Fast Sparse Representation Approaches for the Classification of High-Dimensional Biological Data

Yifeng Li and Alioune Ngom
School of Computer Science, University of Windsor, Windsor, Ontario, Canada
{li11112c, angom}@uwindsor.ca

***Abstract*—Classifying genomic and proteomic data is very important to predict diseases in a very early stage and investigate signaling pathways. However, this poses many computationally challenging problems, such as curse of dimensionality, noise, redundancy and so on. The principle of sparse representation has been applied to analyzing high-dimensional biological data within the frameworks of clustering, classification, and dimension reduction approaches. However, the existing sparse representation approaches are either inefficient or have the difficulty of kernelization. In this paper, we propose fast active-set-based sparse coding approach and a dictionary learning framework for classifying high-dimensional biological data. We show that they can be easily kernelized. Experimental results show that our approaches are very efficient, and satisfactory accuracy can be obtained compared with existing approaches.**

***Keywords*-sparse coding, dictionary learning, kernel approach, active-set algorithm, classification.**

I. Introduction

In the area of biological and clinical study, huge amount of various data, such as genome-wide microarray data and proteomic mass spectrometry data, have been being produced. These data provide us much richer information which allows us to conduct genome-wide study, and therefore to reach more precise decisions and conclusions than ever before. In this paper we focus on the classification of such data.

Statistical learning and computational intelligence are among the main tools to analyze these data. However, there are many difficulties that preventing an efficient and accurate analysis. For example there are usually tens of thousands of dimensions in microarray gene expression data, while its sample size is usually very small. With this problem, it may be impossible to estimate the parameters of a model, for example the Bayesian classifier, as the number of sufficient samples needed increases exponentially as the number of dimensions. Secondly, the high-dimensional data often have many redundant features when only few (maybe hidden) features correspond to the desired study. This might drown useful information. For example, distance based methods, for example *k-nearest neighbors* (k-NN), suffers from low precision on such data. In addition to the small sample size, the noise present in biological data and uncertainty in target variable often lead to overfitting of some models sensitive to noise and uncertainty, for example decision tree learning.

On one hand, many sophisticated models have been directly applied to classifying high-dimensional data. The most famous family is the basis-expanded linear models: $f(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}^{\mathrm{T}}\phi(\boldsymbol{x}) + b)$ where $\phi(\bullet)$ is a map function and b is bias. Given n training sample $\boldsymbol{D} \in \mathbb{R}^{m\times n}$ and the corresponding class information $\boldsymbol{c}$, its risk minimization can be expressed as $\min_{\boldsymbol{w}} r(\boldsymbol{w}) + C\xi(\boldsymbol{w}, \boldsymbol{D}, \boldsymbol{c})$, where $r(\boldsymbol{w})$ is the regularization term, $\xi(\boldsymbol{w}, \boldsymbol{D}, \boldsymbol{c})$ is loss function, and C is the trade-off parameter. The former term is to increase the generalization of the model, and the later term is to decrease empirical error of classification. Most linear models can be kernelized as long as the optimization and prediction only use the inner products between samples. Kernel trick can make the optimization dimension-free. An appropriate kernel can linearize complex patterns and avoid overfitting. When $r(\boldsymbol{w})$ is the l_2-norm, and the loss function is hinge loss, this is the well-known *support vector machine* (SVM) [1]. If the regularization term is l_1-norm, then it becomes l_1 SVM [2].

On another hand, dimension reduction techniques such as the *principal component analysis* (PCA) and *independent component analysis* (ICA) has been applied to biological data analysis [3], [4]. The basis vectors of PCA are orthogonal, which may be not suitable for biological data analysis, because the hidden patterns may not be orthogonal. ICA can be more suitable as it produces statistically independent basis vectors. However, it is computationally very costly.

