



Fast Backward Iterative Laplacian Score for Unsupervised Feature Selection

Qing-Qing Pang¹  and Li Zhang^{1,2}  

- ¹ School of Computer Science and Technology and Joint International Research Laboratory of Machine Learning and Neuromorphic Computing, Soochow University, Suzhou 215006, Jiangsu, China
20184227025@stu.suda.edu.cn, zhangliml@suda.edu.cn
- ² Provincial Key Laboratory for Computer Information Processing Technology, Soochow University, Suzhou 215006, Jiangsu, China

Abstract. Iterative Laplacian Score (IterativeLS), an extension of Laplacian score (LS) for unsupervised feature selection, iteratively updates the nearest neighborhood graph for evaluating the importance of a feature by its local preserving ability. However, LS and IterativeLS separately measure the importance of each feature and do not consider the association of features. To remedy it, this paper proposes an enhanced version of IterativeLS, called fast backward iterative Laplacian score (FBILS). The goal of FBILS is to fast remove some unimportant features by taking into account the association of features. The proposed FBILS evaluates the feature importance according to the joint local preserving ability that reflects the association of features. In addition, FBILS deletes more than one feature in an iteration, which would speed up the process of feature selection. Extensive experiments are conducted on UCI and microarray gene datasets. Experimental results confirm that FBILS can achieve a good performance.

Keywords: Unsupervised learning · Feature selection · Laplacian score · Local preserving · Iteration algorithm

1 Introduction

With the development of technology and storage, the dimensionality of data could be very high in many applications, such as image annotation [1], object tracking [5] and image classification [15]. Usually, data may contain irrelevant and redundant information, which would have a negative effect on learning algorithms owing to the curse of dimensionality. As a technique of dimensionality reduction, feature selection has attracted a lot of attention in pattern recognition,

Supported by the Natural Science Foundation of the Jiangsu Higher Education Institutions of China under Grant No. 19KJA550002, the Six Talent Peak Project of Jiangsu Province of China under Grant No. XYDXX-054, and the Priority Academic Program Development of Jiangsu Higher Education Institutions.

machine learning and data mining. Feature selection can eliminate irrelevant and redundant features, which promotes the computational efficiency and keeps the interpretation of reduced description [13, 15].

According to the situations of data labels, feature selection methods can be divided into three types: supervised, unsupervised and semi-supervised ones [2]. Both supervised and semi-supervised methods for feature selection, to some extent, depend on the label information to guide the feature evaluation by encoding features' discriminative information contained in labels [10]. For unsupervised methods, feature importance is assessed by the ability to maintain specific attributes of data, such as the variance value [4], and Laplacian score (LS) [9]. LS was proposed based on the spectral graph theory and uses a neighborhood graph to determine optimal features. Zhu et al. [17] proposed an iterative Laplacian score (IterativeLS), which progressively changes the neighborhood graph by discarding the least important features in each iteration and assesses the importance of the feature by its Laplacian score value. In each iteration, IterativeLS would reconstruct a neighborhood graph using the rest features. In doing so, higher time complexity is required for IterativeLS. Moreover, both LS and IterativeLS separately measure the importance of each feature and ignore the association of features.

To enhance both LS and IterativeLS, this paper presents a fast backward iterative Laplacian score (FBILS) method for unsupervised feature selection. Inspired by IterativeLS, FBILS also adopts a recursive scheme to select features. FBILS differs from IterativeLS in that it deletes more than one feature at each iteration, greatly reducing the number of iterations. The criterion of evaluating the feature importance in FBILS is to calculate the joint local preserving ability of features, which is totally different from those in both LS and IterativeLS. In summary, FBILS could speed up the process of iterative feature selection and take into account the association of features. The validity and stability of FBILS is confirmed by experimental results.

The remainder of this paper is organized as follows. In Sect. 2, we review two unsupervised methods for feature selection. Section 3 proposes the FBILS method and discusses its properties. In Sect. 4, we conduct experiments on UCI and gene datasets to compare the proposed method with the existing unsupervised methods. This paper is summarized in Sect. 5.

