Knowledge-Based Systems 215 (2021) 106748

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Unsupervised feature selection via transformed auto-encoder

Yunhe Zhang, Zhoumin Lu, Shiping Wang *

College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350116, China
Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, Fuzhou 350116, China

ARTICLE INFO

Article history:
Received 17 August 2020
Received in revised form 1 November 2020
Accepted 1 January 2021
Available online 9 January 2021

Keywords:
Machine learning
Deep learning
Feature selection
Unsupervised learning
Auto-encoder

ABSTRACT

As one of the fundamental research issues, feature selection plays a critical role in machine learning. By the removal of irrelevant features, it attempts to reduce computational complexities of upstream tasks, usually with computation accelerations and performance improvements. This paper proposes an auto-encoder based scheme for unsupervised feature selection. Due to the inherent consistency, this framework can solve traditional constrained feature selection problems approximately. Specifically, the proposed model takes non-negativity, orthogonality, and sparsity into account, whose internal characteristics are exploited sufficiently. It can also employ other loss functions and flexible activation functions. The former can fit a wide range of learning tasks, and the latter has the ability to play the role of regularization terms to impose regularization constraints on the model. Thereinafter, the proposed model is validated on multiple benchmark datasets, where various activation and loss functions are analyzed for finding better feature selectors. Finally, extensive experiments demonstrate the superiority of the proposed method against other compared state-of-the-arts.

1. Introduction

Data dimensionality reduction [1] is rapidly becoming a popular research area with the increasing dimension of data. The explosion of data dimension called the curse of dimensionality raises several issues. For instance, as the number of features increases, the sample data of the corresponding feature space could exhibit exponential growth. Because of the reasons mentioned above, high dimensional data may increase the time cost of feature analysis and training models. This phenomenon also leads to the high complexity of models. Therefore, dimensionality reduction is a significant and promising theme, especially in the era of big data.

Data dimensionality reduction is generally used to process raw high-dimensional data by leaving more critical information for better behavior in the subsequent processing steps. The dimensionality reduction methods can be roughly divided into categories: feature selection [2–7], feature extraction [8–10], and feature fusion [11]. Feature fusion copes with the selection and combination of features to remove redundant and irrelevant features. It aims at processing data with different types of features frequently. Feature extraction is defined as the method that makes the high dimensional data project to a new low-dimensional feature space. The major problem in feature extraction is whether the semantics of original data can be maintained without loss or not. Compared with feature extraction, feature selection aims to approximate the original data by selecting a set of essential features. This type of method can make a model more reliable and be furnished with better interpretability. Moreover, feature selection is essential for a wide range of technologies, such as data mining [12,13], machine learning [14], and so on.

From the perspective of the availability of label information, feature selection is categorized as supervised [15,16], and unsupervised methods [17–19] in common. Supervised methods require sufficiently labeled training data, and are commonly designed for classification or regression problems. But generally, most of the supervised methods consume more running times and computational cost to maintain their advantages. Most labels of the data are also difficult to obtain in real-world applications. Unsupervised feature selection methods are precisely designed to work without label information. Therefore, in some practical situations, the proposed model in the unsupervised feature selection type is encouraging.

Feature selection is also divided into wrapper [20,21], filter [22,23] and embedded methods [24–26] from the angle of search strategies. Wrapper methods select features by generating feature candidates commonly. A main challenge of wrapper methods is that they consume much more training time and severe algorithm complexity. Filter methods select features through unsupervised fitting according to probabilistic and intrinsic characteristics of the dataset. This type of approach usually evaluates each feature by statistical methods. Besides, filter methods generally require two steps to select features. They score features

* Corresponding author at: College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350116, China.
E-mail address: shipingwangphd@163.com (S. Wang).

https://doi.org/10.1016/j.knosys.2021.106748
0950-7051/

firstly and then select features based on predefined thresholds. Different from filter methods, embedded methods integrate the feature selection process and the learning training process into a unified framework. In other words, embedded methods will obtain the accuracy of the model once this model is established. However, traditional embedded methods, like Lasso feature selection, cannot handle non-linear data normally. Actually, most feature selection methods use the kernel mapping method for non-linear learning. The unsupervised feature selection algorithm of kernel mapping could increase the amount of data that needs to be stored. Meanwhile, all training datasets are required for each prediction. Except for the ineffectiveness to handle non-linear data, previous feature selection methods often suffer from several other problems, such as ill-considered orthogonality or non-negative constraints, and weak optimization scalability of optimization loss functions.

Deep learning [27,28] has become a promising and hot research topic in recent years. In an early stage, Hinton et al. [29] combined deep learning with data dimensionality reduction and proposed the deep belief nets (DBN), which is able to learn a joint distribution between test data and labels. This model works in the following phases. The first phase is to extract the features of the data to be processed, then use the classifier to sort them out. Except for DBN, the auto-encoder network is also discovered that can be applied to solve feature selection problems in this paper as a deep learning model.

Inspired by the rapid development of deep learning and the main problems of existing feature selection methods, we revisited the general problem of unsupervised feature selection and proposed a general objective function. Then we come to discover that this objective function can be solved by a deep auto-encoder network. We consider that the auto-encoder network is not only able to deal with both linear and non-linear data through different activation functions, but also has the ability to generalize models better. It can reduce the dimensions of data while maintaining high-quality representation as well.

Consequently, we propose an effective unsupervised feature selection method via transformed auto-encoders (UFS-TAE), which aims to acquire a feature selector constrained by orthogonality and non-negativity. The proposed method is mainly divided into three phases. First and foremost, we obtain the indicator matrix constrained by orthogonality via deep auto-encoder. Then we use the non-negative least squares method to obtain the approximate and non-negative indicator matrix. Ultimately, we select the feature selection matrix according to the indicator matrix and evaluate the proposed model with K-means. The architecture of the proposed model is presented in Fig. 1. In experiments, we use nine unsupervised feature selection algorithms and six datasets for comparison. For the fairness of the experiment, two common evaluation metrics are utilized, including clustering accuracy and normalized mutual information.

Except for the encouraging performance, the proposed model also takes advantage of several beneficial aspects. Traditional machine learning algorithms generally use only one certain objective function to solve the corresponding problem. However, the proposed model has the ability to handle different issues with different loss functions. More than that, it can also satisfy different needs through varying activation functions. The activation function is capable of replacing regularization terms to some extent. For instance, the ReLU activation function could make the output of some neurons sparse, which causes the sparsity of the network and reduces the interdependence of parameters, thereby alleviating the occurrence of overfitting. The ℓ_1-norm is commonly used in general machine learning algorithms and also has the leaves a similar effect on the model. These advantages make the proposed model more convenient and promising. We also add $\ell_{2,1}$-norm to sparse rows of the indicator matrix with joint sparsity. In summary, the main contributions of this paper are summed up as follows:

- Propose the model with non-negativity and orthogonality, which selects discriminative features skillfully through auto-encoder, where loss and activation functions are flexible when handling different learning tasks.
- Different from other auto-encoder based methods, the lifted transformed net ranks original features instead of generating hidden representation, while maintaining the ability to preserve the non-linear structure of data.
- The proposed algorithm offers a new perspective for feature selection, where implementation platforms and embedding characteristics are convenient and efficient, validated by superior performance in comparative experiments.

This paper is structured as follows on the whole. First of all, some related works are reviewed in Section 2. Then we present the problem formulation and the optimization strategy of the feature selection problem in Section 3. Furthermore, we report the performance of the proposed algorithm against several excellent methods and analyze its advantages in Section 4. Finally, this paper is concluded in Section 5.

2. Related works

In this section, we provide an overview of the related feature selection methods. Laplacian score (LS) [30] as a filter method, gets compressed data through evaluating features according to their Laplacian scores obtained by the degree of consistency of the Gaussian Laplacian matrix and features. But LS does not consider the relationship between features so that it is possible to select redundant features. Except for LS, Cai et al. proposed an unsupervised feature selection method for multi-cluster data (MCFS) [31]. MCFS can well preserve the structure of multi-cluster through using spectral analysis techniques and obtain the final result. Cluster indicator is obtained via spectral clustering and then uses the indicator matrix to perform feature selection. This method can also well solve the sparse eigenproblem and the L1-regularized least-squares problem in the optimization. But MCFS ignores the non-negative constraint, increasing difficulty in getting the cluster labels. Yang et al. [32] proposed an unsupervised discriminative feature selection method (UDFS) by selecting the most discriminative features for data representation, which is learning with the consideration of manifold structure, and using $\ell_{2,1}$-norm in the process. After UDFS, Li et al. constructed the non-negative discriminative feature selection (NDFS) [33] framework, which obtains compressed data through a one-step strategy. This framework can learn both the pseudo-class labels and compressed data simultaneously. NDFS is more flexible than UDFS through experiments in that paper. However, both UDFS and NDFS are not robust and are vulnerable to outliers or noise. Then another feature selection method named robust unsupervised feature selection (RUFS) [34] utilized the robust clustering with local learning and the robust feature selection with $\ell_{2,1}$-norm regularized minimization to solve this problem.