Sparse representation (SR) [5] produces non-orthogonal (may redundant) basis vectors, and only very few basis vectors are associated to a sample due to sparsity. Because of this sparse structure, SR has theoretical advantages [6] including i) it increases the capability of associative memories; ii) it describes a signal using explicit structures; iii) it provides a simple representation of complex signal for subsequent processing; iv) it saves energy efficiently; v) it can reduce redundancy as it seeks for a specific set of independent patterns for a specific signal; and vi) it is robust to noise [7]. Therefore SR may be more applicable for biological data analysis than PCA and ICA. Its optimization involves *dictionary learning* (to learn the basis vectors) and *sparse coding* (to learn the sparse loading). Generally speaking, sparsity can regularize a model for better interpretation and decreasing model complexity, for example l_1-SVM. Sparse result can be obtained by many

978-1-4673-2560-8/12/$31.00 ©2012 IEEE

methods such as non-negativity and l_1-norm. In fact, the non-negative SR is the well-known *non-negative matrix factorization* (NMF) [8] and the l_1 SR is one of the most popular sparse models in signal and image processing [5]. As far as we know in this fast-developing direction, there are two *kernel sparse representation* (KSR) approaches. One of these approaches was proposed in [9] and [10]. The approach maps all training samples in a high feature space and then reduces the dimensions to a lower space. Dictionary learning is not consider in this approach. The second approach was proposed in [11]. This approach is inefficient because the dictionary is learned over training samples iteratively. In each iteration, updating the sparse coefficients for a training sample is a sparse coding problem. Therefore many sparse coding problems have to be solved separably. Moreover, the issue encountered in [11] is that the dictionary learned in the feature space is difficult to be represented as the feature space is intractable. SR models have been applied to analyzing biological data and indeed exhibit advantages over PCA and ICA. [12] applied NMF as clustering method to find disease subtypes. [13] employed NMF and high-order NMF to extract feature from two-way and three-way gene expression data. [14] applied NMFs for the inference of transcriptional regulatory network. [15] applied l_1 *least squares* (l_1LS) sparse coding for the classification of gene expression data. [16] provided a method which uses SVD to learn a dictionary and then used l_1LS sparse coding to classify gene expression data of tumor subtypes.

Unfortunately, the optimization of SR is non-convex and is optimized in a block coordinate descent fashion. This is usually very slow. For example, the multiplicative update-rules for NMF and high-order NMF is usually extremely slow to converge. Also, as pointed in [15], the interior-point sparse coding algorithm goes very slow as the sample size increases. The second issue is that the kernel SR has not yet been well-addressed. Moreover, the practical issue is that the non-negative and l_1 SR models have not been experimentally compared with respect to precision and efficiency in the biological community.

In this paper we propose kernel sparse coding and dictionary learning approaches in order to classifying high-dimensional biological data. Although there exists sparse coding approaches applied to classify biological data, this is the first time that SR methods are comprehensively investigated in this field. The contributions of this study are listed in the following:

1) we propose fast kernel sparse coding approaches for direct classification of biological data;
2) we propose an efficient framework of kernel dictionary learning for dimension reduction;
3) we compare the non-negative and l_1 SR models in computational experiments.

The rest of this paper is organized as follow. In Section II, the sparse-coding-based classification approach is introduced and revised active-set algorithms are proposed. Their kernel extension are also given. In Section III, the generic optimization framework of (kernel) dictionary learning is proposed. Then dictionary-learning-based classification is proposed as well. In Section IV, we show the experiment results, including accuracy and computing time. Finally, we draw conclusions and mention future works.

II. Sparse Coding Methods

The l_1LS sparse coding is a two-sided symmetric model which indicates that a coefficient is allowed to be a real number [17]. In the case of a single new sample $\boldsymbol{b} \in \mathbb{R}^m$, it is expressed as:

$$\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1, \tag{1}$$

where $\boldsymbol{A} \in \mathbb{R}^{m\times k}$ is the given dictionary, each column of which is an atom or basis vector, $\boldsymbol{x} \in \mathbb{R}^k$ is sparse coefficient vector. l_1LS sparse coding have been applied for classifying gene expression data in [15]. The main idea is that first training samples are collected in the dictionary, then a new sample is regressed by l_1LS. Thus its sparse coefficient vector is obtained. After that, the regression residual of this sample to each class is computed, and this sample is assigned to the class with the minimal residual.

