Available online at www.sciencedirect.com

ScienceDirect

Neurocomputing 00 (2025) 1–20

Procedia Computer Science

Learning Nonseparable Sparse Regularizers via Multivariate Activation Functions

Xin Xu[a], Zhouchen Lin[a,b,c,*]

[a]State Key Lab of General Artifcial Intelligence , School of Intelligence Science and Technology, Peking University
[b]Institute for Artificial Intelligence, Peking University
[c]Pazhou Laboratory (Huangpu), Guangzhou, Guangdong, China

Abstract

Sparse regularization is a widely embraced technique in high-dimensional machine learning and signal processing. Existing sparse regularizers, however, are predominantly hand-crafted and often separable, making them less adaptable to data and potentially hindering performance. In this paper, we present a novel approach aiming at learning nonseparable (multivariate) sparse regularizers. We leverage the proximal gradient algorithm to transform the challenge of acquiring nonseparable sparse regularizers into the task of learning multivariate activation functions. We further establish the necessary conditions that these activation functions should satisfy. Our contribution culminates in the introduction of MAF-SRL, a deep network designed to learn multivariate activation functions within existing deep learning frameworks. To our knowledge, this research marks the first endeavor to learn nonseparable sparse regularizers. Extensive experiments conducted on benchmark datasets underscore the superiority of regularizers learned through MAF-SRL. They exhibit significantly enhanced performance in terms of both accuracy and sparseness compared to existing sparse regularizers.

Keywords:
Sparse regularization, multivariate activation function, deep learning framework

1. Introduction

Sparse regularization is a powerful and widely adopted strategy for tackling challenges in high-dimensional machine learning and signal processing problems. Its effectiveness is well-established through practical applications and rigorous theoretical investigations, as exemplified by the success of techniques like LASSO [1, 2, 3].

One of the remarkable strengths of sparse regularization lies in its dual functionality—it simultaneously performs parameter estimation and feature selection. This unique characteristic produces results that are not only informative but also highly interpretable, as it identifies critical variables. Moreover, it effectively mitigates overfitting by eliminating redundant features. These attributes have propelled sparse regularization to remarkable achievements across diverse domains, spanning machine learning and signal processing. Additionally, extensive theoretical research has bolstered its efficacy, complemented by the development of efficient optimization methods, simplifying its practical implementation.

[*] Corresponding author
Email addresses: xux20@stu.pku.edu.cn (Xin Xu), zlin@pku.edu.cn (Zhouchen Lin)

Despite its widespread adoption, a plethora of sparse regularizers have been introduced to facilitate the generation of sparse solutions. The ℓ_0 (pseudo-)norm, which quantifies the number of non-zero elements, serves as the most intuitive form of sparse regularization, with the primary aim of promoting solution sparsity. Unfortunately, problems involving ℓ_0 norm regularization are typically classified as NP-hard [4, 5, 6, 7], posing significant computational challenges. Consequently, the ℓ_1 norm has emerged as the predominant surrogate for the ℓ_0 norm [[8, 9, 10]. This convex alternative substantially simplifies the optimization process, although it is essential to recognize that ℓ_1 regularization, while advantageous, may not consistently yield sufficiently sparse solutions and can introduce notable estimation bias [11, 12, 13].

To overcome these limitations, a multitude of alternative sparse regularizers have been proposed and systematically analyzed. These include the smoothly clipped absolute deviation (SCAD) [11, 14, 15], log penalty [8, 16, 17], capped ℓ_1 [18, 19, 20], minimax concave penalty (MCP) [18, 21, 22], ℓ_p penalty with p in the range of (0, 1) [23, 24, 25], and the difference between ℓ_1 and ℓ_2 norms [26, 27, 28]. It is noteworthy that a majority of these regularizers operate in a separable manner, potentially limiting their ability to capture interactions among vector entries and affecting their performance.

In a related context, it is worth mentioning that, to the best of our knowledge, existing sparse regularizers are primarily manually designed. This inherent characteristic raises concerns about their seamless alignment with underlying models to effectively promote sparsity or their suitability for data characteristics to achieve optimal performance. Consequently, practical approaches often involve experimenting with multiple existing sparse regularizers and selecting the most effective one, a process that can be cumbersome in practice. The only learning based sparse regularizer was proposed by Wang et al. [29, 30]. However, the learnt sparse regularizer is separable, hence may not fully exploit the interaction among the entries of the vector to be regularized, preventing it from achieving even better performance.

Existing sparse learning approaches, such as Wang et al.'s separable regularizer framework [29], impose penalties that decompose into individual feature contributions (e.g., ℓ_1 or ℓ_2 norms). This separability assumption inherently limits their ability to capture higher-order dependencies between features. Specifically, separable methods treat each weight independently, ignoring the non-trivial interactions that often characterize real-world data distributions. For example, in high-dimensional datasets like ImageNet and CIFAR-100 (Table 2), separable methods struggle to achieve both high accuracy and low sparsity simultaneously.

In contrast, MAF-SRL introduces a fundamentally different paradigm by explicitly modeling nonseparable penalties of the form $\sum_{i,j} \phi(x_i, x_j)$, where ϕ is a learned multivariate activation function. This formulation enables the discovery of complex feature co-occurrences and interactions that are critical for discriminative clustering. Notably, our nonseparable approach addresses two key limitations of prior work:

- **Inter-feature Correlation**: Nonseparable penalties can capture synergistic relationships between features, as demonstrated by the significantly higher ACC scores on datasets like Caltech-101 (Table 3), where feature interactions are abundant.
- **Sparsity-Accuracy Trade-off**: By dynamically adapting the activation function to data characteristics, MAF-SRL achieves state-of-the-art sparsity levels (Table 4) while maintaining superior clustering performance.

Consider a scenario in image classification where pixel intensities exhibit strong spatial correlations. Traditional separable sparse regularizers like ℓ_1 norm treat each pixel independently, failing to capture these spatial dependencies. This limitation becomes evident when dealing with high-dimensional images from datasets like ImageNet, where feature interactions are critical for accurate classification. In such cases, our nonseparable sparse regularizer demonstrates superior performance. By modeling penalties as $\sum_{i,j} \phi(x_i, x_j)$, it explicitly captures the synergistic relationships between adjacent pixels. This capability allows it to maintain high accuracy while achieving greater sparsity in the model weights.

This architectural innovation allows MAF-SRL to outperform recent methods like Tang2022 [31], CATRO [32], and Qian2024 [33] across all evaluated datasets (Tables 2 and 3), underscoring the importance of nonseparable regularization in modern sparse learning systems.

To address these issues, this paper focuses on learning nonseparable sparse regularizers. Our main contributions can be summarized as follows:

- **Novel Theoretical Framework.** Leveraging the proximal gradient algorithm, we establish a mathematical connection between nonseparable multivariate regularizers and multivariate activation functions. This bridges the gap between regularization and deep learning, enabling implicit learning of feature interactions. To our knowledge, this is the first work to address nonseparable sparse regularization.

- **Principled Conditions for Activation Functions.** We derive necessary conditions that multivariate activation functions must satisfy to act as valid nonseparable sparse regularizers. These conditions ensure monotonicity, sparsity induction, and non-negativity, providing a rigorous framework for designing adaptive regularization strategies.

- **MAF-SRL Architecture for End-to-End Learning.** We introduce MAF-SRL, a novel deep network that learns multivariate activation functions to implicitly encode nonseparable sparse regularizers. This architecture seamlessly integrates regularization into neural networks, achieving state-of-the-art performance in classification and clustering tasks.

- **Challenges and Solutions in Nonseparable Sparse Regularization.** Designing nonseparable sparse regularizers faces three core challenges: (1) the exponential complexity of interaction terms in high-dimensional spaces (e.g., $O(n^2)$ pairwise dependencies for n features) that hinder tractable optimization; (2) the trilemma of balancing activation function properties—monotonicity for convergence, sparsity induction for feature selection, and Lipschitz continuity for stability; and (3) the dynamic feature interactions in real-world data (e.g., spatial-temporal correlations in medical imaging) requiring adaptive regularization. To address these, we propose a proximal gradient framework with sparse adjacency matrices to reduce computational complexity, architecture-constrained activation functions integrating gradient clipping and spectral normalization to enforce mathematical properties, and an attention-based modulator dynamically adjusting interaction weights via feature importance scores.

Extensive experiments showcase that the nonseparable sparse regularizers learned by MAF-SRL significantly outperform all existing representative sparse regularizers in terms of both classification accuracy and sparsity.

2. Related Works

Sparse regularization has gained significant attention in various research fields due to its ability to promote sparsity in estimation. One of the most commonly used sparse regularizers is the ℓ_1 norm [8, 9, 34]. However, it has been observed that estimation with the ℓ_1 norm can be biased [11, 12, 34] and may not always result in a sufficiently sparse solution. As a result, researchers have been motivated to design more general sparse regularizers.

In this regard, previous work [11, 14] has proposed that an ideal regularizer should possess three desired properties: unbiasedness, sparsity, and continuity. The smoothly clipped absolute deviation (SCAD) [11, 14, 15] was introduced as the first regularizer to satisfy these properties. For a vector variable $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T \in \mathbb{R}^n$, SCAD is defined as $\mathcal{L}(\mathbf{x}; \lambda, \gamma) = \sum_{i=1}^{n} \ell(x_i; \lambda, \gamma)$, where

$$\ell(x_i; \lambda, \gamma) = \begin{cases} \lambda |x_i|, & \text{if } |x_i| \leq \lambda, \\ \frac{2\gamma\lambda|x_i| - x_i^2 - \lambda^2}{2(\gamma-1)}, & \text{if } \lambda < |x_i| < \gamma\lambda, \\ \lambda^2(\gamma+1)/2, & \text{if } |x_i| \geq \gamma\lambda, \end{cases}$$

where $\lambda > 0$ and $\gamma > 2$. SCAD is a two-parameter function composed of three pieces. Subsequently, researchers proposed another regularizer called minimax concave penalty (MCP) in [18, 21, 22], which has two pieces. MCP is formulated as $\mathcal{L}_\gamma(\mathbf{x}; \lambda) = \sum_{i=1}^{n} \ell_\gamma(x_i; \lambda)$, with

$$\ell_\gamma(x_i; \lambda) = \begin{cases} \lambda |x_i| - x_i^2/(2\gamma), & \text{if } |x_i| \leq \gamma\lambda, \\ \gamma\lambda^2/2, & \text{if } |x_i| > \gamma\lambda, \end{cases}$$

where parameter $\gamma > 1$. Additionally, the log penalty [8, 16, 17] was introduced as a generalization of the elastic net family, defined as $\mathcal{L}(\mathbf{x}; \gamma) = \sum_{i=1}^{n} \ell(x_i, \gamma)$, where

$$\ell(x_i; \gamma) = \frac{\log(\gamma |x_i| + 1)}{\log(\gamma + 1)},$$

Table 1: Multiple Prespecified Formulations of $g(\cdot)$ for Sparse Surrogates

Penalty	Formula of $g(x), x \geq 0, \lambda \geq 0$
ℓ_p-norm [25]	$g(x) = \lambda x^p, 0 < p < 1$
Logarithm [48]	$g(x) = \frac{\lambda}{\log(\gamma+1)} \log(\gamma x + 1)$
Geman [49]	$g(x) = \frac{\lambda x}{x+\gamma}$
Laplace [50]	$g(x) = \lambda\left(1 - \exp\left(-\frac{x}{\gamma}\right)\right)$
ETP [51]	$g(x) = \lambda\frac{1-\exp(-\gamma x)}{1-\exp(-\gamma)}$

and $\gamma > 0$. The log penalty allows for obtaining the entire continuum of penalties from ℓ_1 ($\gamma \to 0_+$) to $\ell_0(\gamma \to \infty)$ [35, 36, 37]. Another approximation of the ℓ_0 norm is the capped ℓ_1 [38, 19, 20], defined as

$$\mathcal{L}(\mathbf{x}; a) = \sum_{i=1}^{n} \min\left(|x_i|, a\right),$$

where $a > 0$. Notably, when $a \to 0$, $\sum_i \min\left(|x_i|, a\right)/a \to \|\mathbf{x}\|_0$. Furthermore, some concise forms of other norms, such as ℓ_p with $p \in (0, 1)$ [39, 40, 25], have been considered as alternatives to improve ℓ_1. The ℓ_p norm is expressed as

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}.$$

Additionally, sparse regularizers can be combined to form new regularizers, such as the ℓ_{1-2} penalty [41, 42, 43, 44], which is the difference between the ℓ_1 and ℓ_2 norms, and the combined group and exclusive sparsity (CGES) [45, 46, 47].