2 Related Methods

This section briefly reviews two unsupervised feature selection methods: LS and IterativeLS, which are very related to our work.

Assume that there is a set of unlabeled data $X = \{\mathbf{x}_1, \dots, \mathbf{x}_u\}$, where $\mathbf{x}_i \in \mathbb{R}^n$, n is the number of features, and u is the number of samples. Let $F = \{f_1, \dots, f_n\}$ be the feature set with features f_k , $k = 1, \dots, n$ and $\mathbf{X} \in \mathbb{R}^{u \times n}$ be the sample matrix with row sample vectors \mathbf{x}_i , $i = 1, \dots, u$.

2.1 LS

He et al. [9] proposed LS based on manifold learning. The goal of LS is to select features which can keep the local structure of the original data. That is to say that LS concerns the local structure rather than the global structure of data.

For the given X , LS first constructs the neighborhood graph that can be represented by a weight matrix \mathbf{S} :

$$S_{ij} = \begin{cases} \exp\{-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2\}, & \text{if } (\mathbf{x}_i \in KNN(\mathbf{x}_j) \vee \mathbf{x}_j \in KNN(\mathbf{x}_i)) \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

where $\gamma > 0$ is a constant to be tuned, and $KNN(\mathbf{x}_i)$ denotes the set of K nearest neighbors of \mathbf{x}_i .

LS measures the importance of feature f_k by calculating its Laplacian score:

$$J_{LS}(f_k) = \frac{\sum_{i=1}^u \sum_{j=1}^u (x_{ik} - x_{jk})^2 S_{ij}}{\sum_{i=1}^u (x_{ik} - \mu_k)^2 D_{ii}} \quad (2)$$

where x_{ik} denotes the k th feature of the i th sample, $\mu_k = \frac{1}{u} \sum_{i=1}^u x_{ik}$ is the mean of all samples on feature f_k , and \mathbf{D} is a diagonal matrix with $D_{ii} = \sum_j S_{ij}$.

The smaller the score $J_{LS}(f_k)$ is, the more important the k th feature is for keeping the local structure of data. The computational complexity of constructing \mathbf{S} is $O(nu^2)$, and the computational complexity of calculating scores for n features is $O(nu^2)$. Hence, the overall computational complexity of LS is $O(nu^2)$.

2.2 IterativeLS

IterativeLS was presented by introducing the iterative idea into LS [17]. Experimental results in [17] indicated that IterativeLS outperforms LS on both classification and clustering tasks.

The key idea of IterativeLS is to gradually improve the neighborhood graph based on the remaining features. In each iteration, IterativeLS discards the least relevant feature with the greatest score among the remaining features. Similar to LS, IterativeLS evaluates the importance of a feature by its Laplacian score that is calculated according to (2).

Assume that A is the remaining feature subset in the current iteration. Let $|A| = n'$. In the current iteration, the computational complexity of constructing \mathbf{S} is $O(n'u^2)$, and computational complexity of calculating scores for n' features is $O(n'u^2)$. Note that $n' < n$. Hence, the overall computational complexity of IterativeLS is $O(n^2u^2)$ for n iterations.

3 Fast Backward Iterative Laplacian Score

This section presents the novel method for unsupervised feature selection: FBILS, which is an extension of LS. Both LS and IterativeLS measure the importance of features separately using the ability to maintain the local structure that

is measured by the Laplacian score. Similar to both LS and IterativeLS, FBILS also considers maintaining the local structure. However, FBILS measures the importance of feature f_k by considering its joint local preserving ability. Similar to IterativeLS, FBILS adopts a recursive scheme. The difference is that IterativeLS discards the least relevant feature with the greatest Laplacian score in each iteration, and FBILS uses a new criterion to remove more than one feature in each iteration. Therefore, FBILS greatly reduces the running time with less iterations.

3.1 Criterion

FBILS iteratively removes features according to a novel criterion, the joint local preserving ability. We first discuss how to measure the joint Laplacian score for feature subsets before we introduce the criterion to measure the importance of feature f_k .

