Autoencoder Feature Selector

Kai Han, Chao Li, Xin Shi
hankaixyz@163.com

Abstract

High-dimensional data in many areas such as computer vision and machine learning brings in computational and analytical difficulty. Feature selection which select a subset of features from original ones has been proven to be effective and efficient to deal with high-dimensional data. In this paper, we propose a novel AutoEncoder Feature Selector (AEFS) for unsupervised feature selection. AEFS is based on the autoencoder and the group lasso regularization. Compared to traditional feature selection methods, AEFS can select the most important features in spite of nonlinear and complex correlation among features. It can be viewed as a nonlinear extension of the linear method regularized self-representation (RSR) for unsupervised feature selection. In order to deal with noise and corruption, we also propose robust AEFS. An efficient iterative algorithm is designed for model optimization and experimental results verify the effectiveness and superiority of the proposed method.

1 Introduction

With the development of the big data technology, we have been encountering more and more high-dimensional data in the field of computer vision and machine learning. A mass of noisy and useless features existing in high-dimensional space lead to extreme inefficiency in data analysis. In this case, feature selection plays a crucial role by choosing a small subset of features in removing nonsense features and decreasing computational complexity. In real world, sample labels are often unknown while labelling samples is both time-consuming and finance-consuming. Thus in this paper, we mainly focus on feature selection in unsupervised scenario.

According to different search strategies, the common feature selection methods can be divided to filter, wrapper and embedded methods [14] where the embedded method is a research hotspot currently. Unlike filter and wrapper methods which make feature selection process and training process into two separate parts, embedded method combines variable selection in the training process. Thus, embedded method have lots of advantages like being more efficient, interacting with the learning algorithm and saving plentiful time for model training.

However, most of traditional embedded methods such as the famous LASSO [22] method can only explore the linear relationship among features. Therefore, how to make good use of the nonlinear relationship among features (Fig. 1) brings in a great challenge. Kernel based feature selection methods [3, 15, 19] were proposed to allow learning of nonlinear representation, but the representation is limited by the fixed kernel [1]. In this paper, we propose to use neural network to learn flexible nonlinear relationship among features. The powerful

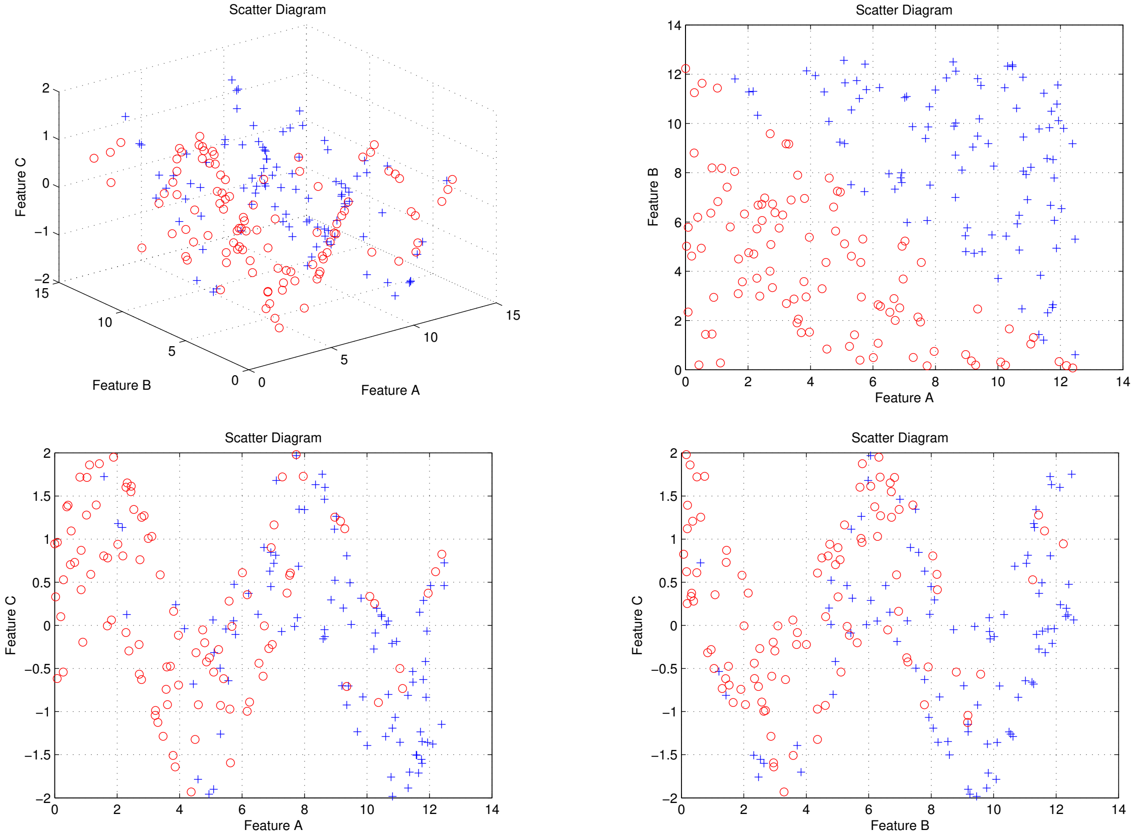

Figure 1: Illustration of nonlinear relationship among features. The synthetic dataset consists of 200 samples with 3 features {A, B, C}. The dataset are split into 2 classes about fifty-fifty. Feature A and B are independent with each other while feature C can be represented by A and B: $C = sin(A) + cos(B)$. In this case, feature C is the redundant and noisy one. Linear feature selection methods is hard to deal with the case.

nonlinear representation approximating ability of neural network makes it successful in a wide variety of tasks. Thus we may expect neural network to help deal with the nonlinearity in feature selection.