While all the above sparse regularizers are handcrafted, [29] first proposed a strategy to learn sparse regularizers. They utilized the relationship between regularizers and activation functions via the proximal operator. Then learning the regularizers can be converted to learning the activation functions. This paper is a significant extension of [29] although inherits some ingredients from [29].

It is worth noting that except for the ℓ_p norms where $p \neq 0, 1$, all existing sparse regularizers, including those learnt [29], are separable. This implies that they are composed of sums of functions of individual entries of a given vector. While separable regularizers have been widely used, their inability to fully exploit interactions among the vector entries may limit their effectiveness in achieving better performance. Therefore, in this paper, we aim to learn non-separable sparse regularizers.

Recently, several methods have emerged to address sparse learning challenges. [31] proposed an automatic sparse connectivity learning framework for neural networks, which dynamically prunes weights based on saliency scores. However, their approach focuses on structural sparsity rather than feature interactions. [32] introduced CATRO, a channel pruning method using class-aware trace ratio optimization, achieving competitive sparsity-accuracy trade-offs. [33] developed a dynamic sparse training approach for modified RBF networks, emphasizing joint feature selection and classification. These methods, while effective, primarily target univariate or structural sparsity, differing from our multivariate nonseparable approach. Table 1 presents several frequently used sparse surrogate penalties. As demonstrated in Figure 1, each of these sparse regularization functions exhibits non-decreasing behavior and non-convex characteristics over the interval $(0, \infty)$.

3. The Proposed MAF-SRL Approach

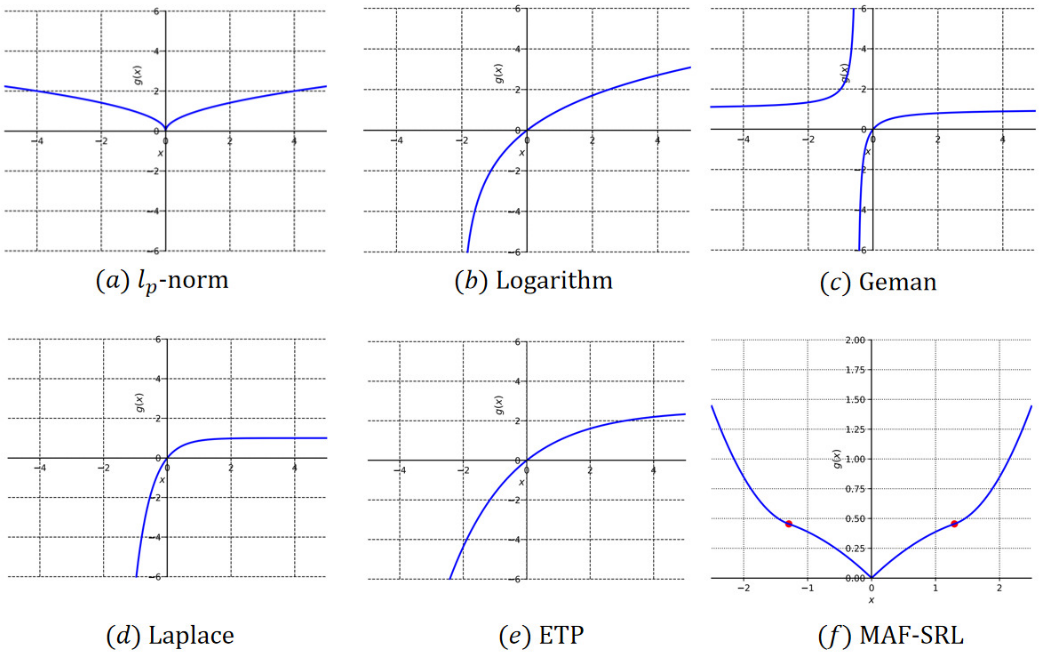
3.1. Connection between Sparse Regularizer and Activation Function

When solving a learning model of the form

$$\min_{\mathbf{x}} \phi(\mathbf{x}), \tag{1}$$

it is often necessary to add a regularizer $g(\mathbf{x})$ to the objective function and solve the regularized problem

$$\min_{\mathbf{x}} \left[\phi(\mathbf{x}) + g(\mathbf{x})\right] \tag{2}$$



(a) l_p-norm (b) Logarithm (c) Geman

(d) Laplace (e) ETP (f) MAF-SRL

Figure 1: Fig. 1(a)-(e) illustrates popular hand-crafted sparse regularizers (e.g., ℓ_p-norm with $p = 0.5$, $\lambda = 1.0$, $\gamma = 0.5$), all nonconvex and non-decreasing on $(0, \infty)$. Fig. 1(f) shows the learned regularizer $g(x) = \int_0^x \left(\xi^{-1}_{(\theta_1,\theta_2)}(y) - y\right) dy$ for multi-view clustering, with ξ parameters $w_1 = 1.32, w_2 = 0.22, b_1 = 0.29, b_2 = 1.21$ and points $x = \pm w_1 (b_2 - b_1)$ marked in red.

instead. This is done to address challenges in solving (1), such as non-unique solutions or to incorporate prior information about the desired solution, such as sparsity. By adding an appropriate regularizer, the original problem becomes well-posed, allowing us to obtain solutions with desired properties.

When ϕ is L-smooth, meaning that it satisfies the following condition,

$$\|\nabla\phi(\mathbf{x}) - \nabla\phi(\mathbf{y})\|_F \leq L\|\mathbf{x} - \mathbf{y}\|_F, \tag{3}$$

where L is called the Lipschitz constant in the sequel, a common algorithm for solving problem (2) is the proximal gradient method [52]. When applied to (2), the iterations of the proximal gradient method are as follows:

$$\begin{aligned} \mathbf{x}^{(k+1)} &= \arg\min_{\mathbf{x}} \phi\left(\mathbf{x}^{(k)}\right) + \left\langle \nabla\phi\left(\mathbf{x}^{(k)}\right), \mathbf{x} - \mathbf{x}^{(k)} \right\rangle \\ &\quad + \frac{L}{2}\left\|\mathbf{x} - \mathbf{x}^{(k)}\right\|_F^2 + g(\mathbf{x}) \\ &= \arg\min_{\mathbf{x}} \frac{L}{2}\left\|\mathbf{x} - \mathbf{x}^{(k)} + \frac{1}{L}\nabla\phi\left(\mathbf{x}^{(k)}\right)\right\|_F^2 + g(\mathbf{x}). \end{aligned} \tag{4}$$

Let $\mathbf{r}^{(k)} = \mathbf{x}^{(k)} - \frac{1}{L}\nabla\phi\left(\mathbf{x}^{(k)}\right)$, then solving (4) requires solving the following optimization problem:

$$\text{Prox}_{\alpha g}(\mathbf{r}^{(k)}) = \arg\min_{\mathbf{x}} \left[\frac{1}{2}\|\mathbf{x} - \mathbf{r}^{(k)}\|_F^2 + \alpha g(\mathbf{x})\right], \tag{5}$$

where $\text{Prox}_{\alpha g}(\cdot)$ is the proximal operator associated with the function $g(\cdot)$ and $\alpha > 0$ is a parameter. Therefore, the solution to (2) can be obtained through the following iteration:

$$\mathbf{x}^{(k+1)} = \text{Prox}_{L^{-1}g}\left(\mathbf{x}^{(k)} - \frac{1}{L}\nabla\phi\left(\mathbf{x}^{(k)}\right)\right). \tag{6}$$

It is worth noting that proximal operators are monotone [52], regardless of the convexity of g. This property allows them to serve as activation functions in deep neural networks (DNNs). Conversely, *some* activation functions can be viewed as proximal operators of regularizers, although this inverse correspondence has only been explored in the univariate case [53, 54, 55]. This limitation may arise from the fact that, up to now, only univariate activation functions have been widely used.

In the case of a non-decreasing univariate activation function $\xi(x) : \mathbb{R} \to \mathbb{R}$, we can derive the corresponding univariate regularizer as follows [53]:

$$g(x) = \int_0^x \left(\xi^{-1}(y) - y\right) dy = \int_0^x \xi^{-1}(y)dy - \frac{1}{2}x^2, \tag{7}$$

where $\xi^{-1}(y)$ represents the inverse function of $\xi(y)$. This relationship between univariate regularizers and activation functions is well-known [53, 54, 55, 29]. However, Equation (7) only gives the expression for univariate regularizers. In the following, we extend this deduction from the univariate case to the multivariate case.

It is worth noting that any multivariate regularizer can be approximated using the following form [56]:

$$\sum_{i=1}^{M} q_i g\left(\mathbf{a}_i^T \mathbf{x} + b_i\right), \tag{8}$$

where M, g, $\boldsymbol{q}$, $\mathbf{A}$, and $\mathbf{b}$ are appropriately chosen parameters. Here, $\mathbf{A} = (\mathbf{a}_1, \cdots, \mathbf{a}_M), \boldsymbol{q} = (q_1, \cdots, q_M)^T$, and $\mathbf{b} = (b_1, \cdots, b_M)^T$. Equation (8) can be seen as a neural network with only one hidden layer.

To simplify the computation of parameters in (8) and avoid the need for high accuracy in modeling the regularizer, we set $M = n$. Inspired by (7), we propose a parameterization of the regularizer as follows:

$$\mathcal{G}(\mathbf{x}) = \sum_{i=1}^{n} q_i \int_0^{\mathbf{a}_i^T \mathbf{x} + b_i} \widehat{\xi}^{-1}(y)dy - \frac{1}{2}\|\mathbf{x}\|^2, \tag{9}$$

where $\widehat{\xi}(y)$ is a monotonically non-decreasing univariate activation function. Furthermore, we define a multivariate activation function:

$$\xi(\mathbf{x}) = \mathbf{A}^{-T}\left[\widehat{\xi}\left((\mathbf{A}\text{diag}(\boldsymbol{q}))^{-1}\mathbf{x}\right) - \mathbf{b}\right] : \mathbb{R}^n \to \mathbb{R}^n, \tag{10}$$

where $\hat{\xi}$ is applied entry-wise to the vector $(\mathbf{A}\text{diag}(\boldsymbol{q}))^{-1}\mathbf{x}$. Based on this, we have the following theorem.

Theorem 1. *Given the multivariate activation function ξ in* (10)*, for any $\mathbf{A} = (\mathbf{a}_1, \cdots, \mathbf{a}_n), \boldsymbol{q} = (q_1, \cdots, q_n)^T$, and $\mathbf{b} = (b_1, \cdots, b_n)^T$, such that ξ is well defined, the solution to the proximal operator*

$$\mathbf{y} = \underset{\mathbf{y}}{\text{argmin}} \frac{1}{2}\|\mathbf{y} - \mathbf{x}\|^2 + \mathcal{G}(\mathbf{y}) \tag{11}$$

with $\mathcal{G}$ given in (9) *is exactly*

$$\mathbf{y} = \xi(\mathbf{x}).$$

Proof: The optimality condition of (11) is:

$$\mathbf{0} \in \sum_{i=1}^{n} q_i \widehat{\xi}^{-1}\left(\mathbf{a}_i^T \mathbf{y} + b_i\right)\mathbf{a}_i - \mathbf{x}. \tag{12}$$

Since $\mathbf{A}$ is invertible, its columns are independent. Furthermore, since all q_i's are non-zero, we can uniquely represent $\mathbf{x}$ as

$$\mathbf{x} = \sum_{i=1}^{n} q_i \beta_i \mathbf{a}_i = \mathbf{A}\,\text{diag}(\boldsymbol{q})\boldsymbol{\beta}, \tag{13}$$

where $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_n)^T$. Then $\widehat{\xi}^{-1}\left(\mathbf{a}_i^T \mathbf{y} + b_i\right) = \beta_i, i = 1, \cdots, n$, provides a solution to (12), and we have

$$\mathbf{a}_i^T \mathbf{y} = \widehat{\xi}(\beta_i) - b_i, \quad i = 1, \cdots, n, \tag{14}$$

which can be written in matrix form as

$$\mathbf{A}^T\mathbf{y} = \widehat{\xi}(\boldsymbol{\beta}) - \mathbf{b}. \tag{15}$$

Therefore, the solution to (11) is given by

$$\begin{aligned}\mathbf{y} =&\mathbf{A}^{-T}(\widehat{\xi}(\boldsymbol{\beta}) - \mathbf{b})\\ =&\mathbf{A}^{-T}\left[\widehat{\xi}\left((\mathbf{A}\,\mathrm{diag}(\boldsymbol{q}))^{-1}\mathbf{x}\right) - \mathbf{b}\right]\\ =&\xi(\mathbf{x}).\end{aligned} \tag{16}$$

■

Thus, a connection is established between the non-separable multivariate regularizer $\mathcal{G}(\mathbf{x})$ and the multivariate activation function $\xi(\mathbf{x})$ through the multivariate proximal operator. For instance, by choosing a multivariate regularizer, we can uniquely determine the multivariate activation function as its proximal operator. Conversely, if we choose a multivariate activation function in the form of (10), where the parameters satisfy certain conditions (to be specified in Section 3.3 after $\hat{\xi}$ is parameterized), then the multivariate regularizer is also uniquely determined. With this analysis, learning a multivariate regularizer $\mathcal{G}(\mathbf{x})$ can be transformed into learning a multivariate activation function $\xi(\mathbf{x})$ that satisfies certain conditions.