Both LS and IterativeLS calculate the Laplacian score of only single feature. Because we want to consider the association between features, we have to calculate the Laplacian score of feature subsets, or joint Laplacian score that is defined as follows.

Definition 1. *Given the weight matrix \mathbf{S} , the centered sample matrix $\tilde{\mathbf{X}}_A$ and its corresponding feature subset A , the joint Laplacian score of A is defined as*

$$J(A) = \frac{\text{trace} \left(\tilde{\mathbf{X}}_A^T \mathbf{L} \tilde{\mathbf{X}}_A \right)}{\text{trace} \left(\tilde{\mathbf{X}}_A^T \mathbf{D} \tilde{\mathbf{X}}_A \right)} \quad (3)$$

where $\text{trace}(\cdot)$ is the sum of the diagonal elements of a matrix, $\mathbf{L} = \mathbf{D} - \mathbf{S}$ is the Laplacian matrix, and \mathbf{D} is a diagonal matrix with $D_{ii} = \sum_j S_{ij}$.

According to Definition 1, we need to construct the weight matrix \mathbf{S} to represent the neighborhood graph. For the given subset of data X_A that is a part of X and consists of features in A , we need to make it centered, which results in a new set with zero mean. Given these conditions, we can calculate the joint Laplacian score of A by (3). A smaller joint Laplacian score of A means that the local structure can be maintained better by using this feature subset, or A is more important. On the basis of Definition 1, we can describe the novel criterion as follows.

Definition 2. *Given the feature set F , a feature subset $A \subseteq F$, and any feature $f_k \in A$, the joint local preserving ability of f_k under A is defined as*

$$L(f_k|A) = J(A - \{f_k\}) - J(A) \quad (4)$$

where $A - \{f_k\}$ denotes that feature f_k is removed from A , and $J(A - \{f_k\})$ and $J(A)$ are the joint Laplacian scores of feature subsets $(A - \{f_k\})$ and A , respectively.

Definition 2 implies that the joint local preserving ability of feature f_k is related to two joint Laplacian scores. One is $J(A)$ that is the same for all $f_k \in A$. The other is $J(A - \{f_k\})$, the joint Laplacian score of the candidate subset $A - \{f_k\}$. We discuss two cases: $L(f_k|A) > 0$ and $L(f_k|A) \leq 0$.

- When $L(f_k|A) \leq 0$, $J(A - \{f_k\})$ is smaller than or equal to $J(A)$. In other words, the joint Laplacian score does not change or becomes small when feature f_k is removed from A . Note that LS-like algorithms prefer features with small Laplacian scores. In this case, we much prefer the feature subset $A - \{f_k\}$ to the one A . That is, f_k is less important.
- When $L(f_k|A) > 0$, $J(A - \{f_k\})$ is greater than $J(A)$. In this case, it is unwise to remove feature f_k from A .

In summary, the greater the value $L(f_k|A)$ is, the stronger the joint local preserving ability of f_k under A is. Thus, what we need to do is to remove these features with a weak joint local preserving ability. We could follow the way of IterativeLS, or deleting the weakest feature in each iteration. However, it is time-consuming if just one bad feature is removed in an iteration.

One goal of FBILS is to speed up the iterative procedure, which can be implemented by using the following deletion rule:

$$\forall f_k \in A, f_k \text{ is deleted from } A, \text{ if } L(f_k|A) \leq 0 \quad (5)$$

The deletion rule allows us to remove all possible bad features from the current feature subset in one time. The feasibility and reasonability of this rule is discussed later in Subject. 3.3.

3.2 Algorithm Description

The detail algorithm description of FBILS is shown in Algorithm 1. Quite simply, FBILS requires constructing a neighborhood graph and calculating joint Laplacian scores of feature subsets in each iteration, which is repeated until all features are ranked.