The autoencoder is a succinct neural network used for unsupervised learning of efficient codings [2], aiming to learn a self-representation for a set of data. In real-world data, we assume that the redundant features can be represented by linear or nonlinear combanition of other relevant features. In this paper, we use autoencoder to capture the self-representation property of features and impose group sparsity on the feature weights to select features. This joint method is efficient and robust for feature selection, and have strong ability to explore the nonlinear relationship among features. Experimental results verify the superiority of the proposed method.

2 Related Work

In this section we introduce the most related previous work to ours including regularization based embedded methods and nonlinear methods.

Embedded feature selection methods can combine feature selection with training process into a whole part. The most widely used embeded methods are regularization models that introduce additional constraints into the optimization of a predictive algorithm that bias the model toward lower complexity. One of famous regularization algorithms is the Lasso proposed by Tibshirani in 1996 [22]. Let X be the feature matrix, y be the label vector. The object function of basic Lasso is $\min_\theta \|y - X\theta\|_2^2 + \lambda\|\theta\|_1$, where θ is the coefficients and λ is the penalty parameter. Some elements of θ may be close to or exact zero, and the important features are selected according to the indexes of non-zero elements. Recently, a regularized self-representation (RSR) model have been proposed by Zhu and Zuo [26], which briefly uses the input data matrix X to reconstruct itself with the group lasso penalization to select significant features as well as reduce the redundant features.

However, most of the previous embeded methods cannot explore the nonlinear relationship among features. Although there exist some nonlinear algorithms by using kernels [3, 15, 19], the specific designed kernels are not sufficient to extract arbitrary nonlinear dependencies of features. In another hand, most of the neural networks used in feature selection are wrapper methods [17, 20, 23], which are inefficient. In order to solve these problems, we propose an embedded feature selection method based on autoencoder which attempt to learn a nonlinear self-representation of input data. The autoencoder is a simple and efficient neural network and is a powerful tool for unsupervised application.

The proposed method combines the autoencoder with the group lasso regularization for feature selection. The group lasso method guarantees the sparse solution for selecting significant features, and autoencoder deals with the nonlinear relationship among features. Within the scope of our knowledge, we firstly use the autoencoder in an embedded feature selection method. Besides, this paper contributes an efficient and robust algorithm for unsupervised feature selection.

3 Autoencoder Feature Selector

3.1 Preliminaries

3.1.1 The problem settings.

The general unsupervised feature selection problem is described as follows. Given the unlabeled sample matrix $X = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m]^T \in \mathbb{R}^{m\times d}$ where m is the number of unlabeled samples and d is the number of features, the task of unsupervised feature selection is to select the most discriminative and informative features among the original ones with the unlabeled data.

3.1.2 Autoencoder.

The autoencoder [9] is a special feedforward neural network attempting to copy its input to its output. As shown in Fig. 2, the typical autoencoder with a h-dimension hidden layer consists of two components: an encoder function $H = f(X) = \sigma_1(XW^{(1)})$ and a decoder that produces a reconstruction $\hat{X} = g(H) = \sigma_2(HW^{(2)})$, where σ_1, σ_2 are activation functions of the hidden layer and the output layer respectively, $\Theta = \{W^{(1)}, W^{(2)}\}$ are weight parameters and $W_{ij}^{(l)}$ denotes the parameter of the connection between neuron i in layer l and neuron j in layer $l+1$. The overall function of the autoencoder could be represented as $g(f(X))$. In learning process, autoencdoer is described simply as minimizing a loss function $\mathcal{J}_{AE}(X, g(f(X)))$, usually in the form of least square or cross entropy loss. In this paper, we just use the least square loss as the fitting error: $\mathcal{J}_{AE}(X, g(f(X))) = \frac{1}{2m}\sum_{i=1}^m \|\boldsymbol{x}_i - g(f(\boldsymbol{x}_i))\|_2^2 = \frac{1}{2m}\|X - g(f(X))\|_F^2$.

3.2 The Proposed Model

Most widely used embedded feature selection methods aim to fit a learning model by minimizing the fitting error and force the coefficients of some features to be small (or exact zero) simultaneously. The object function of the general embedded feature selection methods can be written as

$$\min_{\Theta} \mathcal{L}(\Theta) + \lambda \mathcal{R}(\Theta) \tag{1}$$

where Θ is the set of parameters, $\mathcal{L}$ is the fitting error of the learning model, $\mathcal{R}$ is the regularization term imposed on Θ and λ is the trade-off parameter. As aforementioned in related work, $\mathcal{L}$ and $\mathcal{R}$ vary in different methods, for example, $\mathcal{L}$ is the least-square loss and $\mathcal{R}$ is the ℓ_1 norm regularization in lasso. In this paper, we propose a novel unsupervised embedded feature selection method based on autoencoder and group lasso penalty. To the best of our knowledge, this is the first time to use the neural network or autoencoder in an embedded feature selection method so far.