Combining graphs and feature selection problems is also a good solution. Wang et al. [35] combined the learned sparse graph and feature selection problem into one framework named the sparsity preserving feature selection (SPFS) approach. Then the neighborhood preserving feature selection (NPFS) algorithm is also proposed to strengthen the constraints of SPFS to learn a sparse graph structure, considering the preservation of neighborhoods for locally linear structures. But both SPFS and NPFS are divided into two phases. For a more convenient optimization, the neighborhood embedding feature selection (NEFS) algorithm is

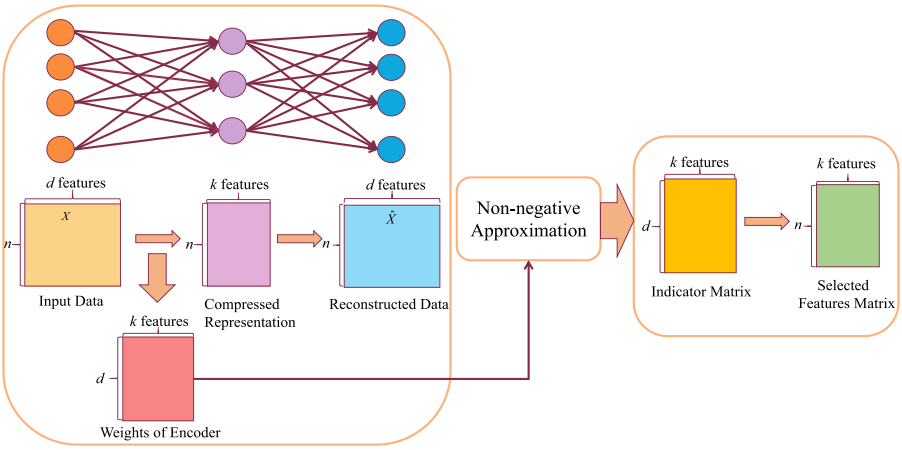


Fig. 1. The proposed model architecture. This model employs the auto-encoder to solve feature selection problems. With appropriate activation and loss functions, the learned weights hold coherence with original indicator matrix. After training, the indicator matrix is restored by non-negative approximation.

proposed in the same paper. NEFS incorporates these processes into one joint framework. Li et al. [36] constructed the generalized uncorrelated regression model with adaptive graph structure (URAFS) to solve and optimize unsupervised feature selection problems.

However, a main problem of existing feature selection methods remains the time cost and computation complexity of processing big data. In order to handle these troubles of big data, many limited deep-neural-network based methods [37–39] have been proposed. Han et al. [40] proposed an auto-encoder feature selection (AEFS) method. AEFS combined group lasso and auto-encoder network into a new unsupervised embedded feature selection model. But this method cannot ensure whether the orthogonal and non-negative constraint of the indicator matrix exists because of the direct acquisition of the weight matrix. And another difference between it and our method is that this model trains an auto-encoder network first, then sorts features and selects k features. Specifically, these two steps are independent. To our knowledge, this property will reduce the interpretability of the model. Our model sets the number of neurons in the hidden layer to k directly. So we can consider the weight of encoder network as the indicator matrix naturally. Another deep feature selection method named concrete auto-encoders for differentiable feature selection (CAE) was proposed by Abid et al. [41]. CAE uses the concrete selector layer to train the whole model and then uses a discrete arg max to test the model. In this paper, we attempt to use a transformed auto-encoder network to solve the feature selection problem with orthogonality and non-negativity constraints. Furthermore, we provide new and different perspectives for deep feature selection problems.

3. Proposed method

In this section, an effective unsupervised feature selection method via transformed auto-encoders is proposed. Firstly we start with the problem formulation of general unsupervised feature selection and obtain the objective function, then propose a deep learning method to recover the formulated problem constrained with non-negativity and orthogonality.

3.1. Problem formulation

A given data point set is denoted as $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$ and data matrix as $\mathbf{X} = [\mathbf{x}_1; \cdots; \mathbf{x}_n] \in \mathbb{R}^{n\times d}$, where $\mathbf{x}_i \in \mathbb{R}^{1\times d}$ is a d-dimensional column vector. Given the number k of selected features, unsupervised feature selection aims to search an optimal feature subset of k elements satisfying some certain criteria. Assume that the variable $\mathbf{I}$ is the index of selected features, then $\mathbf{X}^{\mathbf{I}}$ is the data matrix with dimensionality reduction. Without loss of generality, unsupervised feature selection problems can be written as the following canonical form of

$$\min_{\mathbf{I}} \mathcal{L}(\mathbf{X}; \mathbf{X}^{\mathbf{I}}). \tag{1}$$

Notably, since the variable $\mathbf{I}$ denotes a discrete index set so that it is difficult to optimize, we introduce an indicator matrix $\mathbf{H}$ to substitute $\mathbf{I}$. Equivalently, $\mathbf{H} = [\mathbf{H}_{ij}]_{d\times k}$ can be defined by

$$\mathbf{H}_{ij} = \begin{cases} 1, & \mathbf{I}(j) = i, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Therefore, $\mathbf{XH}$ serves as the selected feature matrix. When integrating a coefficient matrix $\mathbf{Z} \in \mathbb{R}^{k\times d}$ to approximate the initial input matrix, then the feature selection problem can be rewritten as the following form:

$$\min_{\mathbf{H},\mathbf{Z}} \mathcal{L}(\mathbf{X}; \mathbf{XHZ}). \tag{3}$$

The data point set $\mathcal{X}$ can be regarded as a feature space and we consider the $\mathbf{XH}$ as the process of matrix dimension reduction. However, $\mathbf{H}$ needs to be constrained by $\mathbf{H} \geq \mathbf{0}$ apparently according to its definition. Furthermore, $\mathbf{H}$ is constrained to be orthogonal and sparse to ensure the validation of the feature selector. The orthogonality constraint makes $\mathbf{H}$ maintain its property as an indicator matrix in the whole learning process. This characteristic can improve interpretability of the proposed model. After adding the above constraints, the objective function of our proposed method can be formulated as follows:

$$\min_{\mathbf{H},\mathbf{Z}} \mathcal{L}(\mathbf{X}; \mathbf{XHZ}) + \mu \|\mathbf{H}\|_{2,1} \quad \text{s.t.} \quad \mathbf{H} \geq \mathbf{0}, \mathbf{H}^{\mathbf{T}}\mathbf{H} = I, \tag{4}$$

where μ is a parameter of the regularization term. In order to keep a trade-off between sparsity and smoothness, $\ell_{2,1}$-norm is utilized.

Though the aforementioned formulation provides a well -defined objective optimization function, it is tough to develop an efficient algorithm for satisfying flexible loss functions and strict constraints. Hereinafter, we attempt to utilize neural network architectures to search beneficial effective solutions.

3.2. Optimization method

To solve the aforementioned unsupervised feature selection problem, we first alleviate the problem by absorbing the orthogonality constraint into optimization function. Hence, the optimization problem is reformulated as

$$\min_{\mathbf{H},\mathbf{Z}} \mathcal{L}(\mathbf{X}; \mathbf{XHZ}) + \frac{\lambda}{4}\left\|\mathbf{H}^{\mathbf{T}}\mathbf{H} - I\right\|_{\mathbf{F}}^{2} + \mu\|\mathbf{H}\|_{2,1} \quad \text{s.t.} \quad \mathbf{H} \geq \mathbf{0}, \tag{5}$$

where λ is a relatively large parameter for the orthogonality constraint. Here, orthogonality and non-negativity jointly guarantee that each column of $\mathbf{H}$ has only one large element and the others are close to zero. These constraints will make the model more interpretable and have a better feature selection result.

It is observed that the above optimization problem has some internal consistency with auto-encoders. The generalized auto-encoder network contains two parts: An encoder network and a decoder network. The encoder network seeks to learn a feature representation of input data. Differently, the decoder network aims to reconstruct the input data according to the output of the encoder network. In the hidden layers, there exist some latent compressed representations of input data. Therefore, we actually utilize $\sigma(\sigma(\mathbf{XC})\mathbf{W})$ to approximate the original $\mathbf{XHZ}$ during the whole optimization. Here $\mathbf{C}$ is the weight matrix of encoder, $\mathbf{W}$ denotes the weight of decoder and $\sigma(\cdot)$ denotes the activation function of auto-encoder network. In the proposed model, we use the non-linear activation function to make it has the ability to process and save the non-linear structure of the data. We consider the weight matrix $\mathbf{C}$ with sparse and orthogonality constraints as a surrogate of the indicator matrix $\mathbf{H}$ in our objective function and then $\mathbf{H}$ would be recovered from $\sigma(\mathbf{XC})$. So $\mathbf{H}$ would be an indicator matrix with approximate orthogonality and sparsity. In this way, we establish a relationship between the auto-encoder and unsupervised feature selection preliminarily.

