Published in final edited form as:

Proc IEEE Int Conf Data Min. 2009 ; : 1004–1009. doi:10.1109/ICDM.2009.101.

Feature Selection in the Tensor Product Feature Space

Aaron Smalter¹, Jun Huan¹, and Gerald Lushington²

¹Department of Electrical Engineering and Computer Science, University of Kansas Lawrence, Kansas, United States

²Molecular Graphics and Modeling Laboratory, University of Kansas Lawrence, Kansas, United States

Abstract

Classifying objects that are sampled jointly from two or more domains has many applications. The tensor product feature space is useful for modeling interactions between feature sets in different domains but feature selection in the tensor product feature space is challenging. Conventional feature selection methods ignore the structure of the feature space and may not provide the optimal results. In this paper we propose methods for selecting features in the original feature spaces of different domains. We obtained sparsity through two approaches, one using integer quadratic programming and another using L1-norm regularization. Experimental studies on biological data sets validate our approach.

I. Introduction

Interaction prediction is the problem of using a set of known interactions between objects from one or more domains to predict unknown but possible interactions. Formally, the interaction prediction is a supervised learning problem with data sampled from domains $\mathcal{A}$ and $\mathcal{B}$ and a label set $\mathcal{Y}$. Samples take the form $\left\{ \left(\left(x_1^A, x_1^B \right), y_1 \right), \dots, \left(\left(x_n^A, x_n^B \right), y_n \right) \right\}$, $x_A \in \mathcal{A} \subseteq \mathbb{R}^{m^A}$, $x_B \in \mathcal{B} \subseteq \mathbb{R}^{m^B}$, $y_i \in \mathcal{Y}$. The learning task is to construct a model $f_{\text{pair}}: (\mathcal{A} \times \mathcal{B}) \rightarrow \mathcal{Y}$ that correctly classify new samples as interacting or not.

Interaction prediction finds natural applications in many fields. In bioinformatics, protein-protein and protein-chemical interaction prediction have been well studied [1], [9], [7]. In social network analysis, interaction prediction is also known as the link discovery or pairwise classification [17] problem. Problems studied in this area include coauthor prediction [4], [15] and web-site link prediction [31]. Broader areas with link prediction applications include predicting query-document links in information retrieval [22], user-item recommendations in collaborative filtering [21], and links among database records with the same identity (record linkage) [26].

Kernel methods are widely used in interaction prediction [1], [8], [16], [17]. In kernel methods, an interaction between two domains is first mapped to a kernel feature space using a kernel function, and then a linear classifier is obtained in the kernel feature space. The advantage of the kernel classifier such as SVM is that we do not need to explicitly compute the feature mapping, and rather compute the inner product of samples in the kernel feature space, which is computationally efficient for high dimensional data. However, a limitation of applying existing kernel classification methods to interaction prediction is that usually

they do not perform feature selection in a high dimensional feature space. Feature selection in high dimensional kernel feature space is important for improving classification performance since the underlying true model is usually sparse [5], [32]. Furthermore, a parsimonious model is easy to interpret and is preferred in many scientific and industry applications.

In general, feature selection in the kernel space is difficult since (i) data in the two interacting domains may not form a natural feature representation, and (ii) even if the data has a natural feature representation, the connection of the original feature space and the kernel feature space may be complicated. As a first step towards feature selection in kernel space in order to derive sparse models for the interaction prediction problem, we consider special cases in which we assume data in the two interacting domains have a feature vector representation.

The first special case we consider is derived from a vector representation of an interaction for a pair of objects by concatenating (or stacking) the features from each object. This case turns out to be not challenging since we have an implicit representation of the feature vector and hence we may use any feature selection to select highly informative features. Another special case, which has gained popularity recently [25], [17], [8] and which we focus on in this paper, is the use of tensor product features, where each feature in the tensor product space corresponds to the product of a pair of features in the original feature spaces. Recently the application of tensor product features has found use in a number of application areas. For example Tao et al. [25] define a general framework for supervised tensor learning and apply it to image recognition for computer vision. Oyama et al. [17] have developed a kernel based on the tensor product of two linear kernels for learning with feature conjunctions and apply their method to coauthor prediction as well as citation matching. Jacob and Vert [8] also use a kernel based on tensor product features, but apply their method to protein-ligand interaction prediction.

