IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 2, 2019, accepted January 20, 2019, date of publication January 31, 2019, date of current version February 14, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2895884

Learning Robust Auto-Encoders With Regularizer for Linearity and Sparsity

YONG SHI[1,2,3,4], (Member, IEEE), MINGLONG LEI[2,3,5], RONGRONG MA[2,3,6], AND LINGFENG NIU[1,2,3]

[1]School of Economics and Management, University of Chinese Academy of Sciences, Beijing 100190, China
[2]Key Laboratory of Big Data Mining and Knowledge Management, Chinese Academy of Sciences, Beijing 100190, China
[3]Research Center on Fictitious Economy & Data Science, Chinese Academy of Sciences, Beijing 100190, China
[4]College of Information Science and Technology, University of Nebraska at Omaha, Omaha, NE 68182, USA
[5]School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 100049, China
[6]School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing, 100049, China

Corresponding author: Lingfeng Niu (niulf@ucas.ac.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 91546201, Grant 71331005, Grant 71110107026, Grant 11671379, and Grant 11331012, and in part by the UCAS under Grant Y55202LY00.

ABSTRACT Unsupervised feature learning via auto-encoders results in low-dimensional representations in latent space that capture the patterns of input data. The auto-encoders with robust regularization learn qualified features that are less sensitive to small perturbations of inputs. However, the previous robust auto-encoders highly depend on pre-defined structure settings and often learn full-connected networks that are easily prone to over-fitting. To solve the above limitations, we propose in this paper an explicitly regularized framework which improves the sparsity and flexibility of robust auto-encoders. First, our model encourages the activation functions to automatically adjust themselves between linear and non-linear ones. Second, the mapping functions of the encoder are constrained by group sparsity and exclusive sparsity to reduce the redundancy of parameters. The proximal gradient method is used to optimize our model since the objective function contains non-smooth components. We conduct experiments in single-layer and multiple-layer auto-encoders in the classification task. The numerical results show that our model achieves better accuracy than baseline models. Our method also shows better performance in denoising task.

INDEX TERMS Auto-encoder, linearity, robust features, representation learning, sparsity.

I. INTRODUCTION

Representation learning via auto-encoder paradigm is a promising topic which is beneficial for many downstream machine learning tasks such as classification [1] and clustering [2]. The auto-encoder framework is trained to copy the inputs to the outputs [3]. The hidden codes describe the latent information of the data and are often used as features to represent the inputs.

Since the basic auto-encoders are required to make a trade-off between model capacity and representation capacity, regularized auto-encoders [1], [2], [4] are proposed as a more general framework that imposes additional constraints besides the reconstruction task of the basic auto-encoders. The advantages of regularized auto-encoders are two-fold. First, the regularized auto-encoders introduce additional information such as the geometric proximity among data points in the original manifold [2]. Second, the regularized auto-encoders encourage additional properties in the framework. For example, the sparse auto-encoders constrain the hidden codes to be sparse [5], [6].

The goal of the auto-encoder framework and its regularized variants is to learn *good* representations which capture the salient information of the inputs. One general idea of the good representations is that the learned features should be robust [1], [4], [7]. Basically, robust feature learning trains the model on a harder task so that the learned model can handle more challenging inputs. According to different strategies used during the training process, the robust feature extractors can be categorized as models with a data perspective and models with a model perspective. The data perspective focuses on increasing the training difficulty by adding noises to the input data manually. In the denoising auto-encoder [1], the model

The associate editor coordinating the review of this manuscript and approving it for publication was Taufik Abrao.

is trained under the inputs contaminated by binary or Gaussian noises. The learned model discovers useful patterns in the original inputs and is able to distinguish the inputs from the noises. Unlike the models with a data perspective where the regularization is implicit, the model perspective introduces explicit regularization terms. The first derivative measures the sensitivity of the hidden units with respect to the alterations of input units. The contractive auto-encoder [4] directly constrains the first derivatives of the encoder mappings so that the learning features are robust to small changes of the inputs.

The robust feature learning models have two main limitations. First, the implicitly regularized auto-encoders depend on manually added noises which usually come from prior knowledge. The quality of learned features is susceptible to the noise type and corruption level. Second, the general robust feature learning models generate full-connected networks. The dense connections between the input layer and hidden layer are redundant and lack of flexibility. The previous models fail in introducing proper regularization terms to the auto-encoder paradigm to improve the model flexibility.

To solve the above shortcomings, we propose MRAE as an explicitly regularized feature learning framework which learns **M**ixed linearity and sparsity **R**obust **A**uto-**E**ncoders. The MRAE is able to learn robust features while keeping the neural network structure light and flexible. First, MRAE introduces a linear and non-linear flexibility regularizer which enables the model to automatically adjust the type of activation functions of the encoder. In traditional auto-encoders, the non-linearity is directly imposed. In some scenarios, the linear functions are good enough to capture useful information. MRAE incorporates the selection of linear and non-linear functions during the training process, which improves the model flexibility. Second, MRAE introduces a structured sparsity induced regularizer which improves the model performance by lightweight connections. Specifically, the $\ell_{1,2}$ norm and $\ell_{2,1}$ norm are used as regularization terms in addition to the auto-encoder framework. The full-connected auto-encoders easily lead to over-fitting, especially when the data is insufficient. The sparse regularizer can be seen as a feature selection process which only considers significant information of the inputs.

The contributions of this paper can be summarized as follows,

- We present a new regularized auto-encoder framework for feature learning. The proposed model is capable of learning robust features with a flexible and light structure.
- We propose two different strategies to learn a flexible auto-encoder structure. The linearity regularizer selects proper activation functions and the sparsity regularizer encourages lightweight connections.
- We use the proximal gradient method to optimize our objective function which includes non-smooth components. The proposed model can be efficiently solved in a two-stage update procedure.
- We conduct experiments in denoising task and classification tasks with respect to single-layer setting and multiple-layer setting. The results show that our method outperforms baselines.

The remainder of the paper is organized as follows. Section 2 reviews some unsupervised feature learning methods and briefly introduces some structured sparsity methods. In Section 3, we give our model after introducing the regularizer for linearity and the regularizer for sparsity. This leads us to Section 4 which presents the proximal gradient algorithm to optimize the proposed model. The experiments in three tasks are shown in Section 5. We conclude our work in Section 6.

II. RELATED WORK

A. UNSUPERVISED FEATURE LEARNING

Unsupervised feature learning attempts to learn meaningful features from unlabeled data [8]–[10]. It can be applied in many research fields such as computer vision [11], [12], speech recognition [6], [13] and network analysis [14], [15]. There is a bunch of unsupervised feature methods which focus on reconstructing the distribution of the original data [16]. The key challenge of those methods is to define proper measurement in the unsupervised settings. For example, the auto-encoders minimize the reconstruction error, which corresponds to maximizing the mutual information of input and features [17]. The reconstruction measurement lacks the ability to balance the model capacity and the performance. As modifications, the regularized auto-encoders introduce additional properties such as sparsity [18], [19], manifold proximity [2] and robustness [1], [4], [20] into basic auto-encoders. The success of convolutional neural networks [21] in extracting features also inspires several works that combine the convolutional operations with auto-encoders [22]–[24].

There are plenty of generative models which also attempt to learn the representations of data such as deep neural networks based on Restricted Boltzmann Machines [25]–[27]. Recently proposed models such as variational auto-encoders [28], [29] and generative adversarial networks [30], [31] are efficient in recovering the distribution of the data. The idea of generative models can also be incorporated into auto-encoders to learn qualified representations [32]–[34].