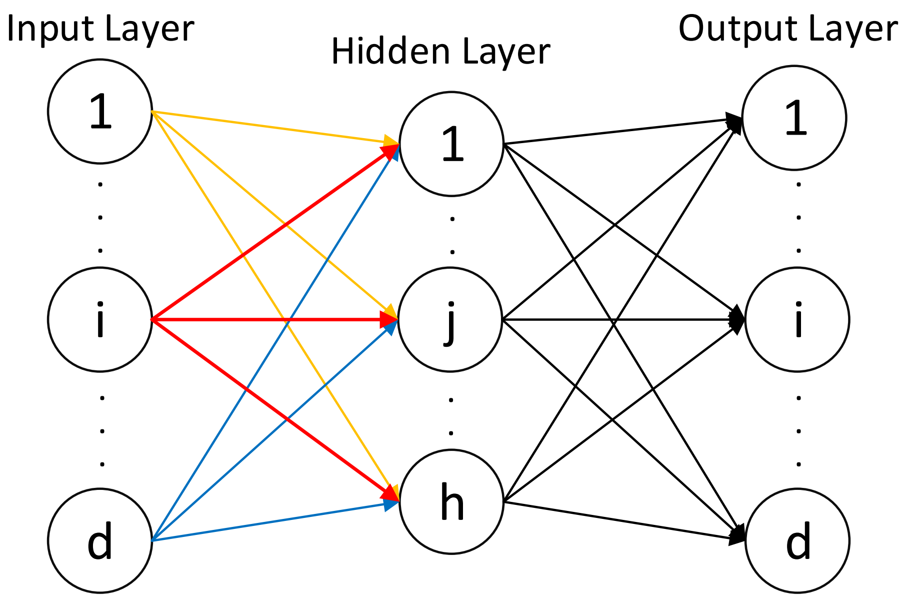

Figure 2: An autoencoder with d input (output) nodes and h hidden nodes.

Denote $X = [\boldsymbol{f}_1, \cdots, \boldsymbol{f}_d]$, and each column $\boldsymbol{f}_i$ represents the ith feature of X. In autoencoder, each feature $\boldsymbol{f}_i$ in X can be well represented by all the features (including $\boldsymbol{f}_i$ itself) and the autoencoder approximates the representation $\hat{X} = g(f(X))$. The weight matrix $W^{(1)}$ connecting the input layer and the hidden layer can be written as $W^{(1)} = [\boldsymbol{w}_1, \cdots, \boldsymbol{w}_d]^T$, where the ith row $\boldsymbol{w}_i$ corresponds to the ith feature $\boldsymbol{f}_i$. As shown in Fig. 2, the yellow lines ($\boldsymbol{w}_1$) represent the connection between the 1st feature and the hidden layer, the red lines ($\boldsymbol{w}_i$) correspond to the ith one, and the blue lines ($\boldsymbol{w}_d$) correspond to the last one. The coefficients set between each input node and the hidden layer reflects the contribution of this input node to the self-reconstruction. Thus the ℓ_2 norm $\|\boldsymbol{w}_i\|_2$ can be used as the criteria to select features because it reflects the importance of the ith feature to self-representation. On the one hand, if $\|\boldsymbol{w}_i\| \approx 0$, the ith feature contributes little to the representation of other features; on the other hand, if ith feature plays important role in the representation of other features, then $\|\boldsymbol{w}_i\|_2$ must be significant. To select the most discriminative features from original ones, we impose row-sparse regularization on $W^{(1)}$. That is to say, we use $\mathcal{R}(\Theta) = \|W^{(1)}\|_{2,1} = \sum_i^d \sqrt{\sum_j^h (W_{ij}^{(1)})^2}$ in the object function (1). Thus we have

$$\min_{\Theta} \mathcal{J}(\Theta) = \frac{1}{2m}\|X - g(f(X))\|_F^2 + \alpha \|W^{(1)}\|_{2,1}, \tag{2}$$

where α is the trade-off parameter between the fitting loss and the regularization term.

Usually in the training process of a neural network, a weight decay term is added in the object function to avoid overfitting and promote convergence. Finally, the overall object function of our model formulates as

$$\min_{\Theta} \mathcal{J}(\Theta) = \frac{1}{2m}\|X - g(f(X))\|_F^2 + \alpha \|W^{(1)}\|_{2,1} + \frac{\beta}{2}\sum_{i=1}^{2}\|W^{(i)}\|_F^2, \tag{3}$$

where β is a penalty parameter. We call the model (3) as AutoEncoder Feature Selector (AEFS).

In real-world data, how to deal with the noise or corruption is a headache in many research areas. Inspired by denoising autoencoder which learn to reconstruct the clean input X from the artificially corrupted counterpart $\tilde{X}$ [24], we apply the thought to autoencoder feature selector to make it robust to partial corruption of the input data. Thus denoising autoencoder feature selector (dAEFS) is to minimize the following object function:

$$\min_{\Theta} \mathcal{J}_{denoising}(\Theta) = \mathbb{E}_{\tilde{X} \sim q(\tilde{X}|X)}[\mathcal{J}(\Theta)], \tag{4}$$

where $q(\tilde{X}|X)$ represents a conditional distribution over corrupted samples $\tilde{X}$, given data samples X. There are various typical corruptions of input data, and in this paper we adopt the additive isotropic Gaussian noise, $\tilde{X} = X + [\boldsymbol{\varepsilon}_1, \cdots, \boldsymbol{\varepsilon}_m]^T$ where $\boldsymbol{\varepsilon}_i \sim \mathcal{N}(0, \sigma^2 I)$ without loss of generality.

