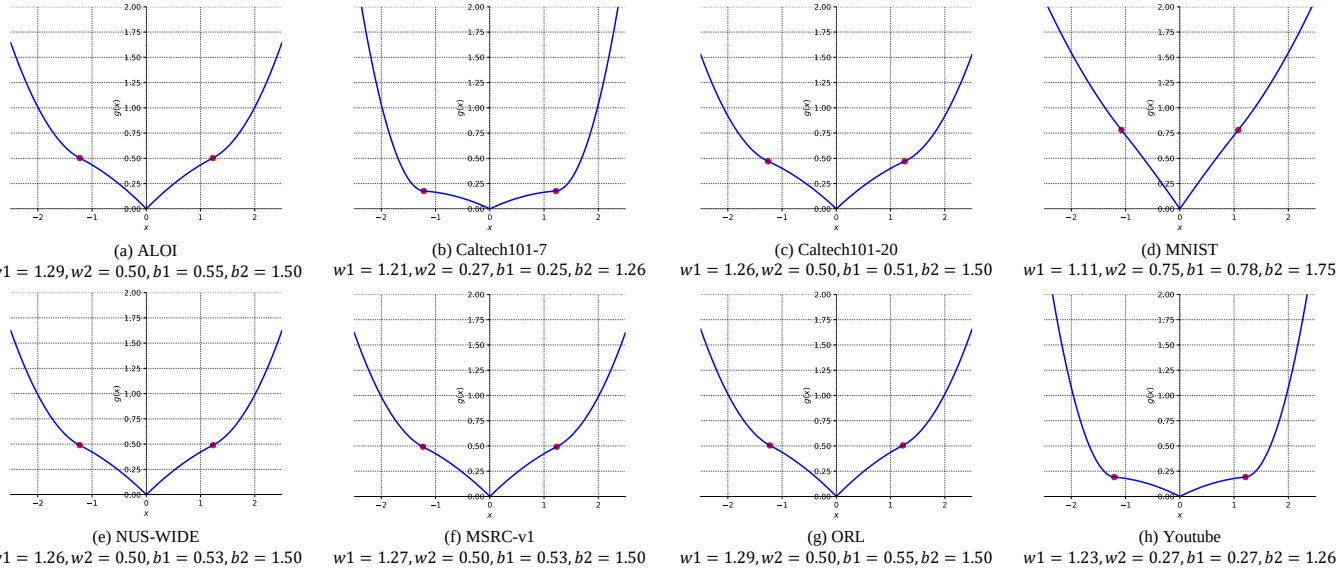
As an application to multi-view semi-supervised learning, we conduct experiments with 10% randomly generated labeled data. The learned sparse regularizers $g(x)$ for semi-supervised classification is illustrated in Figure 6. It can be seen from this figure that although the learned parameters differ in those of clustering scenarios, they share some common properties. We also compare the performance of DSRL with various defined sparse regularizers, as recorded in Table V. Different from 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 by classification accuracy. Although DSRL does not always generate the sparsest outputs, it achieves the best performance in all datasets with considerable sparseness, which partially suggests the importance and necessity of learning data-driven sparse regularizers. Moreover, it is 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 VI compares DSRL with other state-of-the-art semi-supervised classification methods, indicating that DSRL also outperforms compared semi-supervised classification methods in seven of eight test datasets. Further, we compare the performance of all methods as the ratio of labeled data ranges in $\{0.05, 0.10, \cdots, 0.80\}$ in Figure 7. Overall, DSRL performs best on all test datasets, and gains higher accuracy with very limited labeled data points. Desired performance of semi-supervised classification further validates the effectiveness of the proposed DSRL.



(a) Ground Truth (b) K-Means (c) MLAN (d) SwMC

(e) MSC-IAS (f) MCGC (g) BMVC (h) DSRL

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 of different predictive clustering labels are marked in varying colors.



(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 $g(x) = \int_0^x (\xi^{-1}_{(\theta_1,\theta_2)}(y) - y)dy$ in test datasets for multi-view semi-supervised classification. All learned parameters $\xi_{(\theta_1,\theta_2)}(x)$ in activation functions are listed under each subfigure, where the points $x = \pm w_1(b_2 - b_1)$ are marked in red.