The inputs of FBILS include the given dataset and the parameter K that is required for the neighborhood graph. In the first step, the subset of remaining features A is initialized to be a complete set, or the whole feature set. \bar{A} is the ordered set and $\bar{A} = \emptyset$. Step 3 computes the weight matrix \mathbf{S}_A using the dataset X_A in the current iteration. Step 4 computes the joint Laplacian score of the feature subset A by (3). Steps 5–8 get the joint local preserving ability for all features in A . Step 10 finds out the unimportant features in A with $L(f_k|A) \leq 0$ and forms a temporary feature subset B . If B is empty, then FBILS would come to an end. In this case, features in A would be ranked according to the value of $L(f_k|A)$. If B is non-empty, then features in B would be ranked according to their joint local preserving ability. Then the ranked features are inserted in the list \bar{A} . Note that the smaller the value $L(f_k|A)$ is, the later is the insertion of the corresponding feature f_k in the list. If B is non-empty, features in B should be removed from A . Repeat Steps 3–18 until A is empty.

Finally, Algorithm 1 greturns the ranking of all features. Assume that the number of features required by a given task is provided in advance, say r . There are two ways to get r features using FBILS. One way is that we can only pick up the r top features in \bar{A} after we perform Algorithm 1. The other way is to change the termination condition in Step 2 as $|A| > r$ and return the remaining feature subset A instead of \bar{A} . In Step 20, if $|A| < r$, we would get the $r - |A|$ top features in \bar{A} and put them into A .

3.3 Properties Analysis

This part concerns about the properties of FBILS. We give a lemma and a theorem, and do not prove them for limitation of space.

Lemma 1. *For a non-empty feature subset A , its joint Laplacian score is positive, or*

$$J(A) \geq 0. \quad (6)$$

Lemma 1 indicates that the joint Laplacian score of feature subsets are greater than zero. The following theorem is related to our deletion rule (5).

Algorithm 1. Fast Backward Iterative Laplacian Score (FBILS)

Require: Dataset X with u samples and n features, and neighborhood parameter K ;

Ensure: The ordered set of features;

- 1: Initialize $A = \{f_1, \dots, f_n\}$ and the ordered set $\bar{A} = \emptyset$;
 - 2: **while** $A \neq \emptyset$ **do**
 - 3: Calculate the weight matrix \mathbf{S}_A by (1) using X_A with u samples and $|A|$ features;
 - 4: Compute the joint Laplacian score of A , $J(A)$ by (3);
 - 5: **for** each $f_k \in A$ **do**
 - 6: Compute the joint Laplacian score of $A - \{f_k\}$, $J(A - \{f_k\})$ by (3);
 - 7: Compute $L(f_k|A)$ by (4);
 - 8: **end for**
 - 9: Let a temporary feature subset $B = \emptyset$;
 - 10: Find all features with $L(f_k|A) \leq 0$ and add them to B , or $B = \{f_k \in A \mid L(f_k|A) \leq 0\}$;
 - 11: **if** $B = \emptyset$ **then**
 - 12: Rank features in A according to the descending order of $L(f_k|A)$, $f_k \in A$;
 - 13: Insert the ranked features into the head of the list \bar{A} ;
 - 14: **break**;
 - 15: **else**
 - 16: Rank features in B according to the descending order of $L(f_k|A)$, $f_k \in B$;
 - 17: Insert the ranked features into the head of the ordered set \bar{A} ;
 - 18: Update the set A by removing these features in B , or $A = A - B$;
 - 19: **end if**
 - 20: **end while**
 - 21: **return** The ordered set \bar{A} .
-

Theorem 1. *Let A be the feature subset. For $f_k \in A$, given the joint Laplacian score $J(A)$, the joint local preserving ability $L(f_k|A)$ and the Laplacian score $J_{LS}(f_k)$, the following relationships hold true:*

- If $L(f_k|A) > 0$, then $J_{LS}(f_k) < J(A)$;
- If $L(f_k|A) \leq 0$, then $J_{LS}(f_k) \geq J(A)$.

