

# Network-based Support Vector Machine for Classification of Microarray Samples

Yanni Zhu<sup>1</sup>, Xiaotong Shen<sup>2</sup> and Wei Pan<sup>\*1</sup>

<sup>1</sup>Division of Biostatistics, School of Public Health, University of Minnesota, Minneapolis, Minnesota 55455, USA

<sup>2</sup>School of Statistics, University of Minnesota, Minneapolis, Minnesota 55455, USA

Email: Yanni Zhu - yannizhu@biostat.umn.edu; Xiaotong Shen - xshen@stat.umn.edu; Wei Pan\* - weip@biostat.umn.edu;

\*Corresponding author

## Abstract

**Background:** The importance of network-based approach to identifying biological markers for diagnostic classification and prognostic assessment in the context of microarray data has been increasingly recognized. To our knowledge, there have been few, if any, statistical tools that explicitly incorporate the prior information of gene networks into classifier building. The main idea of this paper is to take full advantage of the biological observation that neighboring genes in a network tend to function together in biological processes and to embed this information into a formal statistical framework.

**Results:** We propose a network-based support vector machine for binary classification problems by constructing a penalty term from the  $F_\infty$ -norm being applied to pairwise gene neighbors with the hope to improve predictive performance and gene selection. Simulation studies in both low- and high-dimensional data settings as well as two real microarray applications indicate that the proposed method is able to identify more clinically relevant genes while maintaining a sparse model with either similar or higher prediction accuracy compared with the standard and the  $L_1$  penalized support vector machines.

**Conclusions:** The proposed network-based support vector machine has the potential to be a practically useful classification tool for microarrays and other high-dimensional data.

## Background

The past two decades have witnessed rapid advances in gene expression profiling with the microarray technology, which not only brighten the

prospect of deciphering the complexity of disease genesis and progression at the genomic level, but also revolutionize the diagnostic, therapeutic, and prognostic approaches. Up to recently, diagnostic classification and prognostic assessment have been based on conventional clinical and pathological risk factors, such as patient age and tumor size, many of which are believed to be secondary manifestation [1]. The advent of microarray technology allows researchers to explore primary disease mechanisms by comparing gene expression profiles for malignant and normal cells. The regularity and aberration in the expression patterns of certain genes shed light on their functions and pathological importance [2]. Studies that seek to identify gene markers to refine diagnostic classification and improve prognostic prediction in the context of gene expression data have enriched the literature [3–5].

In recent years, researchers have realized that gene markers identified from microarrays drawn from different studies on the same disease across similar cohorts lack consistency [6, 7]. A possibly more effective means to resolve this problem is to employ a network-based approach, that is, to identify markers as gene subnetworks, defined as groups of functionally related genes based on a gene network, instead of treating individual genes as completely independent and identical a priori as in most existing approaches [1]. A novel network-based approach proposed recently [1, 8] can be summarized as follows: (1) randomly searching subnetworks and assigning a score to each subnetwork that characterizes the subnetwork-wise gene expression level; (2) identifying significant subnetworks that can well discriminate the clinical outcome; (3) constructing a classifier based on the significant subnetworks with a conventional statistical tool, such as logistic regression. Essentially such a network-based approach aggregates gene expression data at the subnetwork level and

then identifies and utilizes some significant sub-networks. It has been shown that such a network-based approach not only improves predictive performance and reproducibility, but also sheds biological insights into molecular mechanisms underlying the clinical outcome. However, the above method is largely heuristic without a formal statistical framework; more importantly, it involves a random search over subnetworks, leading to possibly different results from different runs with no guarantee of the optimality of the final result. Because of the ever-increasing popularity of penalization methods for high-dimensional data, we propose a novel network-based penalty to be used with the hinge loss, leading to a network-based support vector machine (SVM). While maintaining some desirable properties of SVM with the hinge loss function, the network-based penalty directly integrates a biological network to realize more effective variable selection, as compared with generic methods, such as the standard SVM or  $L_1$ -penalized SVM (L1-SVM).

The support vector machine (SVM) is one of the most popular supervised learning techniques with wide-ranging applications [9, 10]. In particular, previous studies have demonstrated its superior performance in gene expression data analysis, especially its ability to handle high dimensional data [11, 12]. Nevertheless, with categorical predictors, both the standard SVM and the L1-SVM may have some shortcomings. [13] applied the concept of grouped variable selection and developed an  $F_\infty$ -norm penalized SVM to realize simultaneous selection/elimination of all the features derived from the same categorical factor (or a group of variables). Their numerical examples showed that the  $F_\infty$ -norm SVM outperformed the L1-SVM in factor-wise variable selection. We extend the idea of variable grouping to gene networks: rather than grouping all the dummy variables created from the same categorical factor, we treat two neighboring genes in a network as one group. The network-based penalty is constructed as the sum of the  $F_\infty$ -norms being applied to the groups of neighboring-gene pairs. With the hinge loss penalized by such a network-based penalty as our objective function, we obtain our network-based SVM.

The later sections are organized as follows. We begin with a brief review of the SVM, and then introduce our proposed network-based SVM. We evaluate its performance by simulation studies in both low dimensional and high dimensional data settings as well as two real data applications. The last section concludes the paper with a brief summary.

## Methods

### Existing methods

Suppose we have training data  $\{(x_i, y_i)\}_{i=1}^N$  with  $x_i \in \mathbb{R}^p$  and  $y_i \in \{1, -1\}$ . Define a hyperplane  $\{x : f(x) = x^T \beta + \beta_0 = 0\}$ . The classification rule induced by  $f(x)$  is  $\text{sign}[\hat{f}(x)]$ . SVM searches for such a hyperplane  $\hat{f}(x) = x^T \hat{\beta} + \hat{\beta}_0$  that maximizes the margin between the training data points for class 1 and class -1:

$$\begin{aligned} & \max_{\beta, \beta_0} \frac{1}{\|\beta\|_2} \\ \text{subject to } & y_i(x_i^T \beta + \beta_0) \geq 1 - \xi_i, \quad \forall i \quad (1) \\ & \xi_i \geq 0, \quad \sum_{i=1}^N \xi_i \leq C \end{aligned}$$

where  $\xi_i$  are slack variables, and  $C$  is a tuning parameter to be determined. The standard SVM has an equivalent *hinge loss + penalty* formulation as an optimization problem [13–15]:

$$\min_{\beta_0, \beta} \left\{ \sum_{i=1}^N [1 - y_i(x_i^T \beta + \beta_0)]_+ + \lambda \|\beta\|_2^2 \right\} \quad (2)$$

where the subscript “+” denotes the positive part, i.e.,  $z_+ = \max\{z, 0\}$ ,  $\|\beta\|_2^2 = \sum_{k=1}^p |\beta_k|^2$ , and  $\lambda$  is the tuning parameter. The solution to (1) is the same as that to (2).

