Second-Order Unsupervised Feature Selection via Knowledge Contrastive Distillation

Han Yue, Jundong Li, *Member, IEEE*, and Hongfu Liu, *Member, IEEE*

***Abstract*—Unsupervised feature selection aims to select a subset from the original features that are most useful for the downstream tasks without external guidance information. While most unsupervised feature selection methods focus on ranking features based on the intrinsic properties of data, most of them do not pay much attention to the relationships between features, which often leads to redundancy among the selected features. In this paper, we propose a two-stage Second-Order unsupervised Feature selection via knowledge contrastive disTillation (SOFT) model that incorporates the second-order covariance matrix with the first-order data matrix for unsupervised feature selection. In the first stage, we learn a sparse attention matrix that can represent second-order relations between features by contrastively distilling the intrinsic structure. In the second stage, we build a relational graph based on the learned attention matrix and perform graph segmentation. To this end, we conduct feature selection by only selecting one feature from each cluster to decrease the feature redundancy. Experimental results on 12 public datasets show that SOFT outperforms classical and recent state-of-the-art methods, which demonstrates the effectiveness of our proposed method. Moreover, we also provide rich in-depth experiments to further explore several key factors of SOFT.**

***Index Terms*—Neural networks, second order, unsupervised feature selection.**

I. Introduction

In THE digital world, huge amounts of high-dimensional data [1], [2], [3], [4], [5], [6] are captured every day. Due to the existence of irrelevant or redundant features, data in high dimensions may significantly increase computational costs and bring challenges for efficient and effective data management. Dimensionality reduction is one of the most well-known techniques to address the above issue, which can be categorized into feature transformation and feature selection. Feature transformation, also known as representation learning, aims to project the original high-dimensional features into a low-dimensional feature space. The new feature space is usually a nonlinear combination of the original features. Although achieving promising performance, it is hard to be interpreted. On the other hand, feature selection methods select a subset of relevant features from all original features based on a predefined criterion to serve the downstream tasks, maintaining physical meanings of the original features for acceleration and interpretability. With the dramatically increasing computational power, acceleration is no more the focus of feature selection, while interpretability becomes the major benefit of feature selection.

Without label information, unsupervised feature selection methods aim to select a feature subset that can preserve the intrinsic structure of the whole feature set accurately. There are many algorithms designed to solve the unsupervised feature selection problem. ReliefF [7], HSIC [8], Laplacian Score [9], SPEC [10], SPFS [11] evaluate features by their capability in preserving the pairwise sample similarity. UDFS [12], FSASL [13], and TSFS [14] employ pseudo labels as the supervision to guide the feature selection along with a sparse constraint. Most of these methods apply the linear feature selection matrices and select the representative features by ranking their feature weight vector. Such operations treat the feature set independently and fail to tackle the complex high-order relationship [15], [16] among original features, which inevitably brings in redundancy among selected features.

Contributions: We propose a two-stage Second-Order unsupervised Feature selection via knowledge contrastive disTillation (SOFT) model that incorporates the second-order covariance matrix with the first-order data. In the first stage, SOFT learns a sparse attention matrix to explore the second-order feature relationships. In the second stage, we perform graph segmentation on the learned attention matrix for feature selection. In summary, we highlight our contributions as follows: (1) We consider the second-order feature relationship in the unsupervised feature selection problem and propose the SOFT algorithm to distill knowledge from both original features and their second-order feature covariance matrix; (2) Our SOFT learns a mask matrix on or off the covariance matrix and obtains the attention matrix and masked matrix. Throughout distilling the structural knowledge, the sparse attention matrix contains such knowledge as much as possible while excluding that from the masked matrix as well; (3) Different from selecting features according to weights, we propose a graph segmentation-based feature selection method on the attention matrix, where only one representative feature is selected from each segment to avoid redundancy; and (4) Experimental results validate that SOFT outperforms classical and recent state-of-the-art models on 12 public datasets. We also provide in-depth explorations for both stages to demonstrate the effectiveness of SOFT.

Manuscript received 7 November 2022; revised 18 July 2023; accepted 27 August 2023. Date of publication 4 September 2023; date of current version 3 November 2023. The work of Jundong Li was supported by the National Science Foundation under Grants IIS-2006844, IIS-2144209, and IIS-2223769. Recommended for acceptance by X. Li. *(Corresponding author: Han Yue.)*

Han Yue and Hongfu Liu are with the Michtom School of Computer Science, Brandeis University, Waltham, MA 02453 USA (e-mail: hanyue@brandeis.edu; hongfuliu@brandeis.edu).

Jundong Li is with the Department of Electrical and Computer Engineering, Department of Computer Science, and School of Data Science, University of Virginia, Charlottesville, VA 22904 USA (e-mail: jundong@virginia.edu).

Digital Object Identifier 10.1109/TPAMI.2023.3311617

II. Related Work

We briefly review related work on unsupervised feature selection and deep feature selection below.

Unsupervised Feature Selection: In the past decades, a large amount of unlabeled data has been generated. To solve the feature selection problem for the unlabeled data, researchers have proposed many unsupervised feature selection methods, which can be divided into three main categories: Filter, Wrapper, and Hybrid. Filter methods evaluate features based on the data itself. [17] propose one of the earliest filter unsupervised feature selection methods, the Sequential backward selection method for Unsupervised Data (SUD). SUD introduces a similarity matrix representing the pair-wise similarity between objects. By measuring the entropy of the data, the relevance of each feature is quantified as a ranking score. Features with the highest scores are selected. Another example is Laplacian Score [9], which weights features according to their ability to preserve a predefined manifold structure represented by the Laplacian matrix. Similarly, SPECtrum decomposition (SPEC) [10] also introduces an object similarity matrix. SPEC measures the consistencies between features and nontrivial eigenvectors of the Laplacian matrix and ranks features based on the consistencies. Wrapper methods select the most relevant features by using a clustering algorithm. [18] introduce a method to select feature subsets using Expectation Maximization (EM) [19] clustering and evaluate them with maximum likelihood and the scatter separability criterion. [20] present an evolutionary multi-objective local selection algorithm to search feature subsets with K-means [21] and EM [19] clustering. Instead of selecting feature subsets, [22] propose to estimate a set of real-valued quantities carried out by an EM algorithm through adopting a minimum message length [23] penalty. Hybrid methods try to take advantage of both approaches by adopting a two-stage process: filter stage and wrapper stage. In the filter stage, the features are scored based on the intrinsic properties of the data. And in the wrapper stage, feature sets are generated by a specific clustering algorithm. For instance, [24] adopts the method of [17] for the filter stage to sort the features and the method of [18] for the wrapper stage to build clusters. [25] combine the spectral feature selection framework using the Laplacian Score [9] ranking and a modified Calinski–Harabasz index [26]. Different from the above-mentioned methods, which are based on ranking, [27] propose a method that starts with the wrapper stage by Least-Square-Estimation based evaluation and then selects feature set through a Bayesian network in the filter stage. InfFS [28] is a graph-based filtering approach, which evaluates the values of paths in a graph and selects discrete input features by exploiting properties of power series of matrices and the concept of absorbing Markov chains. GLLE [29] proposes to preserve the local linear reconstruction relationship among neighboring data points in the feature subspace. and adopts manifold regularization to find the relevant and representative features. MGF ^{2} WL [30] introduces a feature weight matrix to learn the weights of different feature dimensions directly, and uses a graph fusion term to fuse multiple predefined similarity graphs for learning a unified similarity graph.

Deep Feature Selection: Recently, deep learning techniques have gained much attention and brought in some studies on deep feature selection [31], [32]. DFS [33] adds a weight layer to Multi-layer Perceptron (MLP) together with a sparse regularization term so as to take advantage of deep structures to model nonlinearity. [34] propose to combine deep neural networks with sparse representation for grouped heterogeneous feature selection. The model first converts the multi-modal data into a unified representation, then selects features through solving a sparse group lasso [35] problem. In recent years, some studies have also involved data reconstruction error in deep unsupervised feature selection. AEFS [36] jointly learns a self-representation autoencoder model and importance weights of each feature for feature selection. Furthermore, GAFS [37] not only adopts a single-layer autoencoder but also incorporates spectral graph analysis for learning. UDSFS [38] selects the most discriminative features and meanwhile designates appropriate weights to feature dimensions by utilizing group sparsity of features. TSFS [14] presents a teacher-student scheme for deep feature selection, in which a teacher network is used to learn low-dimensional representations, and a student network is employed for feature selection by minimizing the reconstruction error. CAE [39] uses a concrete selector layer as encoder and a standard neural network as decoder, stochastically selecting discrete features by concrete random variables and reparametrization trick to get a subset of features. AARC [40] integrates unsupervised feature selection and determination of a compact network structure into a single framework, and applies a penalty based on dependency between features to control the level of redundancy in the selected features.

