SPARSE REPRESENTATION LEARNING OF DATA BY AUTOENCODERS WITH $L_{1/2}$ REGULARIZATION

F. Li[*], *J.M. Zurada*[†], *W. Wu*[*]

Abstract: Autoencoder networks have been demonstrated to be efficient for unsupervised learning of representation of images, documents and time series. Sparse representation can improve the interpretability of the input data and the generalization of a model by eliminating redundant features and extracting the latent structure of data. In this paper, we use $L_{1/2}$ regularization method to enforce sparsity on the hidden representation of an autoencoder for achieving sparse representation of data. The performance of our approach in terms of unsupervised feature learning and supervised classification is assessed on the MNIST digit data set, the ORL face database and the Reuters-21578 text corpus. The results demonstrate that the proposed autoencoder can produce sparser representation and better reconstruction performance than the Sparse Autoencoder and the L_1 regularization Autoencoder. The new representation is also illustrated to be useful for a deep network to improve the classification performance.

Key words: *autoencoder, sparse representation, unsupervised feature learning, deep network, $L_{1/2}$ regularization*

Received: June 12, 2017 **DOI:** 10.14311/NNW.2018.28.008
Revised and accepted: April 3, 2018

1. Introduction

Unsupervised feature learning can extract the hidden structure of data and learn the representation that produces better performance. The autoencoder (AE) network is one of the popular algorithms for unsupervised feature learning, which is efficient for learning feature representation of the image, document and speech data [1–4]. Various kinds of AEs [2, 5–7] have been proposed by enforcing different constraints on the network to achieve different representations [2, 6, 8].

AEs can also be used to construct a deep network working as a representation-learning method. A deep network uses multi-level non-linear modules to transform the representation from one level (beginning with the raw input) to a representation at a higher and more abstract level [9]. With the representation at enough

[*]Feng Li; Wei Wu – Corresponding author; School of Mathematical Sciences, Dalian University of Technology, Dalian, Liao Ning province, China, E-mail: dscling@126.com, wuweiw@dlut.edu.cn

[†]Jacek M. Zurada; Department of Electrical and Computer Engineering, University of Louisville, Louisville, KY, USA and Information Technology Institute, University of Social Science, Łódz, Poland, E-mail: jacek.zurada@louisville.edu

high and abstract level, the highly nonlinear complex functions can be learned compactly. Many recent theoretical and empirical researches on machine learning have demonstrated that deep networks can produce good generalization performance in detection, prediction or recognition tasks [10–14]. However, initially, deep networks were trained in purely supervised mode, and the results were often found to be worse than shallow networks. This is due to that the gradient-based optimization starting from a random initialization frequently gets stuck near poor solutions [1]. A greedy layer-wise learning algorithm was presented in [11] to choose an appropriate initialization for the deep network.

To suitably initialize a deep AE network, the greedy layer-wise algorithm trains the first AE to minimize the reconstruction error of the raw data in unsupervised mode, followed by training subsequent AE with the hidden activations of previous AE as input. Then, the last hidden activations are taken as input to train a supervised layer. Finally, the algorithm fine-tunes all parameters of this deep network with supervised mode for good performance [1, 11]. Each AE learns a representation of its input (the raw data or the previous hidden activations) at hidden layer. Therefore, an AE learning a more accurate representation can yield a better initialization of corresponding parameters of the deep network.

Many useful constraints [2, 5–8] have been proposed for an AE to learn good representation. One of the common constraints is the sparsity for a sparse representation. The sparsity has attracted more and more attention in machine learning, especially for big data problem. Better sparsity can decrease the computational complexity and improve the accuracy of pattern extracting. Sparse representation in machine learning is inspired by the observation of the sparse representation in the brain: Only around 1 %–4 % of the neurons are active at a given time [15]. In machine learning, sparse representation means that only few modules of the model are active at a given time. Sparse representation is frequently used to improve the interpretability of the input data and the generalization of the model, by eliminating the useless features and extracting the latent structure of data [16, 17]. The well-known sparse autoencoder (SAE) proposed in [6] employs the Kullback-Leibler (KL) divergence function [18] to enforce the sparsity on the activations of hidden nodes.

As a tool for sparsity, L_0 regularization method can produce the sparsest result. But L_0 regularization involves solving an NP-hard optimization problem. Fortunately, it has been shown that L_1 regularization (Lasso [19]) is a good approximation to L_0 regularization [20, 21], while the convexity of L_1 regularization makes the corresponding optimization problem easy to solve. The Lasso method and its variants [22–24] have made L_1 regularization become a popular data analysis algorithm. Later on, an $L_{1/2}$ regularization method was proposed in [25–27], which has some promising properties. Fig. 1 illustrates the sparsification mechanism of $L_{1/2}$, L_1 and L_2 regularizers. As shown in Fig. 1, the sparsity solution is the first place at which the contours touch the constraint region, and this will occur at a corner corresponding to a zero coefficient. It is obvious that the solution of $L_{1/2}$ regularizer occurs at a corner with a higher possibility, implying that it leads to sparser solution compared with L_1 and L_2 regularizers.

In this paper, we use $L_{1/2}$ regularization to enforce sparsity on the activation of hidden nodes for a sparse representation. To this end, we introduce an $L_{1/2}$

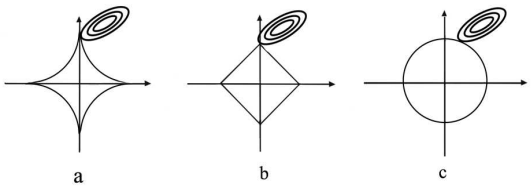


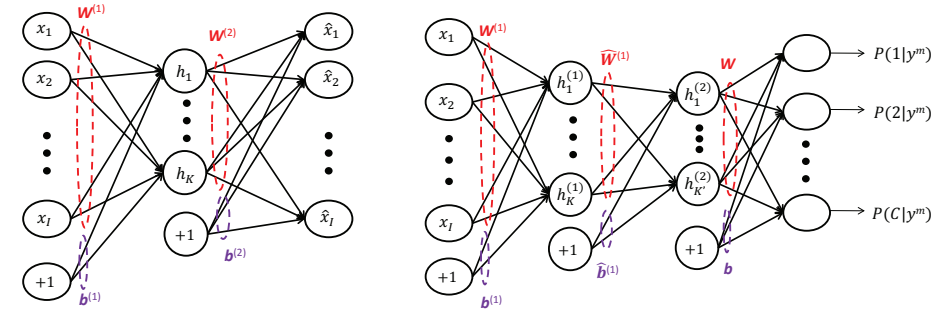
Fig. 1 *Sparsification mechanism of (a) $L_{1/2}$, (b) L_1 and (c) L_2 regularizers.*