B. THE STRUCTURED SPARSITY AND DEEP NEURAL NETWORKS

In statistic learning, the sparsity induced regularization learns sparse models described by fewer parameters [35]. The notion is to predict the output variables by a reduced number of input variables [36], [37]. For example, The Least Absolute Shrinkage and Selection Operator (LASSO) [38] uses ℓ_1 regularization to enforce the parameters to be sparse. However, the ℓ_1 regularizer considers the variables as individual ones and consequently ignores the structures between the variables [35], [39]. The structured sparsity induced norms

consider the structures of variables. The group LASSO [36] regularizer with the combination of ℓ_1 norm and ℓ_2 norm extends LASSO regularizer and sets groups of variables to be zero. For groups with overlaps, the latent group LASSO [37] modifies the norm to deal with variables that not only belong to one group. Unlike the group LASSO, the exclusive LASSO [40] through $\ell_{1,2}$ norm encourages the competition within the group and only sets some elements in the group to be zero.

The structured sparsity induced norms have been used in deep neural networks. For deep models with a large amount of parameters, the sparsity enforces the models to be more compact and makes them interpretable [41]. Recent works such as [42] and [43] apply the group LASSO to deep networks and attempt to remove part of the neural connections by regularization. The work [41] uses the group sparsity and exclusive sparsity in convolutional neural networks to learn compact structures. Moreover, solving the non-smooth objective usually resorts to proximal methods [44] which have been used in many machine learning tasks [45], [46].

III. THE MODEL

We first illustrate the notations used in this paper. The matrices are denoted as boldface uppercase letters and the vectors are denoted as boldface lowercase letters. The elements of the matrix are represented as lowercase letters. The ℓ_1-norm and ℓ_2-norm of the vector $\mathbf{v}$ are denoted as $\|\mathbf{v}\|_1$ and $\|\mathbf{v}\|_2$ respectively. For a matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$, denote $\mathbf{m}_i$ as its i-th row vector, the Frobenius norm is represented as:

$$\|\mathbf{M}\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} m_{i,j}^2} = \sqrt{\sum_{i=1}^{m} \|\mathbf{m}_i\|_2^2} \tag{1}$$

The $\ell_{2,1}$-norm of $\mathbf{M}$ [47] is

$$\|\mathbf{M}\|_{2,1} = \sum_{i=1}^{m} \sqrt{\sum_{j=1}^{n} m_{i,j}^2} = \sum_{i=1}^{m} \|\mathbf{m}_i\|_2 \tag{2}$$

The $\ell_{1,2}$-norm of $\mathbf{M}$ [48] is

$$\|\mathbf{M}\|_{1,2} = \sum_{i=1}^{m} (\sum_{j=1}^{n} |m_{i,j}|)^2 = \sum_{i=1}^{m} \|\mathbf{m}_i\|_1^2 \tag{3}$$

The basic auto-encoder contains an encoder and a decoder. The encoder function f maps the inputs $\mathbf{X} \in \mathbb{R}^{d_x\times n}$ to hidden representations $\mathbf{Y} \in \mathbb{R}^{d_h\times n}$. The decoder function g maps the hidden representations to reconstructions $\mathbf{Z} \in \mathbb{R}^{d_x\times n}$. The weight matrix of the encoder is denoted as $\mathbf{W} \in \mathbb{R}^{d_x\times d_h}$. Based on f and g, the learning objective of a basic auto-encoder is expressed as,

$$\mathcal{J}_{AE} = L(f, g), \tag{4}$$

where L is the reconstruction loss with respect to $\mathbf{X}$ and $\mathbf{Z}$.

The regularized auto-encoders extend the basic auto-encoder and encourage other properties by adding additional regularization terms. The general form of regularized auto-encoders is represented as,

$$\mathcal{J}_{RAE} = L(f, g) + \alpha R(f), \tag{5}$$

where L is the reconstruction loss and R is the regularization term.

The regularized auto-encoders are powerful in learning useful information from the data. Following the regularized auto-encoder form, we present a robust and flexible auto-encoder framework. The proposed model is regularized by two explicit terms and is able to automatically adjust the network structures during the training process. In summary, we introduce two flexibility regularization terms which also correspond to two strategies used in previous robust auto-encoders. The **regularizer for linearity** attempts to adjust the first derivatives of the encoder, which is similar to contractive auto-encoders. The **regularizer for sparsity** attempts to adjust the connections of the encoder, which can be seen as the binary noise driven denoising auto-encoder with a model perspective. Notice that both regularization terms are explicitly revealed in the training process. The detailed description of the model is introduced in the following subsections.

A. REGULARIZER FOR LINEARITY

The under-complete auto-encoder is considered as a non-linear version of Principal Component Analysis [3]. The non-linearity of the encoder is set in advance in order to better approximate the transformation functions. The selection of activation functions will affect the optimization process of the model. For example, the Rectified Linear Unit (ReLU) is proposed to solve the vanishing gradient problem in deep neural networks [49]. The introducing of linearity in ReLU reduces the dependence on non-linearity and has achieved good performance. Notice that the piecewise linear activation function is an all-vs-nothing process which imposes linearity or non-linearity to all neural units. In order to improve the model flexibility, we propose a self-adjust regularizer incorporated in the model that selects the linear and non-linear functions during the training process.

Recall that the contractive auto-encoder (CAE) [4] aims to learn features that are robust to tiny perturbations of the inputs. Generally, the CAE attempts to optimize the objective over all data points,

$$\mathcal{J}_{CAE} = \sum_{\mathbf{x}\in\mathbf{X}} (L(\mathbf{x}, \mathbf{z}) + \alpha \|J_f(\mathbf{x})\|_F^2), \tag{6}$$

where $\|J_f(\mathbf{x})\|_F = \|\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}\|_F$ ($J \in \mathbb{R}^{d_h\times d_x}$) is the Frobenius norm of the Jacobian matrix.

For auto-encoders, the Jacobian matrix describes the first-order partial derivatives of the non-linear transformations from the input layer to the hidden layer. Since the Jacobian matrix is an indicator of non-linearity and linearity, the intuitive method to introduce mixed linearity is imposing the sparsity to the Jacobian matrix and minimizing the

following function,

$$\mathcal{J} = \sum_{\mathbf{x}\in\mathbf{X}}(L(\mathbf{x}, \mathbf{z}) + \alpha R(J_f(\mathbf{x}))), \tag{7}$$

where $R(J_f(\mathbf{x}))$ is the regularization term that introduces sparsity to the Jacobean matrix $J_f(\mathbf{x})$.

Solving the sparsity regularization $R(J_f(\mathbf{x}))$ is not an easy task since the sparsity norms bring non-smoothness to the loss function. Optimizing the objective function is computationally inefficient, especially when the sparsity regularization is a complex function form of the parameter matrix $\mathbf{W}$. Instead of directly endowing sparsity to the Jacobian matrix by sparsity norms, we consider an alternative method to approximate the regularization term. Since the goal is to replace some non-linear activation functions with linear functions, we minimize the loss between the first-order derivative $\frac{\partial hj}{\partial xi}$ and the weight $w_{i,j}$. The objective in (7) can be modified as follows:

$$\mathcal{J} = L(\mathbf{X}, \mathbf{Z}) + \lambda_1 \|J_f(\mathbf{X}) - \mathbf{W}^T\|_F^2 \tag{8}$$

The Jacobian matrix $J_f(\mathbf{X})$ describes the first-order derivatives and indicates the linearity and non-linearity of the activation functions. When the activation function is linear, the corresponding first-order derivative $\frac{\partial h_j}{\partial x_i}$ is equivalent to the weight $w_{i,j}$. $\|J_f(\mathbf{X}) - \mathbf{W}^T\|_F^2$ can consequently control the linearity of the model. In (8), the trade-off should be made between the approximation ability revealed by L and the flexibility of activation functions revealed by R. The model is more flexible since it adjusts the type of mappings and combines both linear and non-linear transformations.