F. Runtime Analyses

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

G. Parameter Sensitivity

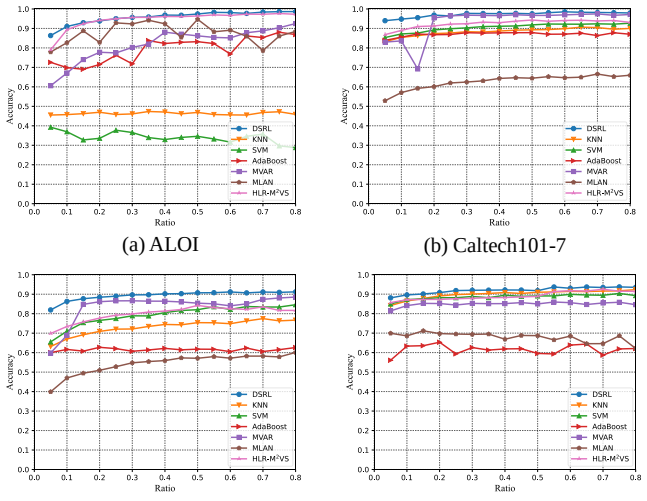
In this subsection, we examine the convergence and robustness of the proposed method DSRL. The loss values of the proposed method DSRL are demonstrated in Figure 9 as the number of iterations varies on different test datasets. It is observed from this figure that the loss objective value of DSRL gradually decreases as the number of iterations increases. Eventually, it will converge to a stable value and

Datasets \ Methods		Baseline	ℓ_p-norm	Logarithm	Geman	Laplace	ETP	DSRL
ALOI	ACC	59.9 (2.6)	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 (1.7)	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	33.8 (1.1)	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	54.9 (0.9)	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	17.9 (1.3)	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	47.1 (5.7)	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	35.6 (3.3)	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	21.2 (1.6)	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 V: 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 I, where the best performance is highlighted in bold.

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)	87.7 (1.7)	**91.6 (1.1)**
Caltech101-7	85.4 (0.8)	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)	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)	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)	44.3 (2.4)
MSRC-v1	55.7 (5.9)	58.9 (5.9)	33.5 (5.2)	49.3 (5.0)	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)	52.0 (4.4)	**54.9 (3.8)**
Youtube	35.9 (1.5)	42.8 (1.0)	27.4 (4.7)	37.6 (1.8)	36.4 (1.0)	32.7 (1.1)	**48.0 (1.1)**

TABLE VI: 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 result is underlined.



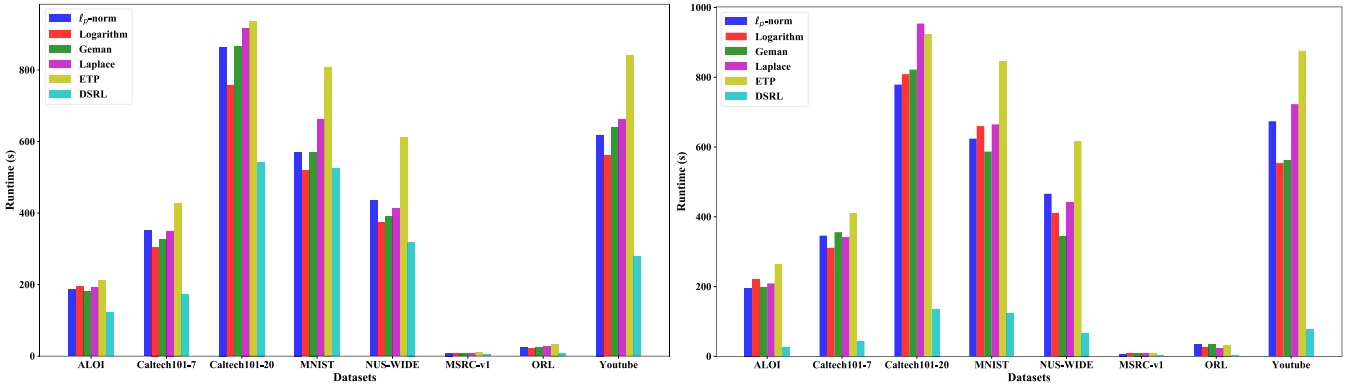
(a) ALOI (b) Caltech101-7

(c) Caltech101-20 (d) MNIST

Fig. 7: The varied performance of all compared methods in semi-supervised classification tasks as the ratio of labeled data ranges in $\{0.05, 0.10, \cdots, 0.80\}$.