regularizer term into the reconstruction error function in the unsupervised learning to drive some hidden activations to zero. The most closely related work to ours is that of Jiang et al. on L_1 regularization AE (L_1-AE) [28], in which L_1 regularization is used to enforce sparsity on the hidden representation. Numerical experiments have been performed on two standard image data sets (the MNIST data set [29] and the ORL database [30]), and one text data set (the Reuters-21578 corpus), to show the efficiency of the algorithm in terms of classification performance and unsupervised feature learning, such as the feature filters, the sparsity of the weights and the reconstruction performance.

The rest of this paper is organized as follows: In Section 2, the algorithm and related notations are described. Supporting numerical experiments are presented in Section 3. Relevant conclusions are given in Section 4.

2. Method

Consider an AE with I input nodes, K hidden nodes and I output nodes as shown in Fig. 2(a). Let $\mathbf{W}^{(1)} \in \mathbf{R}^{K\times I}$ and $\mathbf{W}^{(2)} \in \mathbf{R}^{I\times K}$ be the encoder and decoder weight matrix, respectively. Similarly, let $\mathbf{b}^{(1)} \in \mathbf{R}^{K\times 1}$ and $\mathbf{b}^{(2)} \in \mathbf{R}^{I\times 1}$ be the encoder and decoder biases, respectively. $g : \mathbf{R} \to \mathbf{R}$ and $f : \mathbf{R} \to \mathbf{R}$ represent the given activation functions of hidden layer and output layer, respectively. For an input vector $\mathbf{x} \in \mathbf{R}^{I\times 1}$, an AE network is aimed at learning to approximate the



(a) autoencoder (b) deep network

Fig. 2 *Structures of an AE and a deep network.*

identity function

$$\hat{\mathbf{x}} = F_\theta(\mathbf{x}) = \mathbf{x}, \tag{1}$$

where $\theta = \{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)}\}$. The AE reconstructs the input $\mathbf{x}$ in terms of following neural network

$$\mathbf{h} = g(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}),\ \hat{\mathbf{x}} = f(\mathbf{W}^{(2)}\mathbf{h} + \mathbf{b}^{(2)}), \tag{2}$$

where $\mathbf{h} \in \mathbf{R}^{K\times 1}$ is the hidden representation, and $\hat{\mathbf{x}} \in \mathbf{R}^{I\times 1}$ is the output. For a given training set $\{\mathbf{x}^m\}_{m=1}^M$, the cost function of AE is the average reconstruction error

$$J_{AE}(\theta) = \frac{1}{M}\sum_{m=1}^{M}\frac{1}{2}\|\hat{\mathbf{x}}^m - \mathbf{x}^m\|^2. \tag{3}$$

As mentioned above, an AE can generate different representations from data by enforcing different constraints on its parameters and network structure, such as constraining the number K of the hidden nodes. Sparse representation is frequently used to improve the interpretability of the input data and the generalization of the model. It can be achieved by constraining the average hidden representation $\mathbf{h}$ with the KL divergence function. Accordingly, the cost function of a SAE (see [6] for more details) is

$$J_{SAE}(\theta) = J_{AE}(\theta) + \beta\sum_{k=1}^{K}\left(\rho\log\frac{\rho}{\hat{h}_k} + (1-\rho)\log\frac{1-\rho}{1-\hat{h}_k}\right) + \frac{\alpha}{2}(\|\mathbf{W}^{(1)}\|^2 + \|\mathbf{W}^{(2)}\|^2), \tag{4}$$

where ρ is a small positive constant selected as a sparsity parameter, $\hat{h}_k = \frac{1}{M}\sum_{m=1}^{M} h_k^m$ h_k^m is the average activation of hidden node k over all training samples, h_k^m is the activation of hidden node k with respect to the input $\mathbf{x}^m$, β and α are the parameters of the sparsity penalty term and the weight decay term, respectively.

In this paper, we use $L_{1/2}$ regularization instead of KL divergence function to enforce sparsity constraint on $\hat{h}_k$. Since logistic sigmoid function is used as the hidden layer activation function, $\hat{h}_k$ is positive. The $L_{1/2}$ regularizer ($L_{1/2}$R) of $\hat{h}_k$ is $J_{L_{1/2}R}(\theta) = \sum_{k=1}^{K}(\hat{h}_k)^{1/2}$. A weight decay term as shown in Eq. (4) is used to prevent over-fitting [31]. Therefore, the cost function of the $L_{1/2}$ regularization autoencoder ($L_{1/2}$AE) is defined as

$$J_{L_{1/2}AE}(\theta) = J_{AE}(\theta) + \beta\sum_{k=1}^{K}(\hat{h}_k)^{1/2} + \frac{\alpha}{2}(\|\mathbf{W}^{(1)}\|^2 + \|\mathbf{W}^{(2)}\|^2). \tag{5}$$

We apply the batch gradient descent method [32–35] to solve the resulting optimization problem. The parameters are updated by

$$\theta = \theta - \eta\frac{\partial}{\partial\theta}J_{L_{1/2}AE}(\theta), \tag{6}$$

where $\eta > 0$ is the learning rate. The gradients of parameters are computed as below.

$$\frac{\partial}{\partial\theta}J_{L_{1/2}AE}(\theta) = \frac{\partial}{\partial\theta}J_{AE}(\theta) + \beta\frac{\partial}{\partial\theta}J_{L_{1/2}R}(\theta) + \alpha(\mathbf{W}^{(1)} + \mathbf{W}^{(2)}). \tag{7}$$