We illustrate our derivations in a three-layer auto-encoder with the input size d_x and the hidden size d_h. Consider an input vector $\mathbf{x}$ and a hidden vector $\mathbf{h}$ with a sigmoid non-linearity ϕ. The ji-th element of the Jacobian matrix is:

$$\begin{aligned}\frac{\partial h_j}{\partial x_i} &= \frac{\partial \phi(w_{i,j}x_i)}{\partial x_i} \\ &= \frac{\partial \phi(w_{i,j}x_i)}{\partial w_{i,j}x_i}\frac{\partial w_{i,j}x_i}{\partial x_i} \\ &= h_j(1-h_j)w_{i,j}\end{aligned} \tag{9}$$

Then we further derive $\|J_f(\mathbf{X}) - \mathbf{W}^T\|_F^2$ following the contractive auto-encoder [4],

$$\begin{aligned}\|J_f(\mathbf{X}) - \mathbf{W}^T\|_F^2 &= \sum_i^{d_x}\sum_j^{d_h}(\frac{\partial h_j}{\partial x_i} - w_{i,j})^2 \\ &= \sum_j^{d_h}(h_j(1-h_j)-1)^2\sum_i^{d_x} w_{i,j}^2\end{aligned} \tag{10}$$

B. REGULARIZER FOR SPARSITY

The full-connected networks describe the relationships between the inputs and outputs in a complicated manner. However, the precise relationships also describe the sampling noises of inputs and lead to over-fitting [50]. The Dropout can relief the problem by randomly removing units in neural networks. Notice that Dropout also depends on advanced parameter settings. In order to further improve the model flexibility, we use a regularization based method that can also be treated as a feature selection process. The proposed regularizer brings sparsity to the connections and thus reduces the redundancy.

Generally, for the regularization purpose, the ridge penalty of the coefficients is the ℓ_2-norm which shrinks the parameters to small values with the same scale to avoid over-fitting. On the other hand, the LASSO penalty of the coefficients selects important coefficients and shrinks less important coefficients to zeros to introduce sparsity. For the weight matrices in neural networks, the structured sparsity induced norms that combine ℓ_1-norm and ℓ_2-norm are appropriate to learn robust and lightweight network structures. We mainly use two kinds of sparsity induced regularization terms in this paper, which are group LASSO [47] and exclusive LASSO [48].

Given a weight matrix $\mathbf{W}$, the group LASSO is defined as the $\ell_{2,1}$ norm of the matrix:

$$R_{2,1}(\mathbf{W}) = \|\mathbf{W}\|_{2,1} = \sum_{i=1}^{d_x}\|\mathbf{w}_i\|_2 \tag{11}$$

The $\ell_{2,1}$ norm induced regularizer for the matrix introduces row-wise sparsity where the rows are seen as groups. The elements in the same group will be all zeros or all non-zeros. For the weight matrix $\mathbf{W}$, the row-wise sparsity removes part of the input neurons [41]. It is natural to find that the effect of group sparsity is to mask the inputs by a regularizer, which is similar to the binary denoising auto-encoder that brings binary noises to the inputs. In this way, we can treat the auto-encoder with $\ell_{2,1}$ norm as a robust feature extractor in which the masking process is imposed in the weights rather than the inputs. However, the group sparsity regularizer is more flexible since the masking process is automatically determined during the learning process.

Similarly, the exclusive sparsity is defined as the $\ell_{1,2}$ norm of the matrix:

$$R_{1,2} = \|\mathbf{W}\|_{1,2} = \sum_{i=1}^{d_x}\|\mathbf{w}_i\|_1^2 \tag{12}$$

The way that $\ell_{1,2}$ norm generates sparsity is different from the $\ell_{2,1}$ norm which eliminates the entire rows of a matrix. The $\ell_{1,2}$ norm firstly applies the ℓ_1-norm in each row or group, which leads to sparsity within each row. Then the ℓ_2-norm is applied with respect to the vector of ℓ_1-norms for all groups to avoid entire rows going to zeros [51]. The exclusive LASSO encourages each row or group to have a certain amount of sparse values [40]. For input neurons, the $\ell_{1,2}$ norm only removes part of the connections to the hidden neurons.

The $\ell_{1,2}$ regularizer and $\ell_{2,1}$ regularizer both attempt to describe the hidden layer with fewer neurons. The group LASSO yields sparsity between groups rather than within groups [52]. The competition is between different groups for group LASSO while the competition is between variables within the group for exclusive LASSO [48], [51].

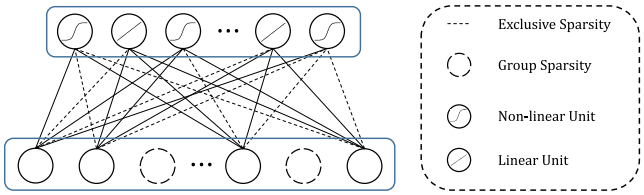


FIGURE 1. The illustration of the effects of the regularization terms. First, the group LASSO removes some input neurons. Second, the exclusive LASSO removes less important connections within the same group. Third, the mixed linear and non-linear mappings make the model more flexible.

We illustrate in Figure 1 the two kinds of regularization strategies used in the model. In summary, we can represent the model, MRAE, as the following objective problem,

$$\min_{\mathbf{W}} L(\mathbf{X},\mathbf{Z})+\lambda_1\|J_f(\mathbf{X})-\mathbf{W}^T\|_F^2+\lambda_2\|\mathbf{W}\|_{2,1}+\lambda_3\|\mathbf{W}\|_{1,2}, \tag{13}$$

where the first term is the auto-encoder loss, the second term is the regularizer which controls the linearity and non-linearity of our model, the last two terms are the regularizers for structured sparsity. λ_1, λ_2 and λ_3 are positive hyper-parameters.

There is a balance between the sparsity norms and the auto-encoder loss since the model removes weights that contribute little to the improvement of approximation. On the other hand, there is a balance between the sparsity norms and the linearity regularization term since the zeros of the weight matrix also affect the values of the Jacobian matrix.

IV. OPTIMIZATION

As mentioned before, the objective function required to be solved is formulated as follows:

$$\mathcal{J}=L(\mathbf{X},\mathbf{Z})+\lambda_1\|J_f(\mathbf{X})-\mathbf{W}^T\|_F^2+\lambda_2\|\mathbf{W}\|_{2,1}+\lambda_3\|\mathbf{W}\|_{1,2} \tag{14}$$

This optimization function can not be optimized by standard gradient descend method since it is not differentiable everywhere. Fortunately, there have been methods proposed to handle with such problems, among which the proximal gradient method draws our attention owning to its ability to deal with the non-smoothness and its remarkable convergent performance [44]. The basic operator of proximal methods is to compute the proximal operator of function, which itself involves solving a small convex optimization problem [44]. In details, when a minimization problem

$$\min_{\mathbf{W}} F(\mathbf{W})+\lambda\Theta(\mathbf{W}) \tag{15}$$

is required to be settled, where $F(\mathbf{W})$ is L-smooth (namely, gradient Lipschitz continuous) and convex, $\Theta(\mathbf{W})$ is convex and non-smooth, and $\lambda > 0$ is the hyper-parameter. The proximal gradient method iteratively minimizes it through the following formula:

$$\min_{\mathbf{W}} \Theta(\mathbf{W})+\frac{1}{2\lambda\eta}\|\mathbf{W}-(\mathbf{W}_t-\eta\nabla F(\mathbf{W}_t))\|_2^2, \tag{16}$$

where η is the step size and $\mathbf{W}_t$ represents the variable obtained after the last iteration. The solution of the above problem is written as $\mathbf{W}_{t+1}$ and will be used in the next iteration. Therefore, solving (15) turns to iteratively dealing with the optimization problem (16). The solution of (16) is called the proximal operator of function $\Theta(\mathbf{W})$ at point $\mathbf{W}_t-\eta\nabla F(\mathbf{W}_t)$, which is represented as $prox_{\lambda\eta\Theta}(\mathbf{W}_t-\eta\nabla F(\mathbf{W}_t))$. Then, the iterative process can be written as,

$$\mathbf{W}_{t+1}=prox_{\lambda\eta\Theta}(\mathbf{W}_t-\eta\nabla F(\mathbf{W}_t)). \tag{17}$$