The above standard SVM forces all nonzero coefficient estimates, which leads to the problem of its inability to conduct variable selection. The L1-SVM was proposed to accomplish the goal of variable selection. It can be formulated as

$$\min_{\beta_0, \beta} \left\{ \sum_{i=1}^N [1 - y_i(x_i^T \beta + \beta_0)]_+ + \lambda \|\beta\|_1 \right\} \quad (3)$$

where  $\|\beta\|_1 = \sum_{k=1}^p |\beta_k|$ . The L1-SVM wins over the standard SVM when the true model is sparse, while the standard SVM is preferred if there are not many redundant noise features [16].

[13] pointed out the shortcoming of the L1-norm penalty: even though it encourages parsimonious models, it fails to guarantee successful models in cases of categorical predictors due to the fact that each dummy variable is selected independently. They applied the concept of grouped variable selection and proposed an  $F_\infty$ -norm SVM to realize simultaneous selection/elimination of features derived from the same factor so as to accomplish automatic factor-wise variable selection. Suppose we have  $G$  factors  $F_1, \dots, F_G$ . From each factor  $F_g$ , we generate a feature vector  $x_{(g)} = (x_1^{(g)}, \dots, x_j^{(g)}, \dots, x_{n_g}^{(g)})^T$ . Correspondingly we have the coefficient vector  $\beta_{(g)} =$

$(\beta_1^{(g)}, \dots, \beta_j^{(g)}, \dots, \beta_{n_g}^{(g)})^T$ . Therefore,

$$f(x) = x^T \beta + \beta_0 = \sum_{g=1}^G x_{(g)}^T \beta_{(g)} + \beta_0 \quad (4)$$

Define the  $F_\infty$ -norm of  $F_g$  as

$$\|F_g\|_\infty = \|\beta_{(g)}\|_\infty = \max_{j \in \{1, \dots, n_g\}} \{|\beta_j^{(g)}|\} \quad (5)$$

The  $F_\infty$ -norm SVM is formulated as

$$\min_{\beta_0, \beta} \left\{ \sum_{i=1}^N \left[ 1 - y_i \left( \sum_{g=1}^G x_{i,(g)}^T \beta_{(g)} + \beta_0 \right) \right]_+ + \lambda \sum_{g=1}^G \|\beta_{(g)}\|_\infty \right\} \quad (6)$$

The most noteworthy property of the  $F_\infty$ -norm SVM is its guarantee of sparsity at the factor level. Due to the singularity property of the infinity norm:  $\|\beta_{(g)}\|_\infty$  is not differentiable at  $\beta_{(g)} = 0$ ,  $\beta_{(g)}$  will be exactly zero if the regularization parameter  $\lambda$  is properly chosen [13]. Therefore, the  $F_\infty$ -norm SVM automatically eliminates factors that are completely irrelevant to the response, and thus achieves the goal of factor-wise selection. The empirical evidence shows that the  $F_\infty$ -norm SVM often outperforms both the L1-SVM and the standard SVM.

## New method

Biological observations reveal that neighboring genes in a network tend to function together in biological processes. To incorporate this prior information, a network-based SVM for classification is proposed to facilitate generating models that extract more biological insight from gene expression data. The penalty term that characterizes the network structure can be specified by implanting the  $F_\infty$ -norm into the context of known functional interrelationships among genes by considering each pair of the functionally related genes as one group.

Consider a gene network with  $S$  denoting the set of all edges, i.e., the pair of connected genes.

$$S = \{(j_1, j_2) : \text{gene } j_1 \text{ and gene } j_2 \text{ are connected}\}$$

Define  $w_k$  as some weight for gene  $k$ . For example,  $w_k = \sqrt{d_k}$  where  $d_k$  is the number of direct neighbors of gene  $k$ , or  $w_k = d_k$ , or simply  $w_k = 1$  for all genes. We propose a novel penalty in the form of

$$\sum_{(j_1, j_2) \in S} \max \left\{ \frac{|\beta_{j_1}|}{w_{j_1}}, \frac{|\beta_{j_2}|}{w_{j_2}} \right\} \quad (7)$$

Thus the network-based SVM solves the optimiza-

tion problem as follows.

$$\min_{\beta_0, \beta} \left\{ \sum_{i=1}^N [1 - y_i (x_i^T \beta + \beta_0)]_+ + \lambda \sum_{(j_1, j_2) \in S} \max \left( \frac{|\beta_{j_1}|}{w_{j_1}}, \frac{|\beta_{j_2}|}{w_{j_2}} \right) \right\} \quad (8)$$