In [28], the L_1-AE was proposed with the cost function as follows:

$$J_{L_1-AE}(\theta) = J_{AE}(\theta) + \beta \sum_{k=1}^{K} \hat{h}_k + \frac{\alpha}{2}\|\theta\|^2. \tag{8}$$

In our paper, we use the greedy layer-wise training algorithm to construct a deep $L_{1/2}$AE network (see Fig. 2(b)) for classification. In this algorithm, the hidden representation of the last pre-trained $L_{1/2}$AE are taken as input to train a supervised layer. We choose the softmax regression classifier [36] as the supervised layer. If the dimensionality of the last hidden representation is s_L and the training samples fall into C categories, let $\mathbf{W} \in \mathbf{R}^{s_L \times C}$ and $\mathbf{b} \in \mathbf{R}^{C \times 1}$ be weights and biases of softmax layer, respectively. The misclassification cost function of the softmax regression classifier with a weight decay is defined as

$$J(\mathbf{W}, \mathbf{b}) = -\frac{1}{M}\Big(\sum_{m=1}^{M} \sum_{c=1}^{C} 1(y^m = c) \log \frac{\exp\left(\mathbf{W}_c^T \tilde{\mathbf{h}}^m + b_c\right)}{\sum_{j=1}^{C} \exp\left(\mathbf{W}_j^T \tilde{\mathbf{h}}^m + b_j\right)} \Big) + \frac{\gamma}{2}\|\mathbf{W}\|^2, \tag{9}$$

where γ is the parameter of the weight decay term of softmax regression classifier, and $\tilde{\mathbf{h}}^m$ and y^m represent the last hidden representation and the label corresponding to $\mathbf{x}^m$, respectively.

The limited-memory BFGS (L-BFGS) quasi-Newton algorithm [37] is used to minimize Eq. (4), Eq. (5), Eq. (8) and Eq. (9).

3. Numerical experiments

This section discusses the performance of our method in feature learning and prediction learning with the MNIST data set, the ORL database and the Reuters-21578 corpus. $L_{1/2}$AEs are trained on all three data sets to demonstrate the ability of feature extraction. For comparison, the corresponding experiments of SAE and L_1-AE are also be performed. Deep $L_{1/2}$AE networks are trained on the MNIST data set and the Reuters-21578 corpus by using the greedy layer-wise algorithm to investigate the classification performance, which is compared with deep networks builded with L_1-AE, SAE, Denoising Autoencoder (DAE) [2], and Dropout Autoencoder (DpAE) [38].

For the sake of easy comparison, we follow the setting in [28] for choosing the parameters as in Tab. I. Similarly, we follow the setting in [12] such that the AEs have 196, 100 and 15 hidden nodes for the MNIST data set, the ORL database and the Reuters-21578 corpus, respectively. The maximum number of iterations for training all networks is 400 when using the L-BFGS quasi-Newton method.

3.1 MNIST digit dataset

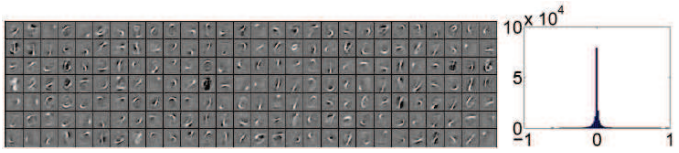
First, we train an $L_{1/2}$AE on the MNIST data set which is a subset of a larger data set NIST. There are 60,000 training and 10,000 testing images in this data set. All images are black and white handwritten digits which are size normalized, and centered in a fixed size image where the center of gravity of the intensity is at the center of the 28×28 pixel box. All the image matrices are reshaped as vectors with

Parameters	SAE	L_1-AE	$L_{1/2}$AE
Sparsity penalty (β)	3	0.01	0.01
Sparsity Parameter (ρ)	0.1	-	-
Weight decay penalty (α)	0.003	0.003	0.003
Weight decay penalty (γ)	0.0001	0.0001	0.0001

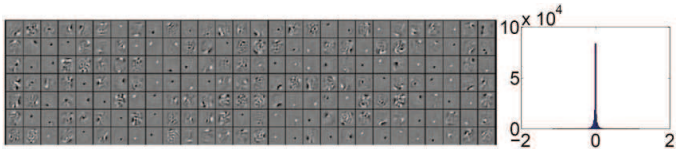
Tab. I *Parameters of all algorithms.*

784 components (pixels). The network contains 784 input nodes and 196 hidden nodes. We use sigmoidal function as activation function for both the hidden and output layer.

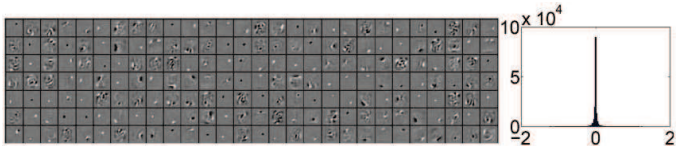
For visualization of the learned features of the images, the weight vectors connected to hidden nodes (the rows of the encoder weight matrix $\mathbf{W}^{(1)}$) are reshaped as a 28×28 matrix and shown in the form of images with 28×28 pixels named receptive fields. Similarly, the columns of the decoder weight matrix $\mathbf{W}^{(2)}$ are also shown as images named decoder filters. We compare the receptive fields, decoder filters and histogram of weights of $L_{1/2}$AE on MNIST data set with those of SAE and L_1-AE in Fig. 3 and Fig. 4. In Fig. 3, it can be seen that the features learned by SAE are whole blurred digits or parts of digits. And the receptive fields of L_1-AE and $L_{1/2}$AE show that many features are compressed to smaller parts of digits such as dots, and that more features of $L_{1/2}$AE compressed as dots. Some other



