2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Data Dimensionality Reduction Approach To Improve Feature Selection Performance Using Sparsified SVD

Pengpeng Lin
Computer Science Department
University of Kentucky
Lexington, KY 40506-0633, USA
Email: M.Lin@uky.edu

Jun Zhang
Computer Science Department
University of Kentucky
Lexington, KY 40506-0633, USA
Email: J.Zhang@uky.edu

Ran An
Department of Plant and Soil Sicences
University of Kentucky
Lexington, KY 40546-0073, USA
Email: ran.an@uky.edu

***Abstract*—Feature selection is a technique of selecting a subset of relevant features for building robust learning models. In this paper, we developed a data dimensionality reduction approach using sparsified singular value decomposition (SSVD) technique to identify and remove trivial features before applying any advanced feature selection algorithm. First, we investigated how SSVD can be used to identify and remove nonessential features in order to facilitate feature selection performance. Second, we analyzed the application limitations and computing complexity. Next, a set of experiments were conducted and the empirical results show that applying feature selection techniques on the data of which the nonessential features are removed by the data dimensionality reduction approach generally results in better performance with significantly reduced computing time.**

I. Introduction

As our world expands virtually at an unprecedented rate, large scale data are collected every day. As a result, analysis of data becomes computationally infeasible when it comes to a point of too much. The existence of large quantities of data is useless if there are no effective data analysis methods [1]. Extensive efforts have been devoted to feature selection research, due to increasing demands for data dimensionality reduction. The general goal of feature selection is to find a minimum set of features such that the resulting probability distribution of data is as close to the original data as possible. Through feature selection, nonessential (irrelevant) features can be removed without affecting learning performance [2]. Otherwise, nonessential features may cause confusion for the mining algorithm employed [3].

To search for the essential features, various search strategies have been developed and can be broadly categorized as exhaustive, heuristic, and randomized. An exhaustive search would certainly find the optimal solution. however, for a dataset with N features, a search on 2^N possible feature combinations is obviously computationally impractical for large value of N [1]. More practical search strategies for large data have been studied. Marill and Green [4] proposed the sequential backward selection, which starts with full feature space and sequentially eliminates features that contribute least to the criterion function one at a time. Whitney [5] introduced sequential forward selection which starts with an empty set and sequentially adds one feature at a time. These greedy sequential search methods generally result in an $O(N^2)$ worst case scenario. Heuristic search methods such as genetic algorithms add some randomness in the search procedure to escape from a local optimum. Individual search methods evaluate each feature individually and select features that either satisfy the condition or are top-ranked. However, these search strategies often trade optimality for efficiency by avoiding searching feature space completely. There is no guarantee that the best possible solution can be found using these techniques. However, one can improve the probability of obtaining the best solution by reducing the proportion of nonessential features, i.e., making the good sampling choices to occur more frequently than that for the bad choices. In this work, we applied a sparsified SVD to reduce dimension of a given data, in an effort to achieve high solution quality and reduce computing time. There have been some works introducing the use of SVD for feature selection methods in recent years. Chen et al., combined SVD and Monte Carlo Decision Tree to fine tune selected biomarker set in terms of classification accuracy [6]. Fallucchi and Zanzotto proposed a novel way of using SVD to compute the pseudo-inverse matrix needed in logistic regression [7]. Phillips et al., claimed that a classification operation that can be applied to a $m \times n$ matrix A can be equivalently applied to a $k \times n$ matrix ΣV^T, where $k < m$ [8]. However, all of them used the general mathematical properties of SVD on various applications without building a systematic model for feature selection or data dimension reduction. The promising results obtained when applying SVD to various applications indicate that more works need to be devoted in order to explore its full potential.

The paper is organized as follows. In Section 2, we proposed to use the sparsified SVD for reducing data dimensionality. In Section 3, we presented and discussed the sparsification strategy. In Section 4, we summarized the approach and analyzed its complexity. In Section 5, we conducted experiments and the empirical results were presented and discussed. Finally, we concluded this work and laid out possible future works in Section 6.

II. Reducing data dimensionality using sparsified Singular Value Decomposition technique

Throughout this paper, the notations are consistent and defined as followings. We use lower case letters for scalar,

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

lower case letters with under bar for vector, and capital letters for data matrix. For example, A is a matrix in $R^{m\times n}$ with $m \geq n$, a is a scalar, $\underline{a}$ is a column vector and the row vector is denoted as $\underline{a}^T$. A_{ij} denotes for elements in the i^{th} row and j^{th} column of A. $R(A)$ denotes for the matrix range and r denotes for the matrix rank where $r \leq n$. We also assume the use of numerical data type and any data can be treated as a matrix. This assumption is made without loss of generality. Since both the quantitative and categorical data domains can be represented in binary form, the results can be easily extended to categorical data.

A. Singular Value Decomposition

SVD is a low rank approximation technique well known for the dimension reduction problem [9]. The insight underlying the use of SVD for dimension reduction is that it takes a data and breaks it into linearly independent components. These components are in some sense an abstraction away from the noisy correlations and close to sets of values that best approximate the underlying data structure. Essentially, SVD factors a matrix A into a product of three matrices: U, D, V^T (Figure 1), where U is an $m \times m$ orthogonal matrix, V is an

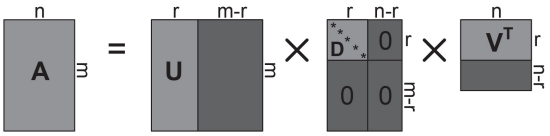

Fig. 1. Singular Value Decomposition

$n \times n$ orthogonal matrix, and D is a $m \times n$ diagonal matrix whose diagonal entries are called singular values such that $\sigma_1 \geq \sigma_2 \geq, \ldots, \geq \sigma_r \geq 0$. Let λ_i be the eighenvalues of $A^T A$, then $\lambda_i \subseteq (\sigma_1^2, \sigma_2^2, \ldots, \sigma_r^2), \forall i \in [1, n]$ [6]. SVD has numerous applications in data mining, information retrieval and image compression in which it is often used to approximate a given matrix by a lower rank matrix with the minimum distance between them.