III. Methodology

Motivation: Unsupervised feature selection aims to select a small portion of the original features that are most useful for the downstream tasks without external guidance. Most previous methods focus on ranking features based on the values of individual features [9], [10], [12] while neglecting the high-order relationships between features. Unfortunately, this might lead to the redundancy of the selected features and further deteriorate downstream tasks. Fig. 1 provides an illustrative example of selection results on *Sonar* [41] and *Waveform* [42] by Laplacian Score (LapScore) [9]. The heatmap shows the initial relationships between features based on the covariance matrix of features. We remove the diagonal values and calculate the absolute values of covariance, which are normalized for visualization. In this example, we select the top four features on *Sonar* and *Waveform* by LapScore, which are highlighted with red lines in Fig. 1. Obviously, features a and b, features c and d on *Sonar* are from two groups within of high similarity, indicating that one feature might be denoted by the other. Similarly, On *Waveform*, features e, f, g, and h contain highly relevant information as well. However, features from the same feature group might lead to redundancy, which disobeys the purpose of feature selection. This drives us to explore the complex relationships derived from the second-order feature covariance matrix in the unsupervised feature selection problem. Motivated by the above situation, we

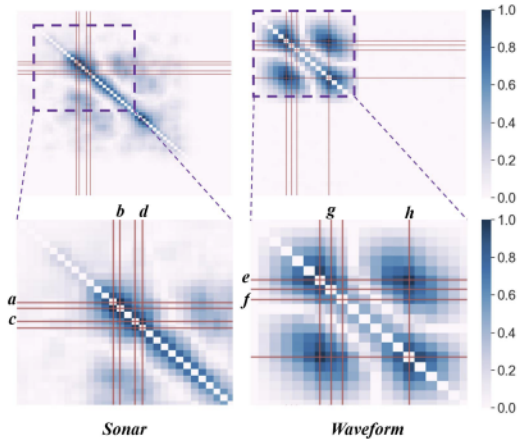


Fig. 1. Visualization of feature relations of *Sonar* and *Waveform* datasets by LapScore [9] with red lines.

design a method that can avoid such redundancy by considering both first-order data and second-order data.

Framework Overview: To incorporate the correlation feature information, we propose a two-stage model SOFT, which takes the first-order data matrix and second-order feature correlation matrix as inputs. In the first stage, SOFT learns a mask on the feature correlation matrix via knowledge contrastive distillation to preserve the data structure. In the second stage, we select the features on the masked correlation matrix via graph segmentation.

Fig. 2 shows the overview of our SOFT framework. With the first-order data matrix and second-order correlation feature matrix as inputs, we aim to learn a mask matrix applying to the feature correlation matrix for feature selection. By this means, we can obtain the attention matrix and masked matrix by the learnable mask on or off the feature matrix, respectively. The key idea of SOFT is to distill the structural knowledge by making the attention matrix contain such knowledge as much as possible while excluding that from the masked matrix as well. The green and red lines in Fig. 2 demonstrate the knowledge contrastive distilling process. To achieve this, we adopt a shared Graph Convolution Network (GCN) [43] to generate attention/original/masked representations from the attention/feature/masked matrices, respectively. Then we use pseudo labels generated by attention representation as positive guidance for original representation and negative guidance for masked representation so that attention representation and original representation are close to each other and far away from masked representation. Throughout the above knowledge contrastive distilling process, we can get a sparse and effective feature relation matrix that represents the second-order correlation, where each node denotes a feature and weights of edges are the corresponding values in the learned attention matrix. In the second stage, different from the existing work, which calculates the weight of each feature and suffers from the redundant feature issue (shown in Fig. 1), our attention matrix delivers weights for pairs of features. To proceed with feature selection, we use graph segmentation to cut the attention matrix into partitions and select one feature that has the highest relationship to others from each partition as the final selection result. By this means, the redundancy among selected features can be mitigated.

TABLE I
NOTATIONS AND DESCRIPTION

Notation	Dimension	Description
n	scalar	Number of input samples
d	scalar	Number of features
c	scalar	Number of clusters
X	$n \times d$	Input data matrix
M_F	$d \times d$	Input feature matrix
M_A	$d \times d$	Attention matrix
M_M	$d \times d$	Masked matrix
G_F	$n \times d$	Representations generated by X and M_F
G_A	$n \times d$	Representations generated by X and M_A
G_M	$n \times d$	Representations generated by X and M_M
P_F	$n \times c$	Predictions of samples by G_F
P_A	$n \times c$	Pseudo labels generated by G_A
P_M	$n \times c$	Predictions of samples by G_M

Attention Matrix Learning: Given n data instances with d features, we have the first-order $n \times d$ data matrix X and the $d \times d$ second-order feature matrix $M_F = X^\top \cdot X$. Our SOFT model calculates a learnable mask for feature selection. In the first stage, SOFT consists of four components, attention layer, shared GCN, pseudo label generation, and contrastive learning. We use $\Theta = \{\theta_M, \theta_G, \theta_C\}$ to denote the learnable parameters set in the SOFT model, where θ_M, θ_G, and θ_C denote the parameters of the learnable mask, shared GCN and contrastive learning, respectively. Note that the pseudo labels are generated from the attention matrix and shared GCN, which are controlled by θ_M and θ_G. Therefore, no learnable parameters are needed for pseudo label generation. Table I shows the notations used in our SOFT model and their descriptions. Each part is detailed as follows.

Attention Layer: The initialized feature matrix M_F may not be good enough to represent the relationship between features, so we add an attention layer to better capture second-order feature interactions by highlighting the important relations and reducing others through a learnable mask θ_M. The attention matrix M_A, which represents the important part of M_F, is calculated as follows:

$$M_A = M_F \odot \theta_M, \tag{1}$$

where $\odot$ is the element-wise product. Due to the symmetry of M_F, θ_M is forced to be symmetric by multiplying a parameter matrix with its transpose. With the learnable mask θ_M and the attention matrix M_A, We assume that the input feature matrix M_F can be decomposed as a summation formulation of attention matrix M_A and masked matrix M_M. Then we define the masked matrix M_M by:

$$M_M = M_F - M_A. \tag{2}$$

To make the learned attention matrix M_A sparse enough to identify crucial and principle relations, we apply $\ell_{2,1}$-norm to learnable mask θ_M on both row and column level:

$$\mathcal{L}_{2,1} = \|\theta_M\|_{2,1} + \|\theta_M^\top\|_{2,1}. \tag{3}$$

Shared Graph Convolutional Network: GCN [43] helps generate embeddings that contain both data information and feature

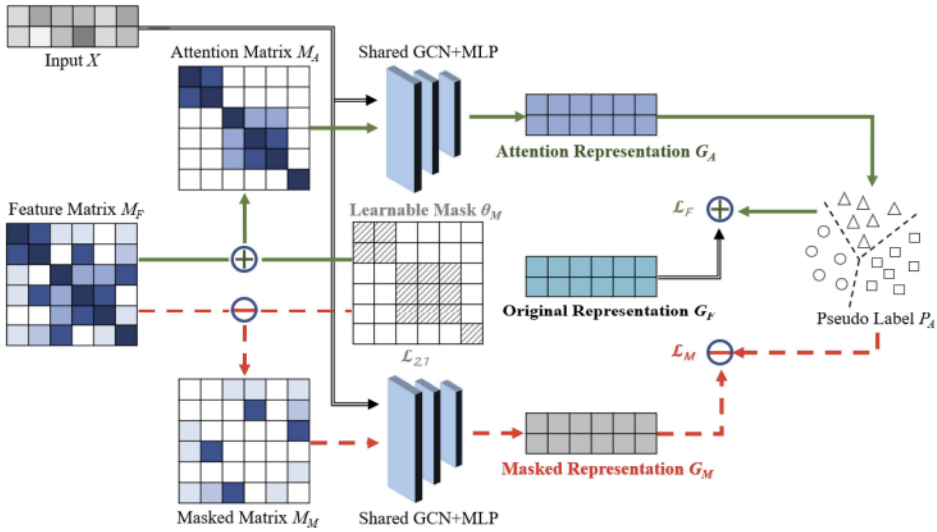


Fig. 2. SOFT model framework. The inputs of our SOFT model consist of the first-order data matrix and second-order correlation feature matrix. The green line shows that attention representation is expected to preserve the intrinsic information from the pseudo labels and get close to original representation, while the red line presents that the masked representation contains little structural information and gets away from the attention representation. Finally, we could get a sparse and effective feature relation matrix to represent the second-order correlation.