(a) SAE

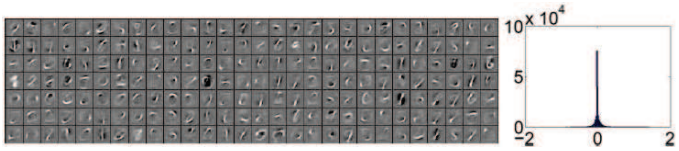


(b) L_1-AE

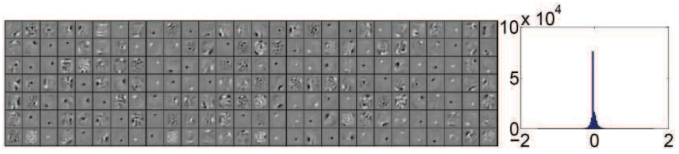


(c) $L_{1/2}$AE

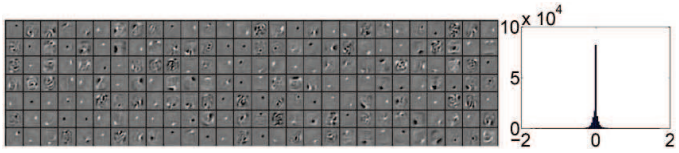
Fig. 3 *Visualization of 196 receptive fields ($\mathbf{W}^{(1)}$) and weight histogram of (a) SAE, (b) L_1-AE and (c) $L_{1/2}$AE for the MNIST data set. Black pixels and white pixels mean negative and positive weights, respectively.*



(a) SAE



(b) L_1-AE



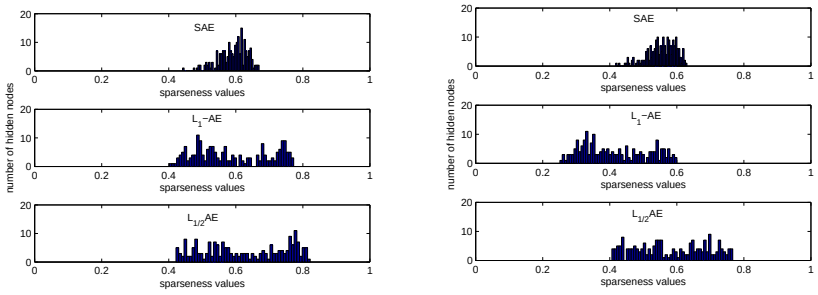
(c) $L_{1/2}$AE

Fig. 4 *Visualization of 196 decoder filters* ($\mathbf{W}^{(2)}$) *and weight histogram of (a) SAE, (b)* L_1*-AE and (c)* $L_{1/2}$*AE for the MNIST data set.*

features are split into several small pieces. The weight histograms of the encoder weights of SAE, L_1-AE and $L_{1/2}$AE show that $L_{1/2}$AE forces more weights near zero. In Fig. 4, the decoder filters and the weight histograms have similar characteristic. To further evaluate the sparsity of weights of $L_{1/2}$AE, we compute the sparseness proposed in [39] of the receptive fields $\mathbf{W}^{(1)}$ and decoder filters $\mathbf{W}^{(2)}$ by using the formula

$$sparseness(\mathbf{x}) = \frac{\sqrt{n} - (\sum_{i=1}^{n} |x_i|)/(\sqrt{\sum_{i=1}^{n} x_i^2})}{\sqrt{n} - 1}, \tag{10}$$

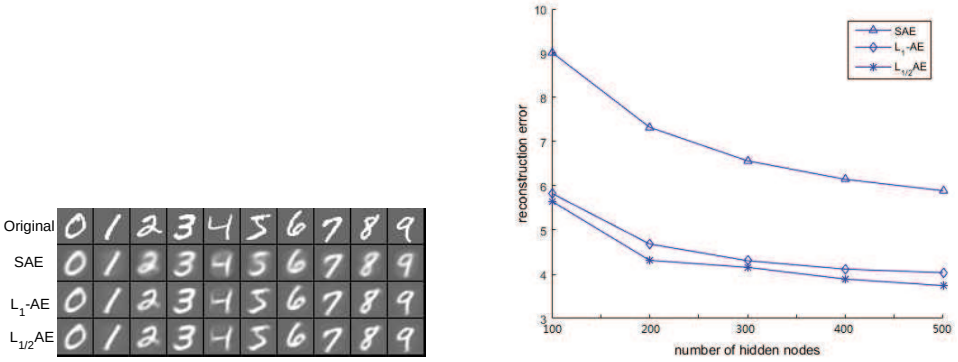
where $\mathbf{x}$ is a vector and n is the dimension of $\mathbf{x}$. The bigger $sparseness(\mathbf{x})$ is, the sparser $\mathbf{x}$ is. Especially, $sparseness(\mathbf{x}) = 1$ means that only one component of $\mathbf{x}$ is non-zero, while $sparseness(\mathbf{x}) = 0$ means all components of $\mathbf{x}$ are equal (up to signs). We compute the sparseness degrees of the 196 receptive fields and the 196 decoder filters, and display them in the histograms in Fig. 5. From Fig. 5, we can see that the sparseness degrees of the receptive fields and decoder filters of $L_{1/2}$AE are bigger, on average, than those of L_1-AE. For the comparison between SAE and $L_{1/2}$AE, in general, some receptive fields and decoder filters of $L_{1/2}$AE have sparseness degree larger than the largest one of SAE. In particular, for the receptive fields, the largest sparseness degree of SAE and $L_{1/2}$AE is 0.6698 and 0.8222, respectively, and there are 84 out of 196 $L_{1/2}$AE's receptive fields with sparseness degree larger than 0.6698. For the decoder filters, the largest sparseness degree of SAE and $L_{1/2}$AE is 0.6302 and 0.7670, respectively, and there are 79 out of 196 $L_{1/2}$AE's decoder filters with sparseness degree larger than 0.6302. Therefore, the $L_{1/2}$AE improves the sparsity of weights of the AE.