Theorem 1 implies the relationship between the Laplacian score of feature f_k and the joint Laplacian score of the feature subset A containing f_k when $L(f_k|A) \leq 0$ or $L(f_k|A) > 0$. When $L(f_k|A) \leq 0$, the Laplacian score of f_k is greater than or equal to the joint Laplacian score of A . The deletion rule states that those features with $L(f_k|A) \leq 0$ would be removed from the current feature subset. In other words, we would delete all $f_k \in A$ with $J_{LS}(f_k) \geq J(A)$, where $J(A)$ is taken as a threshold to make us select more than one feature in one iteration. According to the view of both LS and IterativeLS, the greater $J_{LS}(f_k)$ is, the less important feature f_k is. Thus, our deletion rule is reasonable because it consists with the view.

Now, we analyze the computational complexity of FBILS. Without loss in generality, let A be the remaining feature subset in the current iteration. Let $|A| = n'$. Similar to IterativeLS, the computational complexity of constructing \mathbf{S}_A is $O(n'u^2)$. If we directly compute joint Laplacian scores, which has a computational complexity of $O(n'^2u^2)$, FBILS would be slow. Fortunately, we could speed up this procedure. According to the proof of Theorem 1, the calculation of $J(A - \{f_k\})$ can be reduced. For limitations of space, we do not discuss it any more. Thus, the computational complexity of computing joint Laplacian scores is reduced to be $O(n'u^2)$. Then the overall time complexity of FBILS is between $O(nu^2)$ and $O(n^2u^2)$, which is related to the iteration number. Assume that the iteration number is T , then the complexity is $O(Tnu^2)$.

4 Experimental Analysis

In order to verify the feasibility and effectiveness of FBILS, simulation experiments were carried out on several UCI datasets [7] and microarray gene expression datasets [11]. We compared FBILS with both LS and IterativeLS and used the nearest neighbor classifier to measure the discriminant ability of selected features.

All experiments were performed in MATLAB 2015b and run in a hardware environment with an Intel Core i5 CPU at 2.60 GHz and with 8 GB RAM.

4.1 UCI Dataset

We considered 8 UCI datasets here and compared FBILS with Variance, LS and IterativeLS algorithms. The related information of 8 UCI datasets, including Australian, Glass, Heart, Ionosphere, Segment, soybeanLarge, Vehicle and Wdbc, is shown in Table 1. For these UCI datasets, the original features were normalized to the interval $[0, 1]$. To validate the ability to select features, we

added n noise features to the original data before normalization, where n is the number of features. In the ranked feature list, we only considered the first n features. We thought that a good method for feature selection should rank these noise features towards the end of the list.

In order to obtain more convincing comparison results and eliminate accidental errors, we used 10-fold cross-validation. That is to say, the original dataset was randomly divided into ten equal-sized subsets. Then 9 subsets were used as the training set and the rest one was used as the test set. The 10 subsets were used as test sets in turn, and then the average of 10 times was calculated as the final result of classification. In addition, all compared algorithms require the parameter K for constructing the neighborhood graph. Let it vary in the set $\{1, 2, \dots, 9\}$.

Table 2 shows the highest average accuracy with the corresponding standard deviation and optimal feature number of all compared algorithms, where the best values among compared methods are in bold. We can see that FBILS is superior to Variance, LS and IterativeLS on all datasets. For example, FBILS achieves the accuracy 92.00% on the SoybeanLarge dataset, followed by IterativeLS 90.80%. In a nutshell, FBILS can effectively rank features and make good ones at the top of feature list. Table 3 shows the running time and iteration numbers of FBILS and IterativeLS on the UCI datasets. Owing to the small number of features in UCI datasets, the time required for LS and variance is very small. Thus, we did not list them here. It can be clearly seen that the number of iterations of FBILS is much smaller than IterativeLS, and the running time of FBILS is also less.