In this paper we examine two approaches towards efficient feature selection in the feature tensor product space. In the first, we use a linear kernel-based classifier, Support Vector Machine, and perform tensor product feature selection using an iterative process similar to the method of recursive feature elimination [3]. To begin, an SVM model is trained and the feature weights are obtained from it. These tensor product feature weights correspond to a matrix of weights between the original domain features. Selection of domain features is then formalized as an integer quadratic programming problem and optimized using a standard solver. Our next method leverages L_1 regularization with a non-linear logistic regression model. In this method the objective function is modified so that selection of domain features is enforced by optimizing selection of the domain features and not only the tensor product features.

We have performed a comprehensive experimental study of the two feature selection approaches outlined before using two benchmarks from the bioinformatics domain involving protein-chemical interaction prediction and protein-protein interaction prediction. Our experimental study show that our methods are able to outperform competing methods for feature selection such as recursive feature elimination and L_1 regularized logistic regression with high significance.

II. Background and Related Work

A. Kernel Functions and Connection to Explicit Features

As discussed in [17], there is a relationship between some kernel functions and explicit feature representations such as feature stacking and tensor product features. Consider first

the case where data are heterogenous, that is instances are comprised of features from two domains. One method for interaction prediction with object pairs such as x^A and z^B from domains $\mathcal{A}$ and $\mathcal{B}$ is to stack or concatenate the features and then apply such a polynomial kernel or linear kernel. Let x refer to the concatenation of x^A and z^B , with n and m referring to the number of features in domains $\mathcal{A}$ and $\mathcal{B}$, respectively. We use $\langle x^A, z^B \rangle$ to denote the inner product of x^A , the feature vector of an object in domain $\mathcal{A}$, and z^B , the feature vector of an object in domain $\mathcal{B}$. One possible kernel is the polynomial kernel, which includes single domain features such as $x_i^A z_j^A$ and $z_i^B z_j^B$ that are not informative in the heterogenous case. Another kernel corresponding to the tensor product between features instead of their concatenation is defined as,

$$K\left(\left(x^A, x^B\right), \left(z^A, z^B\right)\right)=\left\langle x^A, z^A\right\rangle\left\langle x^B, z^B\right\rangle \quad (1)$$

$$=\sum_{i=1}^n \sum_{j=1}^m x_i^A z_i^A x_j^B z_j^B \quad (2)$$

This corresponds to using tensor product features and includes only the products of features across domains, and excludes products between features in the same domain. In the case where a linear kernel is used, however, both of the kernels are equivalent. Yet another possible kernel definition is the product of kernels for domains $\mathcal{A}$ and $\mathcal{B}$, $K_A(x^A, z^A) \times K_B(x^B, z^B)$, which also corresponds to the tensor product feature space when K_A and K_B are defined as the inner product. An important note is that the reduction of the kernel function to a product of already established kernel functions is a useful approach, allowing the application of many sophisticated kernel functions for structured data. However, it focuses on the similarity between objects in the same domain, and not the interaction between objects from each domain, which is important for heterogenous data.

In the case where data are homogenous, that is the interacting domains are the same (such as in protein-protein interaction), this kernel is problematic since $K\left(\left(x_1^A, x_2^A\right), \left(z_1^A, z_2^A\right)\right)$ should be equivalent to $K\left(\left(x_2^A, x_1^A\right), \left(z_1^A, z_2^A\right)\right)$ since the order of x_1^A and x_2^A should not matter, however this is not the case. A better formulation for homogenous data is,

$$K\left(\left(x^A, x^B\right), \left(z^A, z^B\right)\right)=\left\langle x^A, z^B\right\rangle\left\langle x^B, z^A\right\rangle+\left\langle x^A, z^A\right\rangle\left\langle x^B, z^B\right\rangle \quad (3)$$

However this approach is problematic for heterogenous data where the inner product between x^A and z^B is not defined.

III. Algorithms

Here we propose two approaches to feature selection in the tensor product feature space. In the first, the feature selection process takes place iteratively, decoupled from the classifier model. In the second, feature selection is incorporated into the optimization criteria of the classifier itself.

One recurring note to keep in mind is the use of the tensor product operator, $\otimes$. This operation between vectors $v \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$, $v \otimes u$, results in a matrix $t \in \mathbb{R}^n \times \mathbb{R}^m$. For our purposes, this matrix is often transformed into a vector $s \in \mathbb{R}^{n \times m}$ using the following mapping: $s_k = v_i \otimes u_j$ where $k = (i-1) \times m + j$.