(a) receptive fields (b) decoder filters

Fig. 5 *Spareness histogram computed by Eq.* (10) *on (a) 196 receptive fields* $\mathbf{W}^{(1)}$ *and (b) 196 decoder filters* $\mathbf{W}^{(2)}$ *from MNIST data set.*

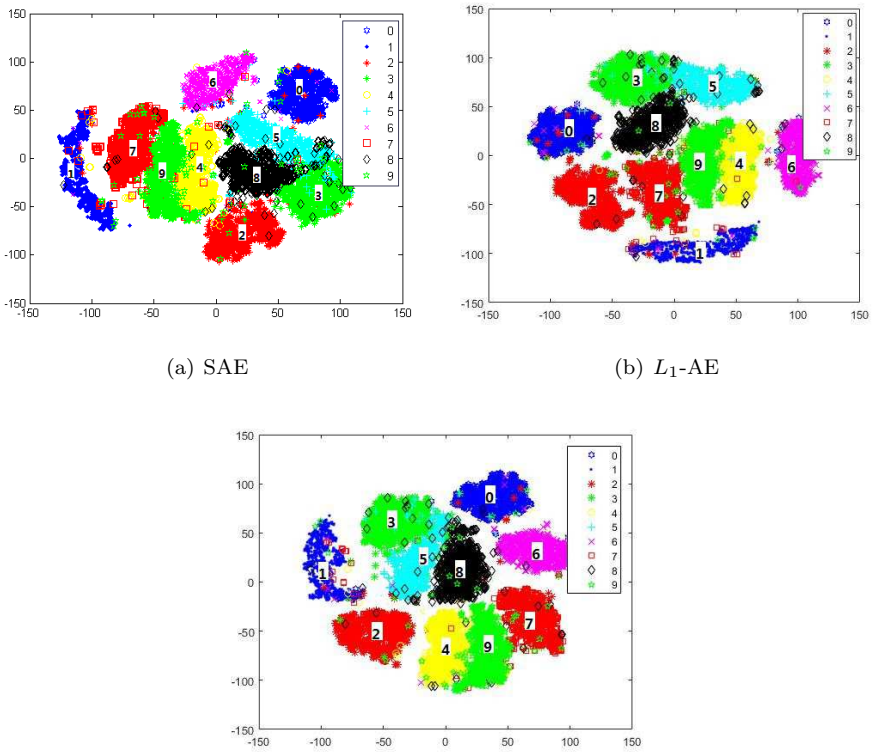
To investigate the reconstruction performance of $L_{1/2}$AE, we test the digit reconstruction with ten chosen digits and compute the reconstruction errors over all testing digits by using $L_{1/2}$AE with different numbers of hidden nodes. A comparison of $L_{1/2}$AE, L_1-AE and SAE is given in Fig. 6. In Fig. 6(a), the original digits and the digits reconstructed by SAE, L_1-AE and $L_{1/2}$AE are displayed in the first, second, third and fourth row, respectively. It can be seen that the digits (especially for digits 2, 4, 5 and 7) reconstructed by L_1-AE and $L_{1/2}$AE are clearer and more similar to the original digits than those reconstructed by SAE. $L_{1/2}$AE achieves the smallest reconstruction error over these ten digits than other two methods. Fig. 6(b) shows that $L_{1/2}$AE induces lower reconstruction errors (computed by Eq. (3)) than SAE and L_1-AE for different numbers of hidden nodes. Therefore, $L_{1/2}$AE has better reconstruction performance.



(a) original and reconstruction digits (b) reconstruction error

Fig. 6 *Reconstruction performance, (a) reconstruction of ten digits of SAE (error=7.6006),* L_1*-AE (error=4.8418) and* $L_{1/2}$*AE (error=4.6792), (b) reconstruction errors computed on all testing digits.*

The hidden structure of data is helpful for improving the interpretability of data. A good AE should be able to extract the hidden structure of data. The t-distributed Stochastic Neighbor Embedding (t-SNE) technique [40] is employed to visualize the hidden representation of 10,000 testing digit images in two-dimensional space in Fig. 7 for SAE, L_1-AE and $L_{1/2}$AE. From the comparison, the manifolds of digits 8, 5, 3 in $L_{1/2}$AE are more linear than in SAE and L_1-AE, and the manifolds of 7, 9, 4 in $L_{1/2}$AE are more linear than in SAE. The separation between the manifolds of digits 7 and 9 in $L_{1/2}$AE is bigger than that in SAE. The manifold of digit 2 in $L_{1/2}$AE is farther away from the other manifolds than in SAE and L_1-AE. In addition, the characteristics of the manifolds of 0, 1, and 6 in SAE, L_1-AE and $L_{1/2}$AE are similar. Therefore, $L_{1/2}$AE can learn a better hidden structure.



(a) SAE (b) L_1-AE

(c) $L_{1/2}$AE

Fig. 7 *Visualization of testing handwritten digits representation: (a) SAE, (b) L_1-AE and (c) $L_{1/2}$AE.*

To investigate the classification performance on the MNIST, a deep $L_{1/2}$AE network is trained by using the greedy layer-wise algorithm. We stack two layers of pre-trained $L_{1/2}$AE with 196, 20 hidden nodes, respectively, and a layer of softmax classifier, which are trained by using the hidden activations of the previous network as input for the next network. Then we fine-tune the resulting deep network to

achieve better prediction performance. The final structure of deep network is 784-196-20-10 (input layer with 784 nodes, first hidden layer with 196 nodes, second hidden layer with 20 nodes and output layer with 10 nodes). The results of deep $L_{1/2}$AE are compared with deep L_1-AE, deep SAE, deep DAE, and deep DpAE in Tab. II. The accuracy results have been averaged over 10 experiments to reduce the influence of random initialization. It can be seen that, before and after fine-tuning, the results of the deep $L_{1/2}$AE are significantly better than the other four deep networks. This is likely due to the sparser representation of $L_{1/2}$AE. As is well-known, sparsity is helpful for automatic feature selection and reducing the interference of the useless features on the classification performance.

	Before fine-tuning		After fine-tuning	
Deep networks	Mean± SD	p-value	Mean± SD	p-value
Deep $L_{1/2}$AE	86.47± 0.1898		97.66± 0.1035	
Deep L_1-AE	86.01± 0.1426	0.0091	97.42± 0.0848	0.0179
Deep SAE	80.21± 0.1370	<0.0001	97.36± 0.1137	0.0248
Deep DAE	83.00± 0.4499	<0.0001	97.03± 0.1543	0.0022
Deep DpAE	75.60± 0.4079	<0.0001	96.43± 0.1430	<0.0001