Four properties of the penalty term are noteworthy. First, the regularization is performed at edge level. In the case of penalized regression problem, it has been proven that this penalty achieves the goal of elimination of  $\beta_{j_1}$  and  $\beta_{j_2}$  simultaneously under certain conditions [17]. The automatic selection of grouped features is due to the singularity nature of  $\max\{|a|, |b|\}$ , that is,  $\max\{|a|, |b|\}$  is nondifferentiable at  $a = b = 0$  [13]. This formulation satisfies our assumption that neighboring genes tend to (or not to) contribute to the same biological process at the same time. Second, the choice of the weight depends on the goal of shrinkage and influences the predictive performance. Consider a network comprised of several subnetworks, each with one regulator and ten target genes. The target genes have the same effect on the outcome and only connect to the regulator. The simulation studies under this setting in the next section show the new method with heavy weight encourages regulators to have larger estimated effect compared with its light weight counterpart. The weighted penalty, in the context of penalized regression, encourages  $\frac{|\beta_{j_1}|}{w_{j_1}} = \frac{|\beta_{j_2}|}{w_{j_2}}$  [17]. Here we examine three weight functions in particular:  $w_k = 1$ ,  $w_k = \sqrt{d_k}$ , and  $w_k = d_k$  where gene  $k$  has  $d_k$  direct neighbors. The new method encourages  $|\beta_{j_1}| = |\beta_{j_2}|$  if  $w_k = 1$ ,  $\frac{|\beta_{j_1}|}{\sqrt{d_{j_1}}} = \frac{|\beta_{j_2}|}{\sqrt{d_{j_2}}}$  if  $w_k = \sqrt{d_k}$ , and  $\frac{|\beta_{j_1}|}{d_{j_1}} = \frac{|\beta_{j_2}|}{d_{j_2}}$  if  $w_k = d_k$ . Therefore, heavier weights (from  $w_k = 1$ ,  $w_k = \sqrt{d_k}$ , to  $w_k = d_k$ ) favor genes with more direct neighbors to have larger coefficient estimates; in other words, heavier weights relax the shrinkage effect for certain genes. Due to this property, the choice of a heavy weight, as a simple strategy, enables us to alleviate the bias in the coefficient estimates from the penalization method and improve the predictive performance. Our default weight is  $w_k = \sqrt{d_k}$ . The weight, considered as another tuning parameter, can be determined from cross-validation or an independent validation data set although we do not consider it here. Third, the penalty term, under certain conditions, tends to encourage a grouping effect, where highly correlated predictors tend to have similar coefficient estimates [18–20]. Fourth, the penalty is linear, which allows the solution to be found by the linear programming (LP) technique that is computationally convenient.

As usual, the fitted classifier is  $\hat{f}(x) = \hat{\beta}_0 + x^T \hat{\beta}$ , and the classification rule is  $\text{sign}(\hat{f}(x))$ . We employ LP to obtain the solutions to (8) by

$$\min_{\beta_0, \beta} \left( \sum_{i=1}^N \xi_i + \lambda \sum_{(j_1, j_2) \in S} M_{(j_1, j_2)} \right) \quad (9)$$

subject to

$$\begin{aligned} y_i (\beta_0^+ - \beta_0^- + x_i^T (\beta^+ - \beta^-)) &\geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \forall i \\ \frac{\beta_j^+}{\sqrt{d_j}} + \frac{\beta_j^-}{\sqrt{d_j}} &\leq M_{(j_1, j_2)}, \quad j = j_1, j_2 \quad \forall (j_1, j_2) \in S \\ \beta_j^+ &\geq 0, \quad \beta_j^- \geq 0, \quad j = j_1, j_2 \quad \forall (j_1, j_2) \in S \end{aligned} \quad (10)$$

where

$$\xi_i = \left[ 1 - y_i \left( \sum_{i=1}^N x_i^T \beta + \beta_0 \right) \right]_+, \quad i = 1, 2, \dots, N \quad (11)$$

$$M_{(j_1, j_2)} = \max \left\{ \frac{|\beta_{j_1}|}{\sqrt{d_{j_1}}}, \frac{|\beta_{j_2}|}{\sqrt{d_{j_2}}} \right\} \quad (12)$$

and  $\beta_j = \beta_j^+ - \beta_j^-$ , in which  $\beta_j^+$  and  $\beta_j^-$  denote the positive and negative parts of  $\beta_j$ . The calculation of the new method can be easily implemented by the R package `lpsolve`, so is the computation of the L1-SVM. The R package `e1071` (with linear kernel) is used to obtain the solution to the standard SVM.

## Results and Discussion

### Simulation

We conducted several simulation studies to numerically evaluate the performance of the network-based SVM along with the standard SVM (STD) and L1-SVM. The simulation setups were similar to those in [18]. We started from a simple network consisting of 5 subnetworks, each having a regulator gene  $t$  ( $t = 1, \dots, 5$ ) that regulated 10 target genes, leading to a total of 55 genes ( $p = 55$ ). We assumed that two out of the five subnetworks were informative; that is, the coefficients of 22 genes were nonzero and thus informative to the outcome, while the remaining 33 noise genes had no effect on the outcome. We generated a simulated data set by the following steps:

- Generate the expression level of regulator gene  $t$ ,  $X_t \sim N(0, 1)$ ,  $t = 1, \dots, 5$ , independently.
- Assume that the expression level of regulator gene  $t$  and each of its regulated genes follow a bivariate normal distribution with correlation 0.7. Thus, the expression level

of each target gene regulated by gene  $t$ ,  $X_l^{(t)} \sim N(0.7X_t, 0.51)$ ,  $l = 1, \dots, 10$  and  $t = 1, \dots, 5$ .

- Generate the outcome  $Y$  from a logistic regression model:  $\text{Logit}(\text{Pr}(Y = 1|X)) = X^T \beta + \beta_0$ ,  $\beta_0 = 2$ , where  $X$  is the vector of the expression levels of all the genes, and coefficient vector  $\beta = (\beta_1^{(1)}, \dots, \beta_{10}^{(1)}, \dots, \beta_1^{(5)}, \beta_{10}^{(5)})$ .

Four sets of true coefficients,  $\beta$ 's, were specified to reflect four scenarios:

$$1. \beta = (5, \underbrace{\frac{5}{\sqrt{10}}, \dots, \frac{5}{\sqrt{10}}}_{10}, -5, \underbrace{\frac{-5}{\sqrt{10}}, \dots, \frac{-5}{\sqrt{10}}}_{10}, 0, \dots, 0).$$

The effect of one informative subnetwork was the same as the other in magnitude but with an opposite direction.

$$2. \beta = (5, \underbrace{\frac{5}{\sqrt{10}}, \dots, \frac{5}{\sqrt{10}}}_{10}, 3, \underbrace{\frac{3}{\sqrt{10}}, \dots, \frac{3}{\sqrt{10}}}_{10}, 0, \dots, 0).$$

Both informative subnetworks had positive effects but in different magnitudes.

$$3. \beta = (5, \underbrace{\frac{5}{\sqrt{10}}, \dots, \frac{5}{\sqrt{10}}}_{7}, \frac{-5}{\sqrt{10}}, \frac{-5}{\sqrt{10}}, \frac{-5}{\sqrt{10}}, 3, \underbrace{\frac{3}{\sqrt{10}}, \dots, \frac{3}{\sqrt{10}}}_{7}, \frac{-3}{\sqrt{10}}, \frac{-3}{\sqrt{10}}, \frac{-3}{\sqrt{10}}, 0, \dots, 0).$$

Target genes in the same informative subnetworks had both positive and negative effects.

$$4. \beta = (5, \underbrace{\frac{5}{\sqrt{10}}, \dots, \frac{5}{\sqrt{10}}}_{6}, \frac{-5}{\sqrt{10}}, \dots, \frac{-5}{\sqrt{10}}, -3, \underbrace{\frac{-3}{\sqrt{10}}, \dots, \frac{-3}{\sqrt{10}}}_{6}, \underbrace{\frac{3}{\sqrt{10}}, \dots, \frac{3}{\sqrt{10}}}_{4}, 0, \dots, 0).$$

It was similar to but more extreme than scenario 3.