B. Eliminating nonessential features with SSVD

1) Define nonessential features: For a dataset A with n features and m tuples, its column vectors form a vector space. Each vector in the space can be built up by applying operations of addition and scalar multiplication on a set of vectors that are referred to as spanning set. Particularly in linear algebra, it is desirable to find a minimal spanning set called 'basis' such that the base vectors in the 'basis' are indispensable to span the linear space. The base vectors form the axis of a plane, where each vector in the plane can be represented by its magnitude and direction with respect to the origin. On one hand, vectors in the plane can be grouped together according to certain criterion to make clusters, i.e., a group of i or more vectors with similar directions and lengths. On the other hand, vectors that do not belong to any group might also represent a unique property of the data. In contrast, those vectors with "relatively short length", either measured by vector norm or in terms of individual element value, can be deemed as having insignificant inference on data utility and hence can be considered as nonessential features. Correspondingly, We define the nonessential features (ε_i, A) as:

Definition 1: A column vector of an $m \times n$ dataset $A = [\underline{a}_1, \underline{a}_2, \ldots, \underline{a}_n]$ is a nonessential feature (ε_i, A), if there exists a small number ϕ (Threshold) such that $(\varepsilon_i, A) = \{\underline{a}_i \mid \underline{a}_i \in [\underline{a}_1, \underline{a}_2, \ldots, \underline{a}_n] \wedge A_{ji} < \phi, \forall j = 1, 2, \ldots, m\}$.

Note that we used the term "relatively short length" to describe nonessential features because, as it will be illustrated in the following sections that not all vectors with small entries can be regarded as nonessential. Sometimes, vectors having small magnitudes measured by vector norms can be as important to the data utility as those with large magnitudes.

2) Identify and remove nonessential features: As our objective is to identify and eliminate the nonessential features, we propose an approach that combines sparsification procedure and SVD technique. Sparsifying a matrix or a vector refers to a procedure that sets the entries to zero with a threshold value. To see how sparsified SVD can be used to identify nonessential features, we look at the following statement.

Statement 2.1: Applying SVD to a given data A to obtain three factor matrices which are then sparsified and multiplied to produce a perturbed data $\tilde{A}$ can help identify nonessential features.

To see the Statement 2.1 is true, let data A be a $n \times m$ matrix with rank r where $m \geq n$ and $A = UDV^T$ by SVD. Then $U_r = [\underline{u}_1, \underline{u}_2, \ldots, \underline{u}_r]$ forms an orthonormal basis for $R(A)$. Similarly, it can be proved that the decomposed matrix V with SVD is also an orthonormal basis for A [9], [6]. Let $U \in R^{m\times m}$, $V \in R^{n\times r}$ and $D \in R^{m\times r}$,

$$D = \begin{bmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_r \\ \vdots & & \vdots \\ 0 & \cdots & 0 \end{bmatrix}$$

it follows that

$$A = [\underline{u}_1, \underline{u}_2, \ldots, \underline{u}_n] \begin{bmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_r \\ \vdots & & \vdots \\ 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} \underline{v}_1^T \\ \underline{v}_2^T \\ \vdots \\ \underline{v}_r^T \end{bmatrix}$$

$$= [\underline{u}_1, \underline{u}_2, \ldots, \underline{u}_n] \begin{bmatrix} \sigma_1 \underline{v}_1^T \\ \vdots \\ \sigma_r \underline{v}_r^T \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

. If we set $\sigma_j = 0$, $j = k+1, \ldots, r$, then we get $\overline{A}$:

$$\overline{A} = \left[\underline{u}_1, \underline{u}_2, \ldots, \underline{u}_k\right] \begin{bmatrix} \sigma_1 \underline{v}_1^T \\ \sigma_2 \underline{v}_2^T \\ \vdots \\ \sigma_k \underline{v}_k^T \end{bmatrix}$$

or equivalently,

$$\overline{A} = \sum_{i=1}^{k} \underline{u}_i \sigma_i \underline{v}_i^T = U \times \widetilde{D} \times V^T.$$

Since $A_i = \underline{u}_i \sigma_i \underline{v}_i^T (\forall i \in [1, r])$, $\overline{A}$ can be represented as:

$$\overline{A} = A_1 + A_2 + , \ldots + A_k.$$

The difference between $\overline{A}$ and original matrix A is:

$$A - \overline{A} = A_{k+1} + A_{k+2} + , \ldots + A_r = \sum_{i=k+1}^{r} A_i.$$

If we use Frobenius norm to map a matrix into a number, then $\|A_i\|_F = \sigma_i \left\|\underline{u}_i \underline{v}_i^T\right\|_F$, $i \in [1, k]$. Since $\sigma_1 \geq \sigma_2 \geq , \ldots, \geq \sigma_r$ by definition, it follows that:

$$\left\|\overline{A}\right\|_F \geq \left\| \sum_{i=1}^{r} \underline{u}_i \sigma_i \underline{v}_i^T - \sum_{j=1}^{k} \underline{u}_j \sigma_j \underline{v}_j^T \right\|_F$$

for some $k \in [1, r]$.