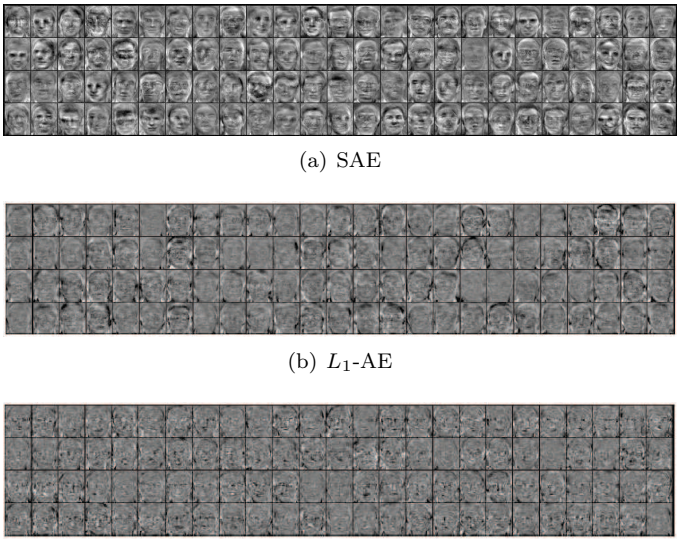
Tab. II *Classification performance of deep networks with structure 784-196-20-10 on the MNIST data set in supervised learning mode.*

3.2 ORL face data set

This experiment aims at evaluating the performance of $L_{1/2}$AE on extracting the facial feature, which is more challenging than extracting the digit feature. The Cambridge ORL (Olivetti Research Lab) face database is used in this experiment. This database consists of 40 distinct individuals, each containing 10 different face images, involving high degree of variation in facial expression, pose, lighting and facial details. Each image is normalized to a resolution of 92×112 pixels with 0 to 255 gray levels. All images are resized to 46×56 pixels to simplify the network structure. Therefore, the network contains 2,576 input nodes and 100 hidden nodes. The sigmoidal activation function is used for both the hidden and output layer.

The receptive fields of SAE, L_1-AE and $L_{1/2}$AE on the ORL data set are shown in Fig. 8. The features learned by SAE have the shape of holistic faces consisted of facial organs, such as eyes, mouth, nose and hair. Parts of features learned by L_1-AE are compressed smaller than those of SAE, while other features are unclear. In the features learned by $L_{1/2}$AE these facial organs are compressed smaller than those of L_1-AE in a blurred form. This demonstrates that $L_{1/2}$AE produces sparser face features. The sparseness histograms of weights are compared among SAE, L_1-AE and $L_{1/2}$AE in Fig. 9. It can be seen that the weights of $L_{1/2}$AE are sparser.

We choose ten face images to show the reconstruction performance for the ORL data set as shown in Fig. 10, and we achieve similar conclusion to that for MNIST data set. It can be seen that $L_{1/2}$AE achieves the smallest reconstruction error, and L_1-AE and $L_{1/2}$AE achieve better reconstructed faces than SAE.

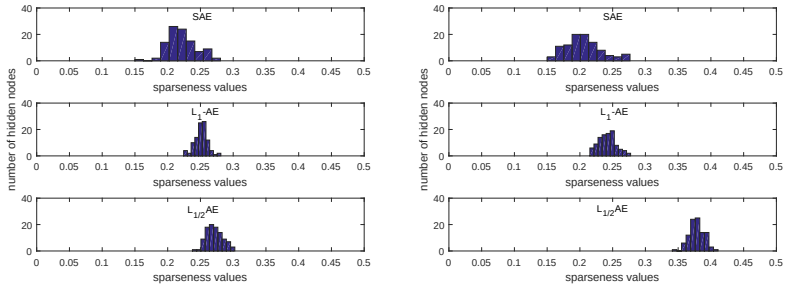


(a) SAE

(b) L_1-AE

(c) $L_{1/2}$AE

Fig. 8 *Visualization of 100 receptive fields of (a) SAE, (b) L_1-AE and (c) $L_{1/2}$AE for the ORL face database.*



(a) receptive fields

(b) decoding filters

Fig. 9 *Spareness histogram computed by Eq. (10) on (a) 100 receptive fields* $\mathbf{W}^{(1)}$ *and (b) 100 decoding filters* $\mathbf{W}^{(2)}$ *from the ORL face database.*

3.3 Reuters-21578 text corpus

The Reuters-21578 text corpus is used to test the capability of $L_{1/2}$AE on semantic feature extraction. The Reuters-21578 corpus contains 21,578 news reported in the Reuters newswire in 1987. We focus on a processed Modified Apte (ModApte) Split of Reuters-21578 corpus available at http://people.kyb.tuebingen.mpg.de/pgehler/rap/. This processed ModApte Split is composed of 11,413 training and 4,024 testing documents with 12,317 words or dimensions. The techniques described in [12] are applied for the dimensionality reduction and the selection of

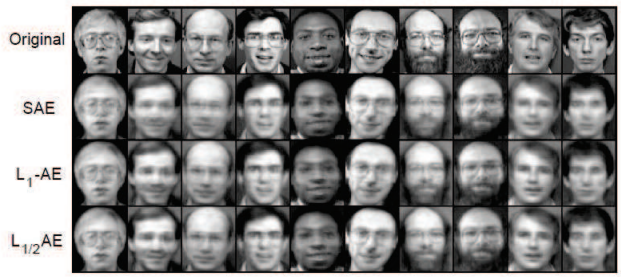
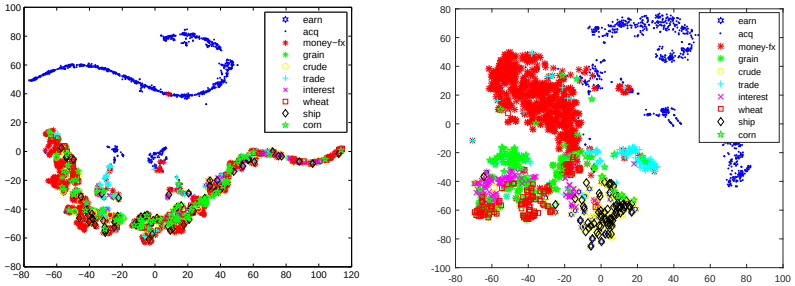


Fig. 10 *Reconstruction performance of ten face images of SAE (error=6.0573), L_1-AE (error=4.8120) and $L_{1/2}$AE (error=4.4084).*

most uncorrelated and representative words in documents. In this experiment, we use the most uncorrelated and representative 200 words to represent each document and choose the documents of the most frequent 10 categories in the processed ModApte Split to train the networks. The $L_{1/2}$AE network for this data set has 200 input nodes and 15 hidden nodes. The sigmoidal activation function is used for the hidden layer, and the linear function for output layer.