Since the non-negative constraint on the weight is difficult to achieve with traditional auto-encoder networks, we temporarily relax the constraints on the weight matrix so that the weight matrix $\mathbf{C}$ is updated with orthogonality constraint and $\ell_{2,1}$-norm. Then the objective function is rewritten as the following form of

$$J(\mathbf{C}, \mathbf{W}) = \mathcal{L}(\mathbf{X}; \sigma(\sigma(\mathbf{XC})\mathbf{W})) + \frac{\lambda}{4}\left\|\mathbf{C}^{\mathbf{T}}\mathbf{C} - I\right\|_{\mathbf{F}}^{2} + \mu\|\mathbf{C}\|_{2,1}, \tag{6}$$

Next, according to the gradient descent method, the weight matrix $\mathbf{C}$ can be obtained by the following updating rule:

$$\mathbf{C}^{(t)} = \mathbf{C}^{(t-1)} - \alpha \times \nabla_{\mathbf{C}^{(t-1)}} J(\mathbf{C}, \mathbf{W}), \tag{7}$$

where $\mathbf{C}^{(t)}$ denotes the tth iteration of $\mathbf{C}$, α indicates the learning rate of the auto-encoder and $\nabla_{\mathbf{C}} J(\mathbf{C}, \mathbf{W})$ is the gradient of $\mathbf{C}$. In addition, the gradient of the weight matrix $\mathbf{C}$ is calculated by

$$\nabla_{\mathbf{C}} J(\mathbf{C}, \mathbf{W}) = \frac{\partial J(\mathbf{C}, \mathbf{W})}{\partial \mathbf{C}} = \frac{\partial \mathcal{L}}{\partial \mathbf{C}} + \lambda\mathbf{C}(\mathbf{C}^{\mathbf{T}}\mathbf{C} - I) + \mu\mathbf{DC}, \tag{8}$$

where $\mathbf{D}$ is a diagonal matrix with the jth diagonal element being $\frac{1}{\|\mathbf{C}_j\|_2}$ and $\mathbf{C}_j$ denotes the jth row of $\mathbf{C}$. Similarly, the weight matrix $\mathbf{W}$ is obtained through iterative calculations as follows:

$$\mathbf{W}^{(t)} = \mathbf{W}^{(t-1)} - \alpha \times \nabla_{\mathbf{W}^{(t-1)}} J(\mathbf{C}, \mathbf{W}) = \mathbf{W}^{(t-1)} - \alpha \times \frac{\partial \mathcal{L}}{\partial \mathbf{W}}. \tag{9}$$

After the training process, the weight matrix $\mathbf{C}$ satisfying an approximative orthogonality constraint and sparsity has been obtained. Furthermore, we utilize $\sigma(\mathbf{XC})$ to approximate $\mathbf{XH}$ so that the indicator matrix $\mathbf{H}$ in the original problem can be restored from the weight matrix $\mathbf{C}$. It is noted that auto-encoder possesses the ability to preserve non-linear structures of data. We consider the non-linear structure of hidden representation obtained by the encoder network is calculated through the weight of encoder. Hence, the recovered indicator matrix can also preserve the non-linear structure to some extent. Similarly, $\mathbf{H}$ can be seen as the sparse and orthogonal. Now our goal is to recover the indicator matrix $\mathbf{H}$ from the hidden layer by

$$\mathbf{XH} \approx \sigma(\mathbf{XC}), \tag{10}$$

where $\mathbf{H} \in \mathbf{R}^{d\times k}$ is restricted with non-negativity. And Equation (10) can be transformed as the following objective subproblem:

$$\min_{\mathbf{H}} \frac{1}{2}\|\mathbf{E} - \mathbf{XH}\|_F^2 \quad \text{s.t.} \quad \mathbf{H} \geq \mathbf{0}, \tag{11}$$

where $\mathbf{E} = \sigma(\mathbf{XC})$ is the output of the encoder network. In this equation, matrix $\mathbf{E}$ and matrix $\mathbf{X}$ have been known. Ultimately, we use a non-negative least square to solve Eq. (11) and obtain the non-negative indicator matrix $\mathbf{H}$.

Actually, for solving the unsupervised feature selection problem with the constraint of indicator matrix $\mathbf{H} \geq \mathbf{0}$, the proposed method restores $\mathbf{H}$ from $\mathbf{C}$ after the training process of the auto-encoder. Our method named unsupervised non-negative feature selection via transformed auto-encoder (UFS-TAE) is summarized in Algorithm 1.

Algorithm 1 Unsupervised Feature Selection via Transformed Auto-Encoder (UFS-TAE)

Input: The input data $\mathbf{X} \in \mathbb{R}^{n\times d}$, the number of selected features k, loss function $\mathcal{L}(\cdot)$, regularization coefficient μ, orthogonal parameter λ and learning rate α.

Output: The selected feature indicator vector $\mathbf{I}$.

1: Initialize the weight parameters $\mathbf{C} \in \mathbb{R}^{d\times k}$ and $\mathbf{W} \in \mathbb{R}^{k\times d}$ of auto-encoders;
2: **repeat**
3: Calculate loss of auto-encoder by Equation (6);
4: Update $\mathbf{C}$ by Equation (7);
5: Update the j-th diagonal element of $\mathbf{D}$ by $\frac{1}{\|\mathbf{C}_j\|_2}$;
6: Update $\mathbf{W}$ by Equation (9);
7: **until** Convergence
8: Calculate the non-negative indicator matrix $\mathbf{H}$ by Equation (11);
9: **return** Calculate $\|\mathbf{H}_i\|_F^2$ of each row of $\mathbf{H}$ and sort it in a descending order. Set $\mathbf{I}$ be the top k largest indices of the ordered vector.

With regard to the concrete loss term $\mathcal{L}(\cdot)$, we can utilize different loss functions according to different characteristics of data. Actually, the main difference between auto-encoder and other models of feature selection is that auto-encoder has the ability to learn non-linear relationships of data efficiently. We list several commonly used loss functions to perform comparative experiments. In the next experimental section, the effectiveness and excellent performance of our method are presented.

4. Experiments

In this section, comprehensive experiments are conducted on six publicly available datasets. In order to provide a fair test bed, parameter settings and evaluation metrics are introduced in details. Then we present the experimental results of our experiment. For the effectiveness of the proposed method, we also show the visualization results of all methods and the learning process of our model. Besides, we conduct the internal experiment to indicate our model have the ability to handle different learning tasks. The effects of different activation functions on different datasets are also displayed in this section. Ultimately, the parameter sensitivity analysis will show that the parameters set in our model are meaningful.

Table 1
Detailed Introduction to Datasets.

Datasets	#Instances	#Features	#Classes
Yale	165	1024	15
Lung	203	3312	5
WarpPIE10P	210	2420	10
Orlraws10P	100	10 304	10
WarpAR10P	130	2400	10
Madelon	2600	500	2

4.1. Dataset

All compared unsupervised feature selection methods are evaluated by six real-world datasets, including Yale, WarpPIE10P, Madelon, WarpAR10P, Lung, and Orlraws10P. These repositories contain one biological dataset, four face image datasets, and one artificial dataset. Hereinafter, the detailed introduction to datasets is demonstrated in Table 1.

Yale consists of 165 gray-scale images. These images are taken from 15 individuals. Besides, there are 11 images per individual, and one image per different facial expressions or configurations, such as happy, normal, sad, sleepy, surprised, and wink. Each face is represented as a 32-by-32 gray-scale image.

WarpPIE10P contains in total of 210 samples from 10 classes. The abbreviation of the pose, illumination, and expression consists of the so-called pie. There are more than 40,000 facial images of 68 people in this dataset. Each person is imaged across 13 poses, under 43 illumination conditions, and with 4 expressions.

Madelon is an artificial dataset that contains data points grouped in 32 clusters. These data points are placed on the vertices of a five-dimensional hypercube and randomly labeled as 1 or -1. This dataset is multivariate and highly non-linear. We use the dataset consisting of the training set and validation set simultaneously.

WarpAR10P contains 126 persons with different facial expressions, illumination conditions and occlusions, including 70 men and 56 women, a total of more than 4,000 color images. Each person participated in two sessions and the same picture was taken in both sessions.

Lung is a popular biological dataset which has 203 instances and each of them has 3,312 features. These samples are classified into 5 classes, all classes with 139, 21, 20, 6, 17 instances respectively.

Orlraws10P is a face image dataset that consists of 10 classes from different countenances such as facial expressions and facial details. Each face is a 92-by-112 gray resolution, reshaped as a 10,304 dimensional feature vector.

4.2. Parameter settings

To verify the effectiveness of the proposed method, several state-of-the-art unsupervised feature selection benchmark methods are adopted, including Laplacian score (LS), non-negative spectral analysis for feature selection (NDFS), $\ell_{2,1}$-norm regularized discriminative feature selection (UDFS), multi-view clustering feature selection (MCFS), robust unsupervised feature selection (RUFS), generalized uncorrelated regression with the adaptive graph for unsupervised feature selection (URAFS), auto -encoder-inspired unsupervised feature selection (AEFS) and concrete auto-encoder (CAE) feature selection.