Since, in the data mining applications, $\sum_{i=k+1}^{r} A_i$ can be considered as noise [10], therefore $\overline{A}$ can be thought as an approximation of A with less noise information. To further surface the nonessential vectors (features), a sparsification process is applied to matrices U and V to obtain $\widetilde{U}$ and $\widetilde{V}$ using a threshold ϕ such that values less than ϕ are set to zero. We then multiply all three sparsified matrices $\widetilde{U}$, $\widetilde{D}$ and $\widetilde{V}$ to get $\widetilde{A}$:

$$\widetilde{A} = \widetilde{U} \widetilde{D} \widetilde{V}^T.$$

If we define S as the sparsification function, then it is easy to see that:

$$\widetilde{A} = S(\overline{A}).$$

And since after sparfication process, many small column vectors in U and V are dropped to zero:

$$\begin{aligned} \widetilde{A} &= S(A_1) + S(A_2) + , \ldots, + S(A_k) \\ &= S(\underline{u}_1 \sigma_1 \underline{v}_1^T) + S(\underline{u}_2 \sigma_2 \underline{v}_2^T) + , \ldots, + S(\underline{u}_k \sigma_k \underline{v}_k^T) = \\ &\sigma_1 S(\underline{u}_1) S(\underline{v}_1^T) + \sigma_2 S(\underline{u}_2) S(\underline{v}_2^T) + , \ldots, + \sigma_k S(\underline{u}_k) S(\underline{v}_k^T) \\ &= S(A_1) + , \ldots, + 0 + S(A_j) + , \ldots, + 0 + , \ldots, + S(A_k). \end{aligned}$$

As the result, these zero vectors can be seen as nonessential features and easily determined, which would otherwise be difficult to identify in the original data set.

Our empirical results show that after applying the above described procedure to a given data A, a set of "small" column vectors measured by 2-norm such that $\|\underline{a}_i^{'}\|_2 \cong 0$ can be easily distinguished from other vectors in the resulting matrix $\widetilde{A} = [\underline{a}_1^{'}, \underline{a}_2^{'}, \ldots, \underline{a}_n^{'}]$. These vectors can be considered as nonessential features by definition and removed from the original data to obtain a new data matrix with smaller feature dimension.

C. Geometric illustration

We name the process of applying the sparsified SVD for eliminating nonessentials S2R, short for sparsified rank reduction process. In this section, we strive to use vector space concept to geometrically interpret how S2R identifies nonessential features. Perhaps, the most elementary vector space is the Euclidean vector space R^n, $n = 1, 2, \ldots$ [11]. For simplicity, we only consider R^2. Given a vector $\underline{v}$ in R^2,

$$\underline{v} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}. \quad (1)$$

we can associate it with a direction and magnitude in a 2-dimensional plane. The direction is defined with respect to the origin (0,0), and the length is defined by Euclidean 2-norm:

$$\|\underline{v}\|_2 = \sqrt{x_1^2 + x_2^2}.$$

In order for a better illustration and analysis, we define a definition of cluster $(\digamma_k, A)$:

Definition 2: A group of i or more column vectors of an $m \times n$ dataset $A = [\underline{a}_1, \underline{a}_2, \ldots, \underline{a}_n]$ is a cluster $(\digamma_n, A)$, if every vector in the group can be represented with similar magnitude and direction in a m-dimensional plane.

For example, $(\digamma_q, A)$ is a cluster containing q or more vectors with similar magnitudes and directions defined in a vector space. Naturally, those vectors that do not belong to any cluster are considered as "outliers". With the concepts of the cluster and outlier, we illustrate nonessential vectors in Figures 2 and 3 where, without loss of generality, only one cluster is involved. In general, short vectors in terms of Euclidean norm can be considered as nonessentials. For example, vectors $\underline{v}_5$ and $\underline{v}_6$ (Figure 2) are considered as nonessentials, since they are the two short vectors in contrast to other vectors. $\underline{v}_7$ does not belong to the cluster since it is in different direction compared to the vectors in cluster $(\digamma_4, A) = \{\underline{v}_1, \underline{v}_2, \underline{v}_3, \underline{v}_4\}$.

Furthermore, it can be shown that nonessential vectors can be produced with other short nonessentials by applying linear combination operations. As an illustration, a short vector $\underline{v}_6$ is resulted from the linear combination of two short vectors: $\underline{u}_4$ and $\underline{u}_5$ (right picture in Figure 3). By comparison, vector $\underline{v}_3$ as a result of the linear combination of $\underline{u}_1$ and $\underline{u}_2$ is considered essential for its larger magnitude (left picture in Figure 3).

D. Restrictions and limitations

Although it has been illustrated that nonessential vectors may be resulted from linearly combination of short base vectors, we can not just simply toss away all the vectors with small magnitude when determining nonessential features. As mentioned in the previous sections, the outliers may also hold great inference on the mining algorithm's behaviors. Those outliers can be of small magnitude and could be members of base that

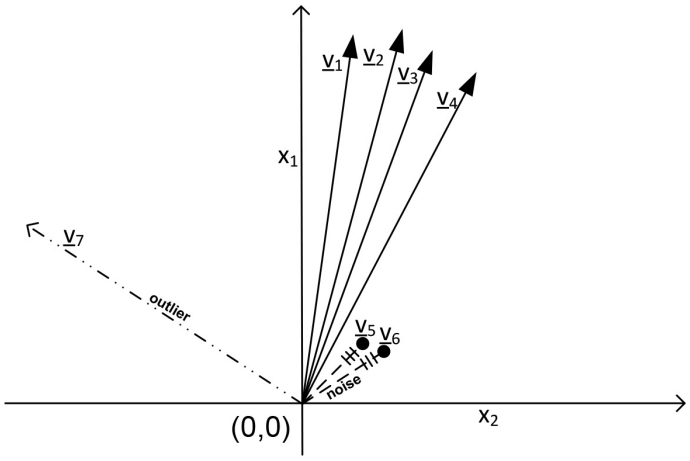

Fig. 2. Illustration of nonessentials and outliers in a 2-D plane.