3.3 Nonlinearity Discuss

As for the nonlinearity property, we compare our method AEFS with a recent work related to ours – regularized self-representation model which solves the following optimization function:

$$\min_{W} \|X - XW\|_F^2 + \lambda \|W\|_{2,1} \tag{5}$$

where W is the feature weight matrix each feature. In RSR, each feature can be represented as the linear combination of its relevant features. By using ℓ_{21}-norm to characterize the representation coefficient matrix, RSR is effective to select representative features. However, if the correlation among features is nonlinear, RSR perhaps does not work very well. In AEFS, the goal is to minimize the object function (3). Since $g(f(X))$ is a nonlinear function, each feature can be nonlinearly represented by its relevant features in AEFS. Thus even if the correlation among features is nonlinear, AEFS could still works well. Moreover, if we set $\sigma_1(X) = X$, $\sigma_2(X) = X$ and leave out the weight decay term, AEFS reduces to a linear form:

$$\min_{W^{(1)}, W^{(2)}} \frac{1}{2m}\|X - XW^{(1)}W^{(2)}\|_F^2 + \alpha \|W^{(1)}\|_{2,1}. \tag{6}$$

We can find that this form is equivalent to (5), so AEFS is a nonlinear extension of RSR.

4 Optimization

Similar to other autoencoder variants, the autoencoder feature selector could be optimized with back-propagation algorithm. Firstly, the error terms of the output layer and the hidden layer are computed as follows.

$$\begin{aligned} \delta^{(o)} &= -(X-\hat{X}) \odot \sigma_2'(H), \\ \delta^{(h)} &= \left((W^{(2)})^T \delta^{(o)}\right) \odot \sigma_1'(X), \end{aligned} \tag{7}$$

where $\odot$ represents element-wise product of two matrix. Then the partial derivative respect to $W^{(2)}$ is given as

$$\nabla_{W^{(2)}} \mathcal{J}(\Theta) = \frac{1}{m}\delta^{(o)}\hat{X}^T + \beta W^{(2)}, \tag{8}$$

and $W^{(2)}$ are optimized by gradient descent method.

However, the partial derivative of the object function respect to $W^{(1)}$ is not available at the zero point, so it can not be directly optimized by gradient descent method. Instead, we use the proximal gradient descent method [5, 16] to solve the problem. The solving process includes two steps:

$$\nabla_{W^{(1)}} \mathcal{J}^-(\Theta) = \frac{1}{m}\delta^{(h)}H^T + \beta W^{(1)}. \tag{9}$$

$$\hat{W^{(1)}} = \Phi^{\#}\left(W^{(1)} - t\nabla_{W^{(1)}}\mathcal{J}^-(\Theta); \alpha t\right) \tag{10}$$

where $\mathcal{J}^-(\Theta)$ denotes the object function leaving out ℓ_{21} norm regularization, $t > 0$ is a step size, $\Phi^{\#}$ is the group soft thresholding operator and the details are described in Definition 1.

Definition 1. The multivariate soft thresholding operator for any vector $\boldsymbol{w} \in \mathbb{R}^d$ is $\overrightarrow{\Phi}(\boldsymbol{w};\lambda) = \boldsymbol{w}^o\Phi(\|\boldsymbol{w}\|_2;\lambda)$ where $\boldsymbol{w}^o = \begin{cases} \frac{\boldsymbol{w}}{\|\boldsymbol{w}\|_2}, & \text{if } \boldsymbol{w} \neq \boldsymbol{0} \\ \boldsymbol{0}, & \text{if } \boldsymbol{w} = \boldsymbol{0} \end{cases}$, and Φ is element-wise soft thresholding operator: $\Phi(x;\lambda) = \operatorname{sign}(x)(|x|-\lambda)_+$. Then we define the group soft thresholding operator for any matrix $W = [\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_n]^T$ as

$$\Phi^{\#}(W;\lambda) = \begin{pmatrix} \overrightarrow{\Phi}(\boldsymbol{w}_1;\lambda)^T \\ \overrightarrow{\Phi}(\boldsymbol{w}_2;\lambda)^T \\ \vdots \\ \overrightarrow{\Phi}(\boldsymbol{w}_n;\lambda)^T \end{pmatrix}$$

The optimization algorithm of autoencoder feature selector is described in Algorithm 1. The algorithm is simple to implement and can easily be extended to other gradient-based optimization method. The denoising autoencoder feature selector can be trained using back-propagation algorithm similar to Algorithm 1 and the differ is that the input data is corrupted artificially.