For LS, MCFS, and UDFS, we fix the size of the neighborhood at 5 for all the datasets. In MCFS, we build a kNN graph with the heat kernel weights and set the parameter $\sigma = 5$. The regularization parameter of UDFS is set to 1. For guaranteeing the orthogonality satisfied, we let the orthogonality parameter equal 1000 in NDFS and UDFS. For RUFS, the graph regularization parameter ν, the parameter on the regression term α and the parameter to control the row-sparsity of the projection matrix β are all set to 1. Except for the parameters mentioned above, the max number of iterations is set to 100 and convergence threshold is set to 1×10^{-4}. For URAFS, we let the coefficients $\alpha = 1$, $\beta = 1$ and the regularization parameter $\lambda = 100$. AEFS is also an auto-encoder based method, we set the number of neutrons in the hidden layer to 256 and the weight of sparsity penalty term $\beta = 0.3$. In addition, the learning rate and iteration are set to 0.03 and 100. The activation function type is set to Sigmoid. For CAE, we use the default settings in its exact package and set the decoder to a one-layer network. The activation function of the decoder is set to Leaky ReLU.

In the proposed method, we use the auto-encoder network contains one hidden layer to perform the experiment and let the model choose the best performance when $k \in \{5, 10, \ldots, 95, 100\}$ features are selected respectively. With regard to other parameters, in the internal experiment, we set the learning rate $\alpha = 0.001$, the orthogonality parameter $\lambda = 10$ and the regularization parameter $\mu = 0.01$. In the comparison between our model and other algorithms, we utilize a hyperparameter optimization framework named optuna to seek better results. The activation function and loss function are fixed to Mish and BCEWithLogits loss. Except for those settings, we set the regularization parameter μ, and learning rate α be in the range of $[1 \times 10^{-5}, 1 \times 10^{-1}]$. The parameter of orthogonality λ varies in $[1 \times 10^{-5}, 100]$.

We also select four loss functions in the experiment: L1 loss function, MSE loss function, BCE With Logits (BCEWithLogits) loss function, and KL divergence loss function (KLDiv loss). It is noted that we use the BCEWithLogits loss function because BCE loss needs input data to vary in [0, 1]. The BCEWithLogits loss function integrates the sigmoid layer into the BCE loss class. This version is more stable in numerical value than using a simple sigmoid layer and BCE loss after combining these two operations into one layer. This technique is used to achieve numerical stability in common.

In addition, the primary influence of activation functions is to transform the current feature space to another space through a certain linear mapping, so that the data can be better classified. Hence, how to deal with the input data is closely related to the selection of activation function as well. We choose five activation functions in total in the experiment. Except for commonly used activation functions (Sigmoid, ReLU, Leaky ReLU, Tanh), Mish function is also employed. Mish function is a new type of activation function, which is called the substitution of ReLU function but performs better than it. Mish activation avoids saturation and slightly allows for negative values. Lately, it is discovered that the combination of Mish activation and ranger optimizer may have an excellent behavior.

In experimental comparison, all feature selection methods are evaluated by their K-means clustering performances. Moreover, clustering results are assessed by two popular evaluation metrics, including accuracy (ACC) and normalized mutual information (NMI). Let $\mathbf{Y}$ be the ground truth label and $\widehat{\mathbf{Y}}$ be the predicted label from K-means clustering algorithm. Then the clustering accuracy metric can be calculated as follows:

$$ACC = \frac{\sum_{i=1}^{n} \delta(\mathbf{y}_i, map(\hat{\mathbf{y}}_i))}{n},$$

where $\mathbf{y}_i$ is the ground truth of the ith instance. Here, $map(\cdot)$ is a permutation mapping that could best match the predicted labels to the given ground truths. Besides, $\delta(\mathbf{y}_i, map(\hat{\mathbf{y}}_i))$ is defined as follows:

$$\delta(\mathbf{y}_i, map(\hat{\mathbf{y}}_i)) = \begin{cases} 1, & \mathbf{y}_i = map(\hat{\mathbf{y}}_i), \\ 0, & \text{otherwise.} \end{cases}$$

Except for ACC, NMI is also a commonly used metric in the machine learning field. According to the definition of NMI displayed as follows, it is obvious that the NMI metric is obtained

through the mutual information (MI). Here the NMI metric in our experiment is calculated by

$$NMI(\mathbf{Y}, \widehat{\mathbf{Y}}) = \frac{MI(\mathbf{Y}, \widehat{\mathbf{Y}})}{\max\{H(\mathbf{Y}), H(\widehat{\mathbf{Y}})\}},$$

where $H(\cdot)$ denotes the entropy. The value of NMI ranges from 0 to 1.

4.3. Experimental results

The whole experiment is divided into two parts. In the first part, we employ five activation functions and four loss functions for the internal comparison. And we select the best result of every combination to compare in this part. It is noted that we just focus on feature selection to obtain internal results in this paper. Ultimately, we choose the combination of Mish and BCEWithLogits loss function to compare with other state-of-art methods because this combination behaves better than others on average.

In the second part, eight unsupervised feature selection algorithms and the baseline of feature selection are compared with the proposed method. The baseline method named All Features means that all features are clustered with K-means. Besides, two metrics, including ACC and NMI, are used to evaluate the effectiveness of all tested methods on six datasets. The experimental results of different algorithms are compared when the same dimensions are selected.

In this section we mainly show three convincing evidence to prove the effectiveness of our model. Except for the experimental results, we also display the visualization results of feature selection of each algorithm and the learning process for our model on the Yale dataset.

First of all, ACC and NMI of different feature selection algorithms are given. As Table 2 presents, UFS-TAE obtains the higher clustering NMI than other algorithms, especially on the dataset WarpAR10P and WarpPIE10P. UFS-TAE also occupies a dominant position in the ACC metric except for Yale, but it is still the second-best. It is noted that we only select the number of features within 100 for the experiment.

Furthermore, in the practical experiment, deep learning models have excellent time consumption behavior because of the advantage of neural networks. And our model converges in 200 iterations within an acceptable loss on each dataset effectively, as Fig. 2 depicts. In fact, our model is close to converging at 75 iterations. The convergence indicates that the result we obtain is a stable result when the overall framework reaches the minimum error.

Figs. 3 and 4 show the learning process of our model on the Yale dataset. Fig. 3 displays the reconstruction results and Fig. 4 displays the feature selection results of the corresponding epoch. From the former we can see that the result reconstructed by our model is getting closer and closer to the original picture. This also means a drop in the loss. Seen from the latter figure, the result of feature selection is gradually concentrated in the vicinity of the eyes, nose, mouth, and facial contours. Here we utilize the symbol $*$ in yellow to denote the selected features.

We also display the feature selection visualizations of each compared method on the Yale dataset in Fig. 5. It can be seen that the selected features of LS are concentrating on the cheeks and forehead of the human face. As for UDFS, it only selects some salient features, such as the nose and mouth, while most of selected results in the eye area are located below. The visualization of NDFS looks like there is more useful information selected than the previous two methods. But several useless features are also selected. Overall, the results are scattered in this photo. Next, most of the features selected by RUFS are close to the contour of the cheeks. It can even be seen that the mouth features are basically not selected. With regard to URAFS, It seems to pay more attention to the feature selection of the contour part, but it also selects obvious areas such as eyes, nose, and mouth. MCFS achieved the highest ACC value on the Yale dataset in the previous experimental results. In this photo, the distribution of features selected by MCFS is very uniform, and the highest accuracy also shows that most of the important features have been selected.

From the figure of AEFS, it is concluded that using autoencoders for feature selection can obtain good results. Here it is inferred that the reason why AEFS is not as good as MCFS and ours is, the selected features are too concentrated, and the important features of the cheek area are not scattered. Most of the CAE results are only distributed on one side of the face as the whole. More than that, the selected features are more distributed on the cheeks and contours, and less on the eyes, nose, and mouth. The features selected by the proposed model include various regions, but the number of features distributed on the cheeks is relatively small. In addition, the areas with more dots have a relatively scattered distribution comparing with AEFS.

4.4. Parameter sensitivity

Different datasets can be seen as various learning tasks. So we simply replace the loss function on different datasets to achieve the goal of solving different learning tasks. We also consider different combinations of various loss functions and activation functions would have different effects on different datasets. Therefore, we perform an internal experiment to prove the usefulness of our model and the impact of different activation functions on a certain dataset.

In this section we utilize experiment results to prove that different loss functions and activation functions leave different impacts on different datasets. Except for that, the influence of other parameters on model performance is also displayed. We examine the clustering performance of the proposed method as the loss functions and activation functions vary. ACC metric is employed to evaluate the performance of clustering with different loss functions and activation functions when keeping the regularization parameter μ, and the orthogonality parameter λ.