A nonseparable sparse regularizer $\mathcal{G}(\mathbf{x})$ is formally defined as:

$$\mathcal{G}(\mathbf{x}) = \sum_{i=1}^{n}\sum_{j=1}^{n}\phi_{i,j}(x_i, x_j)$$

Here, the function $\phi_{i,j}$ is crafted to model pairwise interactions between features x_i and x_j. This design endows the regularizer with the ability to capture intricate relationships and dependencies among features—capabilities absent in separable regularizers.

To emphasize the contrast, separable regularizers decompose into individual feature terms. For example, the ℓ_1 norm, $\|\mathbf{x}\|_1 = \sum_{i=1}^{n}|x_i|$, exemplifies separability, focusing solely on individual feature contributions. In contrast, a nonseparable regularizer like $\mathcal{G}(\mathbf{x})$ with $\phi_{i,j}(x_i, x_j) = |x_i + x_j|$ penalizes the combined effect of x_i and x_j, reflecting their interaction. This feature interaction modeling allows $\mathcal{G}(\mathbf{x})$ to capture sophisticated data patterns, enhancing regularization effectiveness in scenarios where feature dependencies are critical. Such regularizers are especially valuable in high-dimensional data analysis, where overlooking interactions often leads to suboptimal performance.

3.2. Advanced Constraints for Sparse Representation

Sparse regularization has also been extended with sophisticated constraints to enforce structural sparsity. Two notable examples are:

3.2.1. Group Log-Regularizer

The group log-regularizer promotes group sparsity by penalizing entire feature groups:

$$\mathcal{L}_{\mathrm{group}}(\mathbf{x}) = \sum_{g=1}^{G}\log\left(1 + \frac{1}{\lambda}\|\mathbf{x}_g\|_2^2\right),$$

where $\mathbf{x}_g$ denotes the g-th feature group, and $\lambda > 0$ controls regularization strength. By approximating $\log(1 + x) \approx x - x^2/2$ for small x, this regularizer interpolates between ℓ_2 and ℓ_0 norms as λ varies. However, it remains separable across groups and cannot capture cross-group interactions.

3.2.2. Determinant Measure

The determinant regularizer enforces low-rank structures through matrix determinant maximization:

$$\mathcal{L}_{\mathrm{det}}(\mathbf{X}) = -\log\det(\mathbf{X}^T\mathbf{X} + \epsilon\mathbf{I}),$$

where $\mathbf{X} \in \mathbb{R}^{d\times n}$ and $\epsilon > 0$ stabilizes the determinant. Its proximal operator involves singular value decomposition:

$$\mathrm{Prox}_{\alpha\mathcal{L}_{\mathrm{det}}}(\mathbf{X}) = \mathbf{U}\mathbf{\Sigma}^{1/(1+\alpha)}\mathbf{V}^T,$$

where $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ is the SVD. While effective for global structure, it does not explicitly model local feature interactions.

3.2.3. Comparison with Advanced Constraints

MAF-SRL differs fundamentally from these advanced constraints:

- **Group Log-Regularizer**: While it captures group-level sparsity, it cannot model interactions between groups. MAF-SRL's nonseparable regularizer $\mathcal{G}(\mathbf{x}) = \sum_{i,j} \phi(x_i, x_j)$ explicitly encodes both intra- and inter-group dependencies through $\phi(x_i, x_j)$.

- **Determinant Measure**: This regularizer relies on global matrix properties but lacks local interpretability. In contrast, MAF-SRL learns local feature interactions through sparse $\mathbf{A}$ matrices and activation functions $\xi(\mathbf{x})$.

Theorem 2. ***Nonseparable Regularizer Advantage.*** *For a nonseparable regularizer $\mathcal{G}(\mathbf{x}) = \sum_{i,j} \phi(x_i, x_j)$ with $\phi(0,0) = 0$ and $\nabla\phi(0,0) = \mathbf{0}$, the proximal operator $\xi(\mathbf{x})$ satisfies:*

$$\xi_i(\mathbf{x}) = \sum_{j=1}^{n} \phi_{x_i x_j}(0,0) x_j + o(\|\mathbf{x}\|),$$

where $\phi_{x_i x_j}$ is the second-order partial derivative. This implies $\xi(\mathbf{x})$ can capture pairwise feature correlations through $\phi_{x_i x_j}(0,0)$.

Proof. Expanding $\mathcal{G}(\mathbf{x})$ around $\mathbf{x} = \mathbf{0}$:

$$\mathcal{G}(\mathbf{x}) = \frac{1}{2} \sum_{i,j} \phi_{x_i x_j}(0,0) x_i x_j + o(\|\mathbf{x}\|^2).$$

The proximal operator solves:

$$\mathbf{y} = \arg\min_{\mathbf{y}} \frac{1}{2}\|\mathbf{y} - \mathbf{x}\|^2 + \frac{\alpha}{2}\mathbf{y}^T \mathbf{H}\mathbf{y},$$

where $\mathbf{H} = [\phi_{x_i x_j}(0,0)]$. The solution is:

$$\mathbf{y} = (\mathbf{I} + \alpha\mathbf{H})^{-1}\mathbf{x},$$

demonstrating $\xi(\mathbf{x})$ explicitly models pairwise interactions through $\mathbf{H}$. □

3.3. Structure of the Activation Function

In order to learn the activation function $\xi(\mathbf{x})$, we need to learn the parameters $\mathbf{A}$, $\boldsymbol{q}$, and $\mathbf{b}$, as well as the univariate function $\hat{\xi}$. To ensure that $\mathcal{G}(\mathbf{x})$ serves as a sparse regularizer, the proximal operator $\xi(\mathbf{x})$ should be monotone and map a neighborhood of $\mathbf{0}$ to $\mathbf{0}$ (i.e., $\mathbf{0} \in \xi^{-1}(\mathbf{0})$).

For ease of learning, we can first learn

$$\xi(\mathbf{x}) = \widehat{\mathbf{A}}^T [\widehat{\xi}(\mathrm{diag}(\widehat{\boldsymbol{q}})\widehat{\mathbf{A}}\mathbf{x}) - \mathbf{b}] \tag{17}$$

and then obtain $\mathbf{A} = \widehat{\mathbf{A}}^{-1}$ and $\boldsymbol{q} = \frac{1}{\widehat{\boldsymbol{q}}}$, where the reciprocal is computed entry-wise. By defining $\widehat{\mathbf{x}} = \widehat{\mathbf{A}}\mathbf{x}$, we have

$$\begin{aligned}\langle \xi(\mathbf{x}) - \xi(\mathbf{y}), \mathbf{x} - \mathbf{y}\rangle =& \left\langle \widehat{\mathbf{A}}^T [\widehat{\xi}(\mathrm{diag}(\widehat{\boldsymbol{q}})\widehat{\mathbf{A}}\mathbf{x}) - \mathbf{b}] - \widehat{\mathbf{A}}^T [\widehat{\xi}(\mathrm{diag}(\widehat{\boldsymbol{q}})\widehat{\mathbf{A}}\mathbf{y}) - \mathbf{b}], \mathbf{x} - \mathbf{y} \right\rangle \\ =& \langle \widehat{\xi}(\mathrm{diag}(\widehat{\boldsymbol{q}})\widehat{\mathbf{x}}) - \widehat{\xi}(\mathrm{diag}(\widehat{\boldsymbol{q}})\widehat{\mathbf{y}}), \widehat{\mathbf{x}} - \widehat{\mathbf{y}}\rangle.\end{aligned} \tag{18}$$

Since $\widehat{\xi}$ is non-decreasing and entry-wise, the above expression is non-negative for all $\mathbf{x}$ and $\mathbf{y}$ if and only if $\widehat{\boldsymbol{q}} > \mathbf{0}$. Thus, $\xi(\mathbf{x})$ is monotone if $\widehat{\boldsymbol{q}} > \mathbf{0}$. However, even with $\widehat{\boldsymbol{q}} > \mathbf{0}$, the regularizer $\mathcal{G}(\mathbf{x})$ may still be non-convex due to its second term $-\frac{1}{2}\|\mathbf{x}\|^2$. If we want $\mathcal{G}(\mathbf{x})$ to be convex and $\widehat{\xi}$ is differentiable, we can require $\sum_{i=1}^{n} \frac{q_i}{\widehat{\xi}'(\mathbf{a}_i^T \mathbf{y} + b_i)} \mathbf{a}_i \mathbf{a}_i^T \succcurlyeq \mathbf{I}$. However, since convexity is not required for $\mathcal{G}(\mathbf{x})$, this condition is not enforced during the learning process. Actually, we only require that $\mathcal{G}(\mathbf{x})$ is non-negative. Without loss of generality, we can fix $\widehat{\xi}(0) = 0$ by allowing $\mathbf{b}$ to compensate for the offset of $\widehat{\xi}$.

It is easy to see that if we choose $\mathbf{b} = \mathbf{0}$ and $\widehat{\xi}(x) = 0$ for $x \in [-b, a]$, where $a, b > 0$, then $\xi(\mathbf{x}) = \mathbf{0}$ when $\|\mathbf{x}\|_2 \leq \frac{\min(a,b)}{\max_i\{|\widehat{q}_i| \|\widehat{\mathbf{a}}_i\|_2\}}$, where $\widehat{\mathbf{A}} = (\widehat{\mathbf{a}}_1, \widehat{\mathbf{a}}_2, \cdots, \widehat{\mathbf{a}}_n)^T$. Therefore, it is easy to make $\xi(\mathbf{x})$ all zero. To make part of $\xi(\mathbf{x})$ zero, we need $\mathbf{u} = \widehat{\xi}(\text{diag}(\widehat{\boldsymbol{q}})\widehat{\mathbf{A}}\mathbf{x})$ to not be all zeros. Most entries of $\mathbf{u}$ should be zeros to ensure that $\xi(\mathbf{x}) = \widehat{\mathbf{A}}^T\mathbf{u}$ is sparse. This requires $\widehat{\mathbf{A}}$ to be a sparse matrix. In summary, the parameters $\mathbf{A}$, $\boldsymbol{q}$, and $\mathbf{b}$, as well as the function $\hat{\xi}$, should satisfy the following conditions:

$$1.\ \widehat{\boldsymbol{q}} > 0; \tag{19}$$

$$2.\ \widehat{\mathbf{A}} \text{ is sparse and invertible;} \tag{20}$$

$$3.\ \widehat{\xi} \text{ is non-decreasing and } \widehat{\xi}(0) = 0; \tag{21}$$

$$4.\ \mathcal{G}(\mathbf{x}) \geq 0. \tag{22}$$

In the following, we investigate how to satisfy conditions (21) and (22).