In this paper, considering the non-smoothness of objective function (13), we propose to employ the proximal gradient method to solve it. We notice that the objective function is a composite model which can be separately considered as two parts,

$$\mathcal{J}=H_1(\mathbf{W})+H_2(\mathbf{W}), \tag{18}$$

where $H_1 = L(\mathbf{X},\mathbf{Z})+\lambda_1\|J_f(\mathbf{X})-\mathbf{W}^T\|_F^2$ and $H_2 = \lambda_2\|\mathbf{W}\|_{2,1}+\lambda_3\|\mathbf{W}\|_{1,2}$. $H_1(\mathbf{W})$ is a convex and smooth function. Introducing proximal gradient method to (18) involves in implementing a gradient step on $H_1(\mathbf{W})$ firstly. Specially, if we want to update the parameters from $\mathbf{W}_t$ to $\mathbf{W}_{t+1}$, the intermediate value after the gradient step is gained as follows:

$$\mathbf{W}^{m_t}=\mathbf{W}_t-\eta\nabla H_1(\mathbf{W}_t). \tag{19}$$

The gradient $\frac{\partial H_1}{\partial \mathbf{W}}$ can be calculated by standard back propagation with chain rules.

Next, we calculate the proximal operator of $\Theta(\mathbf{W})$ at point $\mathbf{W}^{m_t}$, i.e. $prox_{\eta H_2}(\mathbf{W}^{m_t})$. The $H_2(\mathbf{W})$ is the combination of two separate regularizers. Due to the special property of proximal operator, the proximal operator of $H_2(\mathbf{W})$ can be obtained by computing the proximal operator for each single regularizer and applying them in a row. That is to say,

$$prox_{\eta H_2}(\mathbf{W}^{m_t})=prox_{\lambda_3\eta\ell_{1,2}}(prox_{\lambda_2\eta\ell_{2,1}}(\mathbf{W}^{m_t})). \tag{20}$$

Solving $\ell_{2,1}$:

According to (16), to obtain the proximal operator of $\ell_{2,1}$ at a point $\mathbf{W}^0$, the following problem is required to be solved:

$$\min_{\mathbf{W}} \|\mathbf{W}\|_{2,1}+\frac{1}{2\lambda_2\eta}\|\mathbf{W}-\mathbf{W}^0\|_2^2. \tag{21}$$

The above function can be expanded as:

$$\min_{\mathbf{W}} \frac{1}{2\lambda_2\eta}\sum_i\sum_j(w_{i,j}-w_{i,j}^0)^2+\sum_i\|\mathbf{w}_i\|_2. \tag{22}$$

Then the optimization process is executed for each row. In details, the (22) can be optimized by solving the following problem for each i:

$$\min_{\mathbf{w}_i} \frac{1}{2\lambda_2\eta}\sum_j(w_{i,j}-w_{i,j}^0)^2+\|\mathbf{w}_i\|_2. \tag{23}$$

Since

$$\partial\|\mathbf{w}_i\|_2=\begin{cases}\dfrac{\mathbf{w}_i}{\|\mathbf{w}_i\|_2}, & if\ \mathbf{w}_i\neq 0\\ \{\mathbf{w}|\|\mathbf{w}\|_2\leq 1\}, & otherwise,\end{cases} \tag{24}$$

the sub-gradient of the objective function in (23) can be formulated as:

$$\frac{1}{\lambda_2\eta}(\mathbf{w}_i - \mathbf{w}_i^0) + \partial\|\mathbf{w}_i\|_2. \tag{25}$$

Let 0 belong to this sub-gradient. When $\mathbf{w}_i \neq 0$, we can get the following equation:

$$\left(1 + \frac{\lambda_2\eta}{\|\mathbf{w}_i\|_2}\right)\mathbf{w}_i = \mathbf{w}_i^0. \tag{26}$$

By computing the ℓ_2 norm of vectors on both sides, we get

$$\left(1 + \frac{\lambda_2\eta}{\|\mathbf{w}_i\|_2}\right)\|\mathbf{w}_i\|_2 = \|\mathbf{w}_i^0\|_2. \tag{27}$$

Therefore, the following equation can be obtained:

$$\|\mathbf{w}_i\|_2 = \|\mathbf{w}_i^0\|_2 - \lambda_2\eta. \tag{28}$$

Since a norm is always non-negative, the above equation is tenable only when $\|\mathbf{w}_i^0\|_2 - \lambda_2\eta \geq 0$. By replacing $\|\mathbf{w}_i\|_2$ in (26) with $\|\mathbf{w}_i^0\|_2 - \lambda_2\eta$, we can obtain:

$$w_{i,j} = w_{i,j}^0(1 - \frac{\lambda_2\eta}{\|\mathbf{w}_i^0\|_2}). \tag{29}$$

Considering the restriction $\|\mathbf{w}_i^0\|_2 - \lambda_2\eta \geq 0$, the proximal operator for $\ell_{2,1}$ is expressed as,

$$prox_{\lambda_2\eta\ell_{2,1}}(\mathbf{W}^0) = \begin{cases} w_{i,j}^0 - \lambda_2\eta\frac{w_{i,j}^0}{\|\mathbf{w}_i^0\|_2}, & \|\mathbf{w}_i^0\|_2 \geq \lambda_2\eta \\ 0, & otherwise. \end{cases} \tag{30}$$

Solving $\ell_{1,2}$:

Similarly, the proximal operator for $\ell_{1,2}$ can be obtained by optimizing the following problem:

$$\min_{\mathbf{W}} \frac{1}{2}\|\mathbf{W}\|_{1,2} + \frac{1}{2\lambda_3\eta}\|\mathbf{W} - \mathbf{W}^0\|_2^2. \tag{31}$$

The above function can be rewritten as:

$$\min_{\mathbf{W}} \frac{1}{2}\sum_i\left(\sum_j |w_{i,j}|\right)^2 + \frac{1}{2\lambda_3\eta}\sum_i\sum_j(w_{i,j} - w_{i,j}^0)^2. \tag{32}$$

It can be optimized for each row. Thus, the optimization problem above turns into solving

$$\min_{\mathbf{w}_i} \frac{1}{2}\left(\sum_j |w_{i,j}|\right)^2 + \frac{1}{2\lambda_3\eta}\sum_j(w_{i,j} - w_{i,j}^0)^2. \tag{33}$$

After computing the sub-gradient of this function and let 0 belong to the sub-gradient, we can get

$$\frac{1}{\lambda_3\eta}(w_{i,j} - w_{i,j}^0) + sign(w_{i,j})\|\mathbf{w}_i\|_1 = 0. \tag{34}$$

This equation is equivalent to

$$w_{i,j} = w_{i,j}^0 - \lambda_3\eta sign(w_{i,j})\|\mathbf{w}_i\|_1. \tag{35}$$

Since the aim of (31) is to find the point closer to $\mathbf{W}^0$ whose $\ell_{1,2}$ norm is lower simultaneously, we can use $w_{i,j}^0$ to approximate $w_{i,j}$ for each i and j when solving (35). Therefore, (35) can be approximated as,

$$w_{i,j} = (w_{i,j}^0 - \lambda_3\eta sign(w_{i,j}^0)\|\mathbf{w}_i^0\|_1)_+, \tag{36}$$

where $x_+ = \max(x, 0)$. Thus, the proximal operator of $\ell_{1,2}$ can be expressed as,

$$prox_{\lambda_3\eta\ell_{1,2}}(\mathbf{W}^0) = \begin{cases} (1 - \frac{\lambda_3\eta\|\mathbf{w}_i^0\|_1}{|w_{i,j}^0|})w_{i,j}^0, & \frac{w_{i,j}^0}{\|\mathbf{w}_\mathbf{i}^0\|_1} > \lambda_3\eta \\ 0, & otherwise. \end{cases} \tag{37}$$

To sum up, the optimization procedure of (13) firstly executes a gradient step on H_1 to obtain an intermediate value $\mathbf{W}^{m_t}$, then calculates the $prox_{\eta H_2}(\mathbf{W}^{m_t})$ according to (20) and repeats the process until convergence.