We generalize this methodology in the way that the sparse code can be obtained by many other regularization and constraints. For example, we can pool all training samples in a dictionary and learn the non-negative coefficient vectors of a new sample, which is formulated as an one-sided model:

$$\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \text{ s.t. } \boldsymbol{x} \geq 0. \tag{2}$$

We called this model *non-negative least squares* (NNLS) sparse coding. Inspired by many sparse NMFs, l_1-regularization can be additionally taken to produce more sparse coefficients. The combination of l_1-regularization and non-negativity results in the l_1NNLS sparse coding model as formulated below

$$\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1 \text{ s.t. } \boldsymbol{x} \geq 0. \tag{3}$$

Now, we give the generalized sparse-coding-based classification approach in details. The method is depicted in Algorithm 1. We shall give the optimization algorithms, latter, required in the first step. The MAX rule mentioned in Algorithm 1 is inspired by the usual way of using NMF as clustering method. For a new sample, it selects the maximal coefficient in the coefficient vector, and then assigns the class label of the corresponding training sample to this new sample. Essentially, this rule is equivalent to apply nearest neighbor classifier in the column space of the training samples. As noise increases, the sparsity could decrease. there may not be dominantly large coefficient. If top coefficients are from different classes, incorrect decision might be made by the MAX rule. The *nearest subspace* (NS) *rule* is proposed by [18] to interpret the sparse coding. NS rule takes the advantage of the property of discrimination of sparse coefficients, and is more robust to noise than the MAX rule. Suppose there are C classes with labels $l_1, \cdots, l_C$. For a given new sample $\boldsymbol{b}$, after obtaining its coefficient vector $\boldsymbol{x}$, the regression residual corresponding to the $i-$th class is computed as $r_i(\boldsymbol{b}) = \|\boldsymbol{b} - \boldsymbol{A}\delta_i(\boldsymbol{x})\|_2^2$, where $\boldsymbol{\delta}_i(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}^n$ returns the coefficients for class l_i. Its $j-$th element is defined by $(\delta_i(\boldsymbol{x}))_j = \begin{cases} x_j & \text{if } \boldsymbol{a}_j \text{ in class } l_i, \\ 0 & \text{otherwise.} \end{cases}$ Finally, class label l_j is assigned to $\boldsymbol{b}$, where $j = \min_{1 \leq i \leq C} r_i(\boldsymbol{b})$.

Algorithm 1 *Sparse-Coding-Based Classification*

```
Input: $A_{m \times n}$: $n$ training samples, $c$: class labels, $B_{m \times p}$:
    $p$ new samples
Output: $p$: predicted class labels of the $p$ new samples
    1) normalize each sample to have unit $l_2$-norm.
    2) get the sparse coefficient matrix $X$, of the new
       samples by solving Equation 1, 2, or 3.
    3) Use a sparse interpreter to predict the class labels of
       new samples, e.g. the MAX or NS rule.
```

A. Optimization

The problem in Equation 1 is equivalent to

$$\min_{\boldsymbol{x},\boldsymbol{u}} \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \boldsymbol{\lambda}^{\mathrm{T}}\boldsymbol{u} \text{ s.t. } -\boldsymbol{u} \leq \boldsymbol{x} \leq \boldsymbol{u}, \tag{4}$$

where $\boldsymbol{u}$ is auxiliary variable to squeeze $\boldsymbol{x}$ towards zero. This can be formulated as standard constrained *quadratic programming* (QP) problem. We hence denote this problem as l_1QP. A general active-set algorithm for constrained QP is provided in [19]. Taking into account the sparse structure of l_1QP, we revised the algorithm for more efficiency. Due to page limitation, we do not detail this modification here.

The NNLS and l_1NNLS sparse coding can be rewritten as *non-negative quadratic programming* (NNQP) problem. It is easier to solve than l_1QP. Our algorithm is obtained via generalizing the active-set algorithm of NNLS [20]. The warm-start point is initialized by the solution to the unconstrained QP. The algorithm keeps adding and dropping constraints in the working set until the true active set is found. Again, for page limitation, we decide to omit details.

If we want to classifying multiple new samples, the initial idea in [18] is to optimize the sparse coding one sample at a time. We adapted both algorithms to solve multiple l_1QP and NNQP. The single active-set algorithms can be solved in parallel by sharing the computation of matrix inverses. At each iteration single problems having the same active set have the same or similar systems of linear equations to solve. These systems of linear equations can be solved once only. For a large value p, active-set algorithms hence have dramatic computational advantage over interior-point [21] and proximal [22] methods.

B. Kernel Approaches

As the optimization of QP only requires inner products of the samples instead of the original data, our active-set algorithms can be naturally extended to solve the kernel sparse coding problem by replacing inner products with kernel matrices. The classification approach in Algorithm 1 is thus extended to kernel version. For narrative convenience, we also denote the classification approaches using l_1LS, NNLS, and l_1NNLS sparse coding as l_1LS, NNLS, and l_1NNLS, respectively. Prefix "K" is used for kernel version.