relationship information. We apply a shared 2-layer GCN to extract the attention/original/masked representations from the attention/feature/masked matrices, respectively. In [43], nodes of graphs denote samples, while in our case, nodes of graphs are features. Therefore, the computation of the GCN layer is a little different from [43] and is described by the following equation:

$$G_{\theta_G}^{(l+1)} = \text{ReLU}\left(G_{\theta_G}^{(l)}\tilde{D}^{-\frac{1}{2}}\tilde{M}\tilde{D}^{-\frac{1}{2}}\theta_G^{(l)}\right), \tag{4}$$

where M is the input relationship matrix, $\tilde{M} = M + I_d$, $I_d \in \mathbb{R}^{d\times d}$ is the identity matrix, $\tilde{D}$ is the degree matrix of $\tilde{M}$, and $\theta_G^{(l)}$ is a layer-specific trainable weight matrix. $G^{(l)}$ denotes the embeddings in the lth layer, and $G^{(0)} = X$. In our case, we use a shared 2-layer GCN to process samples with different input relationship matrices M_A, M_F, and M_M. Thus we have attention representation G_A, original representation G_F, and masked representation G_M, respectively.

Pseudo Label Generation: In unsupervised feature selection, pseudo labels are usually used as the criterion to guide feature selection. In our experiments, we use Deep Embedded Clustering (DEC) [44] to generate pseudo labels. The embedding processed by the clustering part is G_A, which is generated by attention matrix M_A, the matrix we expect to learn in the SOFT model. DEC helps learns feature representations, and assigns pseudo cluster labels P_A for the input samples X.

Contrastive Learning: With the above representations, we expect that attention representation and original representation are close to each other but far away from masked representation. To achieve such contrastive learning, we design two losses to measure the predictive abilities of different representations and compare them with pseudo labels. Specially, we employ a 2-layer Multi-Layer Perceptron (MLP) to get predictions of the samples with original representation G_F and masked representation G_M about which clusters they belong to,

$$P_{\theta_C}^{(l+1)} = \Phi\left(P_{\theta_C}^{(l)} \cdot \theta_C^{(l)}\right), \tag{5}$$

where $\theta_C^{(l)}$ is a layer-specific trainable weight matrix, $P^{(l)}$ denotes the embeddings in the lth layer, and $P^{(0)} = G_F$ or $P^{(0)} = G_M$ for the original representation and masked representation, respectively. We use $\Phi(\cdot) = \text{ReLU}(\cdot)$ as the activation function for layers before the last layer, and $\Phi(\cdot) = \text{Softmax}(\cdot)$

Algorithm 1: Second-Order Unsupervised Feature Selection Via Knowledge Contrastive Distillation.

Input: Input Data Matrix X; Feature Matrix $M_F = X^\top \cdot X$;
Output: k selected features;
1: Initialize $\Theta = \{\theta_M, \theta_G, \theta_C\}$;
2: **repeat**
3: Generate Attention Matrix M_A and Masked Matrix M_M by applying Learnable Mask on or off M_F;
4: Use GCN to process X with M_A, M_F and M_M to generate Attention/Original/Masked Representation G_A/G_F/G_M;
5: Adopt clustering on G_A to get pseudo labels P_A;
6: Employ MLP on Original/Masked Representation G_F/G_M to get predictions P_F/P_M;
7: Use P_A as positive guidance for P_F, and negative guidance for P_M to train the objective function;
8: **until** the model is converged.
9: Build a graph based on M_A via removing noisy features;
10: Apply graph segmentation on the graph to get k partitions and select one feature from each partition.

for the last layer. In our experiments, we adopt a shared 2-layer MLP and obtain P_F and P_M for original representation G_F and masked representation G_M.

Pseudo labels P_A generated by the clustering part are used for the following positive and negative training. By positive guidance for P_F, we get a cross-entropy loss $\mathcal{L}_F$ for P_F, which is defined as:

$$\mathcal{L}_F = -\sum_{i=1}^{n}\sum_{j=1}^{c}(P_{Aij}\log P_{Fij} + (1-P_{Aij})\log(1-P_{Fij})), \tag{6}$$

where P_{Fij} denotes the probability that the ith sample belongs to cluster j based on G_F. $P_{Aij} = 1$ if the ith sample belongs to cluster j based on the clustering result, otherwise $P_{Aij} = 0$. Eq. (7) is designed to make P_F similar to pseudo labels P_A. While by negative guidance for predictions of the masked part P_M, we apply attention loss described in [45] to reduce the weights of unimportant relations, which is stated as follows based on our notations:

$$\mathcal{L}_M = \sum_{i=1}^{n}\sum_{j=1}^{c} P_{Aij}P_{Mij}, \tag{7}$$

where P_{Mij} denotes the probability that the ith instance belongs to cluster j based on G_M. Through minimizing (7), the model tends to put instances to a cluster that they do not belong to, thus driving predictions P_M generated by masked representation different from pseudo labels P_A, which would finally lead to the difference of masked representation G_M and attention representation G_A.

Objective Function: Combining (3), (6), and (7), Our overall objective function of SOFT can be written as:

$$\min_{\Theta} \mathcal{L}_F + \alpha\mathcal{L}_M + \beta\mathcal{L}_{2,1}, \tag{8}$$

where α and β are hyperparameters for $\mathcal{L}_M$ and $\mathcal{L}_{2,1}$, respectively. We adopt Adam optimizer [46] to minimize the objective function.

Feature Selection on Attention Matrix: Different from traditional feature selection methods that return a weight vector to choose the top-ranked features, here, our attention matrix provides the weight of feature pairs. To proceed with the feature selection, we build one graph based on the attention matrix learned from the whole data. Nodes of the graph denote features, and edges are the relationship between features in the attention matrix. All nodes are linked to each other. Then we perform graph segmentation for feature selection.

Graph Construction: Instead of transforming the attention matrix M_A to a graph immediately, we add two additional processes so that the generated graph is more suitable for graph segmentation and feature selection. The first one is to set all values in M_A to their absolute values to make sure no negative edges would exist in the constructed graph. The second step is to build a scoring list. We calculate a score for each feature by $S = \sum_{i=1}^{d} M_{Ai}$, where S is the summation of each row or column of the attention matrix, which measures the importance of each feature.

Graph Segmentation: With the processed graph, we apply the graph segmentation method provided by [47] and cut the graph into k parts, where k is the number of features we aim to select. From each partition, we choose the feature that has the largest value in S since we consider features in the same partition are highly related.

Through learning attention matrix in the first stage of our method, we incorporate both first-order data and second-order data and get a refined feature relation matrix that can better reflect feature relations. Then in the second stage, we build a graph based on the learned attention matrix and perform graph segmentation, which groups high-correlated features together. Therefore, by selecting features from each partition, our method reduces redundancies among the selected features. Our proposed method focuses on learning pair-wise relationships of features and uses graph segmentation to select features. The time complexity of our method is $O(nd^2)$. If the number of samples is larger than the number of features ($n > d$), the space complexity of our method is $O(nd)$, otherwise $O(d^2)$. Algorithm 1 describes the whole process of SOFT.