5 Experiments

5.1 Synthetic Dataset Illustration

We construct a synthetic dataset as shown in Fig. 1. The dataset consists of 200 samples with 3 features {A, B, C}. The dataset are split into 2 classes about fifty-fifty. Feature A

Algorithm 1 Optimization Algorithm of Autoencoder Feature Selector

```
Input: Data matrix $X \in \mathbb{R}^{m \times d}$, parameters $\alpha, \beta$.
Initialization: $W^{(1)}, W^{(2)}$ that satisfy Gaussian distribution.
Output: $W^{(1)}, W^{(2)}$
  Repeat
    Compute the feedforward activations of the hidden layer $H$ and the output layer $\hat{X}$.
    Compute the error terms using equations (7).
    Compute the partial derivatives using equations (8), (9).
    Update $W^{(2)}$ by gradient descent.
    Update $W^{(1)}$ by group soft thresholding operator (10).
  Until stopping criterion
  Select features according to the index of the top-$k$ row-norms of $W^{(1)}$ in descending order.
```

and B are randomly generated from uniform distribution in the range of $\{0, 2\pi\}$, and they are independent with each other. Meanwhile feature C is fromed from the formulation: $C = sin(A) + cos(B)$. In this case, feature C has a nonlinear relationship with A and B, and it is the redundant and noisy one.

In order to verify the nonlinear property of our method, we apply RSR and AEFS to select 2 features from {A, B, C} respectively. We repeat 10 times to perform feature selection, the result is that RSR always selects feature B and C while AEFS selects feature A and B in most cases. With the nonlinear representation ability, AEFS can deal with nonlinear relationship among features effectively.

5.2 Real-world Datasets and Experiemntal Settings

Experiments are conducted on 8 benchmark datasets to evaluate the performance of AEFS. The datasets include one spoken letter dataset (i.e., Isolet[1] [6]), one face image dataset (i.e., warpPIE10P[2] [21]), one text dataset (i.e., PCMAC[3] [12]), one artificial dataset(i.e., madelon[4] [7]), two microarray datasets (i.e., lung_discrete[5] [18] and Prostate_GE[6] [10]), one handwritten digits dataset (i.e., MNIST[7] [13]) and one image hand-crafted feature dataset (i.e., AWA[8] [11]). The detailed information of the six datasets used in the experiments is summarized in Table 1. In experimental datasets, The dimension varies from 325 to 5,966 and the feature types include image, text and microarray. All the data is normalized before experiments.

In order to evaluate superiority of our method, We compare AEFS with the following unsupervised feature selection methods.

AllFea: All original features without feature selection.

LS: Laplacian Score [8] feature selection method which selects features that well preserve the data manifold structure.

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

[2] http://www.ri.cmu.edu/research_project_detail.html?project_id=418&menu_id=261

[3] http://featureselection.asu.edu/datasets.php

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

[5] http://featureselection.asu.edu/datasets.php

[6] http://www.ncbi.nlm.nih.gov/pubmed/12381711

[7] http://yann.lecun.com/exdb/mnist/

[8] http://attributes.kyb.tuebingen.mpg.de/

Table 1: Summary of used datasets.

Dataset	Keywords	#Instances	#Features	#Classes
Isolet	Spoken letter,continuous	1560	617	26
warpPIE10P	Face image,continuous	210	2420	10
PCMAC	Text,discrete	1943	3289	2
madelon	Artificial,continuous	2600	500	2
lung_discrete	Biological,discrete	73	325	7
Prostate_GE	Biological,continuous	102	5966	2
MNIST	Handwritten digits,discrete	70000	784	10
AWA	Image feature,continuous	14112	4940	20

MCFS: Multi-Cluster Feature Selection [4] method which selects features using spectral regression with ℓ_1 norm regularization.

UDFS: Unsupervised Discriminative Feature Selection [25] method that selects the most discriminative features by exploiting both the discriminative information and feature correlations.

RSR: Regularized Self-Representation [26] model for feature selection which exploiting the self-representation ability of features with ℓ_{21} regularization.

As for parameters setting, in the methods LS, MCFS and UDFS, the size of the neighbors k is fixed as 5 for all the cases. For fair comparison, the parameters in all the methods are tuned in the range of $\{0.001, 0.01, \cdots, 100, 1000\}$. In AEFS, we set the size of hidden layer in $\{128, 256, 512, 1024\}$ and the activation function $\sigma_1(X) = 1/(1+e^{-X})$, $\sigma_2(X) = X$. For all datasets, we set the number of selected features as $\{50, 100, 150, \cdots, 300\}$ and report the best results from the optimal parameters for all the methods.

5.3 Clustering and Classification Experiments

We conduct clustering experiments using k-means algorithm and classification experiments using nearest neighbor classifier to evaluate the performance of different feature selection methods.

5.3.1 Evaluation metrics.

For clustering experiments, two widely used evaluation metrics, Accuracy (ACC) and Normalized Mutual Information (NMI), are used to measure the clustering performance[9].

Denote p_i as the true label and q_i as the clustering result of the sample $\boldsymbol{x}_i$. ACC is defined as

$$ACC = \frac{\sum_{i=1}^{m} \delta(p_i, map(q_i))}{m} \tag{11}$$

where $\delta(x,y) = 1$ if $x = y$; $\delta(x,y) = 0$ otherwise and $map(q_i)$ is the best mapping function that permutes clustering labels to match the ground truth labels, which can be gotten using the Kuhn-Munkres algorithm. In clustering, a larger ACC is expected.

[9] Since the result of k-means depends on initialization, we repeat the experiments 20 times with random initialization and report the average results with standard deviation.