III. Dictionary Learning Methods

We pursue our dictionary-learning-based approach for biological data based on the following two reasons. First, since sparse-coding-only approach is lazy learning, the optimization can be slow for large training set.Therefore, learning a concise dictionary is more efficient for future real-time applications. Second, dictionary learning might capture hidden independent key factors, the classification performance might hence be improved. Suppose $\boldsymbol{D}_{m \times n}$ is the data of n training samples, and the dictionary $\boldsymbol{A}$ to be learned has k atoms, our dictionary learning models of l_1LS, NNLS, and l_1NNLS are expressed as below, respectively:

$$\min_{\boldsymbol{A},\boldsymbol{Y}} \frac{1}{2}\|\boldsymbol{D} - \boldsymbol{A}\boldsymbol{Y}\|_2^2 + \lambda \sum_{i=1}^{k} \|\boldsymbol{y}_i\|_1 \text{ s.t. } \boldsymbol{a}_i^{\mathrm{T}}\boldsymbol{a}_i = 1,$$

$$\min_{\boldsymbol{A},\boldsymbol{Y}} \frac{1}{2}\|\boldsymbol{D} - \boldsymbol{A}\boldsymbol{Y}\|_2^2 \text{ s.t. } \boldsymbol{a}_i^{\mathrm{T}}\boldsymbol{a}_i = 1; \boldsymbol{Y} \geq 0,$$

and

$$\min_{\boldsymbol{A},\boldsymbol{Y}} \frac{1}{2}\|\boldsymbol{D} - \boldsymbol{A}\boldsymbol{Y}\|_2^2 + \sum_{i=1}^{k} \boldsymbol{\lambda}^{\mathrm{T}}\boldsymbol{y}_i \text{ s.t. } \boldsymbol{a}_i^{\mathrm{T}}\boldsymbol{a}_i = 1; \boldsymbol{Y} \geq 0.$$

Our models above are different from the traditional formulation which enforces $\|\boldsymbol{a}_i\|_2 \leq 1$. We shall show that it is very convenient to extend our models to kernel versions thanks to this modification. We devise block-coordinate-descent-based algorithms for the optimization of the above three models. The main idea is that, in one step, $\boldsymbol{Y}$ is updated while fixing $\boldsymbol{A}$ (a sparse coding procedure); in the next step, $\boldsymbol{Y}$ is fixed, and the unnormalized $\boldsymbol{A}$ is updated via $\boldsymbol{A} = \boldsymbol{D}\boldsymbol{Y}^{\dagger}$, where $\boldsymbol{Y}^{\dagger}$ is pseudoinverse. After that, $\boldsymbol{A}$ is normalized to have unit l_2 norm columns. Note that solving the sparse coding of training samples, while keeping $\boldsymbol{A}$ intact, only needs the inner product $\boldsymbol{R} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$. Therefore, we might learn $\boldsymbol{R}$ instead of $\boldsymbol{A}$. Indeed, we have $\boldsymbol{R} = \boldsymbol{Y}^{\dagger\mathrm{T}}\boldsymbol{D}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{Y}^{\dagger}$. Therefore, updating $\boldsymbol{R}$ only needs the inner product $\boldsymbol{K} = \boldsymbol{D}^{\mathrm{T}}\boldsymbol{D}$ and $\boldsymbol{Y}$. The normalization of $\boldsymbol{R}$ is straightforward. We have $\boldsymbol{R} = \boldsymbol{R}./\sqrt{\mathrm{diag}(\boldsymbol{R})\mathrm{diag}(\boldsymbol{R})^{\mathrm{T}}}$, where $./$ and $\sqrt{\bullet}$ are element-wise operators. Learning inner product $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ instead of $\boldsymbol{A}$ has the benefits of dimension-free computation and kernelization. Due to the above derivation, we have the framework of solving our dictionary learning models as illustrated in Algorithm 2.