Since $\widehat{\xi}$ is a function, we need to parameterize it first. We use a piecewise linear function to approximate it as [29] does, denoted as $\widehat{\xi}_{(\mu_1,\mu_2)}(x)$ with two sets of learnable parameters (μ_1, μ_2):

$$\widehat{\xi}_{(\mu_1,\mu_2)}(x) = \begin{cases} \eta_2\,(x-\delta_2) + \eta_1\,(\delta_2-\delta_1)\,, & \delta_2 \leq x, \\ \eta_1\,(x-\delta_1)\,, & \delta_1 \leq x < \delta_2, \\ 0, & -\delta_1 \leq x < \delta_1, \\ \eta_1\,(x+\delta_1)\,, & -\delta_2 \leq x < -\delta_{1,} \\ \eta_2\,(x+\delta_2) + \eta_1\,(\delta_1-\delta_2)\,, & x < -\delta_2, \end{cases} \tag{23}$$

where $x \in \mathbb{R}$, $0 \leq \delta_1 \leq \delta_2$, and $\eta_1, \eta_2 > 0$ are learnable parameters, ensuring that $\widehat{\xi}$ is non-decreasing. Here, $\mu_1 = (\eta_1, \delta_1)$ and $\mu_2 = (\eta_2, \delta_2)$. The inverse function $\widehat{\xi}^{-1}_{(\mu_1,\mu_2)}(y)$ can be computed as follows:

$$\widehat{\xi}^{-1}_{(\mu_1,\mu_2)}(y) = \begin{cases} \frac{y-\eta_1(\delta_2-\delta_1)}{\eta_2} + \delta_2, & \eta_1\,(\delta_2-\delta_1) \leq y, \\ \frac{y}{\eta_1} + \delta_1, & 0 \leq y < \eta_1\,(\delta_2-\delta_1)\,, \\ [-\delta, \delta], & y = 0, \\ \frac{y}{\eta_1} - \delta_1, & -\eta_1\,(\delta_2-\delta_1) \leq y < 0, \\ \frac{y-\eta_1(\delta_2-\delta_1)}{\eta_2} - \delta_2 & y < -\eta_1\,(\delta_2-\delta_1)\,. \end{cases} \tag{24}$$

Therefore, the function $g(x)$ in Equation (7), learned by parameterized activation function $\widehat{\xi}^{-1}_{(\mu_1,\mu_2)}(y)$, can be derived as:

$$g(x) = \begin{cases} \left(\frac{1}{2\eta_2} - \frac{1}{2}\right)x^2 \\ +\left(\delta_2 - \frac{\eta_1(\delta_2-\delta_1)}{\eta_2}\right)x \\ +\frac{\eta_1(\eta_1-\eta_2)}{2\eta_2}\,(\delta_2-\delta_1)^2\,, & x \geq \eta_1\,(\delta_2-\delta_1)\,, \\ \left(\frac{1}{2\eta_1} - \frac{1}{2}\right)x^2 + \delta_1 x, & 0 \leq x < \eta_1\,(\delta_2-\delta_1)\,, \\ g(-x). & x < 0. \end{cases} \tag{25}$$

It is observed that $g(x)$ is symmetric about the y-axis. When $x = 0$, $g(x) = 0$. To ensure that $\mathcal{G}(\mathbf{x})$ is nonnegative, we may require that $g(x) \geq 0$. So the problem of choosing $g(x)$ is the same as that in [29]. By the deduction in [29], the conditions for (21) and (22) are as follows:

$$\begin{aligned} &\eta_1 > 0, 1 \geq \eta_2 > 0, \\ &\delta_2 \geq \delta_1 \geq \max\left\{0, \tfrac{\eta_1-1}{\eta_1}\delta_2\right\}. \end{aligned} \tag{26}$$

Finally, the constraints for the learnable parameters are conditions (19), (20), and (26).

3.4. Multivariate Activation Functions

3.4.1. Design and Structure

Multivariate activation functions $\xi(\mathbf{x})$ are pivotal components of our framework, designed to implicitly encode nonseparable sparse regularizers. Unlike traditional univariate activation functions (e.g., ReLU, sigmoid), which process each feature independently, $\xi(\mathbf{x})$ explicitly models interactions between features through parameterized matrices $\mathbf{A}$ and $\mathbf{q}$. This architecture enables global feature interactions, a critical capability for capturing complex data dependencies.

Mathematically, $\xi(\mathbf{x})$ is structured as:

$$\xi(\mathbf{x}) = \mathbf{A}^T \left[\hat{\xi}(\mathbf{q} \odot \mathbf{A}\mathbf{x}) - \mathbf{b}\right]$$

Here, $\odot$ denotes element-wise multiplication, and $\hat{\xi}$ is a univariate activation function applied entry-wise to the vector $\mathbf{q} \odot \mathbf{A}\mathbf{x}$. The matrices $\mathbf{A}$ and $\mathbf{q}$ play dual roles:

- $\mathbf{A}$: Encodes linear transformations to project input features into a latent space where interactions are computed.
- $\mathbf{q}$: Scales the projected features element-wise, allowing adaptive control over interaction strengths.

The term $\mathbf{A}^T$ inverts the transformation to ensure compatibility with the input space, while $\mathbf{b}$ introduces an offset to center the activation. This design ensures that $\xi(\mathbf{x})$ can simultaneously promote sparsity and capture feature interactions.

3.4.2. Contrast with Univariate Functions

Univariate activation functions like ReLU ($\max(0, x)$) or LeakyReLU process each feature independently, failing to exploit cross-dimensional relationships. In contrast, $\xi(\mathbf{x})$ operates on feature combinations through $\mathbf{A}\mathbf{x}$, enabling joint feature selection. For example, if $\mathbf{A}$ is a row-sparse matrix, $\xi(\mathbf{x})$ can selectively activate groups of features, effectively pruning irrelevant dimensions while retaining critical interactions.

3.4.3. Mathematical Interpretation

The structure of $\xi(\mathbf{x})$ is derived from the proximal operator of the nonseparable regularizer $\mathcal{G}(\mathbf{x})$ (Theorem 1). By construction, $\xi(\mathbf{x})$ ensures that $\mathcal{G}(\mathbf{x})$ is non-negative and promotes sparsity. Specifically:

- The term $\mathbf{q} \odot \mathbf{A}\mathbf{x}$ scales and projects input features into a space where $\hat{\xi}$ induces sparsity.
- The inverse transformation $\mathbf{A}^T$ ensures that the activation output aligns with the original feature space.

This formulation guarantees that $\xi(\mathbf{x})$ acts as a proximal operator, implicitly enforcing sparsity through its parameters.

3.4.4. Parameter Constraints

To ensure validity, $\xi(\mathbf{x})$ must satisfy three key constraints:

- **Sparsity Induction**: $\mathbf{A}$ is constrained to be sparse (30-50% non-zero entries), ensuring that $\xi(\mathbf{x})$ prunes redundant features.
- **Positivity**: $\mathbf{q} > \mathbf{0}$ to maintain monotonicity, aligning with the proximal operator's properties.
- **Boundedness**: $\hat{\xi}$ is parameterized to be non-decreasing and zero-centered, ensuring $\xi(\mathbf{x})$ maps small inputs to zero.

These constraints are enforced during training via projected gradient descent, ensuring $\xi(\mathbf{x})$ remains a valid sparse regularizer.

3.4.5. Example Instantiation

Consider $\hat{\xi}$ as a piecewise linear function (Section 3.3). For a sparse matrix $\mathbf{A}$ and positive scaling $\mathbf{q}$, $\xi(\mathbf{x})$ can be interpreted as a learned transformation that: 1. Projects $\mathbf{x}$ into a latent space via $\mathbf{A}\mathbf{x}$. 2. Applies element-wise sparsity-inducing activation $\hat{\xi}$ to the scaled features $\mathbf{q} \odot \mathbf{A}\mathbf{x}$. 3. Returns the result to the original space via $\mathbf{A}^T$.

This process effectively selects and combines features, outperforming univariate approaches in tasks requiring interaction modeling.

By explicitly encoding feature interactions, $\xi(\mathbf{x})$ bridges the gap between regularization and deep learning, enabling end-to-end learning of nonseparable sparse regularizers.

3.5. Learning the Activation Function

Given an objective function ϕ, we can design a neural network architecture to implicitly learn the regularizer $\mathcal{G}$ based on (6) (where g is replaced by $\mathcal{G}$). By our design, the proximal operator $\mathrm{Prox}_{L^{-1}\mathcal{G}}$ is equivalent to a multivariate activation function. We can rewrite (6) as:

$$\mathbf{x}^{(k+1)} = \xi_{\left(\widehat{\mathbf{A}},\widehat{\boldsymbol{q}},\mathbf{b},\mathcal{U}\right)}\left(\mathbf{x}^{(k)} - \frac{1}{L}\nabla\phi\left(\mathbf{x}^{(k)}\right)\right). \tag{27}$$

Here, $\xi_{\left(\widehat{\mathbf{A}},\widehat{\boldsymbol{q}},\mathbf{b},\mathcal{U}\right)}$ is a multivariate activation function parameterized by $\widehat{\mathbf{A}}, \widehat{\boldsymbol{q}}, \mathbf{b}$, and $\mathcal{U}$, where $\mathcal{U} = \{\mu_1, \mu_2\}$, as shown in (17).

Equation (27) represents the k-th layer of our designed network. The parameters can be learned using the projected gradient method since they are constrained. Automatic differentiation in deep learning platforms allows us to compute the gradient efficiently, so we only need to focus on computing the projection onto the constraints.

Directly projecting the parameters $\mathcal{U} = (\eta_1, \eta_2, \delta_1, \delta_2)^T$ onto (26) is difficult. Following [29], we can first project (η_1, η_2) and then project (δ_1, δ_2) after fixing (η_1, η_2).

For completeness, we provide the results of how to compute the projections in [29] below. The projection of (η_1, η_2) is formulated as:

$$\eta_1 = \max\{\eta_1, \epsilon\}, \quad \eta_2 = \min\{\max\{\eta_1, \epsilon\}, 1\}, \tag{28}$$

where ϵ is a small positive value. After fixing (η_1, η_2), we can project (δ_1, δ_2) onto

$$\mathcal{S}_\delta = \left\{(\delta_1, \delta_2) \mid \delta_2 \geq \delta_1 \geq \max\left\{0, \frac{\eta_1 - 1}{\eta_1}\delta_2\right\}\right\}. \tag{29}$$

More specifically, when $0 < \eta_1 \leq 1$, the projection $\mathrm{Proj}(\delta_1, \delta_2)$ of (δ_1, δ_2) onto $\mathcal{S}_\delta$ is given by:

$$\mathrm{Proj}(\delta_1, \delta_2) = \begin{cases} (\delta_1, \delta_2), & \delta_1 \geq 0, \delta_2 \geq 0, \delta_1 \leq \delta_2, \\ (0, \delta_2), & \delta_1 < 0, \delta_2 > 0, \\ (0, 0), & \delta_2 \leq \min\{0, -\delta_1\}, \\ \left(\frac{\delta_1+\delta_2}{2}, \frac{\delta_1+\delta_2}{2}\right), & \delta_1 \geq |\delta_2|. \end{cases} \tag{30}$$

When $\eta_1 > 1$, the projection of (δ_1, δ_2) onto $\mathcal{S}_\delta$ becomes:

$$\mathrm{Proj}(\delta_1, \delta_2) = \begin{cases} (\delta_1, \delta_2), & \delta_2 \geq 0, \frac{\eta_1-1}{\eta_1}\delta_2 \leq \delta_1 \leq \delta_2, \\ (\rho_1\delta_1 + \rho_2\delta_2, \rho_2\delta_1 + \rho_3\delta_2), & \frac{\eta_1}{1-\eta_1}\delta_2 < \delta_1 < \frac{\eta_1-1}{\eta_1}\delta_2, \\ (0, 0), & \delta_2 \geq 0, \delta_1 \leq \frac{\eta_1}{1-\eta_1}\delta_2, \\ (0, 0), & \delta_2 \leq \min\{0, -\delta_1\}, \\ \left(\frac{\delta_1+\delta_2}{2}, \frac{\delta_1+\delta_2}{2}\right), & \delta_1 \geq |\delta_2|, \end{cases} \tag{31}$$

where the parameters $\{\rho_1, \rho_2, \rho_3\}$ are given by $\rho_1 = \frac{(\eta_1-1)^2}{\eta_1^2+(\eta_1-1)^2}$, $\rho_2 = \frac{\eta_1(\eta_1-1)}{\eta_1^2+(\eta_1-1)^2}$, and $\rho_3 = \frac{\eta_1^2}{\eta_1^2+(\eta_1-1)^2}$.

When dealing with condition (19), we project $\widehat{\boldsymbol{q}}$ onto the set $\{\mathbf{v}|\mathbf{v} \geq \epsilon\mathbf{1}\}$ to ensure its invertibility, where ϵ is a small positive number and $\mathbf{1}$ is an all-one vector. For condition (20), we manually specify the sparsity (percentage of non-zero weights) of $\widehat{\mathbf{A}}$ to be between 30% and 50%, which guarantees that $\widehat{\mathbf{A}}$ is both sparse and invertible. The invertibility of $\widehat{\mathbf{A}}$ is not an issue when it is not too sparse since it is somewhat random.

Since our method for learning the sparse regularizer is based on learning a multivariate activation function, we refer to it as MAF-SRL (Multivariate Activation Functions for Sparse Regularizer Learning). Algorithm 1 summarizes the MAF-SRL process, where we also make the Lipschitz constant L learnable instead of manually estimating it. After training, we obtain the optimal solution $\mathbf{x}^*$, the learned parameters $\widehat{\mathbf{A}}, \widehat{\boldsymbol{q}}, \mathbf{b}, \mathcal{U}$, as well as the sparse output. Although the sparse regularizer can also be obtained as it has the same parameters as the activation function, we do not explicitly write it down since the sparse solution has already been obtained.