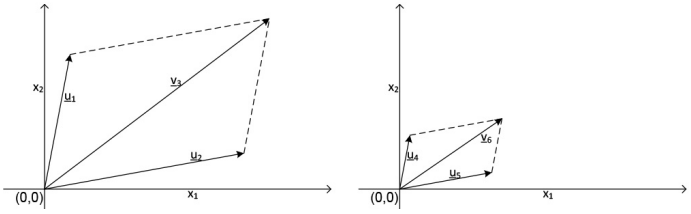

Fig. 3. Two vectors $\underline{v}_3$ (left) and $\underline{v}_6$ (right) are resulted by spanning other vectors ($\underline{u}_1$,$\underline{u}_2$) and ($\underline{u}_4$,$\underline{u}_5$), respectively.

spin to produce essential features. An example is illustrated in Figure 4 which describes the transpose of a 2×8 matrix A in a two dimensional plane. Vector set $\{\underline{u}_1, \underline{u}_2, \underline{u}_3, \underline{u}_4, \underline{u}_5\}$ forms a basis of A, and vectors $\{\underline{v}_1, \underline{v}_2, \underline{v}_3\}$ are linearly dependent on the base set. Let vector set $\{\underline{v}_1, \underline{v}_2, \underline{u}_2, \underline{u}_4\}$ be considered as essentials and form a cluster (F_4, A) based on the definition, then vector set $\{\underline{v}_3, \underline{u}_1, \underline{u}_3, \underline{u}_5\}$ can be treated as outliers. Clearly, we can not throw away the short base vectors $\underline{u}_3$ and $\underline{u}_5$, because they are used to produce $\underline{v}_2$ and $\underline{v}_3$.

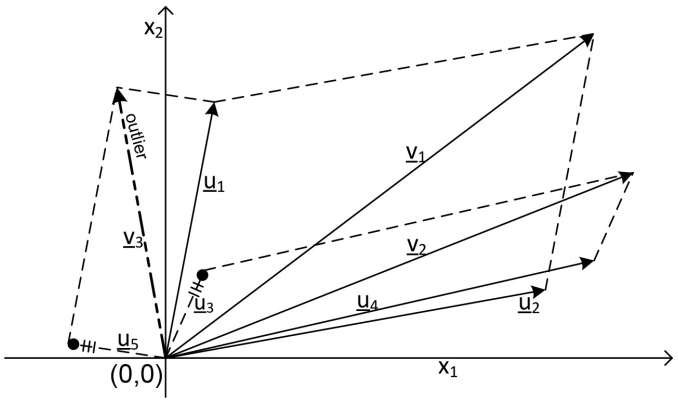

Fig. 4. Spanning outliers.

One immediate intuition from the above example is that those outlier vectors tend to have unbalanced shapes, i.e., one portion of elements in a vector may be extremely larger than the other portion. For example, the projection of $\underline{u}_3$ on x_2 is greater than that on x_1. Accordingly, when trying to separate essential outliers from nonessentials, we could look at the value distribution of entries in a vector. However, in practice, this can not be used as golden rule to distinguish exactly between essential outliers and nonessential vectors. Nevertheless, an effort is made in this work to identify the true nonessential features. Specifically, our strategy is to develop a threshold function that is able to produce customized threshold values according to individual element for the vectors being evaluated.

Furthermore, we should not expect S2R being able to identify all the nonessentials. As an example illustrated in Figure 5, two essential base vectors $\underline{u}_5$ and $\underline{u}_6$ in two almost completely opposite directions in the plane can also produce nonessentials vector $\underline{v}_6$, as opposed to examples illustrated in Figure 3. Clearly, it is hard to identify base vectors that produce $\underline{v}_6$ with S2R. In such a situation, we can rely on the ordinary feature selection methods to identify the remaining nonessentials. We will show, in Section 5, that after applying the proposed S2R approach, the work load for feature search algorithm is significantly reduced whereas the probability of obtaining optimal feature subset is increased.

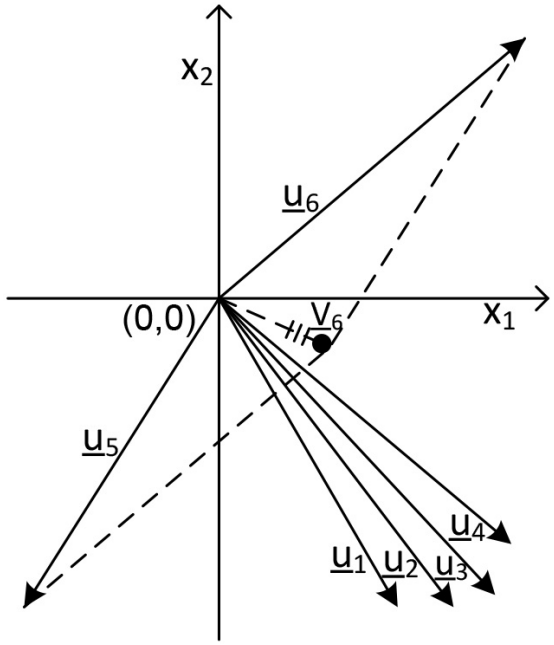

Fig. 5. Linear spanning nonessentials by essential base vectors.