Algorithm 2 Dictionary Learning

```
Input: $\boldsymbol{K} = \boldsymbol{D}^{\mathrm{T}}\boldsymbol{D}$, dictionary size $k$, $\lambda$
Output: $\boldsymbol{R} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$, $\boldsymbol{Y}$
initialize $\boldsymbol{Y}$ and $\boldsymbol{R} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ randomly;
$r_{prev} = Inf$; % previous residual
for $i = 1 : maxIter$ do
  update $\boldsymbol{Y}$ by solving the active-set based $l_1$LS, NNLS, or $l_1$NNLS sparse coding algorithms;
  update $\boldsymbol{R} = \boldsymbol{Y}^{\dagger\mathrm{T}}\boldsymbol{D}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{Y}^{\dagger}$;
  normalize $\boldsymbol{R}$ by $\boldsymbol{R} = \boldsymbol{R}./\sqrt{\mathrm{diag}(\boldsymbol{R})d\mathrm{diag}(\boldsymbol{R})^{\mathrm{T}}}$;
  if $i == maxIter$ or $i \mod l = 0$ then
    % check every $l$ iterations
    $r_{cur} = f(\boldsymbol{A}, \boldsymbol{Y})$; % current residual
    if $r_{prev} - r_{cur} \leq \epsilon$ or $r_{cur} \leq \epsilon$ then
      break;
    end if
    $r_{prev} = r_{cur}$;
  end if
end for
```

Algorithm 3 *Dictionary-Learning-Based Classification*

```
Input: $\boldsymbol{D}_{m\times n}$: $n$ training samples, $\boldsymbol{c}$ the class labels,
       $\boldsymbol{B}_{m\times p}$: $p$ new samples, $k$: dictionary size
Output: $\boldsymbol{p}$: the predicted class labels of the $p$ new samples
  training step:
  1: normalize each training sample to have unit $l_2$ norm.
  2: learn dictionary inner product $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ and sparse coefficient matrix $\boldsymbol{Y}$ of training samples by Algorithm 2.
  3: training a classifier using $\boldsymbol{Y}$ (in the feature space spanned by columns of $\boldsymbol{A}$).
  prediction step:
  1: normalize each new sample to have unit $l_2$ norm.
  2: obtain the sparse coefficient matrix $\boldsymbol{X}$ of the new samples by solving Equation 1, 2, or 3.
  3: predict the class labels of $\boldsymbol{X}$ using the classifier learned.
```

Now, we present the generic dictionary-learning-based classification approach in Algorithm 3. The dictionary learning in the training step should be consistent with the sparse coding in the prediction step. As discussed in the previous section, the sparse coding in the prediction step needs the inner products $\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B}$ and $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{B}$ which actually is $\boldsymbol{Y}^{\dagger\mathrm{T}}\boldsymbol{D}^{\mathrm{T}}\boldsymbol{B}$.

A. Kernel Approaches

The l_1LS based kernel dictionary learning and sparse coding are expressed in the following, respectively:

$$\min_{\boldsymbol{A}_\phi, \boldsymbol{Y}} \frac{1}{2}\|\phi(\boldsymbol{D}) - \boldsymbol{A}_\phi\boldsymbol{Y}\|_2^2 + \lambda\|\boldsymbol{Y}\|_1 \text{ s.t. } \mathrm{diag}(\boldsymbol{A}_\phi^{\mathrm{T}}\boldsymbol{A}_\phi) = \boldsymbol{1},$$

$$\min_{\boldsymbol{X}} \frac{1}{2}\|\phi(\boldsymbol{B}) - \boldsymbol{A}_\phi\boldsymbol{X}\|_2^2 + \lambda\|\boldsymbol{X}\|_1,$$

where $\phi(\bullet)$ is a mapping function. The NNLS and l_1NNLS based kernel SR are defined analogically. Recall that the optimizations of the three pairs of linear models, only require inner products of samples. Therefore, they can be easily extended to kernel versions by replacing these inner products by kernel matrices. In this paper, we use prefix "SR" before l_1LS, NNLS, and l_1NNLS to indicate that dictionary learning is involved in SR. We use prefix "KSR" before them to indicate the kernel versions.

IV. Experiments

We tested our approaches over two high-dimensional datasets. One is a collection of microarray gene expressions of five breast tumor subtypes [23]. It includes 158 samples and 13582 genes. Another is a prostate protein mass spectrometry dataset [24] of 15154 features. There are totally 322 samples including 253 normal and 69 cancerous samples. We divided our experiment into two parts. First, we tested the performance of our sparse-coding-based classification approach in the respects of accuracy and computing time. Then we tested our dictionary-learning-based approach.