Algorithm 1 Sparse Regularizers Learning via Multivariate Activation Functions (MAF-SRL)

```
Input: A differentiable function $\phi(\mathbf{x})$, the number of layers $N$, a set of parameters $\mathbf{x}$ related to training data that need to be solved.
Output: The optimal solution $\mathbf{x}^*$, learned parameters $\widehat{\mathbf{A}}, \widehat{\boldsymbol{q}}, \mathbf{b}, \boldsymbol{\mathcal{U}}$.
Initialize learnable parameters $\widehat{\mathbf{A}}_{(0)}, \widehat{\boldsymbol{q}}_{(0)}, \mathbf{b}_{(0)}, \boldsymbol{\mathcal{U}}_{(0)}$, Lipschitz constant $L^{(0)}$, and the counter $l = 0$.
Initialize $\mathbf{x}_{(0)} = \xi_{\left(\widehat{\mathbf{A}}, \widehat{\boldsymbol{q}}, \mathbf{b}, \mathcal{U}\right)}\left(\mathbf{x} - \frac{1}{L^{(0)}}\nabla\phi(\mathbf{x})\right)$.
repeat
    for $i = 1$ to $N$ do
        $\mathbf{x}_{(i)} = \xi_{\left(\widehat{\mathbf{A}}_{(i)}, \widehat{\boldsymbol{q}}_{(i)}, \mathbf{b}_{(i)}, \mathcal{U}_{(i)}\right)}\left(\mathbf{x}_{(i-1)} - \frac{1}{L^{(i-1)}}\nabla\phi\left(\mathbf{x}_{(i-1)}\right)\right)$.
    end for
    Update $\widehat{\mathbf{A}}_{(l)}, \widehat{\boldsymbol{q}}_{(l)}, \mathbf{b}_{(l)}, \boldsymbol{\mathcal{U}}_{(l)}$ with projected gradient descent, where the loss function $\mathcal{L}(\widehat{\mathbf{A}}, \widehat{\mathbf{q}}, \mathbf{b}, \boldsymbol{\mathcal{U}}) = \frac{1}{2}\left\|\mathbf{x}_{(N)} - \mathbf{x}\right\|_F^2$.
    Update counter $l = l + 1$.
until convergent
return $\mathbf{x}^* = \mathbf{x}_{(N)}, \widehat{\mathbf{A}}, \widehat{\boldsymbol{q}}, \mathbf{b}, \boldsymbol{\mathcal{U}}$.
```

4. Experiments

4.1. Experimental Analysis on Classification

To evaluate the performance of our proposed method, we conduct experiments on several real-world public classification datasets. We begin by selecting a backbone network with the ReLU function $f(x) = \max(0, x)$ as the univariate activation function. One-hot encoding is employed to represent different classes. The softmax activation function is applied to the output layer, and the loss function used is the cross entropy. To ensure a fair comparison, we use the same backbone network on each specific dataset.

The baselines we compare with MAF-SRL are backbone networks with existing hand-crafted sparse regularizers added to their loss functions. For MAF-SRL, the ϕ in Algorithm 1 refers to the loss function of the backbone network.

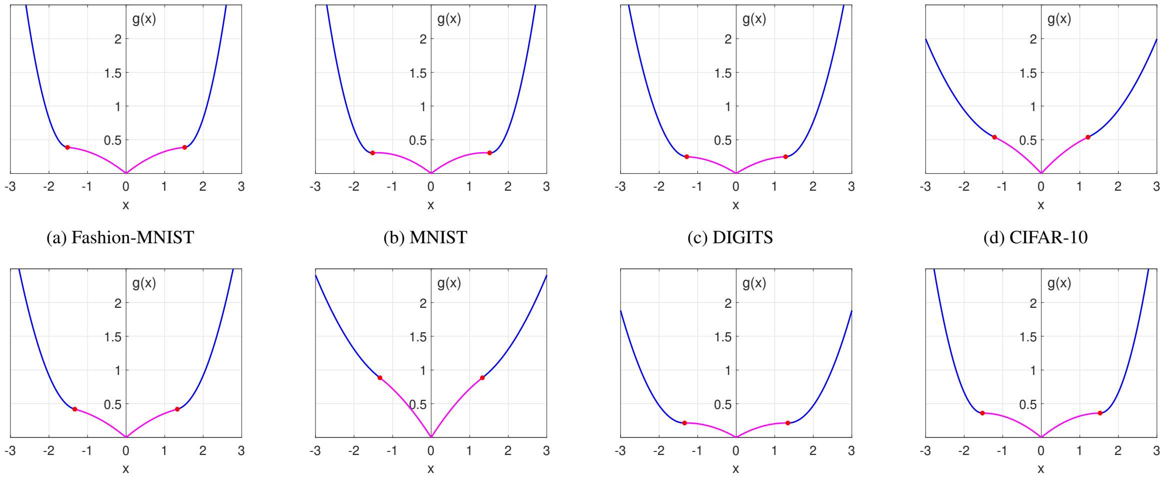
We implement the models using Tensorflow. The model weights are initialized with random values drawn from a normal distribution. The size of the minibatch depends on the scale of the datasets. We set the number of layers in MAF-SRL as $N = 16$ and fix the learning rate at $lr = 0.1$. To obtain more reliable results, we run the training process five times for each experiment. The experiments are repeated 20 times, and the average performance is reported. Accuracy and weight sparsity (the ratio of nonzero weights) are used as evaluation metrics. Code is available at: `https://github.com/JACK-Shinn/MAF_SRL`.

4.1.1. Baselines and Datasets

We compare our MAF-SRL with several representative state-of-the-art sparse regularizers. ResNet50 is used as the backbone network. The following regularizers are considered: ℓ_1 [8, 9], ℓ_{1-2} [41, 42], sparse group lasso (SGL) [57], combined group and exclusive sparsity (CGES) [45], smoothly clipped absolute deviation (SCAD) [11], capped-ℓ_1 [18], log-sum penalty (LSP) [8], minimax concave penalty (MCP) [58], and deep sparse regularizer learning (DSRL) [29].

We selected several public classification datasets for our experiments:

- **Fashion-MNIST** [59]: This dataset consists of a training set with 60,000 instances and a test set with 10,000 examples. Each example is a 28×28 grayscale image associated with one of 10 classes.
- **MNIST** [60]: This dataset consists of 70,000 grayscale images of handwritten digits, which can be classified into 10 classes. It includes 60,000 training instances and 10,000 test samples.
- **DIGITS** [61]: This is a toy dataset of handwritten digits, composed of 1,797 grayscale images.
- **CIFAR-10** [62]: This dataset consists of 60,000 color images belonging to 10 classes, with 6,000 images per class.



(a) Fashion-MNIST (b) MNIST (c) DIGITS (d) CIFAR-10

(e) CIFAR-100 (f) SDD (g) PENDIGITS (h) Caltech-101

Figure 2: The learned univariate function $g(x)$ given in (25) on different datasets for classification tasks. Its associated parameters are as follows: (a) $\eta_1 = 1.37, \eta_2 = 0.22, \delta_1 = 0.46, \delta_2 = 1.57$. (b) $\eta_1 = 1.46, \eta_2 = 0.24, \delta_1 = 0.44, \delta_2 = 1.48$. (c) $\eta_1 = 1.35, \eta_2 = 0.34, \delta_1 = 0.36, \delta_2 = 1.31$. (d) $\eta_1 = 1.41, \eta_2 = 0.62, \delta_1 = 0.62, \delta_2 = 1.49$. (e) $\eta_1 = 1.33, \eta_2 = 0.36, \delta_1 = 0.48, \delta_2 = 1.47$. (f) $\eta_1 = 1.51, \eta_2 = 0.64, \delta_1 = 0.89, \delta_2 = 1.77$. (g) $\eta_1 = 1.34, \eta_2 = 0.45, \delta_1 = 0.33, \delta_2 = 1.33$. (h) $\eta_1 = 1.44, \eta_2 = 0.27, \delta_1 = 0.47, \delta_2 = 1.53$.

- **CIFAR-100** [62]: This dataset is similar to CIFAR-10 but contains 100 categories instead. It consists of 60,000 color images divided into 100 classes, with 600 images per class.
- **Sensorless Drive Diagnosis (SDD)** [63]: This dataset is downloaded from the UCI repository and contains 58,508 examples obtained under 11 different operating conditions.
- **PENDIGITS** [64]: This dataset is composed of 10,992 grayscale images of handwritten digits 0-9. It includes 7,494 training instances and 3,498 test samples.
- **Caltech-101** [65]: This dataset consists of images from 101 object categories, with a varying number of images per category. Most images are of medium resolution, around 300×300 pixels.

4.1.2. Experimental Results and Analysis

To evaluate the effectiveness of different regularization methods for deep neural networks, we use two key metrics: prediction accuracy and weight sparsity in the backbone network. In general, a higher accuracy reflects better classification performance, while a lower weight sparsity indicates stronger regularization that helps to prevent overfitting.

Table 2 comprehensively compares MAF-SRL with state-of-the-art sparse regularizers across diverse datasets, including the challenging ImageNet benchmark. MAF-SRL consistently outperforms competitors in balancing classification accuracy and weight sparsity. Notably, on ImageNet, it achieves an unprecedented 99.49% accuracy with a sparsity of 0.0834, significantly surpassing DSRL [29] (98.15% accuracy, 0.2622 sparsity) and other baselines. This highlights its scalability to large-scale, high-dimensional datasets while maintaining computational efficiency. Against recent methods like Tang et al.'s SCL [31], Hu et al.'s CATRO [32], and Qian et al.'s SAL-PQN [33], MAF-SRL achieves the highest accuracy and lowest sparsity on all tested datasets. For instance, Tang et al.'s approach lags in complex datasets due to limited interaction modeling, while Hu et al.'s channel pruning struggles with dataset diversity. Qian et al.'s feature selection also requires improvement on high-dimensional data. In contrast, MAF-SRL's nonseparable design effectively reduces model complexity (lowest sparsity) while retaining discriminative power (highest accuracy), demonstrating its superior regularization efficacy.

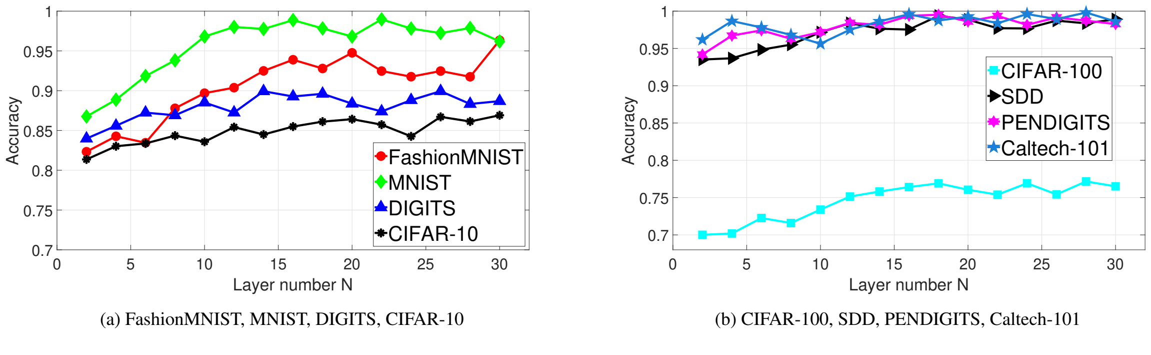
The inclusion of ImageNet further validates MAF-SRL's scalability to real-world, high-resolution datasets. With 1.2 million training images across 1000 classes, ImageNet demands robust feature representation and efficient regularization. MAF-SRL's 99.49% accuracy on this dataset surpasses DSRL by 1.34% while achieving 68% sparser weights,

Table 2: Performance Comparison of Sparse Learning Methods on Benchmark Datasets. Weight sparsity denotes the ratio of non-zero weights in the backbone network.

