

# Discovering Compact and Highly Discriminative Features or Feature Combinations of Drug Activities Using Support Vector Machine

Hwanjo Yu<sup>1</sup>, Jiong Yang<sup>1</sup>, Wei Wang<sup>2</sup>, Jiawei Han<sup>1</sup>

<sup>1</sup>Department of Computer Science, University of Illinois, Urbana-Champaign, IL 61801, USA  
{hwanjoyu,jioyang,hanj}@cs.uiuc.edu

<sup>2</sup>Department of Computer Science, University of North Carolina, Chapel Hill, NC 27599, USA  
weiwang@cs.unc.edu

## Abstract

Nowadays, high throughput techniques such as microarray experiments make it feasible to examine and collect massive data at the molecular level. These data, typically mapped to a very high dimensional feature space, carry rich information about functionalities of certain biological entities and can be used to infer valuable knowledge for the purposes of classification and prediction. Typically, a small number of features or feature combinations play deterministic roles in functional discrimination. The identification of such features or feature combinations is of great importance. In this paper, we study the problem of discovering compact and highly discriminative features or feature combinations from a rich feature collection. We employ the support vector machine as the classification means and aim at finding compact feature combinations. Comparing to previous methods on feature selection, which identify features solely based on their individual roles in the classification, our method is able to identify minimal feature combinations that ultimately have deterministic roles in a systematic fashion. Experimental study on drug activity data shows that our method can discover descriptors that are not necessarily significant individually but are most significant combinatorially.

**Keywords:** Feature Selection, Support Vector Machine, Drug Activity

## 1 Introduction

Due to the high throughput of modern experiment techniques, analyzing and modeling the massive biological data becomes increasingly important and has become a crucial step to derive useful biological knowledge. During the past a few years, many computational models have been built and deployed to perform functional classification and annotation. It is often the case that all potentially relevant features are collected and used to build the classification model. Such model is typically very complicated as it captures the dependency of the target function to features or feature combinations. It has been well known that only a subset of features may play deterministic roles in the target function and that many collected features may be correlated. How to evaluate the importance of a feature or a feature combination with respect to the target function and how to discover the set of essential features has become an active research topic. This will lead to a deeper understanding of the system and to a more succinct model. Correlation mining and feature selection methods are two popular approaches to tackle these issues.

- **Correlation mining techniques** find most discriminative rules (or feature combinations) using a statistical metric (e.g., chi-square test). However, those rules are not very compact nor very discriminative since they do not take into account mutual information between feature

or feature combinations, that is, they evaluate each rule independently. Also the complexity of those methods is fundamentally exponential to the number of features since considering feature combinations is a combinatorial problem.

- **Feature selection methods using support vector machine (SVM)** extract a highly discriminative feature set using a discriminative classifier (i.e., SVM) In terms of discrimination (or classification) quality, these methods are better than correlation mining methods based on statistical metrics. However, due to the fact that SVM model is “hidden”, these methods do not disclose the relations within the feature set. Also the classification accuracy of the derived feature set is unreliable especially when the total number of features are large.

In this paper, we focus on finding a compact set of features or feature combinations that mostly discriminates (or classifies) the target class. For example, in Figure 1(A), let  $d$  be the target class and the others are features. Then a set  $\{\{\bar{f}\}, \{a, b\}\}$  is a most discriminative compact set of feature combinations. The set  $\{\{\bar{f}\}, \{a, b\}, \{a, b, e\}\}$  is also discriminative but not *compact* since the feature combination  $\{a, b, e\}$  is *redundant* of  $\{a, b\}$ .

## 1.1 Key ideas of our method

1. We take advantage of SVM’s kernel trick (polynomial kernel) to find  $k' (\ll k)$ , where  $k$  is the number of features and  $k'$  is the minimum degree of feature combinations that needs to be considered to generate the highest classification performance. In other words, considering more than  $k'$  number of feature combinations would not generate higher classification performance but decrease the *generalization* performance. (An example of “redundant” feature sets is given at the end of Section 2.1.) SVM’s kernel trick allows us to find  $k'$  in linear time in terms of  $k$ .
2. After we generate a SVM model using the polynomial kernel of degree  $k'$ , we extract the ranked features or feature combinations from the model, which would be much less time-consuming since we know  $k'$  which is much smaller than  $k$ .
3. To find a small subset of highly discriminative feature combinations, we can perform the recursive feature elimination of [9] but with the polynomial kernel of degree  $k'$ .