A. Sparse-Coding Approaches

Instead of learning a dictionary from data, we pool all training samples in the dictionary. Our active-set sparse coding algorithms including l_1LS, NNLS, and l_1NNLS were explored. Both linear and kernel versions were tested. *Radial basis function* (rbf) kernel was used. We compared our algorithms with interior-point [21] and proximal [22] algorithms for l_1LS (denoted by l_1LS-IP and l_1LS-PX), kNN, and SVM using rbf kernel. 4-fold *cross-validation* (CV) was employed. It was run 20 times and the average and standard deviation were recorded. The parameters of these approaches were selected by line or grid search.

The mean accuracies and standard deviations on both datasets are shown in Figure 1, from which we have four observations in the following. First, among all the methods, the highest accuracies over Prostate are obtained by l_1LS and its kernel version, Kl_1LS. This convinces us that sparse-coding approaches are worthy of trying to classify high-dimensional biological data. Second, NNLS and l_1NNLS and their kernel counterparts obtained similar accuracies with SVM on both data. Finally, l_1LS based on our active-set algorithm yields the same accuracies as that based on the existing interior-point and proximal methods on Breast. However, proximal method has the worst accuracy on Prostate. This implies that our active-set sparse coding approaches converge to the global minima as interior-point method, but proximal method might fail on some data.

The average computing time (in second) of CV are illustrated in Figure 2. Logarithm of base two is taken. First, it can be obviously seen that the l_1LS sparse coding using interior-point algorithm is very time-consuming. The efficiency of our active-set-algorithm-based approaches are contributed by the facts that active-set algorithms are usually the fastest algorithms for small and median scale constrained

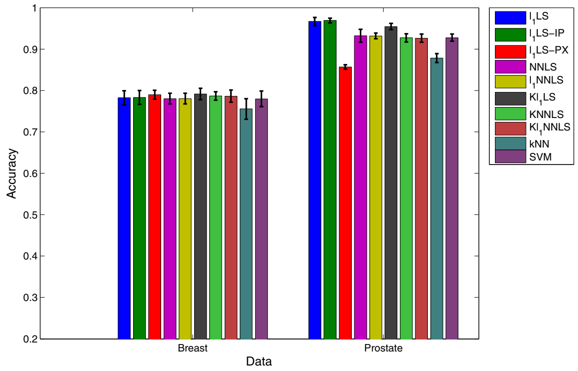

Figure 1. Accuracies of Sparse Coding and Benchmark Approaches

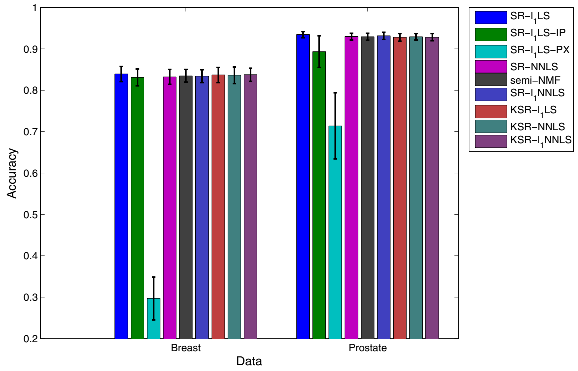

Figure 3. Accuracies of Dictionary Learning Approaches

QP, and the ease of parallel share of computing makes the active-set algorithm more competent. Although proximal method is faster than active set algorithm on Prostate, its accuracy is not competitive. Second, NNLS and l_1NNLS have the same time-complexity. They are even faster than l_1LS. Therefore, in the case of having similar accuracy as l_1LS, for example on Breast data, preference should be given to NNLS and l_1NNLS. Furthermore, they takes less time than SVM on both data.

B. Dictionary-Learning Approaches

We explored the performance of the linear and kernel dictionary-learning-based classification approaches on Breast and Prostate. They are compared with the interior-point [21] and proximal methods [22], and semi-NMF using multiplicative-update-rules [25]. 4-fold CV was employed. After dimension reduction, linear SVM was employed.