Table 1. Description of UCI datasets

No.	Dataset	#Sample	#Attribute	#Class
1	Australian	690	14	2
2	Glass	214	9	6
3	Heart	301	13	2
4	Ionosphere	351	34	2
5	Segment	2310	19	7
6	SoybeanLarge	250	35	14
7	Vehicle	846	18	4
8	Wdbc	569	30	2

4.2 Microarray Gene Datasets

In experiments, FBILS was applied to microarray gene datasets, including Leukemia [8], Novartis [6], St. Jude Leukemia (SJ-Leukemia) [16], Lungcancer [3] and the central nervous system (CNS) [12]. It is well-known that the number of features is much greater than the number of samples in the gene datasets. The

Table 2. Average accuracy and standard deviation of different methods on UCI datasets

Dataset	FBILS	Variance	LS	IterativeLS
Australian	83.33 \pm 5.57 (11)	81.91 \pm 4.06 (10)	81.33 \pm 4.48 (8)	81.77 \pm 4.02(8)
Glass	62.32 \pm 14.41 (7)	42.59 \pm 11.26 (7)	60.74 \pm 11.60 (4)	51.73 \pm 11.41 (9)
Heart	78.52 \pm 7.16 (12)	76.30 \pm 9.59 (7)	77.78 \pm 9.56 (15)	76.30 \pm 9.11 (10)
Ionosphere	93.19 \pm 3.26 (8)	91.46 \pm 4.22 (8)	92.63 \pm 6.62 (9)	92.06 \pm 3.42 (11)
Segment	97.27 \pm 1.08 (18)	97.27 \pm 1.14 (12)	97.14 \pm 1.14 (17)	96.84 \pm 1.65 (12)
SoybeanLarge	92.00 \pm 5.96 (28)	88.00 \pm 5.96 (33)	90.40 \pm 6.31 (31)	90.80 \pm 4.24 (27)
Vehicle	73.50 \pm 4.90 (17)	69.85 \pm 4.57(12)	73.38 \pm 2.27 (16)	73.16 \pm 2.80(12)
Wdbc	96.47 \pm 3.31 (30)	92.43 \pm 1.51 (19)	96.30 \pm 1.81 (29)	96.30 \pm 2.06 (23)

*Numbers in parentheses are optimal feature numbers. The accuracy and standard deviation was showed by percentage values.

Table 3. Running time (sec.) and iteration numbers of FBILS and IterativeLS on UCI datasets

Dataset	FBILS	IterativeLS
Australian	0.494 \pm 0.078 (2.9)	1.967 \pm 0.027 (28)
Glass	0.068 \pm 0.011 (3)	0.129 \pm 0.009 (18)
Heart	0.097 \pm 0.023 (3.3)	0.277 \pm 0.018 (26)
Ionosphere	0.475 \pm 0.025 (4.7)	1.556 \pm 0.126 (68)
Segment	7.677 \pm 0.316 (5.1)	34.501 \pm 0.273 (38)
Soybeanlarge	0.331 \pm 0.025 (7.9)	1.030 \pm 0.037 (70)
Vehicle	1.233 \pm 0.116 (5)	4.448 \pm 0.071 (36)
Wdbc	1.000 \pm 0.034 (6)	3.968 \pm 0.461 (60)

*Numbers in parentheses are mean number of iterations.

Table 4. Description of microarray gene datasets

No.	Dataset	#Sample	#Attribute	#Class
1	Lungcancer	197	1000	4
2	Novartis	103	1000	4
2	SJ–Leukemia	248	985	6
3	Leukemia	38	999	3
4	CNS	42	989	5

gene expression datasets we used have been processed as described in [11]. Further biological details about these datasets can be found in the referenced papers. Most data were processed on the Human Genome U95 Affymetrix ©microarrays. The leukemia dataset was from the previous-generation Human Genome HU6800 Affymetrix ©microarray. The relevant information of these datasets is summaries in Table 4.

Here, we also compared FBILS with the unsupervised methods: Variance, LS and IterativeLS. In order to obtain convincing comparison results and eliminate accidental errors, as in the previous section, we used cross-validation. Because the number of samples in datasets is small, three-fold cross-validation was applied. In each trail, we randomly selected 2/3 of the samples as the training set, and the remaining 1/3 of samples as the test set. The experimental results were reported on the well-defined test sets. According to the statement in [14], we can know that we need 400 genes at most to complete the classification task of microarray gene data. Therefore, we analyze on the first 400 top features.