The complete algorithm is listed in Algorithm 1.

Algorithm 1 Training Algorithm for MRAE

Require: Initialized weight matrix of the encoder $\mathbf{W}$
Hyper-parameters λ_1, λ_2, λ_3
mini-batch size T
step size η
Ensure: Updated weight matrix $\mathbf{W}$
1: Sample T samples from dataset $\mathbf{X} = \{\mathbf{x_1}, \cdots, \mathbf{x_n}\}$
2: **repeat**
3: Update the parameters by standard stochastic gradient with respect to H_1 to obtain an intermediate value: $\mathbf{W}^{m_t} = \mathbf{W}_t - \eta\nabla H_1(\mathbf{W}_t)$
4: Calculate the proximal gradient to update: $\mathbf{W}^{\ell_{2,1}} = prox_{\lambda_2\eta\ell_{2,1}}(\mathbf{W}^{m_t})$ by (30)
5: Calculate the proximal gradient to update: $\mathbf{W}^{\ell_{1,2}} = prox_{\lambda_3\eta\ell_{1,2}}(\mathbf{W}^{\ell_{2,1}})$ by (37)
6: Set $\mathbf{W}_{t+1} = \mathbf{W}^{\ell_{1,2}}$
7: **until** convergence
8: Return the updated weight matrix $\mathbf{W}$

V. EXPERIMENTS

A. DATASETS

MNIST [53]. This dataset is a famous hand-written digit dataset which contains 50,000 training samples, 10,000 validation samples and 10,000 test samples. The MNIST is a gray-scale dataset with a size of 28×28 for each digit image.

MNIST-variants [4]. The MNIST-variants contain 5 datasets transformed from MNIST dataset. The MNIST-variants are more difficult than original MNIST. The MNIST-basic is the basic MNIST dataset without any transformation. The MNIST-bg-rand and MNIST-bg-img transform the MNIST by inserting random backgrounds or background images. The MNIST-rot rotates the MNIST with an angle between 0 to 2π. The MNIST-bg-img-rot is the combination of MNIST-bg-img and MNIST-rot.

MNIST-fashion [54]. This dataset is a fashion product dataset designed to replace MNIST. The MNIST-fashion follows the data organization of MNIST. It contains 70,000 gray-scale product images with the same train-valid-test split of MNIST. Each product can be classified into one of ten classes.

COIL-20 [55]. This is a gray-scale dataset that contains 1440 images from 20 different objects. Each object contains 72 32×32 images. The objects are rotated in every 5-degree interval with a fixed camera to obtain the images.

CIFAR10-bw [56]. The CIFAR-10 dataset is composed of 60,000 32×32 real-world images. Each image can be categorized into one of ten classes. The CIFAR10-bw is a gray-scale dataset transformed form CIFAR10.

SVHN-bw [57]. The SVHN is a digit dataset collected from real-world house numbers by Google Street View. It contains 73,257 training images and 26,032 test images. Each 32×32 image belongs to one of ten categories. Similar to CIFAR10-bw, all images in SVHN are transformed to gray-scale ones to formulate SVHN-bw.

B. SINGLE-LAYER MRAE

The single-layer auto-encoders only contain one hidden layer. They can also be seen as basic blocks for deeper neural networks. In this subsection, we test the performance of single-layer MRAE on MNIST dataset, MNIST-fashion dataset and COIL-20 dataset. Several classic auto-encoder frameworks are used as our baseline models.

- AE: Basic Auto-encoder without any regularization [58];
- w-AE: Weight-decay Auto-encoder [4];
- b-DAE: Binary Denoising Auto-encoder [17];
- CAE: Contractive Auto-encoder [4];
- RBM: Restricted Boltzmann Machine with binary hidden units.

First, we evaluate the proposed model by classification task in three datasets. Our model and all baseline models are initialized with d_h hidden units. After the unsupervised pre-training, the decoder layer is removed and a softmax layer is added for supervised fine tuning. The hyper-parameters that achieve best classification performance in the validation set are selected. The test errors of classification are then calculated in the test set.

In addition, the dimension of hidden units d_h reflects the representation capacity of the models. To demonstrate the stability and performance of MRAE, all models are tested with respect to different values of d_h. For each d_h, we run the training and test process mentioned above. The test errors for MNIST, MNIST-fashion and COIL-20 are drawn in Figure 2, Figure 3 and Figure 4 respectively. Since the input sizes of the three datasets are different, the hidden dimension has different vales in order to cover both over-complete and under-complete settings. For MNIST and MNIST-fashion, d_h varies among [32, 100, 200, 400, 600, 800, 1000]. For COIL-20, d_h varies among [32, 100, 200, 400, 600, 800, 1000, 1200].

MNIST. The results of MNIST are shown in Figure 2. It can be observed that the MRAE has achieved the best performance among all dimensions of hidden units. The results are very stable with the increasing of hidden dimensions. The auto-encoder with weight decay is competitive

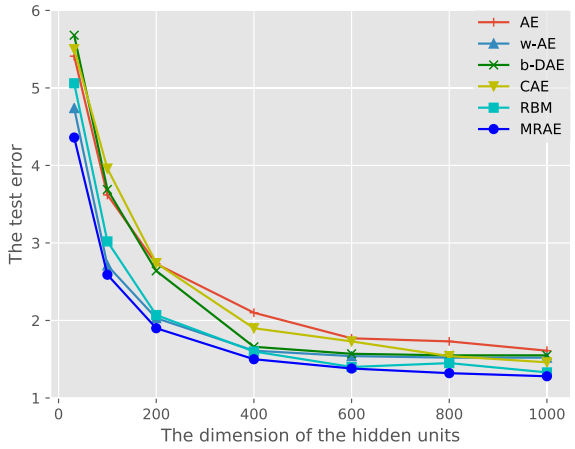


FIGURE 2. The test errors on MNIST dataset with the dimension of the hidden units varying among [32, 100, 200, 400, 600, 800, 1000].

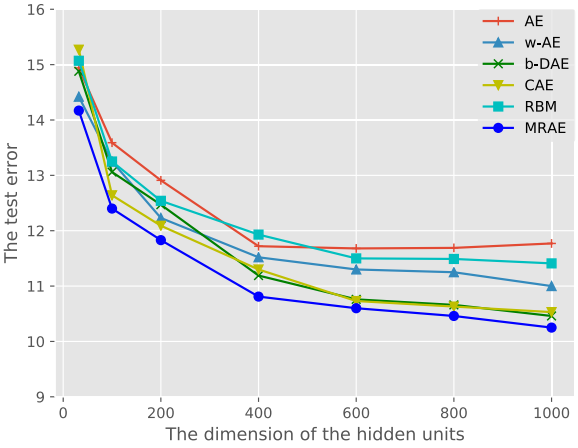


FIGURE 3. The test errors on MNIST-fashion dataset with the dimension of the hidden units varying among [32, 100, 200, 400, 600, 800, 1000].