TABLE II
STATISTICS OF DATASETS

Dataset	Type	#Sample	#Feature	#Class	Ratio	Density
COIL20	Face Image	1440	1024	20	1.41	0.656
Colon	Biological	62	2000	2	0.03	0.584
Gisette	Handwritten	13500	5000	2	2.70	0.130
L.Cancer	Biological	32	56	2	0.57	0.940
Madelon	Artificial	2600	500	2	5.20	1.000
Mov.Libras	Gesture	360	90	15	4.00	1.000
NCI9	Biological	60	9712	9	0.01	0.503
ORL	Face Image	400	1024	40	0.39	1.000
Sonar	Sonar Signal	208	60	2	3.47	0.999
UAV1	Traffic	19380	54	2	358.89	0.972
UAV2	Traffic	17256	54	2	319.56	0.983
Wave.	Artificial	5000	40	3	125.00	0.997

IV. Experimental Results

A. Experimental Settings

Datasets: We select 12 public feature selection benchmark datasets of different types for evaluation including *COIL20* [48], *Colon* [49], *Gisette* [1], *Lung-Cancer* (*L.Cancer*) [50], *Madelon* [51], *MovementLibras* (*M.Libras*) [52], *NCI9* [53], *ORL* [54], *Sonar* [41], *UAV1* and *UAV2* [55], and *Waveform* (*Wave.*) [42]. The instance numbers of these datasets range from 32 to 19937, the feature numbers range from 40 to 9712, and the true cluster numbers are from 2 to 40. The sample/feature ratios of these datasets range from 0.01 to more than 350, indicating the diversity of the datasets. Table II shows the statistics of these datasets.

Comparative Methods and Implementation: We choose 10 classical and recent state-of-the-art unsupervised feature selection methods for comparison. Laplacian Score (LapScore) [9] selects features by scoring features with a Gaussian Laplacian matrix. SPEC [10] is a more general framework for feature selection based on spectral graph theory, where LapScore is a special case of it. MCFS [56] uses spectral analysis and sparse regression to select features and capture the multi-cluster data structure. UDFS [12] selects features by discriminative analysis and $\ell_{2,1}$ minimization. NDFS [57] selects the most discriminative features with a nonnegative constraint and $\ell_{2,1}$ regularization. LRPFS [58] adopts a low-rank constraint to preserve

TABLE III
RESULTS OF DIFFERENT UFS METHODS ON SELECTED 10% FEATURES IN TERMS OF ACCURACY

Dataset	LapScore	SPEC	MCFS	UDFS	NDFS	LRPFS	NSSLFS	TSFS	CAE	InfFS	SOFT
COIL20	0.56±0.03	0.59±0.02	0.63±0.02	0.55±0.03	0.61±0.02	0.57±0.02	0.64±0.04	0.61±0.04	0.65±0.02	0.58±0.05	**0.74**±0.03
Colon	0.54±0.01	0.55±0.00	0.53±0.00	0.52±0.00	0.55±0.00	0.55±0.01	0.55±0.00	0.54±0.03	0.53±0.01	0.55±0.07	**0.58**±0.00
Gisette	0.67±0.00	0.70±0.00	**0.80**±0.00	0.58±0.00	0.74±0.01	N/A	N/A	0.57±0.00	0.62±0.00	0.61±0.04	0.75±0.00
L.Cancer	0.75±0.00	0.59±0.00	0.69±0.00	0.65±0.01	0.69±0.00	**0.78**±0.00	**0.78**±0.00	0.60±0.05	0.56±0.02	0.69±0.10	0.72±0.01
Madelon	0.51±0.00	0.51±0.00	0.53±0.00	0.51±0.00	**0.60**±0.00	0.50±0.00	0.53±0.00	0.57±0.03	0.56±0.00	0.52±0.03	0.57±0.00
M.Libras	0.27±0.01	0.29±0.01	0.38±0.01	0.35±0.02	0.43±0.01	0.39±0.01	0.37±0.01	0.36±0.03	0.43±0.02	0.44±0.03	**0.51**±0.02
NCI9	0.44±0.02	0.43±0.02	0.38±0.02	0.44±0.03	0.43±0.02	0.42±0.03	0.37±0.03	0.44±0.02	0.46±0.04	0.40±0.03	**0.53**±0.03
ORL	0.49±0.02	0.56±0.02	0.56±0.02	0.47±0.02	0.57±0.02	0.43±0.02	0.56±0.03	0.57±0.01	0.51±0.01	0.52±0.02	**0.63**±0.02
Sonar	0.57±0.00	0.54±0.00	0.58±0.00	0.54±0.00	0.52±0.00	0.52±0.00	**0.64**±0.00	0.56±0.04	0.56±0.00	0.58±0.00	0.59±0.00
UAV1	0.56±0.00	0.67±0.00	0.54±0.00	0.55±0.00	0.65±0.00	N/A	0.60±0.00	0.65±0.01	0.56±0.00	0.55±0.00	**0.80**±0.00
UAV2	0.80±0.00	0.60±0.00	0.76±0.00	0.81±0.00	0.58±0.00	0.55±0.00	0.56±0.00	0.64±0.01	0.81±0.00	0.58±0.02	**0.83**±0.00
Wave.	0.51±0.00	0.34±0.00	0.51±0.00	0.52±0.00	0.49±0.00	0.48±0.00	0.37±0.00	0.52±0.00	0.51±0.00	0.51±0.00	**0.56**±0.00
Average	0.55	0.53	0.57	0.54	0.57	0.52	0.54	0.55	0.56	0.54	**0.65**

N/A indicates the out-of-64GB-memory error.

the subspace structure information. NSSLFS [59] learns the feature weight matrix with the $\ell_{2,1}$-norm and the non-negative constraint based on the low-dimensional sparse subspace learning. TSFS [14] employs a teacher-student scheme for deep feature selection. CAE [39] uses the concrete distribution and the reparametrization trick to differentiate through a reconstruction loss and select input features. InfFS [28] is a fast graph-based approach that ranks and selects features by considering the possible subsets of features as paths on a graph.

For LapScore, SPEC, MCFS, UDFS, and NDFS, we adopt implementations and default settings provided by scikit-feature [60]. For LRPFS and NSSLFS, we set the values of hyperparameters in their objective functions to 1.0. For TSFS, CAE (non-linear version), and InfFS, we use default settings provided in their open-source codes. The settings of our SOFT model are as follows. In the stage of Attention Matrix learning, we implement the networks by PyTorch. The Learnable Mask is initialized by the normal distribution. The weights α and β in the objective function are 1 and 0.001 by default, respectively. We adopt Adam optimizer [46] to minimize the objective function and set the learning rate to 1e-4. We run a total of 300 epochs to learn the attention matrix. In the stage of graph segmentation, we first build a graph based on the learned attention matrix, then perform graph segmentation [47] and segment the graph into k partitions, where k is the target number of selected features. We choose one feature from each partition as our selected feature.

Evaluation Metric: We employ k-means++ [61] on samples with selected features, and compare the obtained partition with ground truth by clustering accuracy, settings of which follow scikit-feature [60]. Additionally, we propose to measure the redundancy of selected features by the proportion of feature pairs that are highly correlated, written as *#pairs of features with high correlation/#pairs of features*. While there is no generally acknowledged threshold for high correlation, we evaluate the compared methods with thresholds of 0.5, 0.6, 0.7, and 0.8 based on the correlation coefficient matrix calculated from features selected by the methods.

B. Algorithmic Performance

Table III shows the experimental results of different unsupervised feature selection methods on selected 10% features in terms of accuracy. The best results are highlighted in bold. "N/A" means the corresponding method cannot process the dataset successfully due to out-of-memory error. We can see that our SOFT model achieves the best on 8 of the 12 datasets. One of the possible reasons is that SOFT explores the second-order relationships among features, while other competitive methods only use the first-order data. By grouping features based on the learned attention matrix, our method can avoid the redundancy described in Section III. Moreover, the average accuracy of SOFT is significantly better than other methods with large margins of 5% to 10%, which demonstrates the positive effects of second-order feature exploration on the unsupervised feature selection problem.

Then for each method on each dataset, we calculate the proportion of feature pairs with high correlation when selecting 10% features, which is shown in Fig. 3. Obviously, as the threshold increases, the proportion of highly correlated feature pairs decreases. We can see from the results that with all four thresholds from 0.5 to 0.8, our proposed SOFT achieves the lowest proportion of highly correlated feature pairs on average, indicating that SOFT is effective in reducing redundancy in the feature selection process, which by our analysis comes from the advantage of adopting graph segmentation on learned attention matrix.