Figure 1 gives the classification accuracy vs. feature number on five microarray gene datasets. From Fig. 1, we can see that FBILS is obviously superior to other three methods on CNS, Lungcancer and Novartis datasets. In addition, FBILS can quickly achieve a better classification performance. We summarized the highest accuracy of compared methods in Table 5 according to Fig. 1, where bold numbers are the best results among compared methods. FBILS algorithm achieves better accuracies on all five gene datasets. On the Leukemia datasets, FBILS has the same accuracy as LS and IterativeLS. On the Lungcancer dataset, the accuracy of FBILS is 1.5% higher than LS. Table 6 shows the running time of the four methods on the gene dataset. The Variance and LS methods are fast without iteration. Two iterative methods, FBILS and IterativeLS take more time. However, it can be clearly seen that FBILS runs many times faster than IterativeLS.

Table 5. Average accuracy and standard deviation comparison on five microarray gene datasets

Dataset	FBILS	Variance	LS	IterativeLS
CNS	87.63 ± 5.09 (40)	85.07 ± 14.71 (83)	86.81 ± 8.96 (138)	87.42 ± 15.16 (187)
Leukemia	96.97 ± 5.25 (73)	93.94 ± 10.49 (144)	96.97 ± 5.25 (59)	96.97 ± 5.24 (358)
Lungcancer	96.44 ± 2.29 (142)	94.93 ± 5.90(208)	94.94 ± 3.16 (292)	94.40 ± 7.11 (286)
Novartis	100 ± 0.00 (37)	99.07 ± 0.78(359)	98.96 ± 2.24 (375)	99.05 ± 3.53 (199)
SJ-leukemia	99.20 ± 0.70 (326)	98.01 ± 1.81 (230)	98.01 ± 1.81 (217)	98.01 ± 1.81 (142)

*Numbers in parentheses are optimal feature numbers. The accuracy and standard deviation was showed by percentage values.

Table 6. Running time (sec.) of different methods on five microarray gene datasets

Dataset	FBILS	Variance	LS	IterativeLS
CNS	0.083 ± 0.032	0.003 ± 0.006	0.030 ± 0.010	13.700 ± 0.430
Leukemia	0.073 ± 0.006	0.007 ± 0.012	0.033 ± 0.006	14.577 ± 0.520
Lungcancer	0.780 ± 0.144	0.003 ± 0.006	0.233 ± 0.006	122.940 ± 3.498
Novartis	0.243 ± 0.085	0.007 ± 0.012	0.100 ± 0.017	45.143 ± 0.326
SJ-leukemia	0.950 ± 0.128	0.007 ± 0.006	0.287 ± 0.006	154.700 ± 8.819