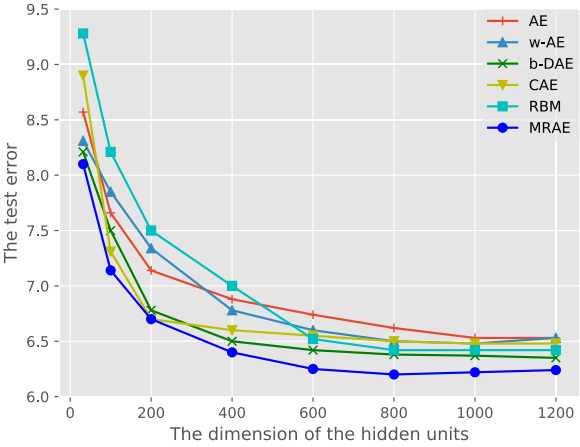


FIGURE 4. The test errors on COIL-20 dataset with the dimension of the hidden units varying among [32, 100, 200, 400, 600, 800, 1000, 1200].

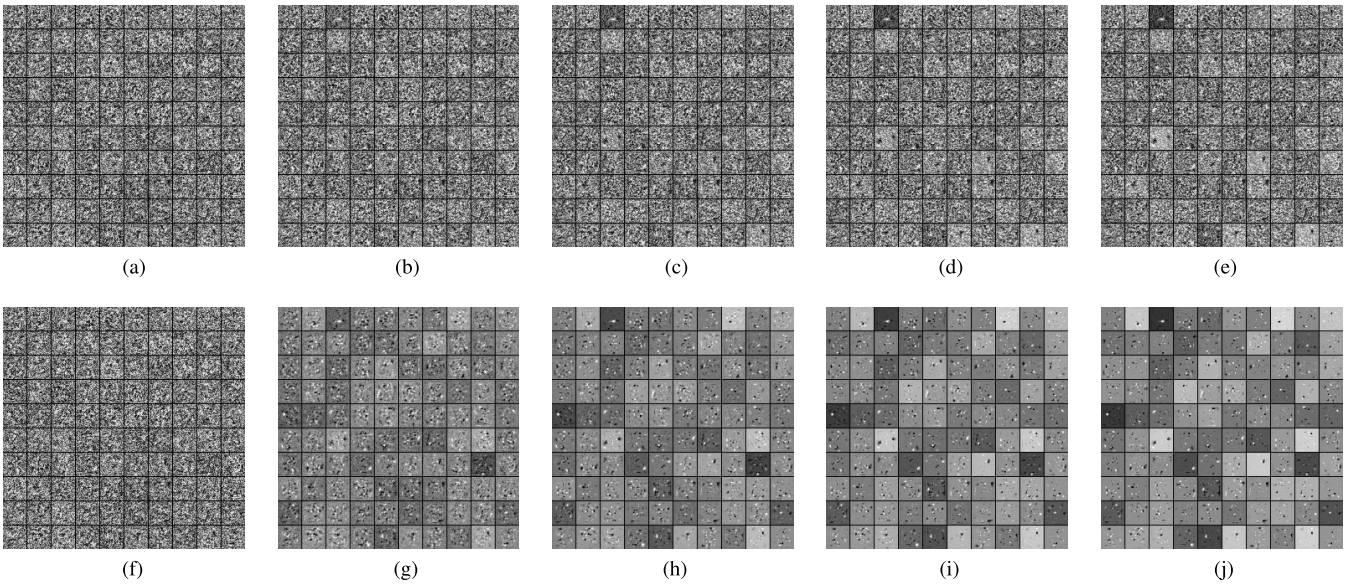


FIGURE 5. The weight matrix visualization during the train process for CAE and MRAE. CAE and MCAE are trained for 20 epoches. We choose the weight every 5 epoches. (a)-(e): CAE; (f)-(j): MRAE.

in low hidden dimensions. When training an over-complete auto-encoder, the weight decay regularizer is not capable to control the model capacity and the performance at the same time. Similarly, the RBM can achieve close results with MRAE in some dimensions of hidden units. However, in some dimensions of hidden units (e.g.$d_h = 32$), the MRAE is more efficient than RBM.

MNIST-fashion. The MRAE has obtained the best performance among all dimensions of hidden units as shown in Figure 3. The test errors of basic AE increase when the dimension of hidden units exceeds the dimension of data inputs. The DAE and CAE have achieved good performance in high dimensions of hidden units, which proves the effectiveness of regularized auto-encoders. The MRAE is able to learn stable features in both over-complete and under-complete settings.

COIL20. The results of COIL20 are drawn in Figure 4. The MRAE has achieved the best performance among all dimensions of hidden units. The test errors of DAE and CAE in some dimensions of hidden units are competitive compared with MRAE. The results of DAE, CAE and our model tend to be stable when d_h exceeds 600.

Second, we visualize the update of the weight matrix in MNIST dataset during the training process. We compare our method with CAE which also calculates the Jacobian matrix. We select the first 20 training epoches for illustration. By recording in every 5 epoches, the filters of the encoder are reported in Figure 5. It is obvious that the MRAE reduces the redundancy of the weights in a few training epoches. Consequently, the unsupervised training of MRAE is more likely to converge without over-fitting.

C. MULTIPLE-LAYER MRAE

In this part, we test the classification performance of multiple-layer MRAE in CIFAR-bw, SVHN-bw and 5 MNIST variant datasets. All comparative methods are built by their corresponding single-layer auto-encoder blocks.

- SAE-3: The three-layer stacked auto-encoder built by basic auto-encoders.
- b-DAE-3: The three-layer stacked auto-encoder built by binary denoising auto-encoders.
- DBN-3: The three-layer deep belief network built by Restricted Boltzmann Machines.
- CAE-2: The two-layer stacked auto-encoder built by contractive auto-encoders.
- R-Linearity-AE-2: The two-layer stacked auto-encoder built by auto-encoders with regularizer for linearity.
- R-Sparsity-AE-2: The two-layer stacked auto-encoder built by auto-encoders with regularizer for sparsity.

All models are pre-trained layer by layer. The test errors of classification are reported in Table 1. The best results are marked in bold. As shown in Table 1, the MRAE achieves lowest test errors among all 7 datasets. The MRAE can obtain competitive results even when the layers of the auto-encoder are not as deep as DBN, DAE and SAE. The results show that the MRAE is capable of capturing useful information of the inputs.

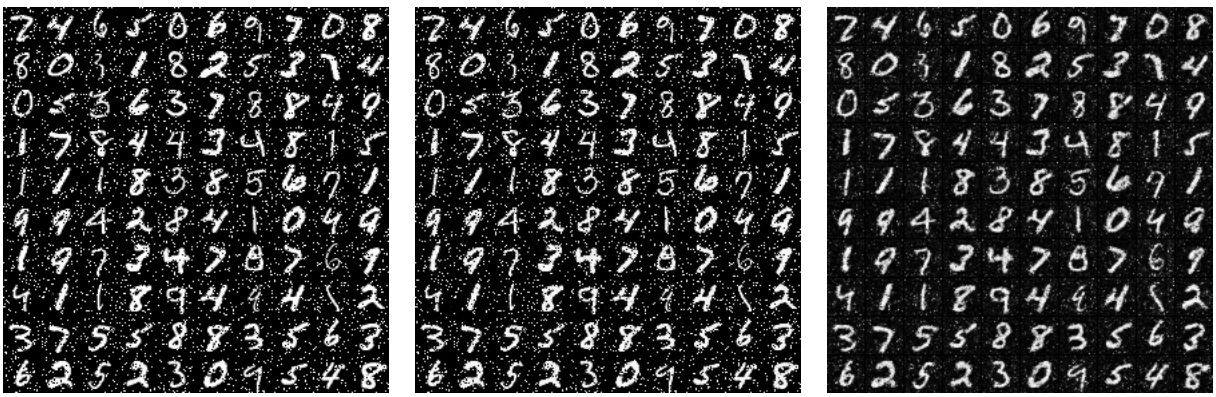
Notice that the auto-encoder only with sparsity regularizer has limited ability to extract features. The two-layer auto-encoders with the regularizer for sparsity achieve similar results as the three-layer stacked auto-encoders. On the other hand, the auto-encoders with regularizer for linearity achieve better performance than most baselines with fewer layers. The reason behind the results may come from the property of the regularizers. The sparsity regularizer describes a feature selection process which only decides the status of neuron connections. However, the linearity regularizer shapes the types of mapping functions and is more powerful to extract features.