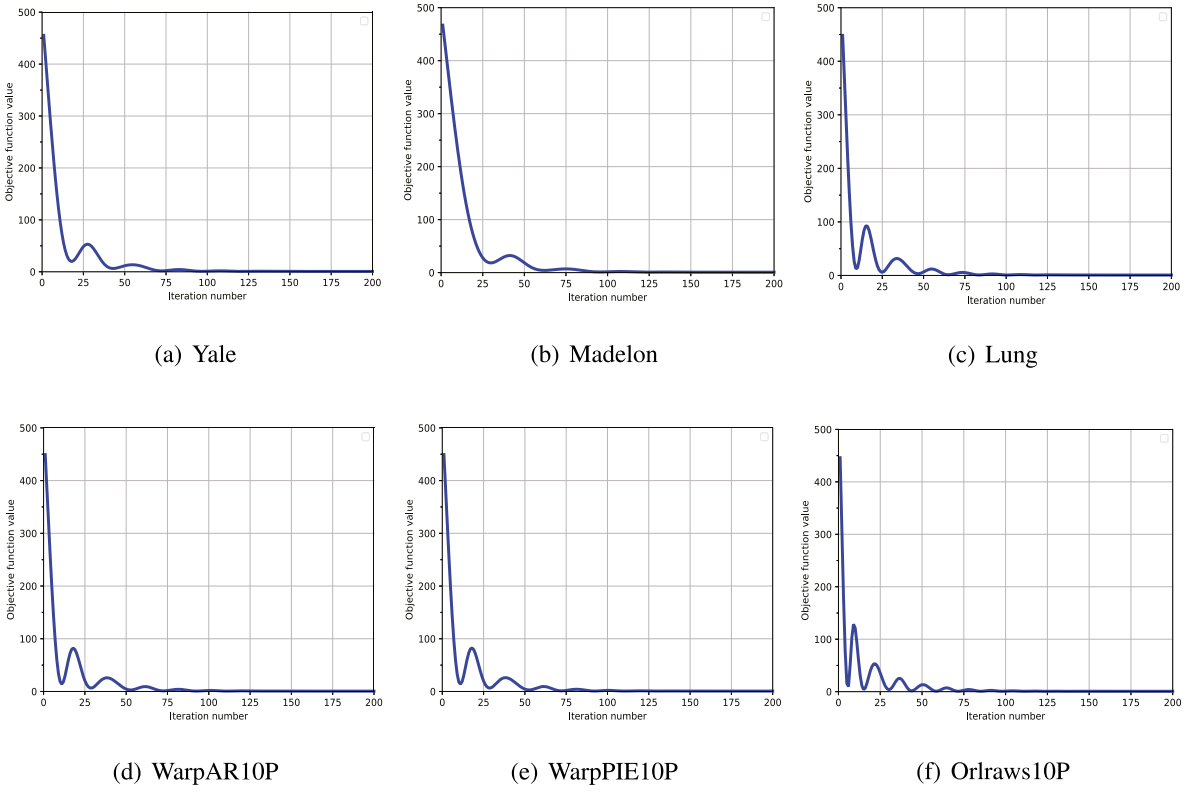
First, we present the performance of five different activation functions using four different loss functions, which can also be seen as handling various issues, in Tables 3–6. Here we provide a new prospect for the deep feature selection model. This type of model can easily be extended to other problems without major modifications. And for data with different structures, it can still approximate the initial data structure by using appropriate activation functions. In this part, we only set fixed parameters, such as learning rate $\alpha = 0.01$, without deliberately adjusting performance. So we can compare the performance of these loss functions and different activation functions on the same dataset as fairly as possible.

From Table 3, most of the results have exceeded the baseline in Table 2 except for the Lung dataset and Yale dataset. Leaky ReLU and Tanh have better behavior on the ACC metric with L1 loss. But measured by the NMI metric, ReLU has an absolute advantage. On the contrary, both Sigmoid and Mish are not good enough with the L1 loss function comparing with other activation functions.

As shown in Table 4, different activation functions with MSE loss function have more advantages on different datasets than L1 loss. For example, from the perspective of ACC, the model with ReLU has the best results on Lung and WarpAR10P, the model with Tanh has the best results on WarpPIE10P and Madelon. As far as our model is concerned, performances of the Lung dataset and Yale dataset are relatively inferior. However, the combination of ReLU and MSE loss obtains the best result in whole internal

Table 2
ACC (%) and NMI (%) of different feature selection algorithms. (Bold and underlined results are the best and the second best.)

	Dataset	Yale	Lung	WarpPIE10P	Orlraws10P	WarpAR10P	Madelon
ACC (%)	All Features	29.45	<u>91.08</u>	27.76	<u>84.80</u>	27.76	50.32
	LS	38.12	59.59	30.57	67.20	21.77	50.34
	UDFS	33.03	59.70	27.71	38.60	22.85	50.20
	NDFS	34.18	65.42	24.57	70.10	19.08	51.15
	RUFS	32.24	84.04	26.67	51.90	25.92	50.23
	URAFS	31.82	70.30	24.57	57.00	27.46	50.19
	MCFS	**39.09**	81.82	37.33	71.00	22.00	50.33
	AEFS	33.34	41.72	<u>41.81</u>	44.35	27.42	<u>57.34</u>
	CAE	25.70	89.46	30.95	63.10	<u>28.62</u>	51.42
	Ours	<u>38.15</u>	**92.71**	**53.86**	**89.80**	**43.69**	**61.96**
NMI (%)	All Features	39.95	<u>73.76</u>	24.37	<u>88.57</u>	24.37	3.10×10^{-3}
	LS	40.55	57.38	32.01	73.97	19.41	3.37×10^{-3}
	UDFS	38.34	41.74	27.47	39.08	17.85	1.70×10^{-3}
	NDFS	40.52	49.34	21.14	78.06	13.83	4.00×10^{-2}
	RUFS	38.40	58.47	26.37	61.47	21.99	1.54×10^{-3}
	URAFS	40.13	52.22	17.52	63.79	<u>25.27</u>	1.00×10^{-2}
	MCFS	<u>46.47</u>	59.44	37.31	78.38	19.46	3.26×10^{-3}
	AEFS	39.52	14.01	<u>44.55</u>	47.72	25.01	<u>1.56</u>
	CAE	34.95	69.15	21.32	82.97	22.19	5.78×10^{-2}
	Ours	**46.58**	**78.91**	**53.32**	**96.22**	**42.82**	**4.17**



(a) Yale (b) Madelon (c) Lung

(d) WarpAR10P (e) WarpPIE10P (f) Orlraws10P

Fig. 2. Convergence curves of the objective function value. The proposed method frequently converges within 200 iterations on each dataset.

experiment. This also confirms what we mentioned earlier that choosing the suitable activation function and loss function for the exact dataset is very important for performance. From the perspective of NMI, Tanh and ReLU perform better than other activation functions together with MSE loss.

Table 5 shows that Leaky ReLU and Tanh with KLDiv loss gain the best results in both the ACC and NMI metrics. However, even if the combination of Tanh activation function with KLDiv loss function obtains the best result on Lung dataset, it still does not exceed the baseline 91.08%. Results of Yale dataset have a similar circumstance: Leaky ReLU achieves the best results but does not exceed the baseline 29.45%. In general, the performance of KLDiv loss is not good enough overall.

Seen from Table 6, the Leaky ReLU and Mish activation functions are more helpful for BCEWithLogits loss. By contrast, some activation functions are not good for this loss function, such as Sigmoid and Tanh. Although Leaky ReLU obtains better results than others, Mish behaves better on Yale and Lung datasets. Except for that, the Mish with BCEWithlogits loss also obtains acceptable performance on other datasets. All of them exceed the baseline and outperform the most compared algorithms. This is the reason why we choose this combination to compare with other methods.

The experimental results of Tables 3–6 are used to verify the effectiveness and potential of our model. It is evidently observed that different loss functions and activation functions may leave

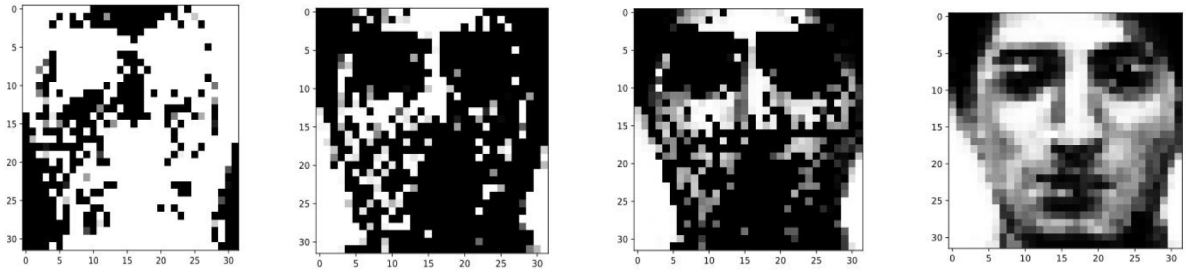


Fig. 3. Reconstructed results of learning process for the proposed method.