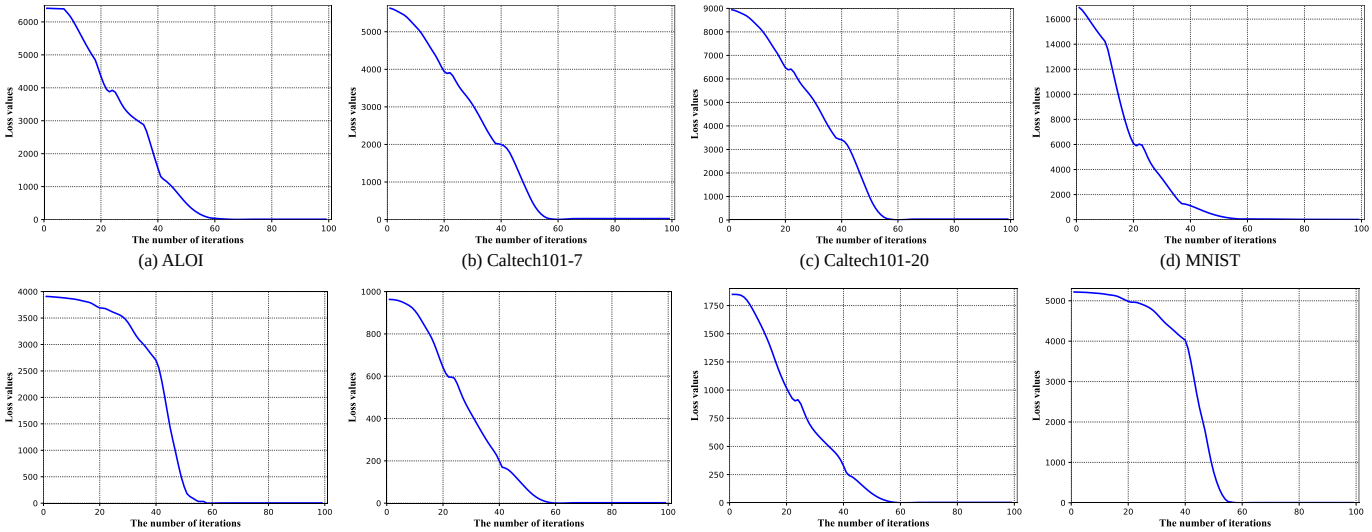
fluctuate lightly when the iteration number is large enough, which partially suggests its convergence.

The performance of DSRL for clustering tasks is reported in Figure 10 with ACC, NMI and ARI as the number of blocks varies. For all figures, the block number ranges in $\{2, 4, \cdots, 24\}$, and the learning rate lr is fixed as 0.02. From Figure 10, we have several beneficial observations. First of all, a small block number t leads to an acceptable result, which indicates that we can set a smaller block number to speed up the computation. Second, the performance on three metrics increases with more blocks for most datasets, and is stable and slightly fluctuates when the block number $t > 10$. This is an underlying empirical explanation that we set $t = 10$ in previous experiments to obtain acceptable results. Third, the influence of the block number is not significant in some datasets (ALOI, Caltech101-7, NUS-WIDE and Youtube), but overall the experimental results follow our previous observations. We also demonstrate the influence of block numbers for multi-view semi-supervised classification in Figure 11. The performance is more robust to the block number for semi-supervised classification, and the experiments further validate our previous analyses.