A. Linear Kernels for Tensor Product Feature Selection

We adopted the idea of SVM RFE [3] to select features in the feature tensor product space. Rather than directly apply RFE to select features in the tensor product space, our approach selects domain $\mathcal{A}$ and domain $\mathcal{B}$ features in the original space and hence obtains a subspace of the tensor product space. Consider an object from domain $\mathcal{A}$ represented by a set of features $A = \{a_1, a_2, \dots, a_n\}$ and an object from domain $\mathcal{B}$ represented by a set of features $B = \{b_1, b_2, \dots, b_m\}$, our goal is to select a subset of domain $\mathcal{A}$ features $A' \subset A$, and a subset of domain $\mathcal{B}$ features $B' \subset B$ to perform fast classification in the feature tensor product space. We formalize this intuition

$$\arg \max_{A', B'} \sum_{i \in A'} \sum_{j \in B'} W_{i,j} \quad (4)$$

subject to $|A'| = q$ and $|B'| = p$ where q and p are the desired number of features selected from each domain and $W_{i,j}$ is the weight of the feature formed by $a_i \times b_j$ in the tensor product space.

Features describing a cross-domain $\mathcal{A} - \mathcal{B}$ interaction are generated by taking the tensor product between the domain $\mathcal{A}$ feature vector and domain $\mathcal{B}$ feature vector. An SVM model is then generated using this combined feature set, and this model gives us the weights corresponding to each $\mathcal{A} - \mathcal{B}$ feature pair. These weights become the matrix W and a subset of features is selected that maximizes the sum over the submatrix W' . The remaining features are then used to again train the SVM model and the process repeats until the desired number of features has been selected.

1) Iterative Tensor Product Feature Selection with Integer Quadratic

Programming—In this section, we show the connection of the iterative tensor product feature selection problem to the mixed-integer quadratic programming problem (MIQP). Below we demonstrate the connection by rewriting the formalization of the bipartite feature selection problem as an integer quadratic programming problem.

$$\min_x \frac{1}{2} z^T H z \quad (5)$$

With $z_i \in \{0, 1\}$, $i = 1, \dots, n+m$ subject to the constraints $A_1 \cdot z = q$ and $A_2 \cdot z = p$. The z vector is a column binary vector. Given n domain $\mathcal{A}$ features and m domain $\mathcal{B}$ features, z has length $n + m$. $A_1 = [1, 1, \dots, 1, 0, 0, \dots, 0]^T$ is a binary column vector with a leading n number of ones followed by m zeros. $A_2 = [0, 0, \dots, 0, 1, 1, \dots, 1]^T$ is a binary column vector with a leading n number of zeros following by m ones. The matrix H corresponds to weights between domain $\mathcal{A}$ and $\mathcal{B}$ pairs, but we cannot use the weight matrix constructed from the SVM model directly since it is dimension $n \times m$. Instead, we must use a matrix that is $(n+m) \times (n+m)$, and embed the $n \times m$ weight matrix twice. The regions of H that correspond to within-domain $\mathcal{A} - \mathcal{A}$ or $\mathcal{B} - \mathcal{B}$ pairs are empty and add nothing to the minimization problem. The regions corresponding to $\mathcal{A} - \mathcal{B}$ pairs are filled with the proper weight from the SVM model. This weight is negated since the QP problem works on minimization while we are interested in maximization.

If the original weight matrix between domain $\mathcal{A}$ and domain $\mathcal{B}$ features is a $n \times m$ matrix, then H is a $(n + m) \times (n + m)$ matrix. MATLAB was used for solving the quadratic programming optimizations.

For learning on homogenous data, in order to enforce selection of only a single set of features in a single domain, the MIQP problem can be simplified. Instead of mapping W into H twice, we may simply use the negated W instead of H and use a single constraint.

B. Regularized Logistic Regression for Tensor Product Feature Selection

Given features derived from two domains, $A = f_1^A \dots f_{m^A}^A$ and $B = f_1^B \dots f_{m^B}^B$ our goal is to select subsets of those features, $r^A \in \{0, 1\}^{n^A}$ and $r^B \in \{0, 1\}^{n^B}$ which are bit vectors where each bit represents inclusion/exclusion of a feature. Features from each domain are then combined by taking the tensor product, $A \otimes B$. Here we explore the integration of feature selection into the logistic regression problem. The approach described here rests on the manipulation of r^A and r^B in the optimization problem to enforce domain space feature selection.