Next, we show the performance of all the methods on selecting different percents of features in Fig. 4. LRPFS achieves the best performance with selected 25% features on *Madelon*, MCFS is always the best among these methods on *Gisette*, NSSLFS outperforms other methods with selected 15% and 30% features on *Colon*, and UDFS with selected 20% and 25% features is ranked the first on *Waveform*. It is worth noting that sometimes the performance of SOFT is not as stable as other methods, such as on *Waveform* when the number of selected features increases. This is because the comparison methods generate a ranking list for features and select features based on the feature ranking scores, while SOFT selects features based on the graph segmentation result. With the number of selected features increasing, the feature partitions might change notably, which results in a different selection. In general, SOFT delivers compromising features with different percentages compared with others, especially when the percentage of selected features is relatively small. For results with all percentages and datasets,

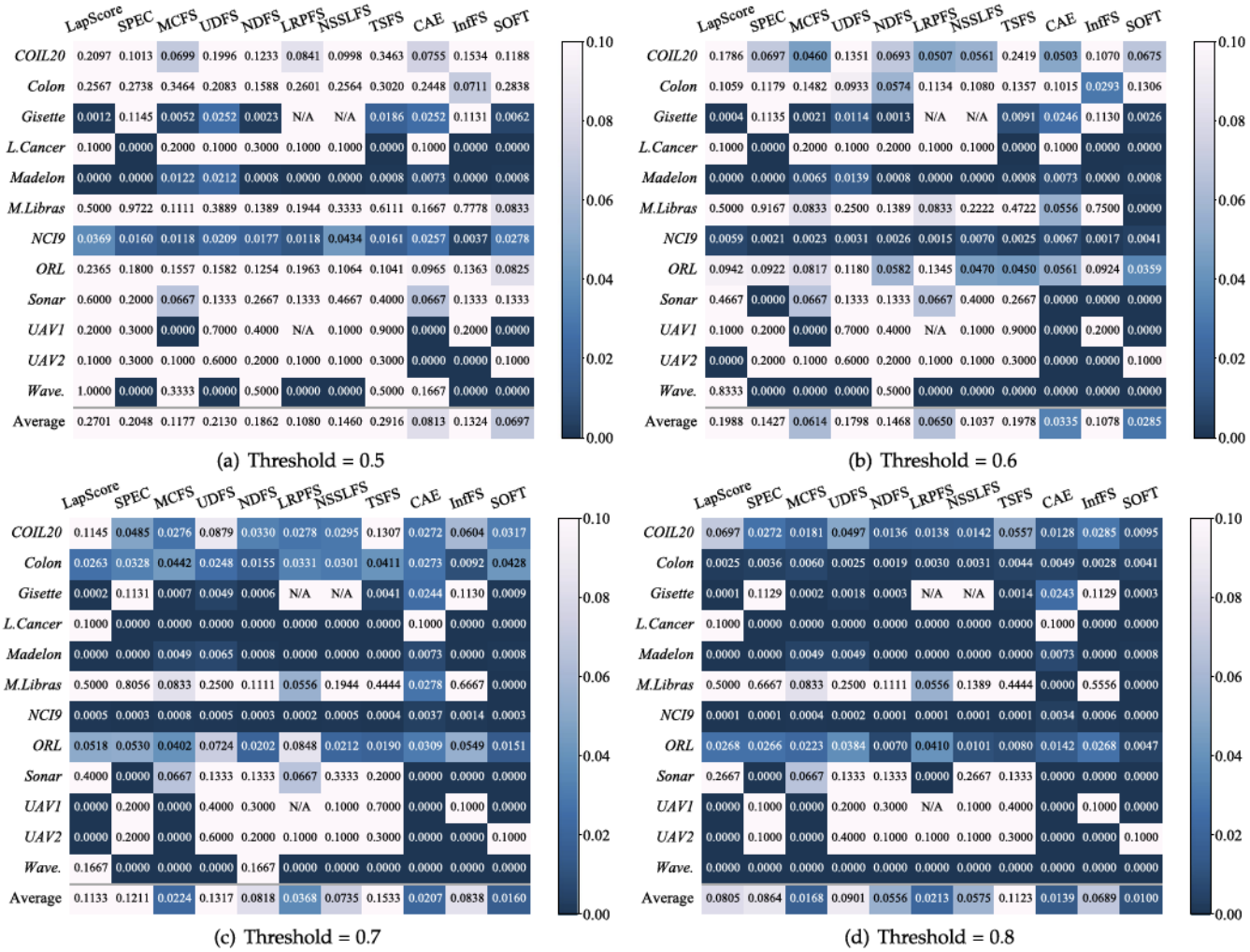


Fig. 3. Proportion of feature pairs with high correlation by different UFS methods on selected 10% features.

SOFT outperforms all other methods with an average accuracy of 66%, which is significantly better than two recent methods (CAE achieves 54%, and InfFS gets 51%).

C. In-Depth Exploration of SOFT

Parameter Analysis: There are two parameters in the objective function of SOFT, α and β, denoting the weight of attention loss and $\ell_{2,1}$-norm loss, respectively. We vary α and β from $1e-3$ to $1e+3$ to explore the impact of these two parameters on the final performance. Fig. 6(a) shows the results on $Colon$ with selected 10% features. We can see that despite the large range of parameter values, the final performance does not change much except when β is significantly larger than α, indicating that the Learnable Mask is well learned and SOFT might not be sensitive to the value of α and β in a large range.

Visualization of Attention Matrix and Learnable Mask: To further analyze the performance of SOFT, we visualize the Attention Matrix and the corresponding Learnable Mask in different epochs on *MovementLibras*, which is shown in Fig. 5. The darker color indicates the stronger correlation of the corresponding pair of features. To better recognize feature relations, we remove the diagonal values for visualization. In the beginning, there is no identified pattern in the Learnable Mask because the learnable mask is randomly initialized. As the training epochs increase, the learnable mask becomes sparser and seeks the dataset-dependent patterns. While the attention matrix is generated by an element-wise product of the original feature matrix and the learnable mask, the attention matrix became sparser as well. Therefore, the most important part of the attention matrix for representing samples was highlighted through network training.

Then we do a further step to explore the effectiveness of the learned attention matrix by comparing our results with two designated baseline methods. The first one is First Order method, which only uses intrinsic properties of the data. We remove the GCN part of SOFT and utilize a vectorial learnable mask for First Order. The First Order learns the ranking scores for features, which are denoted by the Learnable Mask. The second one is Covariance method, which uses the original feature matrix directly for the second stage of SOFT. Fig. 6(b) shows the experimental result in terms of accuracy. On almost all datasets, both SOFT and Covariance performed better than First Order, which demonstrates the effectiveness of incorporating feature relation in feature selection. While Covariance sometimes achieved