TABLE 1. Test errors for classification.

Data Set	MRAE-2	R-Linearity-AE-2	R-Sparsity-AE-2	DBN-3	CAE-2	b-DAE-3	SAE-3
MNIST-basic	**2.32**	2.54	3.38	2.94	2.88	2.96	3.65
MNIST-bg-rand	**11.12**	11.43	12.94	11.31	11.49	11.23	12.37
MNIST-bg-img	**18.97**	20.30	22.45	26.14	19.42	19.25	27.83
MNIST-rot	**10.35**	11.56	12.60	10.97	11.08	10.74	11.78
MNIST-bg-img-rot	**48.68**	50.33	54.04	51.89	50.76	49.38	55.89
CIFAR-bw	**49.05**	51.73	54.44	56.25	50.09	49.86	54.71
SVHN-bw	**15.87**	16.52	16.97	19.32	17.29	16.20	16.91

(a) (b) (c)

FIGURE 6. The denoising results under Gaussian noise ($\mu = 0, \delta = 0.2$). (a): The corrupted digits; (b): The reconstruction results of MRAE; (c): The reconstruction results of AE.



(a) (b) (c)

FIGURE 7. The denoising results under Binary noise ($n_{corrupt} = 50$). (a): The corrupted digits; (b): The reconstruction results of MRAE; (c): The reconstruction results of AE.

D. DENOISING TASK

We test the denoising ability of MRAE blocks and AE blocks with one hidden layer. The single hidden layer auto-encoders are limited in capturing higher features. The denoising task gives us insights into the representation abilities of MRAE and AE. This experiment uses the MNIST digits for corruption. First, the original digits are corrupted with Gaussian noise. A random generator is used to generate noise that is subject to Gaussian distribution with mean μ and standard deviation δ. To avoid the overflow, the values of corrupted digits are clipped between 1 to 255. Second, following the procedure in [46], we introduce the binary corruption. For each digit image, $n_{corrupt}$ pixels are randomly selected as the corruption targets. The pixel value is set to be 0 if the original value is larger than 0.5. The pixel value is set to be 1 otherwise.

For MRAE and AE, the number of hidden layer units is set to be 1000. Both models are trained with corrupted images in an unsupervised way. The reconstruction results of Gaussian noise and binary noise are displayed in Figure 6, Figure 7, and Figure 8.

In Figure 6 and Figure 7, the denoising results under Gaussian noise ($\mu = 0, \delta = 0.2$) and binary noise ($n_{corrupt} = 50$) are reported. One hundred digits in the test set are randomly selected after the unsupervised training is completed. It can be observed that the MRAE converges and achieves better denoising results in a few training epoches. Moreover, in both noise settings, the MRAE always attempts to capture the

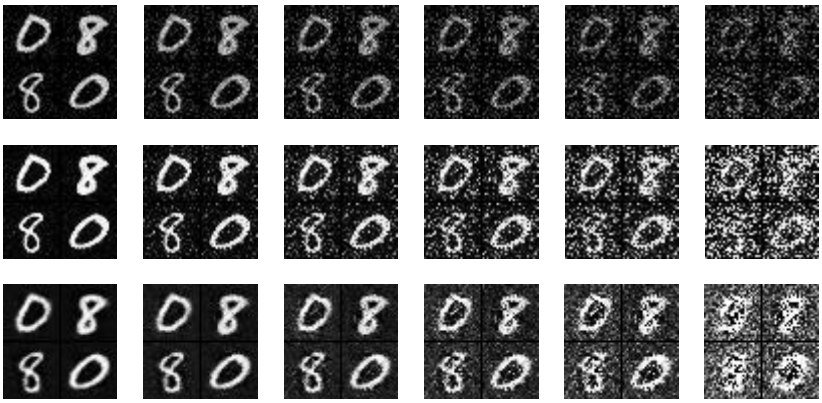


FIGURE 8. The denoising results in four digits under Gaussian noise by changing the δ among [0.1, 0.2, 0.3, 0.4, 0.5, 0.8] from left to right. The first row is the corrupted images, the second row is the reconstruction results of AE, and the third row is the reconstruction results of MRAE.

salient information in the digits while the AE only attempts to copy the inputs.

In Figure 8, the different levels of corruption are presented in the Gaussian noise setting. As the δ changes among [0.1, 0.2, 0.3, 0.4, 0.5, 0.8], four digits are selected for each δ. After δ exceeds 0.5, the single hidden layer auto-encoder can hardly denoise the inputs. It can be observed that the AE recovers the noises in the digits and the background. In other words, the AE augments the information in the inputs, no matter the original digits or the later added noises. On the contrary, the MRAE eliminates the unnecessary information in the background and recovers meaningful information. The training of MRAE is a process of feature selection and linear and non-linear adjustment, which makes the model more sensitive to the salient information.

VI. CONCLUSION

In this paper, we present a novel regularized auto-encoder framework MRAE. In the perspective of representation learning, the proposed model has several advantages. First, the basic MRAE block is a robust feature extractor which learns useful information from the original inputs. The experiments in the denoising task show that the MRAE attempts to capture the salient structures of the inputs instead of purely reconstruct the inputs. Second, the single hidden layer MRAE can be used as a pre-train tool for multiple-layer neural networks. The classification tasks in both single hidden layer and multiple hidden layers networks have shown the efficiency of the MRAE. Lastly, the MRAE is a structure flexible framework which contains fewer parameters than traditional auto-encoders. As shown in the visualization results of the weight matrix, most redundant parameters are removed in a few training epoches. Only mappings that are useful in capturing salient information are kept. The training of MRAE is consequently a process of feature selection, which makes the network more robust and flexible.

The idea of model robustness is not a trivial task, our future working directions focus on introducing the idea of mixed linear and non-linear activation functions in other neural network structures. In addition, the linearity of mappings in our model only works in two adjacent and connected layers. The MRAE can be extended to deeper networks and applied to handle unconnected layers. Finally, the sparsity induced norms in our model can be replaced by more coherent norms in the future works.