Five methods, the standard SVM (STD), L1-SVM, and network-based SVM with  $w_k = 1$ ,  $w_k = \sqrt{d_k}$ , and  $w_k = d_k$ , were compared based on the results averaged over 100 runs under each of the above four scenarios. For each run, 100 observations were simulated as training data to build a classifier (with any given  $\lambda$ ), another 100 for tuning the regularization parameter  $\lambda$ , and the last 10,000 as test data. Each predictor was normalized to have mean 0 and standard deviation 1. Given any value of  $\lambda$ , we obtained the coefficient estimates from the training set, then applied the classifier to the tuning set to find the classification error. We

searched for  $\hat{\lambda}$ , from a wide range of prespecified values, which produced the smallest classification error. The classifier corresponding to  $\hat{\lambda}$  was identified as the fitted classifier  $\hat{f}$ . Then we applied  $\hat{f}$  to the test set and calculated the test error, the number of misclassifications divided by the test sample size. Table 1 reports the mean classification error of the test set and its standard error (SE in parentheses), the standard deviation of the classification errors divided by the square root of the number of runs, for each method over 100 runs under each scenario. To evaluate each method’s ability to select informative genes, we examined the false negatives, defined as the number of informative genes whose coefficients were estimated to be zero. In addition, we also considered a smaller sample size: we repeated the entire process with 50 training data points, 50 tuning data points, and again 10,000 test data points. The network-based SVM is named as "New" in the table.

[Table 1 about here.]

[Table 2 about here.]

According to our simulation setups, the correct weight function should be  $w = \sqrt{d}$ . However, we find that the new method with  $w = d$  overwhelmingly beat all other methods in all the setups. It consistently made the most accurate classifications and missed no informative genes. The new method with  $w = \sqrt{d}$  performed the second best: in most cases, it improved the classification accuracy over the standard SVM and L1-SVM; and under all the settings, it produced models that identified more informative genes than the L1-SVM. In contrast,  $w = 1$  did not bring much gains over the STD or the L1-SVM. The L1-SVM led to models that were too sparse, missing about 14 and 11 informative genes for  $n = 50$  and  $n = 100$  respectively. The superior performance and the larger model size of the heavy weight ( $w = d$ ) compared with its counterparts ( $w = 1$  and  $w = \sqrt{d}$ ) is presumably due to its relaxation of the shrinkage effect. The penalization methods shrink the  $\hat{\beta}$  toward zero by imposing the constraints (the penalty term) and therefore introduces bias to  $\hat{\beta}$ . By grouping neighboring genes, the new method encourages the pairwise weighted coefficients to be equal. Therefore, a heavy weight leads to larger  $|\hat{\beta}|$  for regulator genes. By choosing a heavier weight, we may overcome over-shrinkage, alleviate biases, and achieve better classification accuracy to some extent at the expense of model sparsity. As shown by Table 2,  $w = d$  produced the largest  $|\hat{\beta}|$  for regulators than its two counterparts. The L1-SVM estimates were treated as a yardstick for comparison as to provide us an idea of the shrinkage effect from each weight function. For example,  $w = 1$  and  $w = \sqrt{d}$  overly shrank all the regulators under all scenarios compared with

the L1-SVM estimates. Note that the binary outcome  $Y$  was obtained from the logistic regression model while  $\hat{\beta}$  was estimated by the linear model. Hence  $\beta$  and  $\hat{\beta}$  are noncomparable.

Next, we evaluated the performance of the new method for high-dimensional data with large  $p$ . We used the setup of 50 observations for training, 50 for tuning, and 10,000 for test data. We assumed that (1) the network was composed of either 50 or 100 subnetworks, each having one gene regulating 10 target genes; (2) the first 2 subnetworks were informative resulting in 22 informative genes; (3) the rest of the genes had no effect on the outcome, leading to 528 noise genes when  $p = 550$  and 1,078 noise genes when  $p = 1,100$ ; and (4) the true  $\beta$  was specified as in scenario 3. Table 3 shows the simulation results averaged over 100 runs.

[Table 3 about here.]

Again, we see the gains from using a heavy weight ( $w = d$ ). It prevailed over all the other methods in making accurate classifications and selecting informative genes. The  $w = \sqrt{d}$  ranked the second. However,  $w = d$  generated models much larger than those from other methods except STD. In this case, the performance of  $w = 1$  is no better than L1-SVM possibly due to over shrinkage of the effects of the regulator genes.

### Applications to microarray data

To evaluate its performance in the real world, we applied the new method to two microarray gene expression data sets related to the Parkinson’s disease (PD) [21] and breast cancer metastasis (BC) [1, 4] respectively.

#### *Parkinson’s Disease*

The data set includes the Parkinson’s disease status and the expression levels of 22,283 genes from 105 patients (50 cases and 55 controls) [22]. We used the same network structure as [18]. The network combines 33 Kyoto Encyclopedia of Genes and Genomes (KEGG) regulatory pathways and contains a total of 1,523 genes and 6,865 edges. The data were randomly split into training (40 observations), tuning (20 observations), and test (45 observations) sets. The expression level of each gene was normalized to have mean 0 and standard deviation 1 across samples. The tuning parameter was identified from the tuning set and the performance of the method was evaluated on the test set by the mean classification error and its standard error averaged over 10 runs. Five methods were compared: standard SVM, L1-SVM, network-based SVM with  $w = 1$ ,  $w = \sqrt{d}$ , and  $w = d$ . To obtain a final model based on the new method with

$w = \sqrt{d}$ , we combined, for each run, the previous tuning and test data as the new tuning set leading to a sample size as large as 65 observations, on which the classification errors were calculated for wide-ranging values of the tuning parameter. Then after 10 runs, we had an averaged classification error corresponding to each tuning parameter value. The value that generated the minimal averaged error was the one we selected to fit the final model to all the data. Note that the classification error rate from the final model was likely to be biased due to the double use of the data for training/tuning and test; the main purpose of fitting the final model was to see the selected genes at the end.