(a) (b)

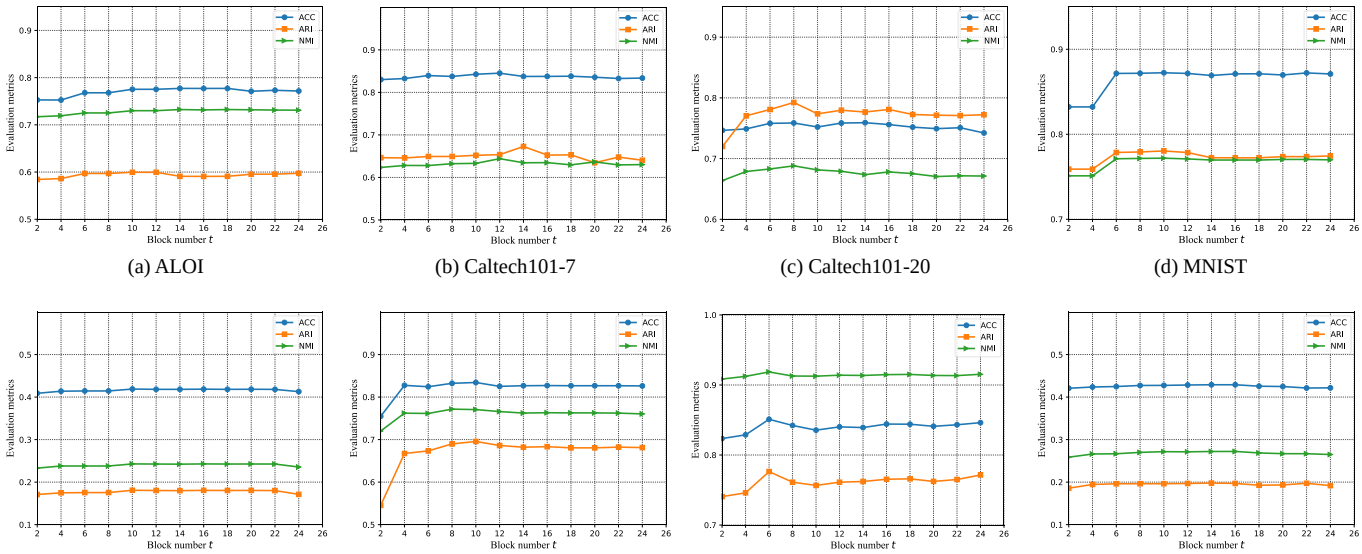
Fig. 8: Runtime comparison for all sparse surrogates and DSRL in (a) clustering and (b) semi-supervised classification.



(a) ALOI (b) Caltech101-7 (c) Caltech101-20 (d) MNIST

(e) NUS-WIDE (f) MSRC-v1 (g) ORL (h) Youtube

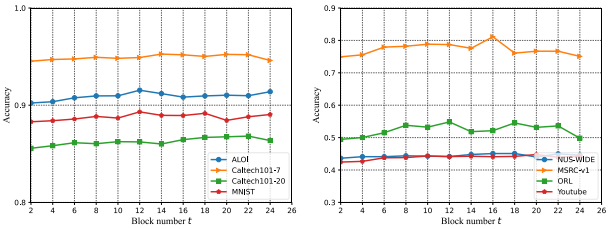
Fig. 9: The convergence curves of the proposed method DSRL on all test datasets.



(a) ALOI (b) Caltech101-7 (c) Caltech101-20 (d) MNIST

(e) NUS-WIDE (f) MSRC-v1 (g) ORL (h) Youtube

Fig. 10: The relations among clustering performance (ACC, ARI and NMI) and block number in $\{2, 4, \cdots, 24\}$ of the proposed method DSRL.



(a) (b)

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

V. Conclusions and Future Work

In this paper, we proposed an effective neural network model DSRL for learning data-driven sparse regularizers adaptively, which was a block-wise deep neural network with learnable activation functions. In this model, we exploited a correspondence between sparse regularizers and parameterized activation functions via proximal operators, where sparse regularizers could be obtained from an integration of activation functions. This provided a solid theoretical justification for DSRL. The proposed DSRL succeeded in solving optimization problems with adaptive sparse regularizers, which was not limited to hand-crafted sparse weights or outputs. Finally, we compared the proposed DSRL with hand-crafted sparse regularizers in eight real-world multi-view datasets and achieved superior performance in terms of multi-view clustering and semi-supervised classification. It 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 the future work, we will further explore learnable multivariate sparse regularizers.