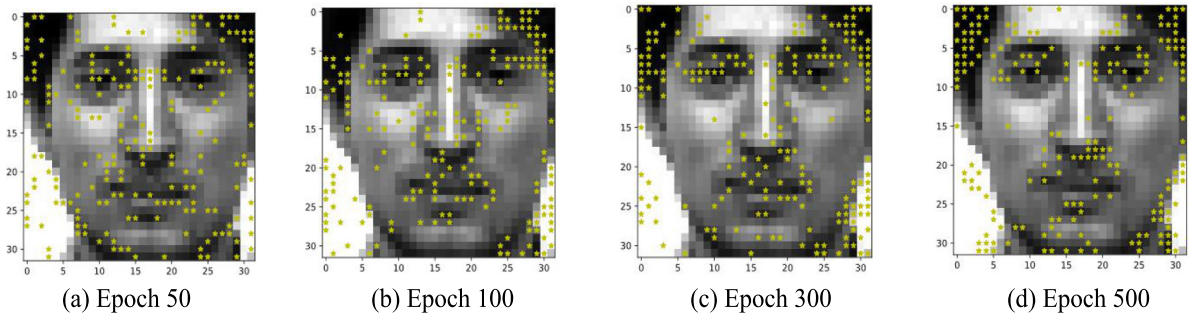


Fig. 4. Feature selection results of learning process for the proposed method.

Table 3
ACC (%) and NMI (%) of Different Activation Functions with L1 Loss. (Bold and underlined results are the best and the second best.)

	Dataset	Sigmoid	ReLU	Leaky ReLU	Tanh	Mish
ACC (%)	Yale	28.79	34.18	**35.39**	33.21	34.91
	Lung	83.60	88.28	**91.18**	89.66	88.97
	WarpPIE10P	41.62	38.76	37.29	**45.33**	39.81
	Orlraws10P	72.80	85.50	85.30	**85.90**	84.20
	WarpAR10P	40.46	41.46	42.08	**44.07**	40.08
	Madelon	60.72	57.31	**60.75**	58.62	59.04
NMI (%)	Yale	35.58	**44.44**	42.80	43.37	43.38
	Lung	60.30	69.55	**72.19**	71.60	68.39
	WarpPIE10P	44.78	37.67	36.79	**46.99**	37.29
	Orlraws10P	77.98	**91.41**	89.94	90.23	90.02
	WarpAR10P	38.56	**45.56**	42.87	43.17	41.00
	Madelon	3.34	**3.69**	3.36	2.44	2.38

Table 4
ACC (%) and NMI (%) of different activation functions with MSE Loss. (Bold and underlined results are the best and the second best.)

	Dataset	Sigmoid	ReLU	Leaky ReLU	Tanh	Mish
ACC (%)	Yale	29.21	35.39	34.00	32.55	**36.79**
	Lung	87.39	**92.02**	91.13	90.79	89.66
	WarpPIE10P	42.90	44.57	40.43	**49.10**	38.71
	Orlraws10P	73.70	84.50	**88.00**	84.50	86.40
	WarpAR10P	40.46	**43.23**	40.77	38.69	40.00
	Madelon	60.73	60.73	58.80	**61.77**	60.73
NMI (%)	Yale	36.91	43.00	42.34	40.98	**43.93**
	Lung	67.77	73.70	70.70	**70.95**	70.91
	WarpPIE10P	44.57	46.12	40.22	**54.56**	38.49
	Orlraws10P	77.90	**92.47**	89.53	88.18	89.56
	WarpAR10P	38.55	**43.74**	40.77	38.24	36.92
	Madelon	3.35	3.35	2.25	**4.03**	3.35

Table 5
ACC (%) and NMI (%) of different activation functions with KLDiv Loss. (Bold and underlined results are the best and the second best.)

	Dataset	Sigmoid	ReLU	Leaky ReLU	Tanh	Mish
ACC (%)	Yale	30.79	34.73	**36.30**	31.88	35.39
	Lung	85.42	83.50	84.68	**88.57**	83.99
	WarpPIE10P	43.52	36.76	37.76	**46.76**	38.38
	Orlraws10P	74.60	83.70	83.50	78.20	**84.80**
	WarpAR10P	39.00	**40.62**	38.69	37.08	32.54
	Madelon	61.40	60.69	**61.73**	61.24	58.59
NMI (%)	Yale	36.17	43.47	**43.51**	42.21	41.14
	Lung	60.37	57.85	64.71	**66.69**	60.92
	WarpPIE10P	45.35	35.65	36.49	**52.74**	34.46
	Orlraws10P	79.31	86.77	86.66	83.18	**88.91**
	WarpAR10P	35.86	**39.68**	37.56	34.68	32.58
	Madelon	3.78	3.33	**4.01**	3.68	2.15

Table 6
ACC (%) and NMI (%) of Different Activation Functions with BCEWithLogits Loss. (Bold results are best, and underlined results are the second best.)

	Dataset	Sigmoid	ReLU	Leaky ReLU	Tanh	Mish
ACC (%)	Yale	28.79	34.91	35.82	32.85	**38.15**
	Lung	84.14	90.69	91.43	91.63	**91.72**
	WarpPIE10P	43.43	47.95	**48.47**	45.29	43.86
	Orlraws10P	72.40	**87.80**	87.20	85.40	87.50
	WarpAR10P	40.46	39.69	**40.77**	39.85	38.54
	Madelon	60.81	59.92	**60.87**	60.69	59.53
NMI (%)	Yale	36.36	45.12	43.79	41.53	**46.58**
	Lung	61.15	73.52	72.75	74.86	**75.35**
	WarpPIE10P	45.73	**49.19**	52.26	47.61	45.91
	Orlraws10P	79.10	**93.45**	92.76	89.46	91.38
	WarpAR10P	41.62	40.34	**45.49**	38.81	37.20
	Madelon	3.40	2.86	**3.44**	3.32	2.64

different effects on different datasets. The closer the non-linear structure of an activation function fits the original data structure, the loss function is more suitable for the data characteristics, so that the better this model would perform. For loss function only, on the Yale dataset, the BCEWithLogits loss acquires a better result than the other three loss functions. With regard to KLDiv loss, it occupies a dominant position on the Lung and WarpAR10P datasets. For Madelon, the results with L1 loss have three ACC values less than 60%, which means that L1 loss behaves worse than others on this dataset. In addition, the results of four loss

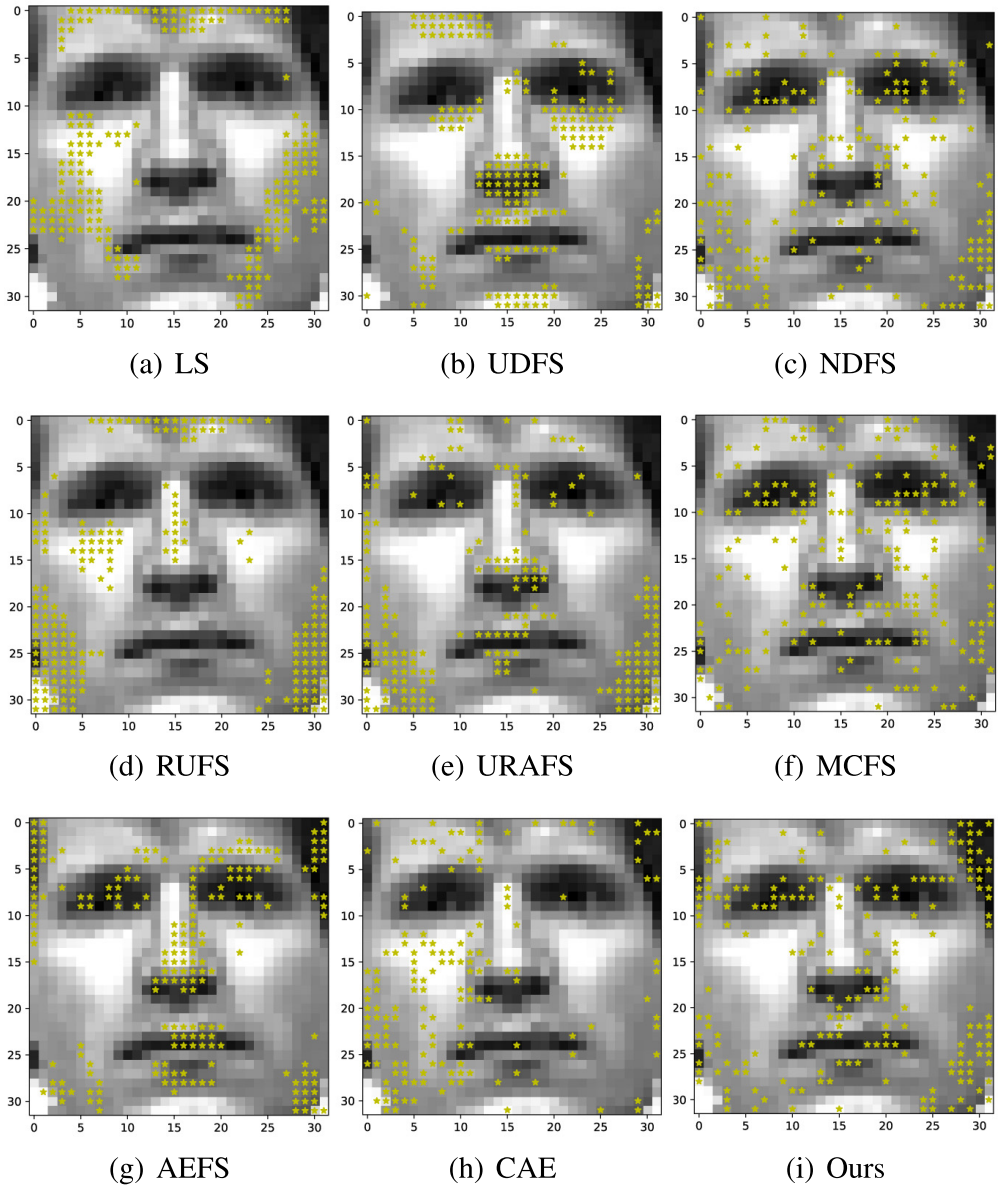


Fig. 5. Visualizations of all compared algorithms and the proposed method on the Yale dataset. The number of selected features k of each compared method is set to 200.