The mean accuracies and standard deviations are shown in Figure 3. First, compared with Figure 1, it can be seen that the accuracies of the SR approaches were improved by dictionary learning on Breast data. Second, on Prostate data, SR-NNLS and SR-l_1NNLS, as shown in Figure 3, have similar performance as NNLS and l_1NNLS (Figure 1). This implies that dictionary learning is a promising dimension reduction technique. The accuracy of SR-l_1LS is slightly lower than l_1LS on Prostate, which could be explained by the unsupervised fashion of dictionary learning. Third, using the same parameter, the accuracy of SR-l_1LS is four percent higher than SR-l_1LS-IP on Prostate. Also, SR-l_1LS is also slightly higher than SR-l_1LS-IP with respect to accuracy on Breast. Very poor accuracies was achieved by SR-l_1LS-PX on both data. SR-l_1LS-IP and SR-l_1LS-PX exhibit larger variances. Thus SR-l_1LS is more stable than SR-l_1LS-IP and SR-l_1LS-PX. Fourth, SR-NNLS has the same accuracy as semi-NMF. This implies that our dictionary learning framework performs well. Finally, the kernel versions obtained similar accuracies as their linear ones. The accuracy may be improved by a suitable kernel.

The mean computing time is compared in Figure 4. First, it can be seen that SR-l_1LS-IP, took much more time than SR-l_1LS. Most of our active-set algorithms are also faster than the proximal algorithm. Though promising for hierarchical dictionary learning in other fields, proximal method is not advantageous over active-set method for biological data where usually only few dictionary atoms are sufficient. Secondly, semi-NMF consumed much more time than the active-set-algorithm-based SR-NNLS. Finally, we can see

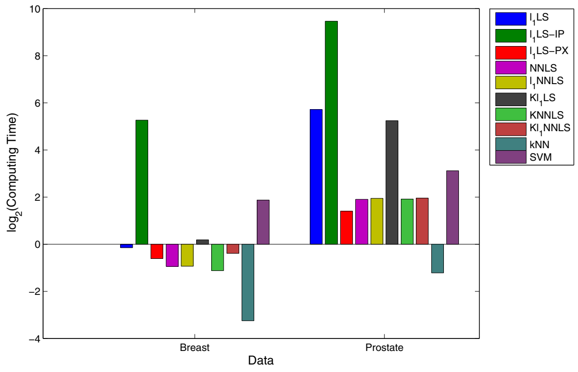

Figure 2. Computing Time of Sparse Coding and Benchmark Approaches

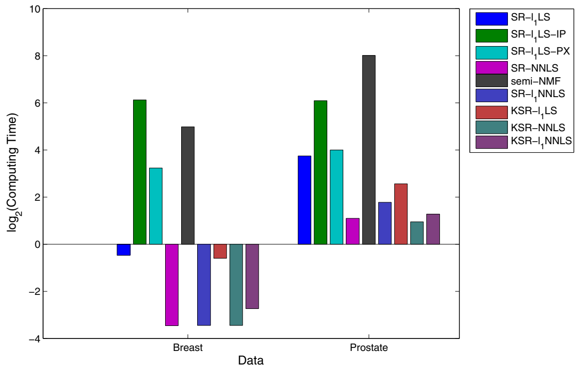

Figure 4. Computing Time of of Dictionary Learning Approaches

that the linear and kernel non-negative methods are more efficient than the l_1 methods. Thus in the case of similar accuracy, the former approaches should be preferred.

V. Conclusion

In this paper, we propose fast l_1 regularized and non-negative kernel SR approaches to classify high-dimensional biological data. Our experimental results show that our approaches are very efficient compared with existing approaches. Furthermore, similar or higher accuracies are obtained than the existing approaches. The SR approaches experimented in this paper are publicly available at cs.uwindsor.ca\~li11112c\sr. Our kernel SR can be applied to classify high-way data where a sample is not a vector but a tensor [13]. It is also competent to classify biomedical text data or relational data where only similarity between samples are known. We will focus our future works on supervised SR and comparison with Bayesian decomposition and Bayesian factor regression modelling on biological data.

Acknowledgment

This research has been supported by IEEE CIS Walter Karplus Summer Research Grant 2010, Ontario Graduate Scholarship 2011-2012, and Canadian NSERC Grants #RGPIN228117-2011.

References

[1] T. Furey, N. Cristianini, N. Duffy, D. Bednarski, M. Schummer, and D. Haussler, “Support vector machine classification and validation of cancer tissue samples using microarray expression data,” *Bioinformatics*, vol. 16, pp. 906–914, 2000.