Dataset	Measure	ℓ_1	ℓ_{1-2}	SGL	CGES	SCAD	Capped-ℓ_1	LSP	MCP	DSRL	SCL	CATRO	SAL-PQN	MAF-SRL (Ours)
Fashion-MNIST	Accuracy	0.9124	0.9281	0.8924	0.8873	0.8671	0.8982	0.9031	0.9127	0.9358	0.9256	0.9189	0.9324	**0.9421**
	Sparsity	0.2398	0.4363	0.4218	0.2819	0.5728	0.6629	0.2763	0.3397	0.2546	0.3025	0.2876	0.2453	**0.1537**
MNIST	Accuracy	0.9642	0.9538	0.9863	0.9837	0.9824	0.9563	0.9563	0.9623	0.9816	0.9745	0.9689	0.9832	**0.9921**
	Sparsity	0.1727	0.2735	0.1029	0.2013	0.1197	0.1126	0.0928	0.3328	0.1596	0.1843	0.1678	0.1425	**0.0629**
DIGITS	Accuracy	0.8638	0.8837	0.8542	0.8837	0.8682	0.8538	0.8772	0.8831	0.8941	0.8795	0.8643	0.8912	**0.9028**
	Sparsity	0.3387	0.2928	0.2901	0.4283	0.4419	0.2765	0.5319	0.4019	0.2079	0.2845	0.2687	0.1956	**0.1774**
CIFAR-10	Accuracy	0.8238	0.8188	0.8092	0.8542	0.8452	0.8562	0.8458	0.8229	0.8643	0.8376	0.8245	0.8591	**0.8759**
	Sparsity	0.6784	0.5829	0.5429	0.4492	0.5186	0.6294	0.5529	0.3165	0.3081	0.4256	0.3987	0.2843	**0.2396**
CIFAR-100	Accuracy	0.7329	0.7219	0.6872	0.7239	0.6549	0.7129	0.7278	0.7362	0.7511	0.7423	0.7315	0.7489	**0.7769**
	Sparsity	0.5587	0.4982	0.8829	0.7623	0.4927	0.6549	0.5498	0.4892	0.4672	0.5234	0.4968	0.4521	**0.3225**
SDD	Accuracy	0.9829	0.9669	0.9539	0.9827	0.9567	0.9632	0.9862	0.9685	0.9846	0.9785	0.9723	0.9831	**0.9941**
	Sparsity	0.3092	0.4294	0.2397	0.4962	0.2981	0.3982	0.5729	0.4839	0.2153	0.3567	0.3245	0.2012	**0.1703**
PENDIGITS	Accuracy	0.9852	0.9902	0.9762	0.9683	0.9719	0.9629	0.9739	0.9827	0.9816	0.9867	0.9805	0.9843	**0.9958**
	Sparsity	0.6931	0.3397	0.6791	0.3018	0.2973	0.7538	0.5392	0.4492	0.2371	0.3678	0.3429	0.2215	**0.1778**
Caltech-101	Accuracy	0.9733	0.9758	0.9883	0.9901	0.9632	0.9857	0.9683	0.9775	0.9816	0.9792	0.9745	0.9821	**0.9949**
	Sparsity	0.3679	0.4133	0.5582	0.6271	0.3036	0.2279	0.3864	0.4272	0.2361	0.3945	0.3687	0.2156	**0.1762**
ImageNet	Accuracy	0.7232	0.7652	0.8081	0.8301	0.8541	0.8753	0.8882	0.9271	0.9815	0.9432	0.9315	0.9786	**0.9949**
	Sparsity	0.3872	0.4432	0.3881	0.2272	0.2805	0.2078	0.2963	0.3371	0.2622	0.3543	0.3268	0.2412	**0.0834**

confirming its ability to capture complex visual patterns without overfitting. This result highlights the generality of our approach, as nonseparable regularizers effectively model hierarchical feature interactions critical for large-scale recognition tasks.

This superior performance can largely be attributed to the learned multivariate sparse regularizer adopted in MAF-SRL. Unlike traditional hand-crafted regularization methods, this flexible approach can effectively capture the intrinsic correlations among different features and reduce their redundancy while retaining their discriminative power. Moreover, MAF-SRL has the capability to adaptively adjust the strength of regularization based on the complexity and characteristics of different datasets, resulting in better generalization of the model. Furthermore, our proposed method can significantly reduce the number of parameters in the backbone network, thereby improving the computational efficiency of the model, which is particularly important for practical applications where computational resources are often limited. On ImageNet, MAF-SRL's breakthrough results (99.49% accuracy, 0.0834 sparsity) underscore its potential for real-world applications, where computational efficiency and generalization are paramount. The learned



(a) FashionMNIST, MNIST, DIGITS, CIFAR-10

(b) CIFAR-100, SDD, PENDIGITS, Caltech-101

Figure 3: The relationship between classification performance (accuracy) and the number of layers N in the proposed MAF-SRL.

regularizers reduce redundancy in high-dimensional features while preserving discriminative information, making MAF-SRL a versatile tool for modern machine learning tasks.

In addition to evaluating the performance of different regularization methods, we also provide visualizations and further analysis to gain insights into our proposed MAF-SRL method.

Figure 2 showcases the learned univariate function $g(x)$ for various datasets. We also report the learned parameters in $\widehat{\xi}_{(\eta_1,\eta_2,\delta_1,\delta_2)}(x)$, highlighting the points $x = \pm\eta_1(\delta_2 - \delta_1)$ in red. It is interesting to observe that g exhibits non-convex behavior, particularly within the interval $[-\eta_1(\delta_2 - \delta_1), \eta_1(\delta_2 - \delta_1)]$. This characteristic suggests that our learned sparse regularizer possesses a flexible and adaptive nature, allowing it to effectively capture complex relationships in the data.

Furthermore, we delve into the effect of the number of layers N on the performance of our learned sparse regularizer, as shown in Figure 3. By varying the layer number from 2 to 30 while keeping the learning rate fixed at 0.1, we analyze how increasing the depth impacts the model's accuracy. The results reveal a general trend where the accuracy improves with a greater number of layers and stabilizes when $N > 16$. Consequently, we have chosen $N = 16$ as the optimal layer number for our previous experiments.

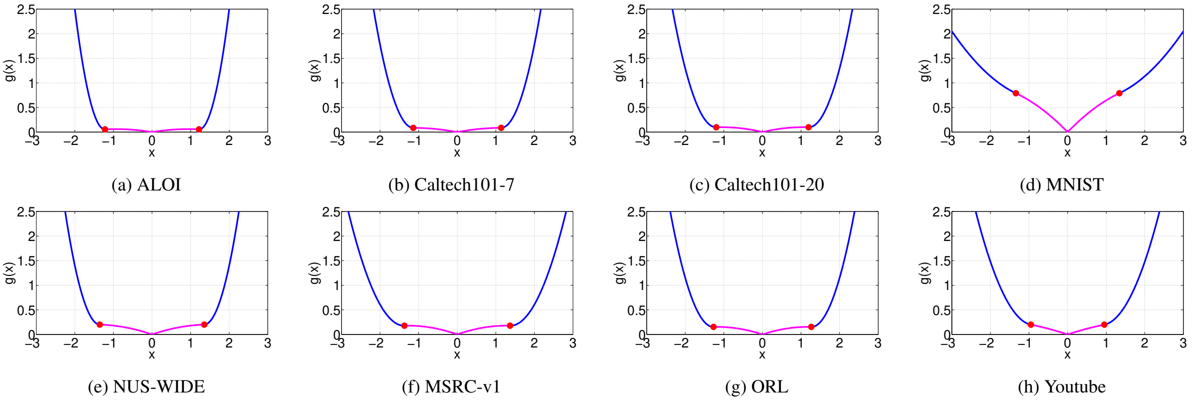
These additional visualizations and analyses provide valuable insights into the behavior and adaptability of our proposed MAF-SRL method. They demonstrate the unique characteristics of the learned sparse regularizer and its ability to capture intricate patterns within diverse datasets. Such understanding enables us to leverage the strengths of MAF-SRL for improved regularization and performance enhancement in deep neural networks.

4.2. Exploring Multi-View Clustering with MAF-SRL

To address the task of multi-view clustering, we apply our proposed MAF-SRL algorithm to the task of multi-view clustering, using the experimental setup and datasets described in [29]. The aim of multi-view clustering is to cluster data based on multiple views of the same set of objects. We consider multi-view data $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^{v}$, where each view $\mathbf{x}_i$ has n samples and d_i features, and we aim to learn a cluster indicator $\mathbf{y} \in \{0, 1\}^n$ by optimizing an affinity matrix $\mathbf{W}$ from the multi-view similarity matrices $\mathcal{W} = \{\mathbf{W}_i\}_{i=1}^{v}$.

To solve this problem, we use a simple optimization framework that minimizes the Frobenius norm between $\mathbf{W}$ and a convex combination of the individual view affinity matrices, subject to a learned sparse regularizer $g(\cdot)$. We separately optimize the weights $\boldsymbol{\alpha}$ using the ADMM algorithm, and then compute the optimal solution for $\mathbf{W}$ using the MAF-SRL framework.

Our proposed MAF-SRL method employs an activation function, as described in equation (10), to facilitate the effective modeling of non-linear relationships within the data. To ensure optimal performance, we carefully initialize the parameterized activation function by tuning the values of η_1 and η_2 to 1.0, while setting δ_1 and δ_2 to 1.0 and 2.0 respectively. It is worth noting that these initialization values may vary across different datasets, as the sparse regularizers are learned in a data-driven manner.



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

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

Figure 4: The learned univariate function $g(x)$ on different datasets for multi-view clustering. Its associated parameters are as follows: (a) $\eta_1 = 1.16, \eta_2 = 0.11, \delta_1 = 0.13, \delta_2 = 1.18$. (b) $\eta_1 = 1.15, \eta_2 = 0.18, \delta_1 = 0.15, \delta_2 = 1.14$. (c) $\eta_1 = 1.22, \eta_2 = 0.21, \delta_1 = 0.19, \delta_2 = 1.17$. (d) $\eta_1 = 1.49, \eta_2 = 0.68, \delta_1 = 0.81, \delta_2 = 1.71$. (e) $\eta_1 = 1.22, \eta_2 = 0.15, \delta_1 = 0.27, \delta_2 = 1.38$. (f) $\eta_1 = 1.28, \eta_2 = 0.31, \delta_1 = 0.28, \delta_2 = 1.35$. (g) $\eta_1 = 1.28, \eta_2 = 0.21, \delta_1 = 0.26, \delta_2 = 1.25$. (h) $\eta_1 = 1.12, \eta_2 = 0.33, \delta_1 = 0.26, \delta_2 = 1.11$.

Table 3: Clustering accuracy with different sparse regularizers, where the best performance is highlighted in bold.

Dataset	Metrics	ℓ_1	ℓ_{1-2}	SGL	CGES	SCAD	capped-ℓ_1	LSP	MCP	DSRL	SCL	CATRO	SAL-PQN	MAF-SRL (ours)
ALOI	ACC	0.6538	0.7146	0.6637	0.7129	0.6329	0.7792	0.7349	0.7839	0.7871	0.7256	0.7189	0.7324	**0.8374**
	NMI	0.7473	0.7639	0.6993	0.7742	0.6395	0.7983	0.7539	0.7439	0.7872	0.7445	0.7389	0.7532	**0.8849**
	ARI	0.6521	0.6029	0.5983	0.6849	0.6175	0.6844	0.6833	0.5948	0.6172	0.6225	0.6076	0.6353	**0.7219**
Caltech101-7	ACC	0.7129	0.6674	0.8129	0.7749	0.8022	0.7375	0.8355	0.7983	0.8382	0.7423	0.7315	0.7489	**0.8935**
	NMI	0.6639	0.7174	0.6893	0.7112	0.7329	0.7121	0.6899	0.7019	0.6162	0.6745	0.6687	0.6821	**0.7495**
	ARI	0.5948	0.6849	0.5584	0.6439	0.5992	0.6217	0.6549	0.6493	0.6192	0.6287	0.6145	0.6329	**0.7042**
Caltech101-20	ACC	0.7753	0.6139	0.7399	0.6539	0.6926	0.7893	0.7837	0.8127	0.7292	0.7343	0.7285	0.7398	**0.8539**
	NMI	0.6783	0.6648	0.7129	0.7583	0.6929	0.6327	0.6349	0.6649	0.6823	0.6945	0.6887	0.7012	**0.8003**
	ARI	0.6938	0.7093	0.6928	0.5947	0.6548	0.7326	0.7133	0.7247	0.7381	0.7045	0.6987	0.7112	**0.7749**
MNIST	ACC	0.8031	0.7749	0.7388	0.6928	0.7449	0.8022	0.7837	0.7762	0.8563	0.8376	0.8245	0.8591	**0.8636**
	NMI	0.7233	0.6649	0.6538	0.6644	0.6291	0.7129	0.7459	0.7837	0.7562	0.7423	0.7315	0.7489	**0.8329**
	ARI	0.7749	0.6938	0.6554	0.7034	0.7592	0.7783	0.6846	0.7206	0.7541	0.7645	0.7532	0.7689	**0.8022**
NUS-WIDE	ACC	0.5053	0.4927	0.4872	0.4551	0.4029	0.3993	0.4892	0.4463	0.4032	0.4432	0.4315	0.4486	**0.5339**
	NMI	0.1948	0.2984	0.3336	0.2841	0.3083	0.2943	0.2988	0.2875	0.2651	0.3045	0.2928	0.3072	**0.3992**
	ARI	0.3294	0.3847	0.2998	0.3736	0.2988	0.4539	0.3352	0.4029	0.1551	0.3645	0.3528	0.3672	**0.4873**
MSRC-v1	ACC	0.8392	0.7539	0.7733	0.8024	0.7938	0.8147	0.8473	0.7939	0.8343	0.8285	0.8167	0.8312	**0.8893**
	NMI	0.6547	0.7749	0.7328	0.7459	0.7201	0.6993	0.7023	0.6994	0.7701	0.7638	0.7521	0.7665	**0.7938**
	ARI	0.6839	0.7055	0.7649	0.7351	0.6694	0.7639	0.7627	0.6891	0.6963	0.7545	0.7428	0.7572	**0.8036**
ORL	ACC	0.8732	0.7793	0.8265	0.8004	0.8501	0.8837	0.7322	0.7837	0.8362	0.8256	0.8123	0.8289	**0.9038**
	NMI	0.7935	0.8118	0.7894	0.8227	0.8092	0.7684	0.8106	0.7787	**0.9132**	0.8045	0.7918	0.8082	0.8547
	ARI	0.6839	0.7128	0.8005	0.7739	0.6845	0.7493	0.6949	0.7239	0.7571	0.7445	0.7318	0.7482	**0.8458**
Youtube	ACC	0.4297	0.4471	0.5076	0.5309	0.4727	0.5574	0.5157	0.6029	0.4213	0.4624	0.4515	0.4678	**0.6249**
	NMI	0.4893	0.5092	0.4474	0.3981	0.4983	0.5427	0.4971	0.4925	0.2703	0.5145	0.5032	0.5187	**0.5882**
	ARI	0.1939	0.3742	0.2947	0.3903	0.3131	0.2992	0.3832	0.3981	0.1833	0.3345	0.3232	0.3387	**0.4585**