To investigate the semantic features extracted by $L_{1/2}$AE, the distributed representation of testing documents in hidden layer is visualized by using t-SNE projection and compared with that of SAE in Fig. 11. In Fig. 11(a) for SAE, all documents except those of acq category are very much overlapping each other, while in Fig. 11(b) for $L_{1/2}$AE, all documents belong to each category are more concentrated, and much less overlapping each other. Therefore, $L_{1/2}$AE produces a better semantic representation of documents.



(a) SAE (b) $L_{1/2}$AE

Fig. 11 *Visualization of Reuters-21578 documents: (a) SAE and (b) $L_{1/2}$AE.*

To further test the semantic representation, a softmax classifier is trained by using the hidden representation of $L_{1/2}$AE as input. Then, we fine-tune the stacked layers of $L_{1/2}$AE and the softmax classifier, that is, we construct a shallow $L_{1/2}$AE network by using the greedy layer-wise algorithm for classification of the Reuters-

21578 corpus. The classification results of the shallow $L_{1/2}$AE, L_1-AE, SAE, DAE, and DpAE are displayed in Tab. III. It shows that, after fine-tuning, the shallow $L_{1/2}$AE produces a significantly better result than the other four networks.

	Before fine-tuning		After fine-tuning	
Shallow networks	Mean± SD	p-value	Mean± SD	p-value
Shallow $L_{1/2}$AE	72.37± 0.1373		82.92± 0.2604	
Shallow L_1-AE	72.33± 0.2010	0.8463	82.04± 0.1650	0.0003
Shallow SAE	57.42± 0.2721	<0.0001	81.97± 0.2064	0.0010
Shallow DAE	68.83± 0.6253	0.0002	81.45± 0.4158	0.0054
Shallow DpAE	50.99± 0.5390	<0.0001	82.27± 0.1664	0.0029

Tab. III *Classification performance of shallow networks with structure 200-15-10 on the Reuters-21578 corpus.*

4. Summary

In this paper, an $L_{1/2}$AE is proposed for learning sparse representation of data and improving the classification capability. This has been achieved by using $L_{1/2}$ regularization method as a sparsity constraint on the average hidden activations. The performance of $L_{1/2}$AE for feature extraction and classification is compared with other popular AEs such as L_1-AE, SAE, DAE, and DpAE. Numerical experiments have been conducted on the MNIST data set, the ORL database and the Reuters-21578 corpus. The results in terms of reconstruction performance and sparseness of weights in unsupervised learning mode demonstrate that the $L_{1/2}$ regularization as a constraint on the hidden activations helps $L_{1/2}$AE achieve a sparser representation of data and smaller reconstruction error than SAE. It is also shown that a deep $L_{1/2}$AE network achieves better classification accuracy than deep L_1-AE, deep SAE, deep DAE and deep DpAE, due to the better representation learned by $L_{1/2}$AE.

Acknowledgement

This research was supported by the National Science Foundation of China (NO: 61473059, 61403056), the Fundamental Research Funds for the Central Universities of China and China Scholarship Council (CSC).

References

[1] BENGIO Y. Learning deep architectures for AI. *Foundations and trends in Machine Learning.* 2009, 2, pp. 1–127, doi: 10.1561/2200000006.

[2] VINCENT P., LAROCHELLE H., LAJOIE I., BENGIO Y., MANZAGOL P.A. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research.* 2010, 11(Dec), pp. 3371–3408.

[3] COATES A., NG A.Y., LEE H. An analysis of single-layer networks in unsupervised feature learning. In: *Proceedings of the 14th International conference on artificial intelligence and statistics*, Fort Lauderdale, FL, USA, 2011, pp. 215–223.

[4] DENG L., SELTZER M.L., YU D., ACERO A., MOHAMED A.R., HINTON G.E. Binary coding of speech spectrograms using a deep auto-encoder. In: *Proceedings of the 11th Annual Conference of the International Speech Communication Association*, Makuhari, Chiba, Japan, 2010, pp. 1692–1695.

[5] RIFAI S., VINCENT P., MULLER X., GLOROT X., BENGIO Y. Contractive auto-encoders: Explicit invariance during feature extraction. In: *Proceedings of the 28th international conference on machine learning*, Bellevue, WA, USA, 2011, pp. 833–840.

[6] NG A. Sparse autoencoder. In: *CS294A Lecture notes* [online], 2011. Available on: http://web.stanford.edu/class/cs294a/sae/sparseAutoencoderNotes.pdf.

[7] WANG W., HUANG Y., WANG Y., WANG L. Generalized autoencoder: A neural network framework for dimensionality reduction. In: *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, Columbus, OH, USA, 2014, pp. 490–497.

[8] SETIONO R., LU G. A neural network construction algorithm with application to image compression. *Neural Computing & Applications*. 1994, 2(2), pp. 61-68, doi: 10.1007/BF01414350.

[9] LECUN Y., BENGIO Y., HINTON G.E. Deep learning. *Nature*. 2015, 521(7553), pp. 436–444, doi: 10.1038/nature14539.

[10] KRIZHEVSKY A., SUTSKEVER I., HINTON G.E. Imagenet classification with deep convolutional neural networks. In: *Advances in neural information processing systems 25*, Lake Tahoe, NV, USA, 2012, pp. 1097–1105.

[11] HINTON G.E., OSINDERO S., TEH Y.W. A fast learning algorithm for deep belief nets. *Neural Computation*. 2006, 18(7), pp. 1527-1554, doi: 10.1162/neco.2006.18.7.1527.

[12] HOSSEINI-ASL E., ZURADA J.M., NASRAOUI O. Deep learning of part-based representation of data using sparse autoencoders with nonnegativity constraints. *IEEE Transactions on Neural Networks and Learning Systems*. 2016, 27(12), pp. 2486–2498, doi: 10.1109/TNNLS.2015.2479223.