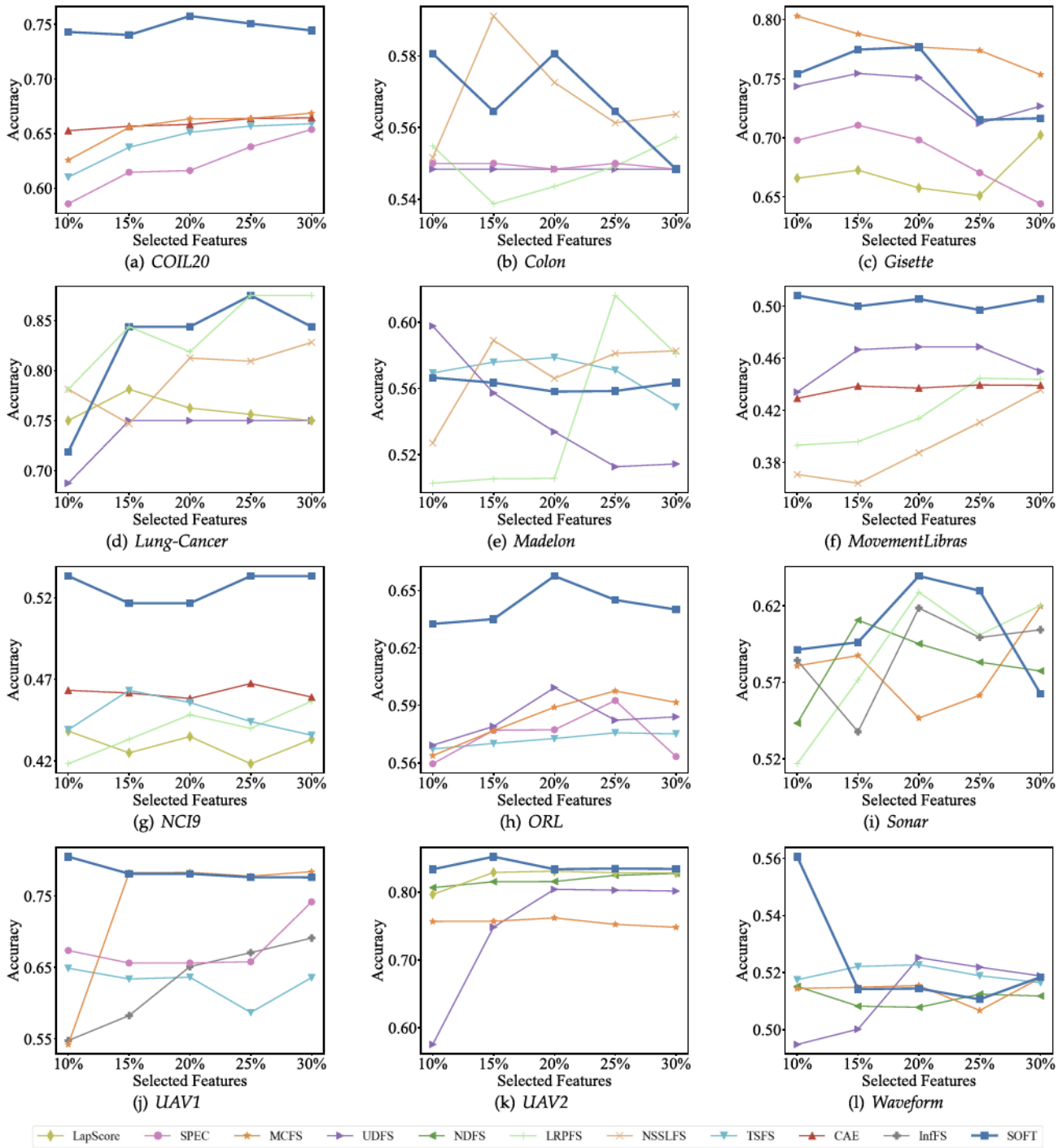


Fig. 4. Performance of 12 UFS methods on different percents of selected features. On each dataset, only top 5 methods on average performance are displayed for better visualization.

better results than SOFT, SOFT performed the best in most cases and achieved the best on average, proving that SOFT could learn a sparse and effective feature relation matrix to represent the second-order correlation.

Feature Selection Strategy: We adopt two baseline methods for feature selection from Attention Matrix as the comparison with the graph segmentation method in SOFT. The first baseline method, Weight Sum, utilizes a row-sum method to get the total relation value of each feature to all other features as the ranking scores for features. Features with the highest scores are selected. The second baseline method, Largest Weight, each time finds the largest value in Attention Matrix and selects the corresponding feature pair until k features are selected. Comparison results among Weight Sum, Largest Weight, and the graph segmentation in SOFT are shown in Fig. 6(c). Overall, the graph segmentation method in SOFT has a significant advantage over the baseline methods. This results from that SOFT avoids selecting highly correlated features by graph segmentation, thus reducing

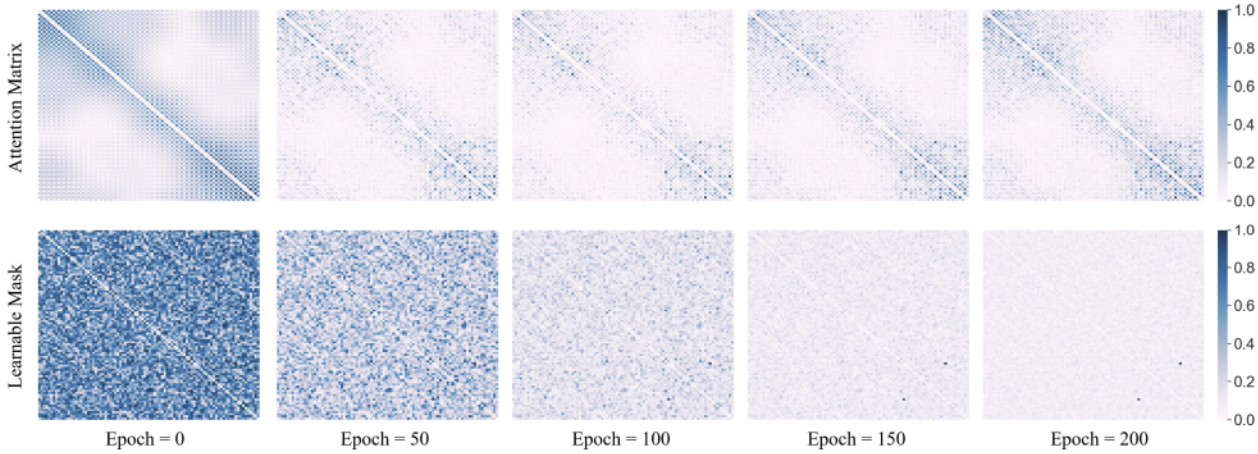
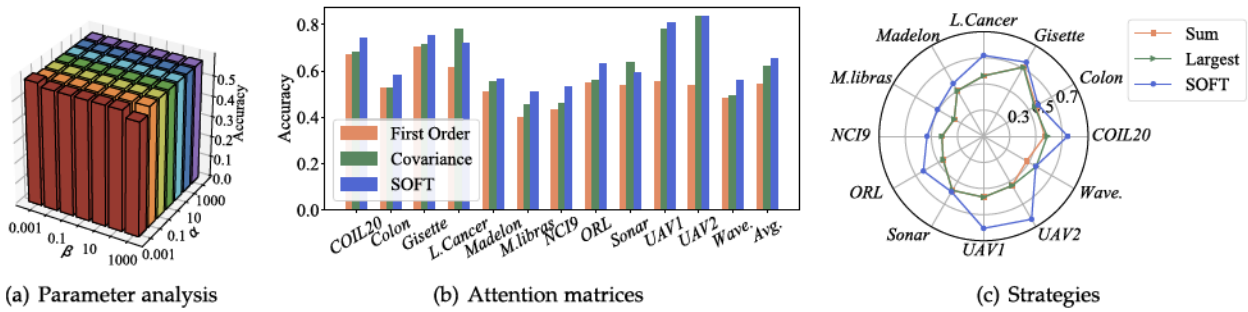
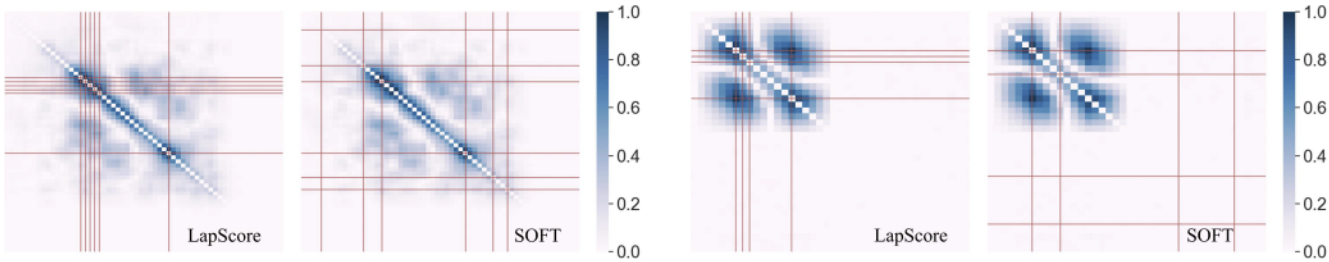


Fig. 5. Visualization of attention matrix and learnable mask during iterations on *MovementLibras*.



(a) Parameter analysis (b) Attention matrices (c) Strategies

Fig. 6. In-depth exploration of SOFT. (a) Parameter analysis of α and β on *Colon*, (b-c) performance of feature selection with different attention matrices and different feature selection strategies.



(a) *Sonar* (b) *Wave.*

Fig. 7. Visualization of feature relations. Red lines denote the features selected by LapScore and SOFT.

redundancies and bringing in more complementary features for performance boosting.