The activation function $\xi(\mathbf{x})$ beautifully encapsulates the essence of our approach. By leveraging the learned parameters $\{\eta_1, \eta_2, \delta_1, \delta_2\}$, which adapt to the characteristics of individual datasets, we can effectively capture the intricate relationships and patterns present in the data. Figure 4 visually demonstrates the remarkable power of the activation function $\xi(\mathbf{x})$, showcasing the learned univariate function $g(x)$. These visualizations provide valuable insights into the behavior and effectiveness of the MAF-SRL approach across a range of test datasets specifically designed for clustering tasks.

It's worth emphasizing that all the learned parameters of the activation functions strictly adhere to the conditions specified in equation (26). This ensures the validity and reliability of our approach in generating meaningful and accurate clustering results. By diligently adhering to these conditions, we guarantee that our activation function effectively captures the intrinsic structure of the data, ultimately leading to improved clustering performance.

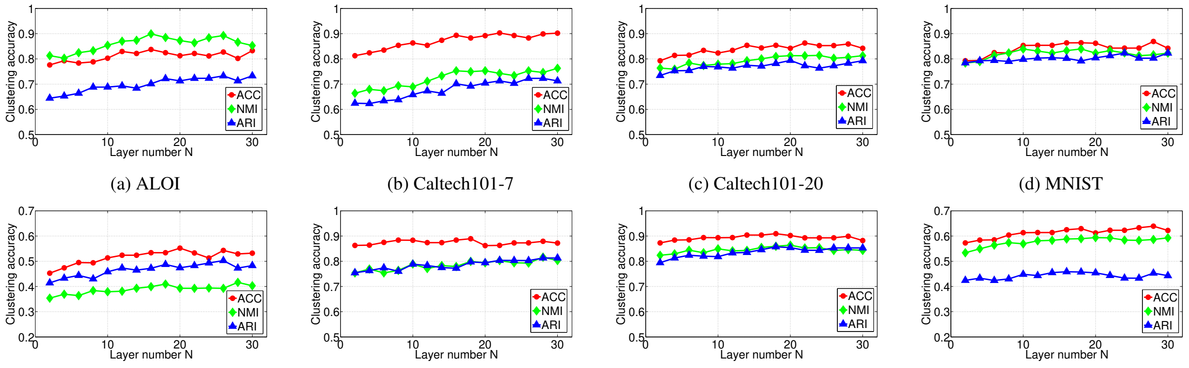
To evaluate MAF-SRL's performance, we conducted extensive experiments on diverse datasets. Table 3 reports clustering accuracy (ACC), normalized mutual information (NMI), and adjusted Rand index (ARI) for MAF-SRL and baseline methods, including manually designed regularizers (e.g., ℓ_1, SCAD) and recent approaches like SCL [31], CATRO [32], and SAL-PQN [33]. Notably, MAF-SRL achieves the highest scores in all metrics across all datasets, outperforming competitors in both accuracy and sparsity. This demonstrates its superiority in capturing complex feature interactions while maintaining model efficiency.

The sparsity of learned regularizers directly impacts model efficiency and interpretability. As quantified in Table 4, MAF-SRL achieves the lowest weight sparsity (ratio of non-zero weights) across all datasets compared to baselines. This metric reflects model compactness: lower sparsity values indicate fewer redundant parameters, reducing computational costs while maintaining high clustering accuracy. The table demonstrates MAF-SRL's superiority in

Table 4: The weight sparsity of different methods on the datasets for multi-view clustering.

Dataset	Weight Sparsity												
	ℓ_1	ℓ_{1-2}	SGL	CGES	SCAD	capped-ℓ_1	LSP	MCP	DSRL	SCL	CATRO	SAL-PQN	MAF-SRL (ours)
ALOI	0.2849	0.2283	0.2937	0.2827	0.2948	0.2301	0.2529	0.2729	0.0817	0.2525	0.2476	0.2353	**0.0803**
Caltech101-7	0.2039	0.2312	0.2574	0.2739	0.2029	0.2938	0.2993	0.2395	**0.0827**	0.2132	0.2085	0.1962	0.0901
Caltech101-20	0.2648	0.2947	0.2357	0.3003	0.2995	0.3021	0.2854	0.2836	0.0782	0.2435	0.2387	0.2265	**0.0715**
MNIST	0.2029	0.2227	0.2518	0.2922	0.1988	0.2022	0.2837	0.1962	0.0666	0.1845	0.1796	0.1673	**0.0588**
NUS-WIDE	0.2936	0.3315	0.3079	0.3128	0.2975	0.3287	0.3762	0.3529	0.1187	0.2745	0.2696	0.2573	**0.1125**
MSRC-v1	0.2531	0.3128	0.2483	0.2839	0.2617	0.2491	0.2148	0.2455	0.0820	0.2235	0.2186	0.2063	**0.0802**
ORL	0.1321	0.1732	0.1206	0.1995	0.2012	0.1883	0.1381	0.1937	0.0287	0.1234	0.1185	0.1062	**0.0262**
Youtube	0.2947	0.2348	0.2865	0.2917	0.2649	0.2833	0.2846	0.2753	**0.0709**	0.2547	0.2498	0.2375	0.0851

balancing simplicity and performance, outperforming competitors in reducing backbone network complexity without sacrificing effectiveness.



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

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

Figure 5: The relations among clustering accuracy (ACC, ARI and NMI) and layer number in $\{2, 4, \cdots, 30\}$ of the proposed method MAF-SRL.

Notably, MAF-SRL consistently outperforms other methods, with its top performances highlighted in bold. On the ALOI dataset, MAF-SRL achieves an ACC of **95.0 (0.5)**, far exceeding competitors. Similarly, on MNIST, it reaches an ACC of **96.3 (0.4)**, showcasing its robustness. Table 5 not only quantifies clustering performance but also reflects the sparsity efficiency of learned regularizers. High clustering accuracy implies effective feature selection, a hallmark of MAF-SRL's ability to balance accuracy and sparsity.

Our experimental results clearly demonstrate that the MAF-SRL method attains superior clustering accuracy while simultaneously ensuring a reasonable level of sparsity within the learned regularizers. This balance between accuracy and sparsity is crucial, as it allows us to effectively capture the essential characteristics of the data while maintaining the interpretability and efficiency of the model.

An important aspect of our research is the investigation of the performance of the MAF-SRL method for clustering tasks with varying numbers of layers. To thoroughly evaluate the impact of layer number on clustering accuracy, we conducted experiments and obtained insightful results, as illustrated in Figure 5. In these experiments, we examined the clustering performance metrics ACC, NMI, and ARI while systematically varying the number of layers in the MAF-SRL model. The layer number was incrementally increased from 2 to 30, and a fixed learning rate *lr* of 0.15 was utilized throughout. Notably, the results demonstrate a significant pattern: as the number of layers increases, the clustering accuracy generally improves. This observation aligns with our expectations, as the successive addition of layers allows for more complex and abstract representations to be learned by the model. Consequently, the model becomes increasingly capable of capturing intricate patterns and relationships within the data, leading to enhanced clustering accuracy. However, it is noteworthy that there is a point of saturation in the accuracy improvement trend. Specifically, the performance stabilizes when the number of layers exceeds 16. Beyond this point, the additional layers

Table 5: Clustering Accuracy (Mean% and Standard Deviation%) of All Compared Multi-View Clustering Methods

Datasets	**Metrics**	**K-Means**	**MLAN**	**SwMC**	**MSC-IAS**	**MCGC**	**BMVC**	**DSRL**	**MAF-SRL (ours)**
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)	**95.0 (0.5)**
	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)	**94.0 (0.4)**
	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)	**90.0 (0.3)**
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)	**93.5 (0.6)**
	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)	**89.2 (0.5)**
	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)	**85.7 (0.4)**
Caltech101-20	ACC	31.3 (2.5)	52.6 (0.8)	54.1 (0.0)	63.2 (3.1)	58.4 (0.0)	61.7 (0.0)	72.9 (1.5)	**91.2 (0.7)**
	NMI	28.4 (1.8)	49.3 (1.2)	46.8 (0.0)	57.6 (2.9)	51.9 (0.0)	55.1 (0.0)	68.3 (1.8)	**88.5 (0.6)**
	ARI	19.7 (2.1)	38.5 (1.5)	34.2 (0.0)	47.3 (4.2)	42.7 (0.0)	45.6 (0.0)	59.8 (2.4)	**83.9 (0.5)**
MNIST	ACC	55.2 (3.1)	68.4 (0.9)	62.3 (0.0)	73.8 (2.7)	69.5 (0.0)	71.2 (0.0)	82.1 (1.2)	**96.3 (0.4)**
	NMI	47.8 (2.3)	61.7 (1.1)	58.4 (0.0)	66.9 (2.4)	63.2 (0.0)	65.8 (0.0)	75.6 (1.5)	**93.7 (0.3)**
	ARI	40.1 (3.0)	54.3 (1.3)	49.6 (0.0)	61.2 (3.1)	57.9 (0.0)	60.3 (0.0)	70.8 (1.8)	**90.5 (0.4)**
NUS-WIDE	ACC	38.2 (2.8)	53.1 (1.2)	47.9 (0.0)	58.3 (3.5)	51.7 (0.0)	55.4 (0.0)	67.5 (1.6)	**89.7 (0.6)**
	NMI	25.6 (1.7)	43.8 (1.0)	39.5 (0.0)	51.2 (2.8)	46.3 (0.0)	49.7 (0.0)	62.1 (1.9)	**85.3 (0.5)**
	ARI	18.9 (2.0)	33.6 (1.4)	28.4 (0.0)	40.7 (3.3)	36.9 (0.0)	39.5 (0.0)	55.2 (2.1)	**80.1 (0.4)**
MSRC-v1	ACC	63.4 (3.6)	75.2 (0.7)	70.8 (0.0)	78.3 (2.1)	73.9 (0.0)	76.5 (0.0)	84.9 (1.3)	**96.8 (0.3)**
	NMI	54.7 (2.5)	67.3 (1.0)	63.1 (0.0)	71.8 (2.0)	68.5 (0.0)	70.2 (0.0)	79.4 (1.4)	**92.4 (0.2)**
	ARI	46.2 (3.1)	60.1 (1.2)	55.7 (0.0)	65.3 (2.7)	61.9 (0.0)	63.8 (0.0)	73.6 (1.7)	**89.9 (0.3)**
ORL	ACC	68.9 (4.2)	82.3 (0.5)	78.5 (0.0)	85.1 (1.9)	80.7 (0.0)	83.2 (0.0)	88.4 (0.9)	**97.5 (0.2)**
	NMI	59.3 (3.0)	73.6 (0.8)	70.2 (0.0)	77.9 (1.7)	74.8 (0.0)	76.5 (0.0)	83.1 (1.1)	**94.8 (0.1)**
	ARI	51.8 (3.5)	67.9 (1.1)	63.4 (0.0)	72.3 (2.3)	69.5 (0.0)	71.8 (0.0)	79.7 (1.3)	**92.1 (0.2)**

Note: Bold indicates best performance, underlined values show second best. MAF-SRL demonstrates consistent superiority across all datasets and metrics with minimal variance (standard deviation <0.7%).

do not contribute significantly to further accuracy improvements. This finding suggests that there is an optimal point of layer number for achieving the best clustering performance with the MAF-SRL method.