References

[1] Y. Li, C. Chen, F. Yang, and J. Huang, "Hierarchical sparse representation for robust image registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 9, pp. 2151–2164, 2018.

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

[3] D. Zou, X. Chen, G. Cao, and X. Wang, "Unsupervised video matting via sparse and low-rank representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 6, pp. 1501–1514, 2020.

[4] H. Gao and H. Huang, "Stochastic second-order method for large-scale nonconvex sparse learning models," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pp. 2128–2134, 2018.

[5] F. Sun, J. Guo, Y. Lan, J. Xu, and X. Cheng, "Sparse word embeddings using l1 regularized online learning," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pp. 2915–2921, AAAI Press, 2016.

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

[7] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proceedings of the Twenty-Seventh International Conference on Machine Learning*, pp. 399–406, 2010.

[8] J. Zhang and B. Ghanem, "Ista-net: Interpretable optimization-inspired deep network for image compressive sensing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1828–1837, 2018.

[9] Z. Wang, Q. Ling, and T. S. Huang, "Learning deep ℓ_0 encoders," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pp. 2194–2200, 2016.

[10] P. Sprechmann, A. M. Bronstein, and G. Sapiro, "Learning efficient sparse and low rank models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1821–1833, 2015.

[11] G. Tanaka, R. Nakane, T. Takeuchi, T. Yamane, D. Nakano, Y. Katayama, and A. Hirose, "Spatially arranged sparse recurrent neural networks for energy efficient associative memory," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 1, pp. 24–38, 2020.

[12] W. Luo, W. Liu, and S. Gao, "A revisit of sparse coding based anomaly detection in stacked rnn framework," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 341–349, 2017.

[13] A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, and J. Z. Kolter, "Differentiable convex optimization layers," in *Advances in Neural Information Processing Systems*, pp. 9558–9570, 2019.

[14] H. Yang, W. Wen, and H. Li, "Deephoyer: Learning sparser neural network with differentiable scale-invariant sparsity measures," in *Proceedings of the Eighth International Conference on Learning 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 *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pp. 12087–12094, 2020.

[16] J. Sun, H. Li, Z. Xu, *et al.*, "Deep admm-net for compressive sensing mri," in *Advances in Neural Information Processing Systems*, pp. 10–18, 2016.

[17] X. Xie, J. Wu, G. Liu, Z. Zhong, and Z. Lin, "Differentiable linearized ADMM," in *Proceedings of the Thirty-Sixth International Conference on Machine Learning*, pp. 6902–6911, 2019.

[18] L. Bertinetto, J. F. Henriques, P. H. S. Torr, and A. Vedaldi, "Meta-learning with differentiable closed-form solvers," in *Proceedings of the Seventh International Conference on Learning Representations*, 2019.

[19] A. Bibi, B. Ghanem, V. Koltun, and R. Ranftl, "Deep layers as stochastic solvers," in *Proceedings of the Seventh International Conference on Learning Representations*, 2019.

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

[21] J. Li, C. Fang, and Z. Lin, "Lifted proximal operator machines," in *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, pp. 4181–4188, 2019.

[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," *International Journal of Forecasting*, vol. 28, no. 3, pp. 722–738, 2012.

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

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

[26] C. Gao, N. Wang, Q. R. Yu, and Z. Zhang, "A feasible nonconvex relaxation approach to feature selection," in *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*.

[27] C. Lu, C. Zhu, C. Xu, S. Yan, and Z. Lin, "Generalized singular value thresholding," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 1805–1811, 2015.

[28] M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien, "ℓ_p-norm multiple kernel learning," *Journal of Machine Learning Research*, vol. 12, no. Mar, pp. 953–997, 2011.

[29] J. Trzasko and A. Manduca, "Highly undersampled magnetic resonance image reconstruction via homotopic ℓ_0-minimization," *IEEE Transactions on Medical Imaging*, vol. 28, no. 1, pp. 106–121, 2008.