First, we focused on the 1,070 genes that appeared in the network with the largest variations of expression levels (i.e., SD of expression levels across the 105 samples  $\geq 15$ ). According to the KEGG pathway of Parkinson’s disease [23], 20 genes play a role in the disease progression, five of which (*UBE1*, *PARK2*, *UBB*, *SEPT5*, and *SNCAIP*) belong to the 1,070 genes. In addition to the classification error, we added two additional criteria for method comparison: the number of disease genes identified, and the number of genes identified. Table 4 shows that STD made the most accurate classification, even though the difference with other methods was perhaps non-significant. The  $w = d$  ranked the second in predictive performance while produced a model including 70.6 genes on average. In this case, the  $w = \sqrt{d}$  gained advantage: it selected more disease genes by a relatively sparse model with a classification error non-significantly larger than STD. From the 1,070 genes, with the final model the new method identified 75 genes including one disease gene.

[Table 4 about here.]

Next, to better integrate the biological observation of the KEGG pathway and the known network structure of [18], we restricted our analysis to the first- and second-order-neighbors of the 8 disease genes on the Parkinson’s disease KEGG pathway whose expression levels and network structure are available. The first-order-neighbor subnetwork (PD-1nb-net) was composed of the 8 disease genes and their 8 direct neighbors. The second-order-neighbor subnetwork (PD-2nb-net) comprised the PD-1nb-net as well as the direct neighbors of the 8 direct neighbors of the disease genes, leading to a total of 26 genes. Figure 1 displays the two subnetworks. We conducted the analysis in the same way as described above. The only difference resided in that this time only genes appearing in the PD-1nb-net and PD-2nb-net were included in the analysis. Table 5 shows the results.

[Figure 1 about here.]

[Table 5 about here.]

We see the gains from employing the new

method when narrowing down our focus on the PD-1nb-net and PD-2nb-net. For the PD-1nb-net,  $w = 1$  and  $w = \sqrt{d}$  performed equally well. They had the smallest classification error and identified one more disease gene through a model slightly larger than the one obtained from L1-SVM. The new method with  $w = d$  won over in the case of PD-2nb-net with the best accuracy and most selected disease genes. The  $w = \sqrt{d}$  ranked the second in terms of the prediction accuracy while detecting 3 more disease genes by a model with 3 more genes than that of the L1-SVM. This means that the new method was able to identify more clinically relevant genes while keeping the same number of noise genes in the model as L1-SVM. In both subnetworks, the final models included all the genes.

### Breast Cancer Metastasis

The breast cancer metastasis data set [1,4] contains expression levels of 8,141 genes for 286 patients, 106 of whom were detected to develop metastasis within a 5-year follow-up after surgery. *TP53*, *BRCA1*, and *BRCA2* are three human genes that belong to the class of tumor suppressor genes, which are known to prevent uncontrolled cell proliferation, and to play a critical role in repairing the chromosomal damage. Certain mutations of these genes lead to increasing risk of breast cancer. We explored the protein-protein interaction (PPI) network previously used by [1]. The PPI network comprises 57,235 interactions among 11,203 proteins, obtained by assembling various sources of experimental data and curation of the literature [1]. We confined our analysis to the direct or first-order neighbors (BC-1nb-net) of the three cancer genes, and the subnetwork composed of two parts (BC-2nb-net): the direct neighbors of *TP53*, and the second-order neighbors of *BRCA1* and *BRCA2*. We fit the final model and compared the four methods in terms of classification error, selection of cancer genes, and sparsity of the model. The cancer genes are the 227 known or putative cancer genes with estimated mutation frequencies in cancer samples (Supplementary Table 10 of [1]). A total of 294 genes that fell into the BC-1nb-net had observed expression levels, among which were 40 cancer genes and 7 cancer genes (*ABL1*, *JAK2*, *p53*, *PTEN*, *p14ARF*, *PTCH*, and *RB*) with mutation frequencies larger than 0.10. The BC-2nb-net was composed of 2,070 genes, 1,718 of them with observed expression levels, including 107 cancer genes. Besides the 7 included in BC-1nb-net, 7 additional cancer genes (*ACH*, *APC*, *EGFR*, *KIT*, *NICD*, *RAS*, and *CTNNB1*) that had mutation frequencies larger than 0.10 belonged to BC-2nb-net.

[Table 6 about here.]

For BC-1nb-net,  $w = d$  had the advantage in selecting cancer genes and those with large mutant frequencies (Table 6). The  $w = \sqrt{d}$  detected more clinically relevant genes by a sparser model while reaching a comparable classification error rate to that of L1-SVM. Even though the final model was parsimonious, it included 4 cancer genes, one of which had a large mutation frequency. For BC-2nb-net, the new method with  $w = \sqrt{d}$  detected more cancer genes with equally accurate predictions while maintaining a sparse model compared with L1-SVM. The final model included only 23 genes out of 1,718, two of which were cancer genes with one having a large mutation frequency.

## Conclusions

The advancement in the microarray technology has enriched the tool kit of researchers to decipher the complexity of disease mechanisms at the genomic level. Studies have been widely conducted to identify genetic markers to better the diagnostic classification and prognostic assessment, largely by ignoring biological knowledge on gene functions and treating individual genes equally and independently a priori. The downside of such an endeavor has been realized; for example, gene markers identified across similar patient cohorts for the same disease in such a way often lack consistency. As a viable alternative, network-based approaches have been gaining popularity. In addition to improving predictive performance and reproducibility, the network-based approach extracts more biological insights from high-throughput gene expression data. Here we have proposed a network-based SVM, with a penalty term incorporating gene network information, as a practically useful classification tool for microarray data. Our simulation studies and two real data applications indicate that the proposed method is able to better identify clinically relevant genes and make accurate predictions.

## Competing interests

The authors, YZ, XS, and WP, declare that they have no competing interests.