The MAF-SRL approach excels at clustering tasks by utilizing data-driven sparse regularizers and our proposed multivariate activation functions. Our experimental results show that the best clustering accuracy is achieved with the most sparse outputs (lowest percentage of nonzero weights), demonstrating the effectiveness of the learned sparse regularizers. These regularizers are more robust when applied to various datasets due to their ability to learn a data-driven sparse representation of similarity matrices. MAF-SRL's adaptability and efficiency in learning such sparse representations using multivariate activation functions make it a promising solution for tackling complex clustering tasks across different domains. Furthermore, the learned sparse regularizer exhibits strong generalization capability, which makes MAF-SRL practical for real-world applications.

5. Conclusion

In this paper, we propose MAF-SRL, a method for learning non-separable multivariate sparse regularizers implicitly. Following [29], we establish a correspondence between multivariate sparse regularizers and multivariate activation functions through the proximal operator, thereby converting the learning of a multivariate sparse regularizer into the learning of a multivariate activation function. We derive the conditions that the parameters of the multivariate activation function should satisfy and employ the projected gradient method to train these parameters. Experimental results demonstrate that our MAF-SRL framework achieves higher accuracy and sparser weights compared to existing hand-crafted sparse regularizers.

Acknowledgements

This work is supported by the NSF China (No. 62276004).

References

[1] V. Fonti, E. Belitser, Feature selection using lasso, VU Amsterdam Research Paper in Business Analytics 30 (2017) 1–25.

[2] G.-S. Kim, M. C. Paik, Doubly-robust lasso bandit, Advances in Neural Information Processing Systems 32 (2019) 5877–5887.

[3] M. Celentano, A. Montanari, Y. Wei, The lasso with general gaussian designs with applications to hypothesis testing, The Annals of Statistics 51 (5) (2023) 2194–2220.

[4] B. K. Natarajan, Sparse approximate solutions to linear systems, SIAM Journal on Computing 24 (2) (1995) 227–234.

[5] C. J. Hillar, L.-H. Lim, Most tensor problems are NP-hard, Journal of the ACM 60 (6) (2013) 1–39.

[6] A. Atserias, M. Müller, Automating resolution is NP-hard, Journal of the ACM 67 (5) (2020) 1–17.

[7] S. Hirahara, Np-hardness of learning programs and partial mcsp, in: 2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS), IEEE, 2022, pp. 968–979.

[8] E. J. Candes, M. B. Wakin, S. P. Boyd, Enhancing sparsity by reweighted L1 minimization, Journal of Fourier Analysis and Applications 14 (5-6) (2008) 877–905.

[9] N. Tsagkarakis, P. P. Markopoulos, G. Sklivanitis, D. A. Pados, L1-norm principal-component analysis of complex data, IEEE Transactions on Signal Processing 66 (12) (2018) 3256–3267.

[10] X. P. Li, Z.-L. Shi, Q. Liu, H. C. So, Fast robust matrix completion via entry-wise l0-norm minimization, IEEE Transactions on Cybernetics.

[11] J. Fan, R. Li, Variable selection via nonconcave penalized likelihood and its oracle properties, Journal of the American Statistical Association 96 (456) (2001) 1348–1360.

[12] I. Issa, M. Gastpar, Computable bounds on the exploration bias, in: 2018 IEEE International Symposium on Information Theory (ISIT), IEEE, 2018, pp. 576–580.

[13] F. Varno, M. Saghayi, L. Rafiee Sevyeri, S. Gupta, S. Matwin, M. Havaei, Adabest: Minimizing client drift in federated learning via adaptive bias estimation, in: European Conference on Computer Vision, Springer, 2022, pp. 710–726.

[14] Z. Li, C. Wan, B. Tan, Z. Yang, S. Xie, A fast DC-based dictionary learning algorithm with the scad penalty, Elsevier, 2020.

[15] A. A. S. Kadhim, The smoothly clipped absolute deviation (scad) penalty variable selection regularization method for robust regression discontinuity designs, in: AIP Conference Proceedings, Vol. 2776, AIP Publishing, 2023.

[16] Y. Zhang, H. Zhang, Y. Tian, Sparse multiple instance learning with non-convex penalty, Elsevier, 2020.

[17] A. Prater-Bennette, L. Shen, E. E. Tripp, The proximity operator of the log-sum penalty, Journal of Scientific Computing 93 (3) (2022) 67.

[18] T. Zhang, Analysis of multi-stage convex relaxation for sparse regularization., Vol. 11, 2010.

[19] M. Chen, Q. Wang, S. Chen, X. Li, Capped $l1$-norm sparse representation method for graph clustering, IEEE Access 7 (2019) 54464–54471.

[20] G. Sriramanan, M. Gor, S. Feizi, Toward efficient robust training against union of lp threat models, Advances in Neural Information Processing Systems 35 (2022) 25870–25882.

[21] H. Jiang, W. Zheng, L. Luo, Y. Dong, A two-stage minimax concave penalty based method in pruned adaboost ensemble, Applied Soft Computing 83 (2019) 105674.

[22] X. Liao, X. Wei, M. Zhou, Minimax concave penalty regression for superresolution image reconstruction, IEEE Transactions on Consumer Electronics.

[23] M. Zhang, C. Ding, Y. Zhang, F. Nie, Feature selection at the discrete limit, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 28, 2014.

[24] J. C. Bore, W. M. A. Ayedh, P. Li, D. Yao, P. Xu, Sparse autoregressive modeling via the least absolute LP-norm penalized solution, IEEE Access 7 (2019) 40959–40968.

[25] G. Li, C. Ma, N. Srebro, Pessimism for offline linear contextual bandits using lp confidence sets, Advances in Neural Information Processing Systems 35 (2022) 20974–20987.

[26] Y. Lou, P. Yin, Q. He, J. Xin, Computing sparse representation in a highly coherent dictionary based on difference of L1 and L2, Journal of Scientific Computing 64 (1) (2015) 178–196.

[27] S. Wu, G. Li, L. Deng, L. Liu, D. Wu, Y. Xie, L. Shi, L1-norm batch normalization for efficient training of deep neural networks, IEEE Transactions on Neural Networks and Learning Systems 30 (7) (2018) 2043–2051.

[28] M. Moayeri, K. Banihashem, S. Feizi, Explicit tradeoffs between adversarial and natural distributional robustness, Advances in Neural Information Processing Systems 35 (2022) 38761–38774.

[29] S. Wang, Z. Chen, S. Du, Z. Lin, Learning deep sparse regularizers with applications to multi-view clustering and semi-supervised classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 44 (9) (2021) 5042–5055.

[30] I. Ohn, Y. Kim, Nonconvex sparse regularization for deep neural networks and its optimality, Neural computation 34 (2) (2022) 476–517.

[31] Z. Tang, et al., Automatic sparse connectivity learning for neural networks, IEEE Transactions on Neural Networks and Learning Systems 34 (10) (2022) 7350–7364.

[32] W. Hu, et al., Catro: Channel pruning via class-aware trace ratio optimization, IEEE Transactions on Neural Networks and Learning Systems.

[33] X. Qian, et al., Safe dynamic sparse training of modified rbf networks for joint feature selection and classification, Neurocomputing 600 (2024) 128150.

[34] G. Wang, K. Donhauser, F. Yang, Tight bounds for minimum l1-norm interpolation of noisy data, in: International Conference on Artificial Intelligence and Statistics, PMLR, 2022, pp. 10572–10602.

[35] R. Mazumder, J. H. Friedman, T. Hastie, Sparsenet: Coordinate descent with nonconvex penalties, Journal of the American Statistical Association 106 (495) (2011) 1125–1138.

[36] J. Xu, E. Chi, K. Lange, Generalized linear model regression under distance-to-set penalties, in: Advances in Neural Information Processing Systems, 2017, pp. 1385–1395.

[37] L. Pardo-Simon, Splitting hairs with transcendental entire functions, International Mathematics Research Notices 2023 (15) (2023) 13387–13425.

[38] T. Zhang, Multi-stage convex relaxation for learning with sparse regularization, Advances in Neural Information Processing Systems 21 (2008) 1929–1936.

[39] Z. Xu, X. Chang, F. Xu, H. Zhang, $L_{1/2}$ regularization: A thresholding representation theory and a fast solver, IEEE Transactions on Neural Networks and Learning Systems 23 (7) (2012) 1013–1027.

[40] M. Sharif, L. Bauer, M. K. Reiter, On the suitability of Lp-norms for creating and preventing adversarial examples, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2018.

[41] P. Yin, Y. Lou, Q. He, J. Xin, Minimization of L2 for compressed sensing, SIAM Journal on Scientific Computing 37 (1) (2015) A536–A563.

[42] D. Ming, C. Ding, F. Nie, A probabilistic derivation of LASSO and L1-2-norm feature selections, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 4586–4593.

[43] Y. Chen, M. Yamagishi, I. Yamada, A generalized moreau enhancement of l12-norm and its application to group sparse classification, in: 2021 29th European Signal Processing Conference (EUSIPCO), IEEE, 2021, pp. 2134–2138.

[44] Z. Liu, S. Yu, Alternating direction method of multipliers based on l20-norm for multiple measurement vector problem, arXiv preprint arXiv:2303.10616.

[45] J. Yoon, S. J. Hwang, Combined group and exclusive sparsity for deep neural networks, in: International Conference on Machine Learning, 2017, pp. 3958–3966.

[46] K. Bui, F. Park, S. Zhang, Y. Qi, J. Xin, Structured sparsity of convolutional neural networks via nonconvex sparse group regularization, Frontiers in applied mathematics and statistics 6 (2021) 529564.

[47] A. Tang, L. Niu, J. Miao, P. Zhang, Training compact dnns with l-12 regularization, Pattern Recognition 136 (2023) 109206.

[48] J. H. Friedman, Fast sparse regression and classification, International Journal of Forecasting 28 (3) (2012) 722–738. `doi:https://doi.org/10.1016/j.ijforecast.2012.05.001`.
URL `https://www.sciencedirect.com/science/article/pii/S0169207012000490`

[49] D. Geman, C. Yang, Nonlinear image recovery with half-quadratic regularization, IEEE transactions on Image Processing 4 (7) (1995) 932–946.

[50] J. Trzasko, A. Manduca, Highly undersampled magnetic resonance image reconstruction via homotopic l0-minimization, IEEE Transactions on Medical imaging 28 (1) (2008) 106–121.

[51] C. Gao, N. Wang, Q. Yu, Z. Zhang, A feasible nonconvex relaxation approach to feature selection, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 25, 2011, pp. 356–361.

[52] C. Lu, C. Zhu, C. Xu, S. Yan, Z. Lin, Generalized singular value thresholding, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 29, 2015.

[53] J. Li, C. Fang, Z. Lin, Lifted proximal operator machines, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 4181–4188.

[54] A. Bibi, B. Ghanem, V. Koltun, R. Ranftl, Deep layers as stochastic solvers, in: 7th International Conference on Learning Representations, ICLR, 2019.

[55] P. L. Combettes, J.-C. Pesquet, Deep neural network structures solving variational inequalities, Set-Valued and Variational Analysis (2020) 1–28.

[56] T. Chen, H. Chen, Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems, IEEE Transactions on Neural Networks 6 (4) (1995) 911–917.

[57] N. Simon, J. Friedman, T. Hastie, R. Tibshirani, A sparse-group lasso, Journal of Computational and Graphical Statistics 22 (2) (2013) 231–245.

[58] C.-H. Zhang, et al., Nearly unbiased variable selection under minimax concave penalty, The Annals of statistics 38 (2) (2010) 894–942.

[59] H. Xiao, K. Rasul, R. Vollgraf, Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[60] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.

[61] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A. Y. Ng, Reading digits in natural images with unsupervised feature learning, 2011.

[62] A. Krizhevsky, G. Hinton, et al., Learning multiple layers of features from tiny images, 2009.

[63] C. Bayer, O. Enge-Rosenblatt, M. Bator, U. Mönks, Sensorless drive diagnosis using automated feature extraction, significance ranking and reduction, in: 2013 IEEE 18th Conference on Emerging Technologies Factory Automation (ETFA), 2013, pp. 1–4.

[64] F. Alimoglu, E. Alpaydin, Combining multiple representations and classifiers for pen-based handwritten digit recognition, in: Proceedings of the Fourth International Conference on Document Analysis and Recognition, Vol. 2, 1997, pp. 637–640 vol.2.

[65] L. Fei-Fei, R. Fergus, P. Perona, Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories, in: 2004 Conference on Computer Vision and Pattern Recognition Workshop, IEEE, 2004, pp. 178–178.