Given two variables P and Q, NMI between them is defined as

$$NMI(P,Q) = \frac{I(P,Q)}{\sqrt{H(P)H(Q)}} \tag{12}$$

where $I(P,Q)$ is the mutual information between P and Q and $H(P)$, $H(Q)$ are the entropies of P and Q, respectively. In clustering evaluation, P and Q are the ground truth labels and the clustering labels respectively. The lager NMI is, the better clustering result is.

For classification experiments, we use the accuracy as evaluation metric and a high classification accuracy is expected.

5.3.2 Experimental results.

The clustering results is shown in Table 2 and 3, and the classification results are listed in Table 4. From the results, we observe that feature selection can not only reduce the dimension of features, but also greatly improve both the clustering and the classification performance. We also see that AEFS outperform other methods almost in all the cases. This benefits from the ability to capture the most import features which could represent all the features and the nonlinearity transformation inside the representation of AEFS.

Table 2: Clustering results (NMI% ± std) of different feature selection methods. The best results are highlighted in bold.

Dataset	**AllFea**	**LS**	**MCFS**	**UDFS**	**RSR**	**AEFS**
Isolet	72.3±1.5	68.6±1.1	73.5±1.5	65.5±1.1	69.3±1.8	**74.6±1.6**
warpPIE10P	29.5±4.5	32.4±2.3	44.4±5.6	54.6±5.2	36.0±3.0	**55.3±4.8**
PCMAC	0.88±0.7	2.14±1.0	2.08±1.4	3.09±1.7	0.93±0.9	**4.19±0.6**
madelon	1.95±0.0	2.05±0.2	2.43±0.2	2.18±0.1	0.80±0.1	**3.54±0.1**
lung_discrete	62.5±5.0	63.1±4.4	66.7±6.7	66.3±5.7	68.6±5.4	**69.5±5.4**
Prostate_GE	3.91±0.0	1.65±0.2	3.96±3.2	7.10±0.9	6.02±6.4	**18.6±8.6**
MNIST	42.3±1.0	27.5±2.1	47.6±1.2	38.3±2.2	22.5±0.3	**46.1±3.2**
AWA	9.9±0.4	7.5±0.3	7.4±0.1	7.5±0.2	6.9±0.2	**7.6±0.3**

Table 3: Clustering results (ACC% ± std) of different feature selection methods. The best results are highlighted in bold.

Dataset	**AllFea**	**LS**	**MCFS**	**UDFS**	**RSR**	**AEFS**
Isolet	54.0±4.6	51.6±3.1	56.5±3.1	45.8±3.2	54.3±3.4	**58.7±3.5**
warpPIE10P	28.7±3.1	32.9±2.8	38.8±4.1	50.4±5.2	35.5±2.5	**50.7±5.3**
PCMAC	50.5±0.2	50.8±0.2	50.9±0.7	51.6±1.0	51.1±0.9	**51.7±1.1**
madelon	58.2±0.5	58.4±0.2	59.1±0.3	58.7±0.2	51.3±1.1	**61.0±0.1**
lung_discrete	64.3±7.1	65.1±9.7	70.3±8.4	68.9±6.8	71.6±5.8	**71.6±7.2**
Prostate_GE	59.9±1.9	57.5±4.6	59.9±5.0	64.5±3.8	60.5±5.2	**73.1±6.4**
MNIST	46.8±2.6	31.4±1.7	50.9±2.3	49.0±2.7	29.3±0.8	**51.8±4.8**
AWA	14.4±0.3	11.6±0.3	13.2±0.2	12.2±0.3	12.2±0.3	**13.4±0.4**

Table 4: Classification results (ACC%) of different feature selection methods. The best results are highlighted in bold.

Dataset	**AllFea**	**LS**	**MCFS**	**UDFS**	**RSR**	**AEFS**
Isolet	90.128	83.562	89.615	82.436	85.0	**89.167**
warpPIE10P	100.0	94.286	**100.0**	99.524	99.048	**100.0**
PCMAC	77.458	65.878	70.201	74.472	66.341	**76.531**
madelon	52.962	68.423	64.346	70.192	51.462	**70.769**
lung_discrete	83.562	85.256	89.041	89.041	87.671	**90.411**
Prostate_GE	80.392	62.745	81.373	**88.235**	79.412	87.255
MNIST	95.006	60.591	95.257	91.921	75.479	**96.204**
AWA	22.825	17.425	18.998	17.141	17.942	**21.514**

5.4 Denoising Experiments

In order to evaluate the robustness of denoising AEFS, we compare the clustering and classification performance of denoising AEFS with AEFS. We add Gaussian noise with zero mean value and 0.1 standard deviation to all the datasets and conduct clustering and classification experiments similar to the above. The experimental results are shown in Table 5. We can see that under noisy circumstance dAEFS perform much better than AEFS. This is because that dAEFS tries to reconstruct the uncorrupted data from the corrupted input so as to undo the corruption in the training process.

Table 5: Clustering and classification results of AEFS and dAEFS. The best results are highlighted in bold.