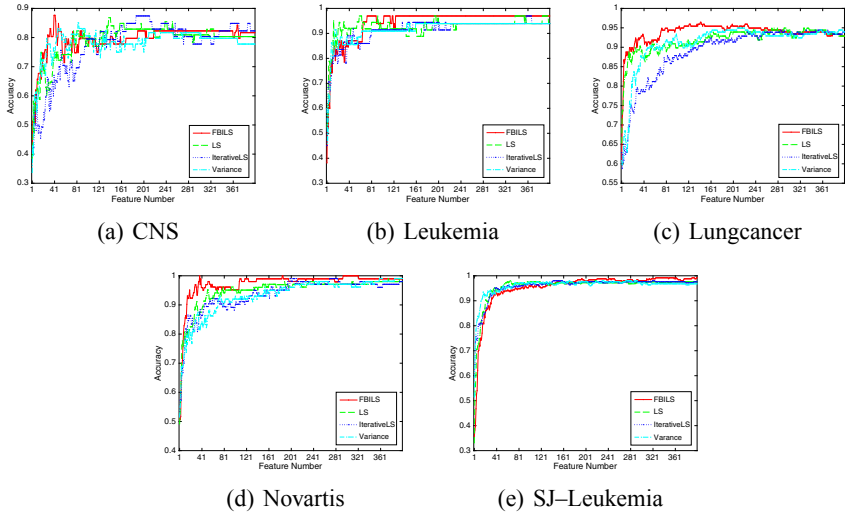


Fig. 1. Accuracy vs. feature number on five gene datasets

5 Conclusion

This paper concentrates on unsupervised feature selection and proposes an algorithm called FBILS. FBILS aims to speed up the iterative process and maintains the local manifold structure. Different from existing LS-like methods, FBILS evaluates the joint locality preserving ability of features instead of Laplacian score, and picks up more than one features in one time. On eight UCI and five microarray gene datasets, a series of experiments were conducted for evaluating the proposed method. FBILS retains the highest classification accuracy on most datasets. From the running time of the UCI and gene dataset, we know that FBILS is much faster than of IterativeLS.

References

1. Amiri, S.H., Jamzad, M.: Automatic image annotation using semi-supervised generative modeling. *Pattern Recogn.* **48**(1), 174–188 (2015)
2. Benabdeslem, K., Hindawi, M.: Constrained laplacian score for semi-supervised feature selection. In: Gunopulos, D., Hofmann, T., Malerba, D., Vazirgiannis, M. (eds.) *ECML PKDD 2011. LNCS (LNAI)*, vol. 6911, pp. 204–218. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23780-5_23
3. Bhattacharjee, A., Richards, W.G., Staunton, J., et al.: Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinomas sub-classes. *Proc. Natl. Acad. Sci.* **98**(24), 13790–13795 (2001)
4. Bishop, C.M.: *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford (1995)
5. Collins, R.T., Liu, Y., Leordeanu, M.: Online selection of discriminative tracking features. *IEEE Trans. Pattern Anal. Mach. Intell.* **27**(10), 1631–1643 (2005)

6. Cooke, M.P., Ching, K.A., Hakak, Y., et al.: Large-scale analysis of the human and mouse transcriptomes. *Proc. Natl. Acad. Sci.* **99**(7), 4465–447 (2002)
7. Dheeru, D., Karra Taniskidou, E.: UCI machine learning repository (2017). <http://archive.ics.uci.edu/ml>
8. Golub, T.R., Slonim, D.K., Tamayo, P., et al.: Molecular classification of cancer: class discovery and class prediction by gene expression. *Science* **286**(5439), 531–537 (1999)
9. He, X., Cai, D., Niyog, P.: Laplacian score for feature selection. In: *International Conference on Neural Information Processing Systems*, pp. 507–541 (2005)
10. Luo, M., Nie, F., Chang, X., Yang, Y., Hauptmann, A.G., Zheng, Q.: Adaptive unsupervised feature selection with structure regularization. *IEEE Trans. Neural Netw. Learn. Syst.* **29**(4), 944–956 (2018)
11. Monti, S., Tamayo, P., Mesirov, J., Golub, T.: Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data. *Mach. Learn.* **52**(1–2), 91–118 (2003)
12. Pomeroy, S., Tamayo, P., Gaasenbeek, M., et al.: Gene expression-based classification and outcome prediction of central nervous system embryonal tumors. *Nature* **415**(6870), 436–442 (2001)
13. Sheikhpour, R., Sarram, M.A., Gharaghani, S., Chahooki, M.A.Z.: A survey on semi-supervised feature selection methods. *Pattern Recogn.* **64**, 141–158 (2017)
14. Shieh, M.D., Yang, C.C.: Multiclass SVM-REF for product from feature selection. *Expert Syst. Appl.* **35**(1–2), 531–541 (2008)
15. Song, X., Zhang, J., Han, Y., Jiang, J.: Semi-supervised feature selection via hierarchical regression for web image classification. *Multimedia Syst.* **22**(1), 41–49 (2014). <https://doi.org/10.1007/s00530-014-0390-0>
16. Yeoh, E.J., Ross, M.E., Shurtleff, S.A., et al.: Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling. *Cancer Cell* **1**(2), 133–143 (2002)
17. Zhu, L., Miao, L., Zhang, D.: Iterative laplacian score for feature selection. In: Liu, C.-L., Zhang, C., Wang, L. (eds.) *CCPR 2012. CCIS*, vol. 321, pp. 80–87. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33506-8_11