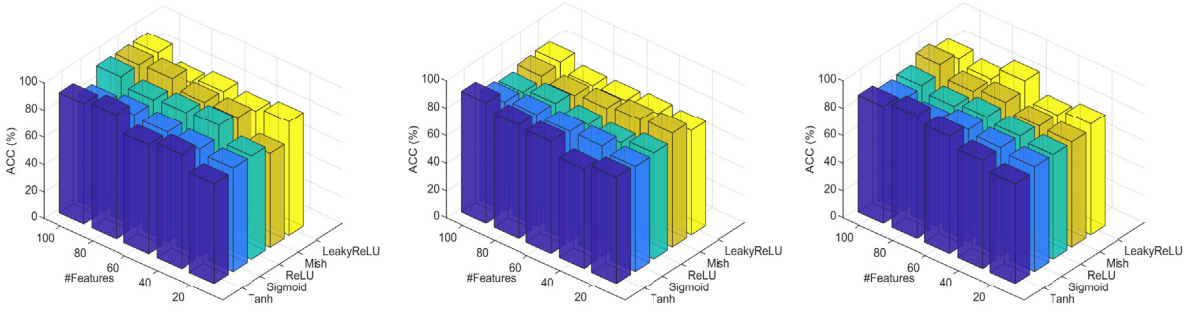
functions are very close on the Orlraws10P dataset. These results also prove that the proposed model has a strong generalization ability, as mentioned in Section 1. More than that, flexible activation functions can help the proposed model approximate different data structures better and impose the same effect as some regularization terms. This advantage can make the model easier to optimize, especially for deep learning models, without considering whether the regularization term is differentiable or not.

Then we select two datasets to display the corresponding histograms of all parameters, including regularization parameter μ, orthogonality parameter λ, different loss functions and flexible activation functions. In order to avoid data overflowing, we set $\lambda \in \{10^{-3}, 10^{-2}, \ldots, 10^{2}\}$, $\mu \in \{10^{-3}, 10^{-2}, \ldots, 10^{2}\}$. The number of selected features varies in $\{20, 40, \ldots, 100\}$. And we choose four loss functions (BCEWithLogits loss, KLDiv loss, L1 loss, MSE loss) and five activation functions (Sigmoid, ReLU, Leaky ReLU, Tanh, Mish). Since the performance is more excellent on different datasets than other parameter settings when $\lambda = 10$ and $\mu = 0.01$, we set them in the experiment of different loss functions.

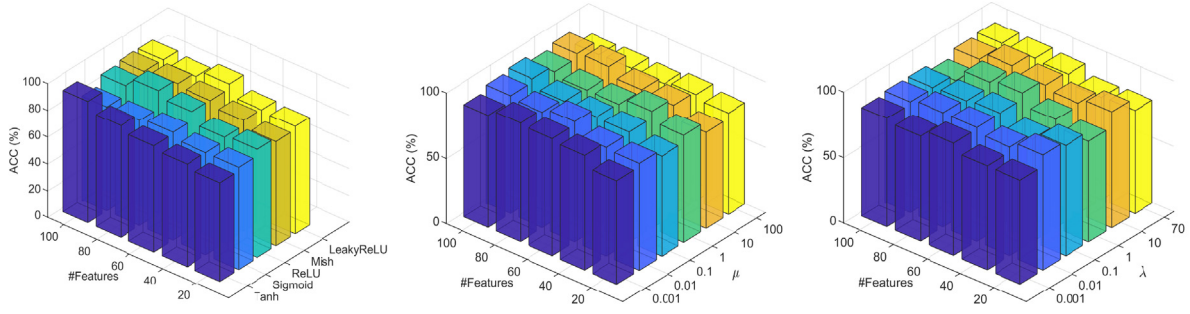
Hereinafter, two figures are shown to indicate the parameter sensitivity on Lung and Madelon datasets. The x-axis represents different activation functions, μ and λ in (a)–(f), respectively, while y-axis represents the number of selected features, and z-axis represents the clustering accuracy.

Firstly, as shown in Fig. 6, several activation functions have little differences in the performance of Lung dataset except for ReLU. The dataset Lung has a curvilinear distribution in the low dimensional projection space, which corresponds to the structure of a one-dimensional flow pattern in the high dimensional space. According to this figure, the capability of BCEWithLogits loss and MSE loss obviously performs better than other loss functions, especially when the number of selected features is small.

Madelon is a highly non-linear dataset. As shown in Fig. 7, it is evident that the combination of the Mish activation function and MSE loss function is more outstanding than other combinations. Actually, the combination of the ReLU activation function and KLDiv loss function can also achieve this effect.

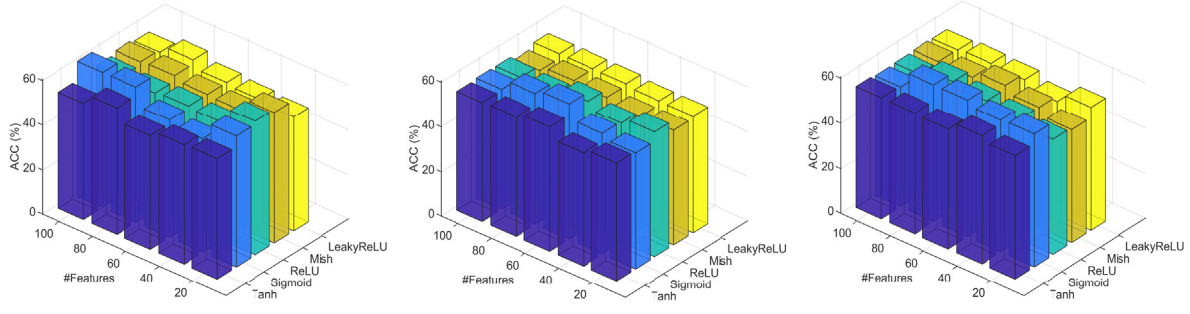


(a) BCEWithLogits loss. (b) KLDiv loss. (c) L1 loss.

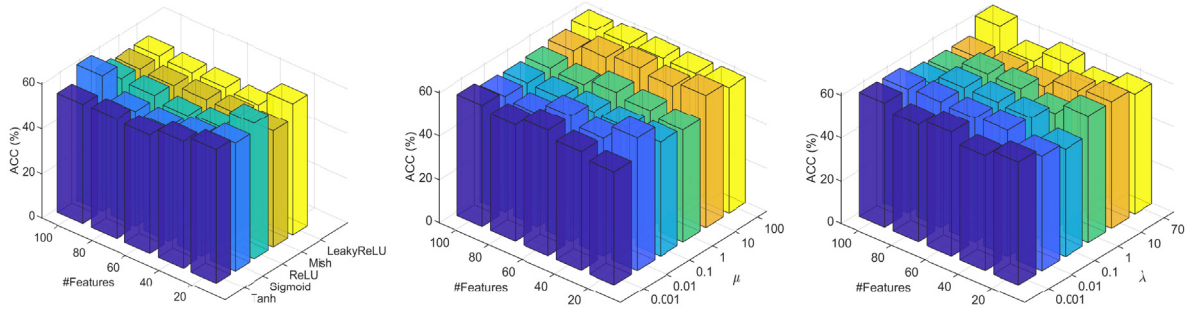


(d) MSE loss. (e) $\lambda = 10$. (f) $\mu = 0.01$.

Fig. 6. Parameter sensitivity demonstration on Lung dataset. (a)–(d) Performance of different loss functions with $\mu = 0.01$ and $\lambda = 10$. (e)–(f) Performance of the fixed loss function and activation function with different μ and λ, where BCEWithLogits loss and Mish activation function are available.



(a) BCEWithLogits loss. (b) KLDiv loss. (c) L1 loss.



(d) MSE loss. (e) $\lambda = 10$. (f) $\mu = 0.01$.

Fig. 7. Parameter sensitivity demonstration on Madelon dataset. (a)–(d) Performance of different loss functions with $\mu = 0.01$ and $\lambda = 10$. (e)–(f) Performance of the fixed loss function and activation function with different μ and λ, where BCEWithLogits loss and Mish activation function are available.

These two figures more intuitively demonstrate that the combination of different activation functions and different loss functions is crucial and has a significant performance increase. Besides, the performance fluctuation of our model affected by parameter λ and μ is shown in (e) and (f) of Figs. 6 and 7. It can be concluded that both parameters leave big effects on the experimental performance of the model. Note that we set the upper bound of μ to 70 because the element of NAN will appear when μ exceeds this value in the practical experiment.