## Authors contributions

YZ implemented the methods, did all the experiments and drafted the paper. XS and WP initiated the project. All participated in the writing of the article.

## Acknowledgements

YZ and WP were partially supported by NIH grants HL65462 and GM081535; XS supported by NIH grant GM081535 and NSF grants IIS-0328802 and DMS-0604394. We thank Dr Hongzhe Li and Dr Trey Ideker for providing the KEGG network and PPI network data respectively.

## References

1. Chuang HY, Lee EJ, Liu YT, Lee DH, Ideker T: **Network-based classification of breast cancer metastasis**. *Mol Syst Biol* 2007, **3**. [Article number 140, doi: 10.1038/msb4100180].
2. Frolov AE, Godwin AK, Favorova OO: **Differential gene expression analysis by DNA microarray technology and its application in molecular oncology**. *Mol Biol* 2003, **37**:486–494.
3. Yang TY: **The simple classification of multiple cancer types using a small number of significant genes**. *Mol Diagn Ther* 2007, **11**:265–275.
4. Wang Y, Klijn JG, Zhang Y, Sieuwerts AM, Look MP, Yang F, Talantov D, Timmermans M, Meijer van Gelder ME, Yu J, Jatkoa T, Berns EM, Atkins D, Foekens JA: **Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer**. *The Lancet* 2005, **365**:671–679.
5. Xiong MM, Li WJ, Zhao JY, Li J, Boerwinkle E: **Feature (gene) selection in gene expression-based tumor classification**. *Mol Genet Metab* 2001, **73**:239–247.
6. Ein-Dor L, Kela I, Getz G, Givol D, Domany E: **Outcome signature genes in breast cancer: is there a unique set?** *Bioinformatics* 2005, **21**:171–178.
7. Ein-Dor L, Zuk O, Domany E: **Thousands of samples are needed to generate a robust gene list for predicting outcome in cancer**. *Proc Natl Acad Sci U S A* 2006, **103**:5923–5928.
8. Liu M, Liberzon A, Kong SW, Lai WR, Park PJ, Kohane IS, Kasif S: **Network-based analysis of affected biological processes in type 2 diabetes models**. *PLoS Genet* 2007, **3**:958–972.
9. Cortes C, Vapnik V: **Support-vector networks**. *Machine Learning* 1995, **20**:273–297.
10. Vapnik V: *The Nature of Statistical Learning Theory*. New York: Springer 1995.
11. Brown MPS, Grundy WN, Lin D, Cristianini N, Sugnet CW, Furey TS, Ares M, Haussler D: **Knowledge-based analysis of microarray gene expression data by using support vector machines**. *Proc Natl Acad Sci U S A* 2000, **97**:262–267.
12. Furey T, Cristianini N, Duffy N, Bednarski DW, Schummer M, Haussler D: **Support vector machine classification and validation of cancer tissue samples using microarray expression data**. *Bioinformatics* 2000, **16**:906–914.

13. Zou H, Yuan M: **The  $F_\infty$ -norm Support Vector Machine**. *Stat Sin* 2008, **18**:379–398.
14. Wahba G, Lin Y, Zhang H: **GACV for support vector machines**. In *Advances in Large Margin Classifiers*. Edited by Smola A, Bartlett P, Scholkopf B, Schuurmans D, Cambridge, MA: MIT Press 2000:297–311.
15. Hastie T, Tibshirani R, Friedman JH: *The Elements of Statistical Learning*. New York: Springer 2001.
16. Friedman JH, Hastie T, Rosset S, Tibshirani R, Zhu J: **Discussion of boosting papers**. *Ann Appl Stat* 2004, **32**:102–107.
17. Pan W, Xie B, Shen X: **Incorporating predictor network in penalized regression with application to microarray data**. [Manuscript submitted to Biometrics].
18. Li C, Li H: **Network-constrained regularization and variable selection for analysis of genomic data**. *Bioinformatics* 2008, **24**:1175–1182.
19. Zou H, Hastie T: **Regularization and variable selection via the elastic net**. *J R Statist Soc B* 2005, **67**:301–320.
20. Wang L, Zhu J, Zou H: **The doubly regularized support vector machine**. *Stat Sin* 2006, **16**:589–615.
21. **Gene Expression Omnibus: GSE6613** [<http://www.ncbi.nlm.nih.gov/geo/>].
22. Scherzer CR, Eklund AC, Morse LJ, Liao Z, Locascio JJ, Fefer D, Schwarzschild MA, Schlossmacher MG, Hauser MA, Vance JM, Sudarsky LR, Standaert DG, Growdon JH, Jensen RV, Gullans SR: **Molecular markers of early Parkinson's disease based on gene expression in blood**. *Proc Natl Acad Sci U S A* 2007, **104**:955–960.
23. **KEGG: Parkinson's disease** [<http://www.genome.ad.jp/kegg/pathway/hsa/hsa05020.html>].

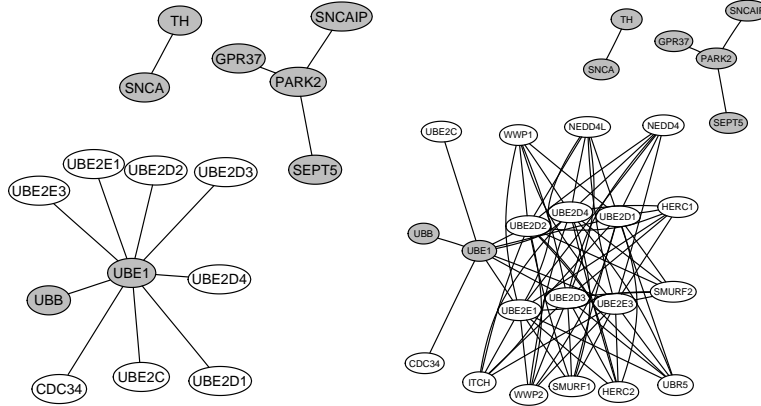


Figure 1: PD subnetworks. Left: PD-1nb-net, including 8 Parkinson disease genes (gray) and their 8 direct neighbors (white). Right: PD-2nb-net, including 8 Parkinson disease genes (gray), their 8 direct and 10 second-order neighbors (white).

Table 1: Simulation Results averaged over 100 runs for  $p = 55$  (22 informative and 33 noise genes).