REFERENCES

[1] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 1096–1103.
[2] K. Jia, L. Sun, S. Gao, Z. Song, and B. E. Shi, "Laplacian auto-encoders: An explicit learning of nonlinear data manifold," *Neurocomputing*, vol. 160, pp. 250–260, Jul. 2015.
[3] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning*, vol. 1. Cambridge, MA, USA: MIT Press, 2016.
[4] S. Rifai, P. Vincent, X. Müller, X. Glorot, and Y. Bengio, "Contractive auto-encoders: Explicit invariance during feature extraction," in *Proc. 28th Int. Conf. Int. Conf. Mach. Learn.*, 2011, pp. 833–840.
[5] Y.-L. Boureau and Y. L. Cun, "Sparse feature learning for deep belief networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 1185–1192.
[6] J. Deng, Z. Zhang, E. Marchi, and B. Schuller, "Sparse autoencoder-based feature transfer learning for speech emotion recognition," in *Proc. Hum. Assoc. Conf. Affect. Comput. Intell. Interact.*, Sep. 2013, pp. 511–516.
[7] B. Du, X. Tang, Z. Wang, L. Zhang, and D. Tao, "Robust graph-based semisupervised learning for noisy labeled data via maximum correntropy criterion," *IEEE Trans. Cybern.*, to be published. doi: 10.1109/TCYB.2018.2804326.
[8] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
[9] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.
[10] Q. V. Le, "Building high-level features using large scale unsupervised learning," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2013, pp. 8595–8598.
[11] Y. Liao, Y. Wang, and Y. Liu, "Graph regularized auto-encoders for image representation," *IEEE Trans. Image Process.*, vol. 26, no. 6, pp. 2839–2852, Jun. 2017.
[12] C. Hong, J. Yu, J. Wan, D. Tao, and M. Wang, "Multimodal deep autoencoder for human pose recovery," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5659–5670, Dec. 2015.
[13] X. Feng, Y. Zhang, and J. Glass, "Speech feature denoising and dereverberation via deep autoencoders for noisy reverberant speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2014, pp. 1759–1763.

[14] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 1225–1234.
[15] F. Tian, B. Gao, Q. Cui, E. Chen, and T.-Y. Liu, "Learning deep representations for graph clustering," in *Proc. AAAI*, 2014, pp. 1293–1299.
[16] Z. Wu, Y. Xiong, X. Y. Stella, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3733–3742.
[17] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, no. 12, pp. 3371–3408, Dec. 2010.
[18] J. Li, T. Zhang, W. Luo, J. Yang, X.-T. Yuan, and J. Zhang, "Sparseness analysis in the pretraining of deep neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 6, pp. 1425–1438, Jun. 2017.
[19] A. Makhzani and B. Frey, "K-sparse autoencoders," in *Proc. Int. Conf. Learn. Represent.*, 2014, pp. 1–9.
[20] S. Rifai *et al.*, "Higher order contractive auto-encoder," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, 2011, pp. 645–660.
[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[22] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *Proc. Int. Conf. Artif. Neural Netw.*, 2011, pp. 52–59.
[23] B. Du, W. Xiong, J. Wu, L. Zhang, L. Zhang, and D. Tao, "Stacked convolutional denoising auto-encoders for feature representation," *IEEE Trans. Cybern.*, vol. 47, no. 4, pp. 1017–1027, Apr. 2016.
[24] W. Luo, J. Li, J. Yang, W. Xu, and J. Zhang, "Convolutional sparse autoencoders for image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 7, pp. 3289–3294, Jul. 2018.
[25] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.
[26] M. Norouzi, M. Ranjbar, and G. Mori, "Stacks of convolutional restricted boltzmann machines for shift-invariant feature learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 2735–2742.
[27] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 609–616.
[28] D. P. Kingma and M. Welling. (2013). "Auto-encoding variational Bayes." [Online]. Available: https://arxiv.org/abs/1312.6114
[29] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1278–1286.
[30] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.
[31] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *CoRR*, vol. abs/1511.06434, pp. 1–16, Nov. 2015.
[32] Y. Bengio, L. Yao, G. Alain, and P. Vincent, "Generalized denoising auto-encoders as generative models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 899–907.
[33] Y. Chen and M. J. Zaki, "Kate: K-competitive autoencoder for text," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 85–94.
[34] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–16.
[35] N. Shervashidze and F. Bach, "Learning the structure for structured sparsity," *IEEE Trans. Signal Process.*, vol. 63, no. 18, pp. 4894–4902, Sep. 2015.
[36] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *J. Roy. Statist. Soc., B (Statist. Methodol.)*, vol. 68, no. 1, pp. 49–67, 2006.
[37] L. Yuan, J. Liu, and J. Ye, "Efficient methods for overlapping group lasso," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 352–360.
[38] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Stat. Soc. B (Methodol.)*, vol. 58, no. 1, pp. 267–288, 1996.
[39] R. Jenatton, J.-Y. Audibert, and F. Bach, "Structured variable selection with sparsity-inducing norms," *J. Mach. Learn. Res.*, vol. 12, pp. 2777–2824, Feb. 2011.
[40] Y. Zhou, R. Jin, and S. C.-H. Hoi, "Exclusive lasso for multi-task feature selection," in *Proc. 13th Int. Conf. Artif. Intell. Stat.*, 2010, pp. 988–995.
[41] J. Yoon and S. J. Hwang, "Combined group and exclusive sparsity for deep neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 3958–3966.
[42] J. M. Alvarez and M. Salzmann, "Learning the number of neurons in deep networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2270–2278.
[43] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2074–2082.
[44] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Optim.*, vol. 1, no. 3, pp. 127–239, Jan. 2014.
[45] B. Du, S. Wang, C. Xu, N. Wang, L. Zhang, and D. Tao, "Multi-task learning for blind source separation," *IEEE Trans. Image Process.*, vol. 27, no. 9, pp. 4219–4231, Sep. 2018.
[46] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 665–674.
[47] F. Nie, H. Huang, X. Cai, and C. H. Ding, "Efficient and robust feature selection via joint $\ell_{2,1}$-norms minimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 1813–1821.
[48] D. Kong, R. Fujimaki, J. Liu, F. Nie, and C. Ding, "Exclusive feature learning on arbitrary structures via $\ell_{1,2}$-norm," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1655–1663.
[49] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 807–814.
[50] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.
[51] F. Campbell and G. I. Allen, "Within group variable selection through the exclusive lasso," *Electron. J. Statist.*, vol. 11, no. 2, pp. 4220–4257, 2017.
[52] N. Simon, J. Friedman, T. Hastie, and R. Tibshirani, "A sparse-group lasso," *J. Comput. Graph. Statist.*, vol. 22, no. 2, pp. 231–245, May 2012.
[53] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[54] H. Xiao, K. Rasul, and R. Vollgraf. (2017). "Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms." [Online]. Available: https://arxiv.org/abs/1708.07747
[55] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (COIL-20)," Dept. Comput. Sci., Columbia Univ., New York, NY, USA, Tech. Rep. CUCS-006-96, 1996.
[56] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009, vol. 1, no. 4.
[57] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *Proc. NIPS Workshop Deep Learn. Unsupervised Feature Learn.*, no. 2, 2011, p. 5.
[58] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 153–160.

YONG SHI received the B.S. degree in mathematics from the Southwest Petroleum Institute, Chengdu, China, in 1982, and the Ph.D. degree in management science from The University of Kansas, Lawrence, KS, USA, in 1991. He is currently the Director of the Key Laboratory of Big Data Mining and Knowledge Management, and the Director of the Research Center on Fictitious Economy and Data Science, Chinese Academy of Sciences, Beijing, China. His current research interests include data mining and multiple criteria decision making.

MINGLONG LEI received the M.S. degree in computer science and technology from The 6th Research Institute of China Electronics Corporation, Beijing, China, in 2016. He is currently pursuing the Ph.D. degree with the Chinese Academy of Sciences, Beijing. His current research interests include machine learning and data mining.

RONGRONG MA received the B.S. degree from the China University of Mining and Technology, Beijing, China, in 2016. She is currently pursuing the M.S. degree with the School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing. Her current research interests include optimization and machine learning.

LINGFENG NIU received the B.S. degree in mathematics from Xi'an Jiaotong University, Xi'an, China, in 2004, and the Ph.D. degree in mathematics from the Chinese Academy of Sciences, Beijing, China, in 2009, where she has been an Associate Professor with the Key Laboratory of Big Data Mining and Knowledge Management, since 2009. Her current research interests include optimization and machine learning.