III. Sparsification strategy

A Sparsification procedure evaluates elements in each column vector of a given matrix with a threshold value ϕ. Elements with values less than ϕ are set to zero. In this work, we define a smooth threshold function similar to the one used in [12], which uses an exponential function in which the threshold value is calculated differently for each column

of the matrix:

$$T_j = \frac{\epsilon}{m} \sum_{i=1}^{m} |a_{ij}| \, e^{j \cdot k^{-2}} \tag{2}$$

where k is the number of the singular values to keep. The computed threshold value for each column is adjustable with a positive scaling factor ϵ, which works similar to the parameter λ in the L1-Regularization, such that smaller threshold values are obtained with lower ϵ values. In the implementation, we set $\epsilon = 0.6$ as its default value for all the test cases. Other practitioners can try different settings that may increase or decrease the sparseness of the resulted matrix. Note that different from the function defined in [12], the absolute value of a_{ij} is computed. This is because SVD process may result in some negative entries in the decomposed matrices U and V. If we add up their entry values directly (without absolute operation), the threshold value calculated may be larger for essential vectors and smaller for nonessential vectors. Therefore, taking absolute value is necessary so that larger value entries would remain whereas smaller value entries would reduce to zero after the sparsification process. There are other sparsification approaches available in the literature such that L1-Regularization [13], etc. However, most of them sparsify a matrix by multiplying matrix entries with a fractional value. In contrast, the smooth threshold function calculates different threshold values which are customized for the column vectors being considered. Intuitively, one would expect more properly sparsified matrices using the smooth threshold function in comparison to that of other sparsification methods.

IV. Analyzing computing complexity

A. S2R procedure

Now, we are ready to summarize the S2R, as described in Procedure 1. The dataset A which consists of large number of features is treated as a matrix and is decomposed into three matrices by the SVD process. The factored matrices are then sparsified and used to compose a new dataset $\widetilde{A}$ with perturbed entry values which helps identify nonessential features. After removing the identified nonessentials, we obtain a new matrix B with fewer number of features, which is expected to facilitate any feature selection algorithm in terms of improved result quality and reduced computing time.

Procedure 1 Combine Feature Selection with S2R

Input: $m \times n$ DataSet A

Ensure: *Numerical Data Type*

1: Decompose A using *SVD*, get $A = UDV^T$
2: Reducing rank of A, get $\widetilde{D}$
3: Sparsify the decomposed matrices, get $\widetilde{U}$, $\widetilde{V}$
4: Compose a new matrix $\widetilde{A} = \widetilde{U}\widetilde{D}\widetilde{V}^T$
5: Label the features (vectors) with norm less then some small number ϵ in $\widetilde{A}$
6: Remove labeled features from the original dataset A, get a $m \times k$ dataset B, where $k \leq n$
7: Apply ordinary feature selection algorithms on B

B. Computing complexity

Procedure 1 takes an $m \times n$ dataset A as input. In step one, the input dataset A is decomposed into 3 matrices U,D, and V by SVD. The total cost for SVD decomposition is $O(n^2m)$. In step two, the rank of A is reduced by dropping small elements on the diagonal of D to zero, which takes a constant cost C^1. In step three, the decomposed matrices U and V are sparsifed with the threshold function described in Section 3. The total cost for the sparsification process is no larger than $2m^2$ and hence can be estimated as $O(m^2)$. In step four, a perturbed dataset $\widetilde{A}$ is composed with the sparsified matrices $\widetilde{U}$, $\widetilde{D}$ and $\widetilde{V}$. Since the running time for multiplication of two $n \times n$ square matrices, if carried out naively, is $O(n^3)$, the cost for composition process of $\widetilde{A}$ is then bounded by $2m^3$, with an assumption that $m \geq n$. In steps five and six, the nonessentials that are manifested in $\widetilde{A}$ as the result of the S2R process are labeled and removed from the original dataset A to give a new matrix B. Assuming the cost for the final two steps is a constant factor C^2, then the total cost can be expressed as:

$$O(n^2m) + C^1 + O(m^2) + O(m^3) + C^2 \tag{3}$$

Since $m \geq n$ by assumption (which is usually true for large datasets), the cost as summarized in equation 3 is asymptotically-bounded by $O(2m^3)$.

V. Experiments and results

A. Setup and dataset

The algorithms and procedures proposed and used in this work were implemented using Oracle Java 7 with Java Runtime Environment JRE 1.7.0.45 (32-bit) and the programs were executed on a Dual Intel E5-2670 8 Core (16 processes), 2.6 GHz with 64 GB of 1600 MHz RAM computer. The tested data sets were first processed to obtain S2R data set. We then applied various feature selection techniques to both original and S2R data sets to compare the performance. Specifically, we applied probabilistic, heuristic and exhaustive feature selection methods. The experiments were conducted on "Wisconsin Breast Cancer (Diagnostic)" data set and Connectionist Bench (Sonar, Mines vs. Rocks) dataset [14], [15]. The Wisconsin Breast Cancer dataset has 32 features including diagnosis, texture, smoothness, concavity, concave points, fractal dimension, etc. These features are computed from a digitized image of a fine needle aspirate (*FNA*) of a breast mass. They described characteristics of the cell nuclei present in the image. The target feature is Diagnosis: "B" = benign, "M" = malignant. The dimension of the data matrix is 569×32. Connectionist Bench dataset has 60 features and 208 instances. This dataset contains patterns obtained by bouncing sonar signals off a metal cylinder or rocks at various angles and under different conditions. Each pattern is a set of 60 numbers in the range from 0.0 to 1.0, which represents the energy within a particular frequency band integrated over a certain period of time. For values of the target feature, the label associated with each record is letter "R" if the object is rock and "M" if it is a metal cylinder.