Scenario	Method	Test Error (SE)		# False Negative (SE)		Model Size (SE)	
		$n = 50$	$n = 100$	$n = 50$	$n = 100$	$n = 50$	$n = 100$
1	STD	0.122 (0.002)	0.096 (0.001)	0.0 (0.0)	0.0 (0.0)	55.0 (0.0)	55.0 (0.0)
	L1	0.134 (0.003)	0.094 (0.002)	13.1 (0.3)	10.9 (0.4)	12.3 (0.6)	15.3 (0.7)
	New ( $w = 1$ )	0.156 (0.003)	0.105 (0.002)	9.3 (0.4)	2.4 (0.3)	17.0 (0.6)	24.3 (0.6)
	New ( $w = \sqrt{d}$ )	0.111 (0.003)	0.068 (0.002)	1.0 (0.3)	0.1 (0.1)	24.7 (0.5)	25.1 (0.4)
	New ( $w = d$ )	0.081 (0.002)	0.059 (0.002)	0.0 (0.0)	0.0 (0.0)	28.6 (0.8)	28.2 (0.8)
2	STD	0.121 (0.002)	0.099 (0.001)	0.0 (0.0)	0.0 (0.0)	55.0 (0.0)	55.0 (0.0)
	L1	0.133 (0.003)	0.096 (0.001)	13.6 (0.3)	11.1 (0.4)	11.4 (0.5)	15.1 (0.7)
	New ( $w = 1$ )	0.156 (0.003)	0.105 (0.002)	9.6 (0.4)	3.9 (0.3)	16.3 (0.7)	24.7 (0.6)
	New ( $w = \sqrt{d}$ )	0.121 (0.003)	0.075 (0.002)	3.0 (0.4)	0.3 (0.1)	22.3 (0.6)	25.2 (0.5)
	New ( $w = d$ )	0.083 (0.002)	0.064 (0.002)	0.0 (0.0)	0.0 (0.0)	28.6 (0.8)	29.0 (0.8)
3	STD	0.162 (0.002)	0.138 (0.001)	0.0 (0.0)	0.0 (0.0)	55.0 (0.0)	55.0 (0.0)
	L1	0.166 (0.003)	0.131 (0.001)	13.9 (0.2)	11.0 (0.3)	11.2 (0.5)	16.6 (0.7)
	New ( $w = 1$ )	0.177 (0.003)	0.140 (0.002)	12.4 (0.4)	7.7 (0.4)	13.5 (0.6)	19.9 (0.8)
	New ( $w = \sqrt{d}$ )	0.164 (0.003)	0.127 (0.002)	4.4 (0.5)	1.2 (0.3)	21.5 (0.6)	26.3 (0.7)
	New ( $w = d$ )	0.137 (0.003)	0.114 (0.001)	0.4 (0.2)	0.1 (0.1)	29.8 (0.9)	33.2 (0.9)
4	STD	0.189 (0.002)	0.157 (0.002)	0.0 (0.0)	0.0 (0.0)	55.0 (0.0)	55.0 (0.0)
	L1	0.186 (0.002)	0.155 (0.002)	14.2 (0.3)	10.5 (0.3)	11.5 (0.6)	18.1 (0.8)
	New ( $w = 1$ )	0.198 (0.003)	0.160 (0.002)	13.8 (0.3)	8.6 (0.4)	11.8 (0.5)	20.9 (0.9)
	New ( $w = \sqrt{d}$ )	0.190 (0.003)	0.147 (0.002)	7.2 (0.6)	1.8 (0.4)	18.8 (0.7)	30.1 (0.9)
	New ( $w = d$ )	0.163 (0.002)	0.139 (0.002)	0.2 (0.2)	0.03 (0.03)	32.2 (1.0)	34.8 (1.0)

Table 2: Coefficient estimates of selected informative genes from 100 runs ( $p = 55$  and  $n = 100$ ).

Scenario	$\beta$	L1		New ( $w = 1$ )		New ( $w = \sqrt{d}$ )		New ( $w = d$ )	
		Mean	SD	Mean	SD	Mean	SD	Mean	SD
1	$\beta_1 = 5$	0.53	0.29	0.04	0.04	0.27	0.26	0.67	0.35
	$\beta_1^{(1)} = \frac{5}{\sqrt{10}}$	0.11	0.17	0.14	0.15	0.10	0.10	0.07	0.08
	$\beta_2 = -5$	-0.55	0.30	-0.04	0.05	-0.28	0.32	-0.68	0.35
	$\beta_1^{(2)} = \frac{-5}{\sqrt{10}}$	-0.08	0.15	-0.18	0.15	-0.11	0.09	-0.08	0.08
2	$\beta_1 = 5$	0.76	0.33	0.09	0.06	0.34	0.16	0.91	0.40
	$\beta_1^{(1)} = \frac{5}{\sqrt{10}}$	0.09	0.14	0.20	0.14	0.14	0.11	0.09	0.08
	$\beta_2 = 3$	0.29	0.23	0.01	0.03	0.15	0.10	0.48	0.23
	$\beta_1^{(2)} = \frac{3}{\sqrt{10}}$	0.08	0.12	0.11	0.13	0.07	0.08	0.04	0.04
3	$\beta_1 = 5$	0.51	0.39	0.03	0.07	0.41	0.70	0.95	0.34
	$\beta_1^{(1)} = \frac{5}{\sqrt{10}}$	0.22	0.21	0.24	0.19	0.20	0.17	0.13	0.11
	$\beta_8^{(1)} = \frac{-5}{\sqrt{10}}$	-0.01	0.07	-0.01	0.11	-0.03	0.21	-0.04	0.12
	$\beta_2 = 3$	0.26	0.27	0.01	0.04	0.15	0.30	0.52	0.27
	$\beta_1^{(2)} = \frac{3}{\sqrt{10}}$	0.09	0.13	0.13	0.16	0.12	0.16	0.07	0.11
	$\beta_8^{(2)} = \frac{-3}{\sqrt{10}}$	0.001	0.07	0.004	0.06	0.01	0.05	-0.01	0.07
4	$\beta_1 = 5$	0.40	0.38	0.03	0.06	0.48	0.80	0.97	0.43
	$\beta_1^{(1)} = \frac{5}{\sqrt{10}}$	0.27	0.26	0.32	0.25	0.30	0.23	0.20	0.20
	$\beta_7^{(1)} = \frac{-5}{\sqrt{10}}$	-0.04	0.12	-0.02	0.14	-0.11	0.24	-0.09	0.16
	$\beta_2 = -3$	-0.23	0.29	-0.004	0.01	-0.21	0.45	-0.56	0.30
	$\beta_1^{(2)} = \frac{-3}{\sqrt{10}}$	-0.15	0.20	-0.16	0.19	-0.17	0.19	-0.09	0.13
	$\beta_7^{(2)} = \frac{3}{\sqrt{10}}$	0.03	0.08	-0.002	0.10	0.05	0.18	0.06	0.15

Table 3: Simulation results averaged over 100 runs for  $p = 550$  or  $1,100$  (22 informative and either 528 or 1,078 noise genes).