First, let the original L_1 -regularized logistic regression optimization problem be defined as,

$$\arg \min_w \frac{1}{n} \sum_{i=1}^n y_i x_i^T w - \log(1 + \exp(x_i^T w)) + \lambda \sum_{i=1}^m |w_i| \quad (6)$$

with data, $(x_i, y_i) \in \mathbb{R}^m \times \{-1, 1\}$, $i=1..n$, optimization variables $w \in \mathbb{R}^m$, $v \in \mathbb{R}$, and regularization parameter $\lambda \geq 0$. To integrate feature selection into the logistic regression problem, we must change both the basic optimization function as well as add penalization terms. With r^A and r^B as domain feature selection vectors, let $n = n^A * n^B$, $m = m^A * m^B$ and define $s \in \{0, 1\}^m$, $s = r^A \otimes r^B$ as the selection vector for the corresponding tensor product space. We will transform this into a diagonal matrix z ,

$$z \in \{0, 1\}^{m^A} \times \{0, 1\}^{m^B} \quad (7)$$

With $z_{i,j} = 0$, $i \neq j$ and $z_{i,j} = s_i$, $i = j$. Now we can rewrite the logistic regression problem as,

$$\arg \min_w \frac{1}{n} \sum_{i=1}^n y_i x_i^T w - \log(1 + \exp(x_i^T z w)) + L_1 \quad (8)$$

$$L_1 = \lambda \sum_{i=1}^m |w_i| * s_i \quad (9)$$

using s and z to enforce proper selection of features in the tensor product space. Finally, we must now add regularization terms to control the number of features selected in r^A and r^B . This final problem must optimize for these new variables:

$$\arg \min_{w, r^A, r^B} \frac{1}{n} \sum_{i=1}^n y_i x_i^T w - \log(1 + \exp(x_i^T z w)) + L_1 \quad (10)$$

$$L_1 = \lambda_1 \sum_{i=1}^m |w_i| * s_i + \lambda_2 \sum_{i=1}^{m^A} |r_i^A| + \lambda_3 \sum_{i=1}^{m^B} |r_i^B| \quad (11)$$

Note that, in this formulation, the number of features selected in each domain cannot be fixed to a specific number. Instead, the parameters λ_2 and λ_3 control the penalty for selecting

more features. Values for these parameters must be selected so the corresponding terms contribute to the minimization process, yet do not dominate it.

This problem formulation is intuitive, but requires several parameters and the use of mixed data types (binary and real-valued) that must be optimized. Instead, we adopt an alternative formulation and remove the binary $r \in \{0, 1\}$ vectors and replace them with $r \in \mathbb{R}^m$. We then set $w = r^A \otimes r^B$ and optimize only r^A and r^B . This problem is written as,

$$\arg \min_{r^A, r^B} \frac{1}{n} \sum_{i=1}^n y_i x_i^T w - \log(1 + \exp(x_i^T w)) + L_1 \quad (12)$$

$$L_1 = \lambda_1 \sum_{i=1}^{m^A} |r_i^A| + \lambda_2 \sum_{i=1}^{m^B} |r_i^B| \quad (13)$$

Where $w = r^A \otimes r^B$. This form of the problem is mathematically more attractive, though perhaps less intuitive. We have implemented both models and found this simpler model to provide better performance, and hence have used this model in our experimental studies.

1) Coordinate Descent for Regularized Logistic Regression—To solve the convex optimization problem, we implemented an algorithm that sequentially optimizes each single variable using a line search, and hence refer to it as a coordinate descent method. To optimize a feature r_j^A the gradients of the loss and penalty terms are calculated as,

$$\nabla r_j^A = \frac{1}{n} \sum_i \frac{e^{-y_i(x_i^T z w)} * (-y_i \sum_k^{m^B} x_{i,l} * w_l * r_k^B)}{1 + e^{-y_i(x_i^T z w)}} + \lambda_1 * \frac{r_j^A}{\sqrt{r_j^{A^2} + \epsilon}} \quad (14)$$

where $l = (j - 1) * m^B + k$, giving the index for w corresponding the the j 'th r^A feature and k 'th r^B feature. The equations are similar for a feature r_k^B . For learning on homogenous data the implementation is altered so that optimization is done to features in only one domain and mirrored in the other (identical) domain.