The sparsification procedure was set differently for the three factor matrices of the decomposed data sets. For the matrix D where singular values σ_i are on its diagonal entries, we used the following function:

$$\sigma_i = \begin{cases} \sigma_i & \text{if } \sigma_i > 1 \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

, where singular values greater than one are kept and the rest are set to zero. For matrices U and V, the smooth threshold function 2 was applied to compute threshold value ϕ for each column vector. The scaling parameter ϵ was set to 0.6. The absolute values in U and V less than ϕ are reduced to zero. Note that the optimal number of singular values and the best scaling parameters to be set vary with the dataset and often depend on specific requirements. To be consistent, both datasets were sparsified using the same parameter setting.

B. Experiment 1

In this experiment, we implemented an Exhaustive Search that enumerates complete feature subset space. Each feature subset is evaluated with the information gain. Since the Exhaustive search takes exponentially long computing time, we set maximum running time to 20 hours on the original data and 40 minutes on the S2R data. Our objective is to test if Exhaustive Search would perform the same or even better on S2R data within a less computing time compared to that of original data. First, S2R process was applied to the original data sets of which the number of features reduced from 31 to 26 for WBC data and from 60 to 48 for Sonar data, resulting in 20.00% and 16.12% feature size reduction respectively (Table I). The experimental results show that Exhaustive Search yielded the same result for both original and S2R datasets with 12 features selected for WBC data and 16 features selected for Sonar data. Furthermore for WBC data, the computing time is significantly reduced with S2R data compared to that with the original data (15 minus vs 1.48 hours). For Sonar data, Exhaustive Search shopped at the maximum allowable running time for both original and S2R data sets. However, with considerably less running time, Exhaustive Search was able to reduce the same number of features from S2R data. This demonstrates that applying Exhaustive Search on the S2R data can produce equally good feature selection results with reduced computing time compared to applying it on the original data.

TABLE I
EXPERIMENTAL RESULTS FOR APPLYING EXHAUSTIVE SEARCH ON ORIGINAL AND S2R DATASETS

	Feature Size		Total Cost	
DataSet:	**WBC**	**Sonar**	**WBC**	**Sonar**
Original	31	60	2^{31}	2^{60}
S2R	26	48	$2 \times 569^3 + 2^{26}$	$2 \times 208^3 + 2^{48}$
Reduction Rate	16.12%	20.00%		
$ExhaustiveOnOriginal$	12	16	1.48 (hour)	20 (hour)
$ExhaustiveOnS2R$	12	16	15 (minute)	40(minute)

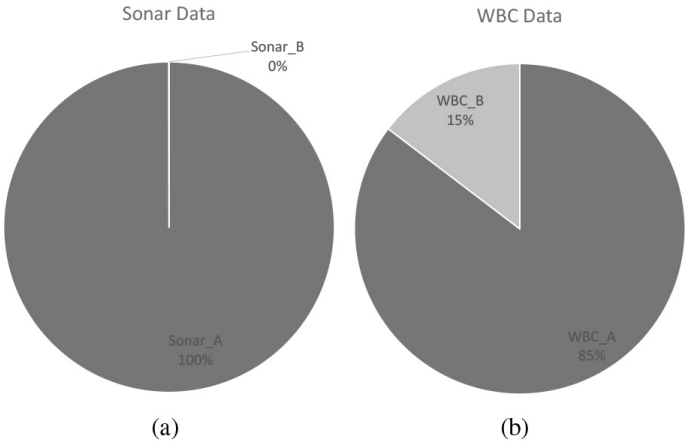

Fig. 6. Cost Comparisons between original and preprocessed Sonar and WBC data sets.

In addition, Table I also lists the number of feature subset that will be evaluated by Exhaustive Search. Theoretically, for an $m \times n$ dataset, the cost of Exhaustive Search on original dataset is estimated as evaluating complete feature space 2^n, whereas the cost for the S2R data is bounded by $2m^3 + 2^n$, which is summation of the costs for S2R process and Exhaustive Search on the S2R data. Filling in the numbers, we get that Exhaustive Search will evaluate 2^{31} feature subsets for original WBC data and 2^{60} feature subsets for Sonar data, and the number of feature subsets to be evaluated for the corresponding S2R datasets are $2 \times 569^3 + 2^{26}$ and $2 \times 208^3 + 2^{48}$. To give a more concrete idea for the differences between these values, we illustrate their proportions using two pie charts (Figure 6). As we can see, Exhaustive Search on S2R data only has to evaluate about 15% of the complete feature subsets for the original WBC data. In the extreme case where the data consists of large feature space and relatively small number of records such as Sonar data, the number of feature subsets to be evaluated for original data greatly dominates that for S2R data, making the cost proportion of S2R data insignificant and close to zero percent. In fact, our experimental results have shown that the cost for S2R can be neglected in that, including the SVD matrix decomposition, it only took 46 milliseconds for WBC data and 49 milliseconds for Sonar data.

C. Experiment 2

In this experiment, we applied a Probabilistic Search named Las Vegas Filter (LVF) [16], which is a specification of the Las Vegas algorithm family [17]. It randomly generates feature subsets with equal probability, and evaluates the feature subset with an evaluation function. LVF stops generating new feature subset after a user defined number of iterations. The objective of this experiment is to test if a random feature search procedure would improve its performance on the S2R data. The correlation-based Feature Selector (CFS) [18] is used to assess the worthiness of a feature subset by considering the individual predictive ability of each feature along with

the degree of redundancy between them, i.e., the bias of the evaluation function is toward subsets that contain features that are highly correlated with the class and uncorrelated with each other. CFS's feature subset evaluation function is shown below:

$$M_S = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}} \quad (5)$$

where M_S is the heuristic merit of a feature subset S containing k features, $\overline{r_{cf}}$ is the mean feature-class correlation, and $\overline{r_{ff}}$ is the average feature-feature inter-correlation. Higher value of M_S indicates better predictive power of a feature subset.