5. Conclusions

This paper attempts to solve the generalized feature selection problem by seeking the non-negative and orthogonal indicator matrix with auto-encoder network. The auto-encoder with several non-linear activation functions allows our model to select both the features with the linear relationship and the potential non-linear relationship features. $\ell_{2,1}$-norm is also used to select features with joint sparsity. The experimental result indicates the effectiveness of our method. The main advantage of our proposed model is that different learning tasks can be handled through different loss functions, which improves the generalization ability of the model. In addition, the flexible activation functions also have the ability to play the role of regularization terms to impose regularization constraints on the model without directly adding them. Hence, it is concluded that the proposed method is promising and encouraging. In the future, we intend to add discrete differentiable layers to the auto-encoder and continue to optimize non-negativity and orthogonality constraint solutions to obtain a more convenient and perfect deep unsupervised feature selection model.

CRediT authorship contribution statement

Yunhe Zhang: Conceptualization, Methodology, Software, Validation, Data curation, Investigation, Visualization, Writing - original draft. **Zhoumin Lu:** Formal analysis, Supervision, Writing - review & editing. **Shiping Wang:** Conceptualization, Resources, Formal analysis, Supervision, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is partly supported by the National Natural Science Foundation of China under Grant Nos. 61502104 and 61672159, the Fujian Collaborative Innovation Center for Big Data Application in Governments, and the Technology Innovation Platform Project of Fujian Province under Grant No. 2014H2005.

References

[1] B. Su, X. Ding, H. Wang, Y. Wu, Discriminative dimensionality reduction for multi-dimensional sequences, IEEE Trans. Pattern Anal. Mach. Intell. 40 (1) (2018) 77–91.
[2] C. Li, X. Wang, W. Dong, J. Yan, Q. Liu, H. Zha, Joint active learning with feature selection via CUR matrix decomposition, IEEE Trans. Pattern Anal. Mach. Intell. 41 (6) (2019) 1382–1396.
[3] S. Wang, W. Pedrycz, Q. Zhu, W. Zhu, Unsupervised feature selection via maximum projection and minimum redundancy, Knowl.-Based Syst. 75 (2015) 19–29.
[4] K. Yu, L. Liu, J. Li, W. Ding, T.D. Le, Multi-source causal feature selection, IEEE Trans. Pattern Anal. Mach. Intell. (2019) 1.
[5] R. Zhang, F. Nie, X. Li, X. Wei, Feature selection with multi-view data: A survey, Inf. Fusion 50 (2019) 158–167.
[6] H. Zhao, P. Zhu, P. Wang, Q. Hu, Hierarchical feature selection with recursive regularization, in: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, 2017, pp. 3483–3489.
[7] P. Zhu, Q. Xu, Q. Hu, C. Zhang, H. Zhao, Multi-label feature selection with missing labels, Pattern Recognit. 74 (2018) 488–502.
[8] B. Perozzi, R. Al-Rfou, S. Skiena, Max-Margin DeepWalk: Discriminative learning of network representation, in: Proceedings of the Twentieth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014, pp. 701–710.
[9] Z. Wu, Y. Xiong, S.X. Yu, D. Lin, Unsupervised feature learning via non-parametric instance discrimination, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 3733–3742.
[10] C. Yang, Z. Liu, D. Zhao, M. Sun, E.Y. Chang, Network representation learning with rich text information, in: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, 2015, pp. 2111–2117.
[11] S. Wang, W. Guo, Sparse multigraph embedding for multimodal feature representation, IEEE Trans. Multimed. 19 (7) (2017) 1454–1466.
[12] J. Li, R. Guo, C. Liu, H. Liu, Adaptive unsupervised feature selection on attributed networks, in: Proceedings of the Twenty-Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2019, pp. 92–100.
[13] T.Q. Tran, J. Sakuma, Seasonal-adjustment based feature selection method for predicting epidemic with large-scale search engine logs, in: Proceedings of the Twenty-Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2019, pp. 2857–2866.
[14] Z. Zhao, H. Liu, Spectral feature selection for supervised and unsupervised learning, in: Proceedings of the Twenty-Fourth International Conference on Machine Learning, 2007, pp. 1151–1157.
[15] J. Jordon, J. Yoon, M. van der Schaar, KnockoffGAN: Generating knockoffs for feature selection using generative adversarial networks, in: Proceedings of the Seventh International Conference on Learning Representations, 2019, pp. 6–9.
[16] F. Nie, H. Huang, X. Cai, C.H. Ding, Efficient and robust feature selection via joint $\ell_{2,1}$-norms minimization, in: Proceedings of the Advances in Neural Information Processing Systems, 2010, pp. 1813–1821.
[17] J.G. Dy, C.E. Brodley, Feature selection for unsupervised learning, J. Mach. Learn. Res. 5 (2004) 845–889.
[18] S. Wang, H. Wang, Unsupervised feature selection via low-rank approximation and structure learning, Knowl.-Based Syst. 124 (2017) 70–79.
[19] L. Zhang, M. Liu, R. Wang, T. Du, J. Li, Multi-view unsupervised feature selection with dynamic sample space structure, in: Proceedings of the IEEE Symposium Series on Computational Intelligence, 2019, pp. 2641–2648.
[20] X. Chen, J.C. Jeong, Enhanced recursive feature elimination, in: Proceedings of the International Conference on Machine Learning and Applications, 2008, pp. 429–435.
[21] I. Guyon, J. Weston, S. Barnhill, V. Vapnik, Gene selection for cancer classification using support vector machines, Mach. Learn. 46 (2002) 389–422.
[22] S. Liao, Q. Gao, F. Nie, Y. Liu, X. Zhang, Worst-case discriminative feature selection, in: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, 2019, pp. 2973–2979.
[23] F. Pisheh, R. Vilalta, Filter-based information-theoretic feature selection, in: Proceedings of the Third International Conference on Advances in Artificial Intelligence, 2019, pp. 207–211.
[24] S.T. Roweis, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (5500) (2000) 2323–2326.
[25] Y. Wan, S. Sun, C. Zeng, Adaptive similarity embedding for unsupervised multi-view feature selection, IEEE Trans. Knowl. Data Eng. pp (2020) 1.
[26] F. Yan, X. Wang, Z. Zeng, C. Hong, Adaptive multi-view subspace clustering for high-dimensional data, Pattern Recognit. Lett. 130 (2020) 299–305.
[27] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, C. Xu, GhostNet: More features from cheap operations, in: Proceedings of the IEEE Internaltional Conference on Computer Vision and Pattern Recogintion, 2020.
[28] Y. LeCun, Y. Bengio, G. Hinton, Deep learn., Nature 521 (7553) (2015) 436–444.
[29] G.E. Hinton, Reducing the dimensionality of data with neural networks, Science 313 (5786) (2006) 504–507.
[30] X. He, D. Cai, P. Niyogi, Laplacian score for feature selection, in: Proceedings of the Advances in Neural Information Processing Systems, 2006, pp. 507–514.
[31] D. Cai, C. Zhang, X. He, Unsupervised feature selection for multi-cluster data, in: Proceedings of the Sixteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2010, pp. 333–342.
[32] Y. Yang, H.T. Shen, Z. Ma, Z. Huang, X. Zhou, $\ell_{2,1}$-norm regularized discriminative feature selection for unsupervised, in: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, 2011.

[33] Z. Li, Y. Yang, J. Liu, X. Zhou, H. Lu, Unsupervised feature selection using nonnegative spectral analysis, in: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.
[34] M. Qian, C. Zhai, Robust unsupervised feature selection, in: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, 2013.
[35] S. Wang, W. Zhu, Sparse graph embedding unsupervised feature selection, IEEE Trans. Syst. Man Cybern.: Syst. 48 (3) (2016) 329–341.
[36] X. Li, H. Zhang, R. Zhang, Y. Liu, F. Nie, Generalized uncorrelated regression with adaptive graph for unsupervised feature selection, IEEE Trans. Neural Netw. Learn. Syst. 30 (5) (2018) 1587–1595.
[37] B. Chandra, R.K. Sharma, Exploring autoencoders for unsupervised feature selection, in: Proceedings of the International Joint Conference on Neural Networks, 2015, pp. 1–6.
[38] N. Gui, D. Ge, Z. Hu, AFS: An attention-based mechanism for supervised feature selection, in: Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, 2019, pp. 3705–3713.
[39] Y. Lu, Y. Fan, J. Lv, W.S. Noble, Deeppink: Reproducible feature selection in deep neural networks, in: Proceedings of the Advances in Neural Information Processing Systems, 2018, pp. 8676–8686.
[40] K. Han, Y. Wang, C. Zhang, C. Li, C. Xu, Autoencoder inspired unsupervised feature selection, in: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, 2018, pp. 2941–2945.
[41] M.F. Balin, A. Abid, J.Y. Zou, Concrete autoencoders: Differentiable feature selection and reconstruction, in: Proceedings of the Thirty-Sixth International Conference on Machine Learning, 2019, pp. 444–453.