IV. Experimental Study

A. Data Sets

Chemical-Protein Interaction—We obtained protein sequences and chemical structures the from DrugBank[27] database. The FDA “approved” drug sets of chemicals and proteins were taken, containing 1,382 chemicals and 1,608 proteins. The number of chemical structures was further cut to 1000 by removing chemicals which contain heavy atoms and compounds that we could not parse. The data set contains 3,115 interactions.

Protein-Protein Interaction—We obtained protein-protein interaction data from the Human Protein Reference Database (HPRD) [19]. This database contains 25,661 proteins and 38,167 protein-protein interactions manually curated from literature. A set of 1000 proteins was randomly selected with 125,888 interactions. This label set is less sparse than the chemical-protein interaction data, but still contains < 15 % positives.

Synthetic Negatives—In both chemical-protein and protein-protein interaction data, the true negatives are not known. For each known positive interaction selected, between say chemical A and protein B , we select an unknown negative interaction between chemical A

and protein C such that the similarity between B and C is low, according their protein feature vectors. A similar process is used in the case of protein-protein interaction.

Feature Extraction—For interaction prediction we must generate features for both the proteins and chemicals. For proteins, we use the set of frequent k -mers [12]. For chemicals, frequent subgraphs were mined and each chemical is also described by a binary feature vector [24], [6]. A chemical-protein pair is then represented by 1) the tensor product between the two feature vectors or 2) the concatenation (stacking) of the feature vectors. Parameters for frequent k -mers and subgraphs are tuned to obtain about 50 features in both cases, although the exact number changes for each trial and cross-validation fold. For frequent k -mers, $k = 3$ was chosen with a support of 27%. For subgraph features, the support threshold was 61% with a subgraph size ranging from 5 to 10 vertices. The entire sets of about 50 features for each domain were used, for a total of about 2500 features in the tensor product space and about 100 using stacked features.

B. Model Construction

1) Training and Testing Samples—For each data set, a random subset of 500 positive and 500 negative interactions are sampled. This subset is then randomly divided into 5 cross-validation folds which are then used for model training and testing. For the SVM linear kernel experiments, in each of the 5 cross-validation experiments, the training data is first used to perform feature selection and next a series of internal 5-fold cross validation experiments are performed on the training data with selected features in order to select the best model parameters. For the logistic regression experiments, since feature selection happens simultaneously with model construction, feature selection and model parameter selection both takes place during the internal 5-fold cross validation process. The final model in both cases is then taken using the best model parameter with selected features. This model is then tested on the cross-validation test data and the performance is recorded. This entire process is repeated over 5 trials, using different random subsets of 500 positives and negatives.

2) Feature Selection

Tensor Product Feature Selection with Linear Kernel—In this approach, we use a linear kernel classifier (SVM) and couple it to a RFE-like feature selection process based on maximizing a weighted submatrix of selected features. We compare this method to SVM-RFE using both tensor product and stacked features. The number of features selected was explicitly set to 50% for all methods, and feature selection is performed before the model parameters are selected.

Logistic Regression with Tensor Product Feature Selection—The second feature selection approach embeds the feature selection process into the regularized logistic regression problem. In order to show the feasibility of solving this new optimization problem, we implemented a coordinate descent algorithm. We compare our tensor product logistic regression formulation to L_1 regularized logistic regression using both stacked features and tensor product features. The number of selected features using this L_1 regularization is not set explicitly, but is controlled through the model parameter λ .

3) Model Parameter Selection—For SVM classification, within each cross-validation trial, after selection of the best 50% of the features, an internal 5-fold cross-validation experiment is performed to select model parameters. A CSVC model is used and the C parameter is chosen through cross validation using a simple grid search.

In the L_1 logistic regression model, the λ parameter must be selected. The coordinate descent algorithm adds another parameter to choose, the step size γ . Using the tensor product logistic regression model, we add an additional λ parameter. Selection of the optimal parameters is a difficult process, requiring many trials testing each parameter combination. Because of this parameter selection difficulty, we set the γ parameter to 1 and set $\lambda_1 = \lambda_2$. The best parameter is selected using grid search.