We run LVF algorithm using CFS as evaluator for 14 times with different stop settings (number of iterations before LVF stops) on both original and S2R-resulted WBC and Sonar datasets. The merit measured by CFS for the best feature subset selected for each run were recored.

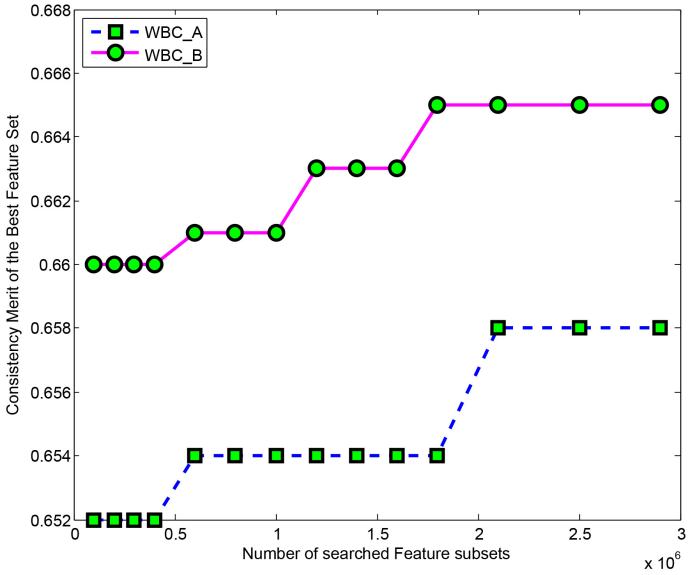

Fig. 7. Correlation merits comparison between original and S2R WBC Data.

We can observe from the results shown in Figure 7 and 8 that, as expected, feature subsets resulted from larger number of iteration settings yield higher CFS values than that from lower number iteration settings. For both WBC and Sonar datasets, CFS values are higher for the S2R data than for the original data for all stop settings, which indicates the utility of original data was well preserved in S2R data and consequently resulted in better feature selection results since the probability of selecting better feature subset by a stochastic algorithm is higher for data with smaller feature dimensionality. Furthermore, the CFS value stops increasing at the sixth iteration number setting for applying LVF on the S2R Sonar data whereas the CFS value for original data continues to increase as the number of iterations to stop is set higher (Figure 8). This indicates potential computing time savings by S2R since fewer number of iterations are required to converge.

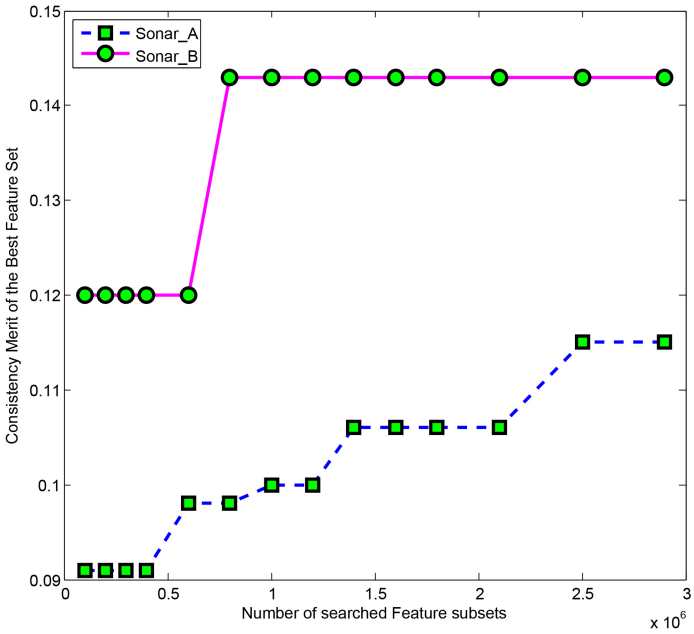

Fig. 8. Correlation merits comparison between original and S2R Sonar Data.

D. Experiment 3

In this experiment, data utility preserved in S2R data is further investigated with heuristic feature search strategies in which the accuracy is often traded for the efficiency. A Backward Sequential Search (BestFirst) algorithm and a Genetic algorithm (GA) were applied. The Sequential Search algorithm searches the space of feature subsets by greedy hill-climbing augmented with a backtracking facility. It starts with the full set of features and searches backward by considering all possible single feature deletions at a given point [19]. Genetic algorithm performs a randomized search and has shown to be less susceptible for being stuck in a local optimal solution [20] as opposed to other heuristic methods that would otherwise produce a result prematurely [1]. In the experiment, we used the well known data mining software Weka [21] which integrates both the Backward Sequential Search and Genetic algorithm. The default parameter settings are used for both heuristic algorithms and the feature subsets were evaluated by the CFS which was used in the experiment 2.

TABLE II
NUMBER OF FEATURES SELECTED USING HEURISTIC FEATURE SELECTION ALGORITHMS FOR ORIGINAL AND S2R DATA

	# of Selected Features			
DataSet:	**Sonar_A**	**Sonar_B**	**WBC_A**	**WBC_B**
SequentialSearch	19	16	12	12
RandomSearch	13	15	13	18

The results show difference in sizes of selected feature sets obtained by Sequential Search and GA in Table II where WBC_A and Sonar_A denote for original data, WBC_B and Sonar_B denote for the resulting data by S2R. Number of

TABLE III
SVM PREDICT RATE (%) FOR ORIGINAL AND S2R DATA

	Correct Predict Rate (Percentage)			
DataSet:	**Sonar_A**	**Sonar_B**	**WBC_A**	**WBC_B**
$SequentialSearch$	77.41	78.37	96.66	96.66
$RandomSearch$	75.58	79.33	96.49	97.89

features selected by Sequential Search for the original data is slightly more than that for the S2R data (19 vs 16 for Sonar, 12 vs 12 for WBC), while GA selects fewer number of features from the original data compared to the S2R data (13 vs 15 for Sonar, 13 vs 18 for WBC). This indicates that different feature subsets were selected between original data and S2R data. The selected feature subsets were then evaluated using support vector machine (SVM) [22]. Ten folds cross validation is set to split the data in 10 approximately equal parts $D_1, \ldots, D_{10}$. Training set D_i^t is obtained by removing part of D_i from D. Feature subsets with the higher predict rate are considered better. Table III shows the predict rate is the same for feature subset obtained by Sequential Search for WBC data (96.66% vs 96.66%), whereas the results are slightly better for S2R in other cases. This indicates there may be more than one feature subset that could produce same predict rate and better feature selection results can be obtained with the S2R data where the nonessential features were removed a priori.