Verification: To verify that SOFT can select less-redundant features as mentioned in Section III, we visualize the selected 10% features on *Sonar* and *Wave.* in Fig. 7. Same to Fig. 1, we highlight the selected features by redlines. The darker area denotes higher relations of features. The highest correlations of selected feature pairs on *Sonar* and *Wave.* are 1.00 and 0.99 by LapScore, respectively, and 0.62 and 0.20 by SOFT, respectively. Obviously, features selected by SOFT have less redundancy than features selected by LapScore on these two datasets, which verifies that SOFT can achieve our target.

Stability: We further demonstrate the algorithmic stability, which means how a small change in data leads to large changes in the chosen feature subset [62]. A typical approach to measure stability is to first take M bootstrap samples of the provided data set, apply feature selection to each one of them, and then measure the variability in the M feature sets obtained. Here we conduct the stability tests on *COIL20* and *L.Cancer*. Specifically, we generate $M = 50$ bootstrap folds and run different unsupervised selection methods on these bootstrap folds. And then, we use the stability measurement proposed by [62] ranging from -1 to 1, which returns the stability score with confidence intervals. The large value means more stable. Fig. 8 shows the stability test of different feature selection methods on *COIL20* and *L. Cancer*. SOFT performs very stable with high scores and small intervals and excels others by a large margin in terms of stability scores.

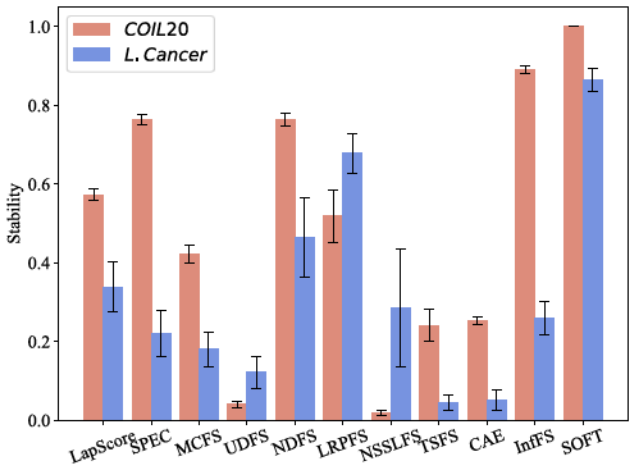


Fig. 8. Stability of different methods with selected 10% features on *COIL20* and *L.Cancer*.

V. Conclusion

In this paper, we proposed a two-stage framework named SOFT for unsupervised feature selection, which incorporated second-order data with first-order data. Specifically, in the first stage, we first obtained the attention matrix and masked matrix by applying a learnable mask on and off the input feature matrix. Then we generated attention/original/masked representations from attention/feature/masked matrices by a shared GCN. To train the learnable mask, we used pseudo labels generated by attention representation as positive guidance for original representation and negative guidance for masked representation, such that attention representation and original representation were similar to each other and different from masked representation. In the second stage, we constructed a graph based on the well-learned attention matrix and utilized graph segmentation to separate the graph into several parts. We chose one feature from each partition as the feature selection result. Experiments on public datasets demonstrated that our method outperformed classical and recent state-of-the-art methods on tackling the unsupervised feature selection problem.

Limitation: We ran experiments on a physical machine with a memory of 64 GB, an AMD Ryzen Threadripper 2920X 12-Core Processor, and an NVIDIA GP102 GPU, honestly, it cannot handle millions of features due to computing resource limitations. The scalability can be handled by more computing resources, using a sparse matrix [63], or focusing on the relationship of local features instead of global features [64].

References

[1] I. Guyon, S. R. Gunn, A. Ben-Hur, and G. Dror, "Result analysis of the NIPS 2003 feature selection challenge," in *Proc. Adv. Neural Inf. Process. Syst.*, 2004, pp. 545–552.
[2] G. Bajwa, R. J. DeBerardinis, B. Shao, B. Hall, J. D. Farrar, and M. A. Gill, "Cutting edge: Critical role of glycolysis in human plasmacytoid dendritic cell antiviral responses," *J. Immunol.*, vol. 196, pp. 2004–2009, 2016.
[3] E. Yang, T. Liu, C. Deng, W. Liu, and D. Tao, "DistillHash: Unsupervised deep hashing by distilling data pairs," in *Proc. Comput. Vis. Pattern Recognit.*, 2019, pp. 2941–2950.
[4] Q. Maoying, Y. Jun, L. Tongliang, W. Xinchao, and T. Dacheng, "Diversified Bayesian nonnegative matrix factorization," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 5420–5427.
[5] H. Behravan et al., "Machine learning identifies interacting genetic variants contributing to breast cancer risk: A case study in finnish cases and controls," *Sci. Rep.*, vol. 8, no. 1, 2018, Art. no. 13149.
[6] H. Behravan, J. M. Hartikainen, M. Tengström, V.-M. Kosma, and A. Mannermaa, "Predicting breast cancer risk using interacting genetic and demographic factors and machine learning," *Sci. Rep.*, vol. 10, no. 1, 2020, Art. no. 11044.
[7] M. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of ReliefF and RReliefF," *Mach. Learn.*, vol. 53, pp. 23–69, 2003.
[8] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf, "Measuring statistical dependence with Hilbert-Schmidt norms," in *Proc. Int. Conf. Algorithmic Learn. Theory*, 2005, pp. 63–77.
[9] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, pp. 507–514.
[10] Z. Zhao and H. Liu, "Spectral feature selection for supervised and unsupervised learning," in *Proc. Int. Conf. Mach. Learn.*, 2007, pp. 1151–1157.
[11] Z. Zhao, L. Wang, H. Liu, and J. Ye, "On similarity preserving feature selection," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 3, pp. 619–632, Mar. 2013.
[12] Y. Yang, H. T. Shen, Z. Ma, Z. Huang, and X. Zhou, "1-norm regularized discriminative feature selection for unsupervised learning," in *Proc. Int. Joint Conf. Artif. Intell.*, 2011, pp. 1589–1594.
[13] L. Du and Y.-D. Shen, "Unsupervised feature selection with adaptive structure learning," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 209–218.
[14] A. Mirzaei, V. Pourahmadi, M. Soltani, and H. Sheikhzadeh, "Deep feature selection using a teacher-student network," *Neurocomputing*, vol. 283, pp. 396–408, 2020.
[15] Z. Zhang, Y. Tian, L. Bai, J. Xiahou, and E. Hancock, "High-order covariate interacted Lasso for feature selection," *Pattern Recognit. Lett.*, vol. 87, pp. 139–146, 2017.
[16] Z. Zhu, T. Liu, and Y. Liu, "A second-order approach to learning with instance-dependent label noise," 2020, *arXiv: 2012.11854*.
[17] M. Dash, H. Liu, and J. Yao, "Dimensionality reduction of unsupervised data," in *Proc. IEEE 9th Int. Conf. Tools Artif. Intell.*, 1997, pp. 532–539.
[18] J. Dy and C. E. Brodley, "Feature selection for unsupervised learning," *J. Mach. Learn. Res.*, vol. 5, pp. 845–889, 2004.
[19] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc.*, vol. 39, pp. 1–22, 1977.
[20] Y. Kim, W. N. Street, and F. Menczer, "Evolutionary model selection in unsupervised learning," *Intell. Data Anal.*, vol. 6, pp. 531–556, 2002.
[21] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A k-means clustering algorithm," *J. Roy. Statist. Soc.*, vol. 28, pp. 100–108, 1979.
[22] M. H. Law, M. A. Figueiredo, and A. K. Jain, "Simultaneous feature selection and clustering using mixture models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1154–1166, Sep. 2004.
[23] C. S. Wallace and D. L. Dowe, "MML clustering of multi-state, poisson, von mises circular and Gaussian distributions," *Statist. Comput.*, vol. 73, pp. 73–83, 2000.
[24] M. Dash and H. Liu, "Feature selection for clustering," in *Proc. Pacific-Asia Conf. Knowl. Discov. Data Mining*, Heidelberg, Berlin, 2000, pp. 110–121.
[25] S. Solorio-Fernández, J. A. Carrasco-Ochoa, and J. F. Mart ínez-Trinidad, "A new hybrid filter–wrapper feature selection method for clustering based on ranking," *Neurocomputing*, vol. 214, pp. 866–880, 2016.
[26] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Commun. Statist.-Theory Methods*, vol. 3, pp. 1–27, 1974.
[27] Y. B. Kim and J. Gao, "Unsupervised gene selection for high dimensional data," in *Proc. IEEE Symp. BioInform. BioEng.*, 2006, pp. 227–234.
[28] G. Roffo et al., "Infinite feature selection: A graph-based feature filtering approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4396–4410, Dec. 2021.
[29] J. Miao, T. Yang, L. Sun, X. Fei, L. Niu, and Y. Shi, "Graph regularized locally linear embedding for unsupervised feature selection," *Pattern Recognit.*, vol. 122, 2022, Art. no. 108299.
[30] C. Tang, X. Zheng, W. Zhang, X. Liu, X. Zhu, and E. Zhu, "Unsupervised feature selection via multiple graph fusion and feature weight learning," *Sci. China Inf. Sci.s*, vol. 66, no. 5, pp. 1–17, 2023.