C. Model Evaluation

Each method is evaluated in terms of accuracy, sensitivity and specificity on testing samples. Accuracy is defined as $(TP + TN)/S$ where TP is number of true positives, TN is number of true negatives and S is the total number of testing samples. Sensitivity $(TP/(TP + FN))$ and specificity $(TN/(TN + FP))$ are also collected. FP is number of false positives and FN is number of false negatives. For statistical significance, we use a two-way ANOVA test and report p-values.

D. Analysis of Results

Table I presents the results of classification experiments using both linear kernel with SVM and logistic regression. Significance of the accuracy of the MIQP-TPFS and L1-TPFS methods compared to the other methods is indicated with asterisks.

1) Chemical-Protein Interaction—The MIQP-TPFS method shows generally improved performance, except in the case of specificity where the RFE-TPF method is slightly better. The accuracy gains from the MIQP-TPFS method are significant with high confidence.

The L1-SF and L1-TPF methods have the same accuracy, however both methods (as well as L1-TPFS) have more balanced sensitivity/specificity with some preference toward sensitivity. The L1-TPFS method shows improved accuracy with high confidence. The specificity of L1-TPFS is also improved over the other methods, however both L1-SF and L1-TPF have better sensitivity.

2) Protein-Protein Interaction—The RFE-SF method shows a 7 % increase in accuracy over RFE-TPF, however, the sensitivity/specificity of RFE-TPF is much more balanced. The MIQP-TPFS method shows significant gains in accuracy over the RFE methods, and has improved sensitivity as well, while the RFE-SF method has slightly better specificity.

For protein-protein interaction with logistic regression, L1-SF and L1-TPF perform almost identically. The L1-TPFS method has significantly improved accuracy and is skewed toward sensitivity where it is better than the other methods, but is worse than the other two with respect to specificity.

V. Future Work

In the future we will explore applications of our feature selection strategies to other classifiers. Of particular interest is the embedding of the tensor product feature selection process into the SVM model. The use of tensor product feature selection as a wrapper for SVM shows promise, and hence integrating this process into the model training process may increase accuracy and efficiency.

Acknowledgments

This work has been partially supported by the KU Center of Excellence for Chemical Methodology and Library Development (NIH/NIGM award P50 GM069663), and the NSF award IIS 0845951.

References

1. Ben-Hur A, Noble WS. Kernel methods for predicting protein-protein interactions. *Bioinformatics*. 2005; 21(Suppl 1):i38–i46. [PubMed: 15961482]
2. Gomez S, Noble W, Rzhetsky A. Learning to predict protein-protein interactions from protein sequences. *Bioinformatics*. 2003; 19(15):1875–1881. [PubMed: 14555619]
3. Guyon I, Weston J, Barnhill S, Vapnik V. Gene selection for cancer classification using support vector machines. *Machine Learning*. Jan.2002 46:389–422.
4. Hasan, MA.; Chaoji, V.; Salem, S.; Zaki, M. Link prediction using supervised learning.. *Proc. of SDM 06 workshop on Link Analysis, Counterterrorism and Security*. 2006.
5. Hastie, T.; Tibshirani, R.; Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag; 2001.
6. Huan, J.; Wang, W.; Prins, J. Efficient mining of frequent subgraph in the presence of isomorphism.; *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM)*. 2003. p. 549-552.
7. Jacob, L.; Hoffmann, B.; Stoven, V.; Vert, J-P. Virtual screening of gpcrs: an in silico chemogenomics approach. Technical Report HAL-00220396. French Center for Computational Biology; 2008.
8. Jacob L, Vert J-P. Protein-ligand interaction prediction: an improved chemogenomics approach. *Bioinformatics*. 2008; 24(19)
9. JL F, M M, S M, K S, R S. Genome scale enzymemetabolite and drug-target interaction predictions using the signature molecular descriptor. *Bioinformatics*. 2007; 24(2):225–33. [PubMed: 18037612]
10. Koh, Kim; Boyd. An interior-point method for large-scale l1-regularized logistic regression. *J. Machine Learning Research*. 2007
11. Kohavi R, John GH. Wrappers for feature subset selection. *Artificial Intelligence*. 1997; 97(1-2): 273–324.
12. Leslie, C.; Eskin, E.; Noble, W. *Proceedings of the Pacific Symposium on Biocomputing 2002*. World Scientific; Singapore: 2002. The spectrum kernel: a string kernel for svm protein classification..
13. Li, Zhang, Wang, Zhang, Chen. Alignment of molecular networks by integer quadratic programming. *Bioinformatics*. 2007
14. Li, F.; Yang, Y.; Xing, EP. From lasso regression to feature vector.. *Advances in Neural Information Processing Systems*. 2005.
15. Liben-Nowell, D.; Kleinberg, J. The link prediction problem for social networks.. *Proceedings of the Twelfth Annual ACM International Conference on Information and Knowledge Management*. 2003.
16. Martin S, Roe D, Faulon J-L. Predicting protein-protein interactions using signature products. *Bioinformatics*. 2005; 21(2):218–226. [PubMed: 15319262]
17. Oyama; Satoshi; Manning; D, C. Using feature conjunctions across examples for learning pairwise classifiers.. *15th European Conference on Machine Learning (ECML2004)*. 2004.
18. Popescul, A.; Ungar, L. Statistical relational learning for link prediction.. *Proc. of the Workshop on Learning Statistical Models from Relational Data*. 2003.
19. Prasad, T. S. K. e. a. Human protein reference database - 2009 update.. *Nucleic Acids Research*.
20. Quanz, B.; Huan, J. Aligned graph classification with laplacian regularized logistic regression.. *Proceeding of the SIAM International Conference on Data Mining (SDM09)*. 2009.
21. Resnick, P.; Iacovou, N.; Suchak, M.; P., B.; Riedl, J. Grouplens: An open architecture for collaborative filtering of netnews.. *ACM Conference on Computer-Supported Cooperative Work*. 1994.
22. Salton, G. *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*. Addison Wesley; Reading, MA: 1989.
23. Shevade, Keerthi. A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*. 2003