[30] C. Lu, J. Tang, S. Yan, and Z. Lin, "Generalized nonconvex nonsmooth low-rank minimization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4130–4137, 2014.

[31] 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 Transactions on Neural Networks and Learning Systems*, vol. 30, no. 10, pp. 2916–2925, 2019.

[32] C. Dan, H. Wang, H. Zhang, Y. Zhou, and P. Ravikumar, "Optimal analysis of subset-selection based l_p low-rank approximation," in *Proceedings of the 33rd Conference on Neural Information Processing Systems*, pp. 2537–2548, 2019.

[33] V. Papyan, Y. Romano, and M. Elad, "Convolutional neural networks analyzed via convolutional sparse coding," *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 2887–2938, 2017.

[34] S. Liu, "Learning sparse neural networks for better generalization," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pp. 5190–5191, 2020.

[35] W. Luo, J. Li, J. Yang, W. Xu, and J. Zhang, "Convolutional sparse autoencoders for image classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 7, pp. 3289–3294, 2018.

[36] E. Tartaglione, S. Lepsøy, A. Fiandrotti, and G. Francini, "Learning sparse neural networks via sensitivity-driven regularization," in *Advances in Neural Information Processing Systems*, pp. 3878–3888, 2018.

[37] X. Liu, W. Li, J. Huo, L. Yao, and Y. Gao, "Layerwise sparse coding for pruned deep neural networks with extreme compression ratio," in *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pp. 4900–4907, 2020.

[38] A. Bhowmik, A. Adiga, C. Seelamantula, F. Hauser, J. Jacak, and B. Heise, "Bayesian deep deconvolutional neural networks," in *the Second Neural Information Processing Systems Workshop on Bayesian Deep Learning*, 2017.

[39] S. Wang, S. Fidler, and R. Urtasun, "Proximal deep structured models," in *Advances in Neural Information Processing Systems*, pp. 865–873, 2016.

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

[41] S. Srinivas, A. Subramanya, and R. Venkatesh Babu, "Training sparse neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 138–145, 2017.

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

[43] F. Nie, J. Li, and X. Li, "Self-weighted multiview clustering with multiple graphs," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pp. 2564–2570, 2017.

[44] F. Nie, J. Li, and X. Li, "Parameter-free auto-weighted multiple graph learning: A framework for multiview clustering and semi-supervised classification," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pp. 1881–1887, 2016.

[45] F. Nie, G. Cai, and X. Li, "Multi-view clustering and semi-supervised classification with adaptive neighbours," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pp. 2408–2414, 2017.

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

[47] K. Zhan, F. Nie, J. Wang, and Y. Yang, "Multiview consensus graph clustering," *IEEE Transactions on Image Processing*, vol. 28, no. 3, pp. 1261–1270, 2019.

[48] Z. Zhang, L. Liu, F. Shen, H. T. Shen, and L. Shao, "Binary multi-view clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 7, pp. 1774–1782, 2019.

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

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

Shiping Wang received his Ph.D. degree from the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China in 2014. He worked as a research fellow in Nanyang Technological University from August 2015 to August 2016. He is currently a Full Professor and Qishan Scholar with 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 his B.S. degree from the College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China in 2019. He is currently pursuing the M.S. degree with the College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China. His current research interests include machine learning, deep learning and recommender systems.

Shide Du received his B.S. degree from the College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China in 2019. He is currently pursuing the M.S. degree with the College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China. 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 with the Key Laboratory of Machine Perception, School of EECS, Peking University. His research interests include computer vision, image processing, machine learning, pattern recognition, and numerical optimization. He is an area chair of CVPR 2014/2016/2019/2020, ICCV 2015, NIPS 2015/2018/2019, and AAAI 2019/2020, and a senior program committee member of AAAI 2016/2017/2018 and IJCAI 2016/2018. He is an associate editor of the IEEE Transactions on Pattern Analysis and Machine Intelligence and the International Journal of Computer Vision. He is a fellow of IAPR and IEEE.