Method	Test Error (SE)		# False Negative (SE)		Model Size (SE)	
	$p = 550$	$p = 1,100$	$p = 550$	$p = 1,100$	$p = 550$	$p = 1,100$
STD	0.305 (0.003)	0.354 (0.002)	0.0 (0.0)	0.0 (0.0)	550 (0.0)	1,100 (0.0)
L1	0.218 (0.004)	0.235 (0.004)	16.6 (0.2)	17.1 (0.2)	16.1 (1.0)	19.2 (1.2)
New ( $w = 1$ )	0.232 (0.003)	0.255 (0.004)	14.9 (0.3)	15.6 (0.3)	20.7 (1.1)	22.6 (1.4)
New ( $w = \sqrt{d}$ )	0.202 (0.004)	0.221 (0.004)	5.7 (0.5)	6.7 (0.6)	32.6 (1.5)	34.6 (1.9)
New ( $w = d$ )	0.170 (0.003)	0.180 (0.004)	0.7 (0.3)	1.3 (0.4)	82.6 (5.4)	98.9 (7.2)

Table 4: PD data with 1,070 genes in the network. Classification error, number of selected disease genes, number of selected genes, and their standard errors (SE in parentheses) obtained by averaging over 10 runs. Five disease genes were *UBE1*, *PARK2*, *UBB*, *SEPT5*, and *SNCAIP*.

Method	Error (SE)	# Disease Genes (SE)	# Genes (SE)
STD	0.424 (0.016)	5.0 (0.0)	1,070.0 (0.0)
L1	0.464 (0.021)	0.1 (0.1)	19.2 (3.8)
New ( $w = 1$ )	0.476 (0.015)	0.1 (0.1)	24.9 (4.3)
New ( $w = \sqrt{d}$ )	0.480 (0.026)	0.2 (0.1)	30.6 (5.2)
New ( $w = d$ )	0.451 (0.028)	0.0 (0.0)	70.6 (14.1)
Final Model	-	1.0	75.0

Table 5: PD-1nb-net/PD-2nb-net. Classification error, number of selected disease genes, number of selected genes, and their standard errors (SE in parentheses) obtained by averaging over 10 runs. Eight disease genes were *UBE1*, *PARK2*, *UBB*, *SEPT5*, *SNCAIP*, *GPR37*, *TH*, and *SNCA*.

Network	Method	Error (SE)	# Disease Genes (SE)	# Genes (SE)
PD-1nb-net	STD	0.476 (0.023)	8.0 (0.0)	16.0 (0.0)
	L1	0.471 (0.017)	2.8 (0.7)	6.1 (1.5)
	New ( $w = 1$ )	0.462 (0.016)	3.4 (0.8)	7.3 (1.7)
	New ( $w = \sqrt{d}$ )	0.462 (0.014)	3.6 (0.7)	8.4 (1.5)
	New ( $w = d$ )	0.482 (0.015)	3.0 (1.2)	7.5 (2.1)
	Final Model	-	8.0	16.0
PD-2nb-net	STD	0.444 (0.016)	8.0 (0.0)	26.0 (0.0)
	L1	0.449 (0.017)	3.1 (0.5)	10.9 (2.1)
	New ( $w = 1$ )	0.464 (0.022)	5.3 (0.9)	13.2 (3.2)
	New ( $w = \sqrt{d}$ )	0.447 (0.023)	6.1 (0.8)	13.7 (2.7)
	New ( $w = d$ )	0.433 (0.016)	6.2 (0.9)	20.0 (2.5)
	Final Model	-	8.0	26.0

Table 6: BC-1nb-net/BC-2nb-net: 294/1,718 genes in total including 40/107 cancer genes, and 7/14 cancer genes with mutation frequencies larger than 0.10. Classification error, number of selected cancer genes with mutation frequencies larger than 0.10 (CA-LMF), number of selected cancer genes (CA), number of selected genes, and their standard errors (SE in parentheses) obtained by averaging over 10 runs.

Network	Method	Error (SE)	# CA-LMF (SE)	# CA (SE)	# Genes (SE)
BC-1nb-net	STD	0.371 (0.014)	7.0 (0.0)	40.0 (0.0)	294.0 (0.0)
	L1	0.357 (0.014)	0.3 (0.2)	4.6 (0.8)	32.3 (4.8)
	New ( $w = 1$ )	0.360 (0.014)	0.4 (0.2)	3.6 (1.1)	25.0 (7.0)
	New ( $w = \sqrt{d}$ )	0.366 (0.012)	0.6 (0.3)	4.7 (1.2)	27.2 (5.2)
	New ( $w = d$ )	0.399 (0.012)	1.2 (0.2)	7.8 (1.7)	40.2 (6.5)
	Final Model	-	1.0	4.0	14.0
BC-2nb-net	STD	0.351 (0.014)	14.0 (0.0)	107.0 (0.0)	1,718.0 (0.0)
	L1	0.360 (0.006)	0.0 (0.0)	2.4 (0.9)	42.9 (11.8)
	New ( $w = 1$ )	0.374 (0.011)	0.1 (0.1)	1.9 (0.5)	51.4 (12.6)
	New ( $w = \sqrt{d}$ )	0.360 (0.007)	0.2 (0.1)	2.5 (0.7)	41.7 (9.2)
	New ( $w = d$ )	0.385 (0.021)	0.3 (0.2)	0.7 (0.3)	34.2 (10.3)
	Final Model	-	1.0	2.0	23.0