[13] AYINDE B.O., HOSSEINI-ASL E., ZURADA J.M. Visualizing and understanding nonnegativity constrained sparse autoencoder in deep learning. In: *Proceedings of 15th International conference on Soft Computing and Artificial Intelligence*, Zakopane, Poland, 2016, doi: 10.1007/978-3-319-39378-0_1.

[14] WANG J., HE H., PROKHOROV D.V. A folded neural network autoencoder for dimensionality reduction. In: *Proceedings of the 3rd International Neural Network Society Winter Conference*, Bangkok, Thailand, 2012, 13, pp. 120–127, doi: 10.1016/j.procs.2012.09.120.

[15] ATTWELL D., LAUGHLIN S.B. An energy budget for signaling in the greymatter of the brain. *Journal of Cerebral Blood Flow And Metabolism*. 2001, 21(10), pp. 1133–1145, doi: 10.1097/00004647-200110000-00001.

[16] LEE H., EKANADHAM C., NG A. Sparse deep belief net model for visual area V2. In: *Advances in Neural Information Processing Systems 20*, Vancouver, British Columbia, Canada, 2007, pp. 873–880.

[17] WRIGHT J., YANG A.Y., GANESH A., SASTRY S.S., MA Y. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2009, 31(2), pp. 210–227, doi: 10.1109/TPAMI.2008.79.

[18] NAIR V., HINTON G.E. 3D object recognition with deep belief nets. In: *Advances in Neural Information Processing Systems 22*, Vancouver, British Columbia, Canada, 2009, pp. 1339–1347.

[19] TIBSHIRANI R. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B*. 1996, 58, pp. 267–288. Available on: http://www.jstor.org/stable/2346178.

[20] DONOHO D.L., HUO X. Uncertainty principles and ideal atomic decomposition. *IEEE Transactions on Information Theory*. 2001, 47(7), pp. 2845–2862, doi: 10.1109/18.959265.

[21] CHEN S., DONOHO D.L., SAUNDERS M. Atomic decomposition by basis pursuit. *SIAM Review*. 2001, 43(1), pp. 129–159, doi: 10.1137/S003614450037906X.

[22] ZOU H. The adaptive Lasso and its oracle properties. *Journal of the American statistical association*. 2006, 101(476), pp. 1418–1429, doi: 10.1198/016214506000000735.

[23] ZHAO P., YU B. Stagewise lasso. *Journal of Machine Learning Research*. 2007, 8(Dec), pp. 2701–2726.

[24] LIU H., MOTODA H., SETIONO R., ZHAO Z. Feature selection: An ever evolving frontier in data mining. In: *Proceedings of The 4th Workshop on Feature Selection in Data Mining*, 2010, pp. 4–13. Available on: http://proceedings.mlr.press/v10/liu10b/liu10b.pdf.

[25] XU Z.B., ZHANG H., WANG Y., CHANG X.Y., LIANG Y. $L_{1/2}$ regularization. *Science China Information Sciences*. 2010, 53(6), pp. 1159–1169, doi: 10.1007/s11432-010-0090-0.

[26] XU Z.B., CHANG X.Y., XU F.M. $L_{1/2}$ Regularization: A Thresholding Representation Theory and a Fast Solver. *IEEE Transactions on Neural Networks and Learning Systems*. 2012, 23(7), pp. 1013–1027, doi: 10.1109/TNNLS.2012.2197412.

[27] ZENG J.S., LIN S.B., WANG Y., XU Z.B. $L_{1/2}$ Regularization: Convergence of Iterative Half Thresholding Algorithm. *IEEE Transactions on Signal Processing*. 2014, 62(9), pp. 2317–2329, doi: 10.1109/TSP.2014.2309076.

[28] JIANG X., ZHANG Y., ZHANG W., XIAO X. A novel sparse auto-encoder for deep unsupervised learning. In: *Proceedings of the 6th International Conference on Advanced Computational Intelligence*, Hangzhou, China, 2013, pp. 256–261.

[29] LECUN Y., BOTTOU L., BENGIO Y., HAFFNER P. Gradient-based learning applied to document recognition. In: *Proceedings of the IEEE*, 1998, 86(11), pp. 2278–2324, doi: 10.1109/5.726791.

[30] SAMARIA F.S., HARTER A.C. Parameterisation of a stochastic model for human face identification. In: *Proceedings of the 2nd IEEE Workshop on Applications of Computer Vision*, 1994, pp. 138–142. Available on: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=341300.

[31] KROGH A., HERTZ J.A. A simple weight decay can improve generalization. In: *Advances in neural information processing systems 4*, Denver, Colorado, USA, 1991, pp. 950–957.

[32] WU W. *Computation of Neural Networks*. Beijing, Higher Education Press, 2003.

[33] ZURADA J.M. *Introduction to Artificial Neural Systems*. St. Paul, Minn., West Publishing Company, 1992.

[34] WU W., FAN Q.W., ZURADA J.M., WANG J., YANG D.K., LIU Y. Batch gradient method with smoothing $L_{1/2}$ regularization for training of feedforward neural networks. *Neural Networks*. 2014, 50, pp. 72–78, doi: 10.1016/j.neunet.2013.11.006.

[35] ZHANG H.S., WU W. Convergence of Split-Complex Backpropagation Algorithm with a Momentum. *Neural Network World*. 2011, 21(1), pp. 75–90, doi: 10.14311/NNW.2011.21.006.

[36] DO C., NG A.Y. Transfer learning for text classification. In: *Advances in Neural Information Processing Systems 18*, Vancouver, British Columbia, Canada, 2005, pp. 299–306.

[37] SCHMIDT M. MinConf: Projection methods for optimization with simple constraints in Matlab. 2008. Available on: http://www.cs.ubc.ca/~schmidtm/Software/minConf.html.

[38] SRIVASTAVA N., HINTON G.E., KRIZHEVSKY A., SUTSKEVER I., SALAKHUTDINOV R. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*. 2014, 15(1), pp. 1929–1958.

[39] HOYER P.O. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*. 2004, 5(Nov), pp. 1457–1469.

[40] MAATEN L.V.D., HINTON G.E. Visualizing data using t-SNE. *Journal of Machine Learning Research*. 2008, 9(Nov), pp. 2579–2605.