24. Smalter, A.; Huan, J.; Lushington, G. Structure-based pattern mining for chemical compound classification.. Proceedings of the 6th Asia Pacific Bioinformatics Conference. 2008.
25. Tao D, Li X, Wu X, Hu W, Maybank SJ. Supervised tensor learning. Journal of Knowledge and Information Systems. 2007; 13
26. Winkler W. Advanced methods for record linkage. Technical report. 1994
27. Wishart DS, Knox C, Guo AC, Shrivastava S, Hassanali M, Stothard P, Chang Z, Woolsey J. Drugbank: a comprehensive resource for in silico drug discovery and exploratin. Nucleic Acids Res. 2006; (1)
28. Wu S, Zou H, Yuan M. Grouped and hierarchical model selection through composite absolute penalties. Annals of Statistics to appear. 2008
29. Yuan M, Lin Y. Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society, Series B. 2006; 68(1)(1-2):4967.
30. Zelenko D, Aone C, Richardella A. Kernel methods for relation extraction. Journal of Machine Learning Research. 2003; 3:1083–1106.
31. Zhu, J.; Hong, J.; Hughes, J. Using markov models for web site link prediction.; Proceedings of the thirteenth ACM conference on Hypertext and hypermedia. 2002. p. 169-170.
32. Zou H, Yuan M. F_{∞} norm support vector machine. Statistica Sinica. 2008; 18:379–398.

Table I

Results for linear kernel and logistic regression methods for chemical-protein (CPI) and protein-protein interaction (PPI) prediction. MIQP-TPFS refers to our tensor product feature selection method implemented as a mixed integer quadratic programming problem, RFE-SF refers to RFE feature selection with stacked features, and RFE-TPF refers to RFE feature selection with tensor product features. L1-TPFS refers to our tensor product feature selection method, L1-SF refers to L_1 regularized regression with stacked features, and L1-TPF refers to L_1 regression with tensor product features. Accuracy values marked with an asterisk are significantly better than the other two methods with p-values both less than 0.01; values marked with a double asterisk are better with p-values less than 0.001.

Data Set	Measurement	MIQP-TPFS	RFE-SF	RFE-TPF	L1-TPFS	L1-SF	L1-TPF
	Accuracy	0.63**	0.43	0.43	0.51**	0.47	0.47
CPI	Sensitivity	0.65	0.33	0.50	0.55	0.57	0.58
	Specificity	0.61	0.62	0.46	0.46	0.42	0.41
	Accuracy	0.59**	0.48	0.45	0.51*	0.47	0.46
PPI	Sensitivity	0.57	0.30	0.44	0.57	0.44	0.45
	Specificity	0.63	0.74	0.57	0.42	0.56	0.55