Dataset	Clustering NMI		Clustering ACC		Classification ACC	
	AEFS	dAEFS	AEFS	dAEFS	AEFS	dAEFS
Isolet	72.6±1.6	**76.3±1.7**	58.4±2.7	**61.2±3.6**	87.885	**90.385**
warpPIE10P	49.5±5.0	**54.1±4.7**	45.9±4.9	**49.5±5.1**	**99.048**	**99.048**
PCMAC	41.0±1.5	**45.6±1.8**	52.1±1.2	**52.5±1.2**	75.142	**75.965**
madelon	29.0±1.0	**40.0±0.1**	59.5±3.0	**61.7±0.1**	69.538	**71.077**
lung_discrete	68.4±6.0	**69.7±5.9**	71.6±8.2	**72.1±8.3**	90.411	**91.781**
Prostate_GE	10.0±9.4	**15.4±10.0**	63.2±8.6	**68.2±9.8**	85.294	**87.255**
MNIST	46.1±3.2	**48.8±2.8**	51.8±4.8	**56.2±4.8**	96.204	**96.771**
AWA	7.6±0.3	**7.7±0.2**	13.4±0.4	**13.7±0.3**	21.514	**22.499**

5.5 Reconstruction Experiments

We conduct reconstruction experiments on the face dataset warpPIE10P using AEFS and RSR. The results are shown in Fig. 3. In Fig. 3(b) and (f), the large weights of features learned by AEFS mainly distribute in the area of eyebrow, eye, nose and mouth which are important for recognition, while the weights learned by RSR is discriminative only in eye position and the eyebrow, nose and mouth are not distinct from other parts. From Fig. 3(c)(d)(g)(h), we can see that both AEFS and RSR can well reconstruct the raw face with much fewer features than the original. However, the reconstructed face of RSR is less similar to raw face than AEFS, especially when the number of selected features is small.

We also evaluate reconstruction ability of denoising AEFS on the face dataset warp-

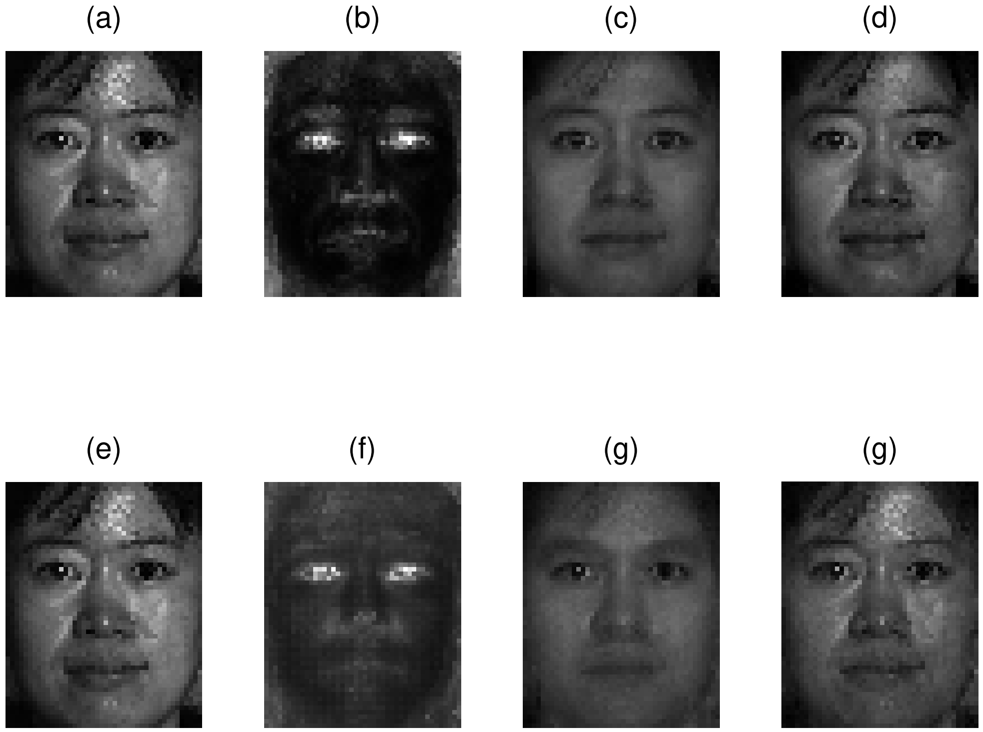

Figure 3: Face reconstruction: (a)(e) raw face (55 × 44px), (b)(f) feature weight map, (c)(g) reconstructed face using 300 feature, (d)(h) reconstructed face using 1000 features. The first row is the results of AEFS and the second row is of RSR.

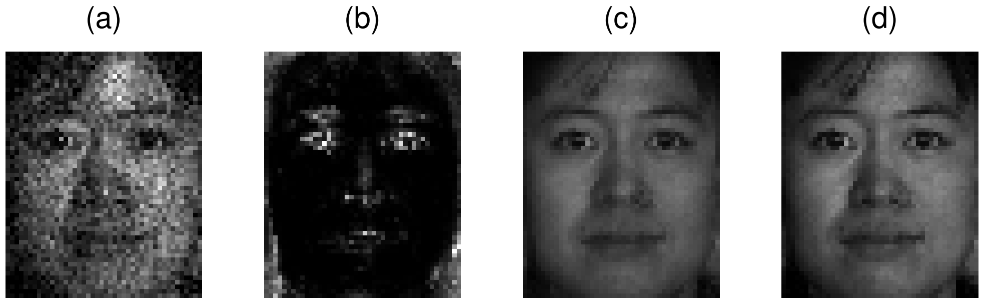

Figure 4: Face reconstruction using dAEFS: (a) raw face with Gaussian noise, (b) feature weight map, (c) reconstructed face using 300 feature, (d) reconstructed face using 1000 features.