[2] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani, “1-norm support vector machines,” in *NIPS*. Cambridge: MIT Press, 2003.

[3] M. Wall, A. Rechtsteiner, and L. Rocha, “Singular value decomposition and principal component analysis,” in *A Practical Approach to Microarray Data Analysis*, D. Berrar, W. Dubitzky, and M. Granzow, Eds. Norwell, MA: Kluwer, 2003, pp. 91–109.

[4] D. Huang and C. Zheng, “Independent component analysis-based penalized discriminant method for tumor classification using gene expression data,” *Bioinformatics*, vol. 22, no. 15, pp. 1855–1862, 2006.

[5] M. Elad, *Sparse and Redundant Representations*. New York: Springer, 2010.

[6] P. Hoyer, “Modeling receptive fields with non-negative sparse coding,” *Neurocomputing*, vol. 52-54, pp. 547–552, 2003.

[7] M. Elad and M. Aharon, “Image denoising via learned dictionaries and sparse representation,” in *CVPR*. Washington DC: IEEE Computer Society, 2006, pp. 895–900.

[8] D. D. Lee and S. Seung, “Learning the parts of objects by non-negative matrix factorization,” *Nature*, vol. 401, pp. 788–791, 1999.

[9] L. Zhang, W. D. Zhou, P. C. Chang, J. Liu, Z. Yan, T. Wang, and F. Z. Li, “Kernel sparse representation-based classifier,” *IEEE Trans. Signal Process.*, vol. 60, pp. 1684 – 1695, 2012.

[10] J. Yin, X. Liu, Z. Jin, and W. Yang, “Kernel sparse representation based classification,” *Neurocmputing*, vol. 77, pp. 120–128, 2012.

[11] S. Gao, I. W. H. Tsang, and L. T. Chia, “Kernel sparse representation for image classification and face recognition,” in *ECCV*. Springer, 2010, pp. 1–14.

[12] J. Brunet, P. Tamayo, T. Golub, and J. Mesirov, “Metagenes and molecular pattern discovery using matrix factorization,” *PNAS*, vol. 101, no. 12, pp. 4164–4169, 2004.

[13] Y. Li and A. Ngom, “Non-negative matrix and tensor factorization based classification of clinical microarray gene expression data,” in *BIBM*. Washington DC: IEEE Computer Society, 2010, pp. 438–443.

[14] M. Ochs and E. Fertig, “Matrix factorization for transcriptional regulatory network inference,” in *CIBCB*. Piscataway: IEEE Press, May 2012, pp. 387–396.

[15] X. Hang and F.-X. Wu, “Sparse representation for classification of tumors using gene expression data,” *J. Biomedicine and Biotechnology*, vol. 2009, 2009.

[16] C.-H. Zheng, L. Zhang, T.-Y. Ng, S. Shiu, and D.-S. Huang, “Metasample-based sparse representation for tumor classification,” *TCBB*, vol. 8, no. 5, pp. 1273–1282, 2011.

[17] B. Olshausen and D. Field, “Sparse coding with an overcomplete basis set: A strategy employed by v1?” *Vision Research*, vol. 37, no. 23, pp. 3311–3325, 1997.

[18] J. Wright, A. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, “Robust face recognition via sparse representation,” *TPAMI*, vol. 31, no. 2, pp. 210–227, 2009.

[19] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York: Springer, 2006.

[20] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*. Piladelphia: SIAM, 1995.

[21] S. J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, “An interior-point method for large-scale $l1$-regularized least squares,” *IEEE J. Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 606–617, 2007.

[22] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach, “Proximal methods for hierarchical sparse coding,” *JMLR*, vol. 12, no. 2011, pp. 2297–2334, 2011.

[23] Z. Hu, “The molecular portraits of breast tumors are conserved across microarray platforms,” *BMC Genomics*, vol. 7, p. 96, 2006.

[24] E. I. Petricoin, “Serum proteomic patterns for detection of prostate cancer,” *J. National Cancer Institute*, vol. 94, no. 20, pp. 1576–1578, 2002.

[25] C. Ding, T. Li, and M. I. Jordan, “Convex and semi-nonnegative matrix factorizations,” *TPAMI*, vol. 32, no. 1, pp. 45–55, 2010.