VI. CONCLUSION & FUTURE WORKS

In this paper, a data dimensionality reduction approach that we call S2R is developed by using sparsified SVD matrix decomposition technique. We investigated how S2R identifies nonessential features and analyzed its computing time complexity. The identified nonessential features are removed from the original data, which forms a new data with smaller feature dimension and facilities to improve performance of feature selection algorithms. We illustrated S2R using the 2-D plane and addressed its limitations. The empirical results showed, when compared to the cost of a feature selection procedure, the extra cost of S2R is a small price to pay for data with large feature size. Although the cost of S2R, which is $O(2m^2)$, can not be neglected, the reduced feature space, as a result, also mitigates work load for the conducted data analysis. In conclusion, S2R procedure can significantly reduce the computational cost inflicted by large scale datasets on feature search related data mining process without compromise of solution quality. In addition, although we used S2R to improve feature selection performance, it could be extended to any data mining algorithm.

The future works should focus on testing the performance of S2R with different parameter settings on more larger data sets and make comparisons with other feature selection techniques and existing feature dimension reduction techniques such as Principal Component Analysis (PCA). In addition, we would like to explore the data utility preserved in the S2R data in terms of different feature selection metrics (apart from the ones used in this work i.e., information gain, SVM and CFS).

REFERENCES

[1] H. Liu and H. Motoda, *Computational methods of feature selection*. Chapman and Hall/CRC, 2007.

[2] G. H. John, R. Kohavi, K. Pfleger *et al.*, "Irrelevant features and the subset selection problem." in *ICML*, vol. 94, 1994, pp. 121–129.

[3] J. Han, M. Kamber, and J. Pei, *Data mining: concepts and techniques*. Morgan kaufmann, 2006.

[4] T. Marill and D. Green, "On the effectiveness of receptors in recognition systems," *Information Theory, IEEE Transactions on*, vol. 9, no. 1, pp. 11–17, 1963.

[5] A. W. Whitney, "A direct method of nonparametric measurement selection," *Computers, IEEE Transactions on*, vol. 100, no. 9, pp. 1100–1103, 1971.

[6] S. Chen, B. Han, L. Li, L. Zhu, H. Lai, and Q. Dai, "Svd based monte carlo approach to feature selection for early ovarian cancer detection," in *Bioinformatics and Biomedical Engineering (iCBBE), 2010 4th International Conference on*. IEEE, 2010, pp. 1–4.

[7] F. Fallucchi and F. M. Zanzotto, "Singular value decomposition for feature selection in taxonomy learning," in *Proceedings of the International Conference RANLP-2009*. Association for Computational Linguistics, 2009, pp. 82–87.

[8] R. D. Phillips, L. T. Watson, R. H. Wynne, and C. E. Blinn, "Feature reduction using a singular value decomposition for the iterative guided spectral class rejection hybrid classifier," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 64, no. 1, pp. 107–116, 2009.

[9] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU Press, 2012, vol. 3.

[10] M. W. Berry, Z. Drmac, and E. R. Jessup, "Matrices, vector spaces, and information retrieval," *SIAM review*, vol. 41, no. 2, pp. 335–362, 1999.

[11] G. Williams, *Linear algebra with applications*. Jones & Bartlett Publishers, 2012.

[12] J. Gao and J. Zhang, "Sparsification strategies in latent semantic indexing," in *Proceedings of the 2003 Text Mining Workshop*, 2003, pp. 93–103.

[13] A. Y. Ng, "Feature selection, l 1 vs. l 2 regularization, and rotational invariance," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 78.

[14] W. H. Wolberg and O. L. Mangasarian, "Multisurface method of pattern separation for medical diagnosis applied to breast cytology." *Proceedings of the national academy of sciences*, vol. 87, no. 23, pp. 9193–9196, 1990.

[15] R. P. Gorman and T. J. Sejnowski, "Analysis of hidden units in a layered network trained to classify sonar targets," *Neural networks*, vol. 1, no. 1, pp. 75–89, 1988.

[16] M. Dash and H. Liu, "Consistency-based search in feature selection," *Artificial intelligence*, vol. 151, no. 1, pp. 155–176, 2003.

[17] G. Brassard and P. Bratley, *Fundamentals of algorithmics*. Prentice Hall Englewood Cliffs, 1996, vol. 524.

[18] M. A. Hall, "Correlation-based feature selection for machine learning," Ph.D. dissertation, The University of Waikato, 1999.

[19] D. W. Aha and R. L. Bankert, "A comparative evaluation of sequential feature selection algorithms," in *Learning from Data*. Springer, 1996, pp. 199–206.

[20] J. Jarmulak and S. Craw, "Genetic algorithms for feature selection and weighting," in *In Proceedings of the IJCAI*, vol. 99. Citeseer, 1999, pp. 28–33.

[21] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[22] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, no. 3, pp. 293–300, 1999.