PIE10P with Gaussian noise. The noise is set zero mean value and 0.5 standard deviation. The result is shown in Fig. 4. Although the noise exists in the face images, we can see that the learned weight map is robust enough and the reconstructed face is noiseless.

6 Conclusion and Future Work

We propose a novel unsupervised feature selection method which could jointly learn a self-reconstruction autoencoder model and the importance weights of each feature. The autoencoder nonlinearly represent each feature using all the features with different weights. With the ℓ_{21} norm regularization on the weight matrix, if a feature is important, then it will participate in the representation of most of other features, leading to a large row-norm of representation weight matrix, and vice versa. As a result, the most representative features which can well reconstruct other features nonlinearly are selected. We also design a denoising AEFS which could be more robust to corruption and noise. Both AEFS and denoising AEFS can be efficiently optimized by gradient projection method with back-propagation algorithm. Experimental results on different real world datasets validate the superiority of our methods. Future work will include the extension to stacked autoencoders and supervised scenarios.

References

[1] Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *ICML*, pages 1247–1255, 2013.

[2] Yoshua Bengio. Learning deep architectures for ai. *Foundations & Trendső in Machine Learning*, 2(1):1–127, 2009.

[3] Paul S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In *Fifteenth International Conference on Machine Learning*, pages 82–90, 1998.

[4] Deng Cai, Chiyuan Zhang, and Xiaofei He. Unsupervised feature selection for multi-cluster data. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 333–342. ACM, 2010.

[5] Patrick L Combettes and Valérie R Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

[6] Mark Fanty and Ronald Cole. Spoken letter recognition. In *Advances in Neural Information Processing Systems*, pages 220–226, 1990.

[7] Isabelle Guyon, Jiwen Li, Theodor Mader, Patrick A. Pletscher, Georg Schneider, and Markus Uhr. Competitive baseline methods set new standards for the nips 2003 feature selection benchmark. *Pattern Recognition Letters*, 28(12):1438–1444, 2007.

[8] Xiaofei He, Deng Cai, and Partha Niyogi. Laplacian score for feature selection. In *Advances in neural information processing systems*, pages 507–514, 2005.

[9] Geoffrey E Hinton and Richard S Zemel. Autoencoders, minimum description length, and helmholtz free energy. *Advances in neural information processing systems*, pages 3–3, 1994.

[10] Emanuel F. Petricoin Iii, David K. Ornstein, Cloud P. Paweletz, Ali Ardekani, Paul S. Hackett, Ben A. Hitt, Alfredo Velassco, Christian Trucco, Laura Wiegand, and Kamillah Wood. Serum proteomic patterns for detection of prostate cancer. *Journal of Urology*, 94(20):1576–8, 2002.

[11] C. H. Lampert, H. Nickisch, and S. Harmeling. *Learning to detect unseen object classes by between-class attribute transfer*. IEEE, 2009.

[12] Ken Lang. *NewsWeeder: Learning to Filter Netnews*. 1995.

[13] Y. L. Lecun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. proc ieee. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[14] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P. Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *CoRR*, abs/1601.07996, 2016. URL http://arxiv.org/abs/1601.07996.

[15] Zhizheng Liang and Tuo Zhao. Feature selection for linear support vector machines. In *International Conference on Pattern Recognition*, pages 606–609, 2006.

[16] Sofia Mosci, Lorenzo Rosasco, Matteo Santoro, Alessandro Verri, and Silvia Villa. Solving structured sparsity regularization with proximal methods. In *Machine Learning and Knowledge Discovery in Databases*, pages 418–433. Springer, 2010.

[17] Erkki Oja. Data compression, feature extraction, and autoassociation in feedforward neural networks. *Artificial neural networks*, 1991.

[18] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 27(8):1226–38, 2005.

[19] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *Artificial Neural NetworksICANN'97*, pages 583–588. Springer, 1997.

[20] Rudy Setiono and Huan Liu. Neural-network feature selector. *Neural Networks, IEEE Transactions on*, 8(3):654–662, 1997.

[21] Terence Sim, Simon Baker, and Maan Bsat. The cmu pose, illumination, and expression (pie) database. In *IEEE International Conference on Automatic Face and Gesture Recognition, 2002. Proceedings*, pages 46 – 51, 2002.

[22] Robert Tibshirani. Regression shrinkage and selection via the lasso. *J of the Royal Statistical Society*, 58(1):267–288, 1996.

[23] Antanas Verikas and Marija Bacauskiene. Feature selection with neural networks. *Pattern Recognition Letters*, 23(11):1323–1335, 2002.

[24] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM, 2008.

[25] Yi Yang, Heng Tao Shen, Zhigang Ma, Zi Huang, and Xiaofang Zhou. l2, 1-norm regularized discriminative feature selection for unsupervised learning. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 1589. Citeseer, 2011.

[26] Pengfei Zhu, Wangmeng Zuo, Lei Zhang, Qinghua Hu, and Simon CK Shiu. Unsupervised feature selection by regularized self-representation. *Pattern Recognition*, 48(2): 438–446, 2015.