[31] C.-H. Chang, L. Rampasek, and A. Goldenberg, "Dropout feature ranking for deep learning models," 2017, *arXiv: 1712.08645*.
[32] A. Taherkhani, G. Cosma, and T. M. McGinnity, "Deep-FS: A feature selection algorithm for deep Boltzmann machines," *Neurocomputing*, vol. 322, pp. 22–37, 2018.
[33] Y. Li, C.-Y. Chen, and W. W. Wasserman, "Deep feature selection: Theory and application to identify enhancers and promoters," *J. Comput. Biol.*, 2016.
[34] L. Zhao, Q. Hu, and W. Wang, "Heterogeneous feature selection with multi-modal deep neural networks and sparse group Lasso," *IEEE Trans. Multimedia*, vol. 23, pp. 322–336, 2015.
[35] J. Friedman, T. Hastie, and R. Tibshirani, "A note on the group Lasso and a sparse group Lasso," 2010, *arXiv:1001.0736*.
[36] K. Han, Y. Wang, C. Zhang, C. Li, and C. Xu, "Autoencoder inspired unsupervised feature selection," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2018, pp. 2941–2945.
[37] S. Feng and M. F. Duarte, "Graph autoencoder-based unsupervised feature selection with broad and local data structure preservation," *Neurocomputing*, vol. 312, pp. 310–323, 2018.
[38] Y. Cong, S. Wang, B. Fan, Y. Yang, and H. Yu, "UDSFS: Unsupervised deep sparse feature selection," *Neurocomputing*, vol. 196, pp. 150–158, 2016.
[39] M. F. Balın, A. Abid, and J. Zou, "Concrete autoencoders: Differentiable feature selection and reconstruction," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 444–453.
[40] X. Gong, L. Yu, J. Wang, K. Zhang, X. Bai, and N. R. Pal, "Unsupervised feature selection via adaptive autoencoder with redundancy control," *Neural Netw.*, vol. 150, pp. 87–101, 2022.
[41] R. Rossi and N. Ahmed, "The network data repository with interactive graph analytics and visualization," in *Proc. AAAI Conf. Artif. Intell.*, 2015, pp. 4292–4293.
[42] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*. Boca Raton, FL, USA: CRC, 1984.
[43] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.
[44] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 478–487.
[45] K. Li, Z. Wu, K.-C. Peng, J. Ernst, and Y. Fu, "Tell me where to look: Guided attention inference network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9215–9223.
[46] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[47] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM J. Sci. Comput.*, vol. 20, pp. 359–392, 1998.
[48] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library: Coil-20," Columbia University, Tech. Rep., 1996.
[49] U. Alon et al., "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," *Nat. Acad. Sci.*, vol. 96, pp. 6745–6750, 1999.
[50] Z.-Q. Hong and J.-Y. Yang, "Optimal discriminant plane for a small number of samples and design method of classifier on the plane," *Pattern Recognit.*, vol. 24, pp. 317–324, 1991.
[51] I. Guyon, J. Li, T. Mader, P. A. Pletscher, G. Schneider, and M. Uhr, "Competitive baseline methods set new standards for the nips 2003 feature selection benchmark," *Pattern Recognit Lett.*, vol. 28, pp. 1438–1444, 2007.
[52] D. B. Dias, R. C. Madeo, T. Rocha, H. H. Bíscaro, and S. M. Peres, "Hand movement recognition for Brazilian sign language: A study using distance-based neural networks," in *Proc. Int. Joint Conf. Neural Netw.*, 2009, pp. 697–704.
[53] D. T. Ross et al., "Systematic variation in gene expression patterns in human cancer cell lines," *Nature Genet.*, vol. 24, pp. 227–35, 2000.
[54] D. Cai, X. He, J. Han, and H.-J. Zhang, "Orthogonal Laplacian faces for face recognition," *IEEE Trans. Image Process.*, vol. 15, no. 11, pp. 3608–3614, Nov. 2006.
[55] L. Zhao, A. Alipour-Fanid, M. Slawski, and K. Zeng, "Prediction-time efficient classification using feature computational dependencies," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 2787–2796.
[56] D. Cai, C. Zhang, and X. He, "Unsupervised feature selection for multi-cluster data," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2010, pp. 333–342.
[57] Z. Li, Y. Yang, J. Liu, X. Zhou, and H. Lu, "Unsupervised feature selection using nonnegative spectral analysis," in *Proc. AAAI Conf. Artif. Intell.*, 2012, pp. 1026–1032.
[58] W. Zheng, C. Xu, J. Yang, J. Gao, and F. Zhu, "Low-rank structure preserving for unsupervised feature selection," *Neurocomputing*, vol. 314, pp. 360–370, 2018.
[59] W. Zheng, H. Yan, and J. Yang, "Robust unsupervised feature selection by nonnegative sparse subspace learning," *Neurocomputing*, 2019, pp. 3615–3620.
[60] J. Li et al., "Feature selection: A data perspective," *ACM Comput. Surv.*, vol. 50, pp. 1–45, 2018.
[61] D. Arthur and S. Vassilvitskii, "K-means: The advantages of careful seeding," in *Proc. 18th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2007, pp. 1027–1035.
[62] S. Nogueira, K. Sechidis, and G. Brown, "On the stability of feature selection algorithms," *J. Mach. Learn. Res.*, vol. 18, pp. 6345–6398, 2017.
[63] K. Thakkar and P. Narayanan, "Part-based graph convolutional network for action recognition," 2018, *arXiv: 1809.04983*.
[64] L. Shi et al., "SGCN: Sparse graph convolution network for pedestrian trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 8994–9003.

Han Yue received the bachelor degree in computer science and technology from the School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China, in 2012, and the master degree in computer science and technology from the Institute of Software, University of Chinese Academy of Sciences, Beijing, China, in 2015. He is currently working toward the PhD degree with the Michtom School of Computer Science, Brandeis University. His research interests include data mining and machine learning.

Jundong Li (Member, IEEE) received the PhD degree in computer science from Arizona State University, in 2019. He is an assistant professor in the Department of Electrical and Computer Engineering, with a joint appointment in the Department of Computer Science, and the School of Data Science. His research interests are generally in data mining and machine learning, with a particular focus on graph mining, causal inference, and algorithmic fairness. As a result of his research work, he has published over 100 papers in high-impact venues (including KDD, WWW, NeurIPS, IJCAI, AAAI, WSDM, EMNLP, CIKM, ICDM, SDM, ECML-PKDD, CSUR, TPAMI, TKDE, TKDD, TIST, etc), with over 6,000 citation count. He has won several prestigious awards, including SIGKDD 2022 Best Research Paper Award, NSF CAREER Award, JP Morgan Chase Faculty Research Award, Cisco Faculty Research Award, and being selected for the AAAI New Faculty Highlights roster.

Hongfu Liu (Member, IEEE) received the bachelor's and master's degrees in management information systems from the School of Economics and Management, Beihang University, Beijing, China, in 2011 and 2014, respectively, and the PhD degree in computer engineering from Northeastern University, Boston, MA, USA, in 2018. He is currently a tenure-track assistant professor affiliated with the Michtom School of Computer Science, Brandeis University, Waltham, MA, USA. His research interests generally focus on data mining and machine learning, with special interests in ensemble learning. He is an associate editor of IEEE Computational Intelligence Magazine and an area chair of ICLR and NeurIPS.