Our experiments in Section 5 show that our method produces better feature sets which give higher classification accuracy than the state-of-the-art feature selection method. In addition, our method provides more information – the relations between those feature sets.

## 1.2 Paper layout

The remainder of this paper is organized as follows. Section 2 gives an overview of research related to our work. The behavior of the SVM polynomial kernel and our method of discriminative feature combination discovery are presented in Sections 3 and 4, respectively. Section 5 shows the experimental study on drug activities and discuss the results. The conclusions are drawn in Section 6.

# 2 Related Work

## 2.1 Correlation mining techniques

For example of Figure 1(A), we want to find a set of features or feature combinations that discriminates a target feature  $d$  (or highly correlated the target class  $d$ ). Each row represents a data record, and each column denotes a feature. “1” indicates the presence of the feature in the row, and “0” indicates the feature’s absence.

$a$	$b$	$c$	$d$	$e$	$f$
1	1	1	1	1	0
1	1	0	1	1	0
1	0	1	0	1	1
1	0	0	0	1	1
0	1	1	0	1	1
0	1	0	0	1	1
0	0	1	1	1	0
0	0	0	1	1	0

(A) Examples of Transactions

$I \rightarrow C$	support	confidence	correlated ?
$\{a\} \rightarrow \{b\}$	25%	50%	No
$\{a\} \rightarrow \{e\}$	50%	100%	No
$\{a, b\} \rightarrow \{d\}$	25%	100%	Yes
$\{a, b, e\} \rightarrow \{d\}$	25%	100%	Yes
$\{f\} \rightarrow \{d\}$	0%	0%	No
$\{\bar{f}\} \rightarrow \{d\}$	100%	100%	Yes
...	..	..	..

(B) Examples of Association Rules

Figure 1: Transactions and Association Rules

### Association rule mining

Let  $I$  be a feature set, and let  $Pr(I)$  denote the ratio of the number of data records that include  $I$  to the number of all records. We call  $Pr(I)$  the *support* of  $I$ . An association rule has the form  $I_1 \rightarrow I_2$ , where  $I_1$  and  $I_2$  are disjoint feature sets. The support of  $I_1 \rightarrow I_2$  is defined as  $Pr(I_1 \cap I_2)$ , while the *confidence* is  $Pr(I_1 \cap I_2)/Pr(I_1)$ . For example, the support and the confidence of  $\{a, b\} \rightarrow \{c\}$  are 12.5% and 50% respectively.

The support of a feature set is *anti-monotone* w.r.t. set-inclusion of feature sets; that is, for any  $I \subseteq J$ ,  $Pr(I) \geq Pr(J)$ . Thus, whenever a feature set is not large w.r.t. a minimum support threshold, neither is any of its supersets. The Apriori algorithm uses this heuristic to effectively prune away a substantial number of unproductive feature sets [2, 3]. There have been numerous further works of the association rule mining based on the support-confidence framework.

The support-confidence framework is weak at catching correlations between feature sets. For example, the first and second rules of Figure 1(B) statistically do not make sense because in each rule the assumptive feature set, say  $I$ , and the conclusive feature set (or target class), say  $C$ , are independent. On the other hand, in the third rule of Figure 1(B), the assumption and the target class are highly and positively correlated.

### Correlation mining

Discovering correlation of feature sets which discriminates the target class typically calls for statistical measures such as chi-squared value, entropy information gain, gini index, or interclass variance. Correlation-based mining methods use such measures, say chi-squared value, for minimum threshold instead of using support and confidence.

The revision of the Apriori algorithm adopting the chi-squared test has been investigated [4, 1]. One suffered from generating too many uncorrelated rules due to still using the support threshold [4] while the other pushed too restrictive constraint to output some desired rules [1]. S. Morishita suggested a scalable statistical pruning method by computing an upper bound of a statistical metric such as chi-squared value, but the upper bound of the statistical metric is only valid for binary feature set [11].

Correlation techniques have the following limitations:

- They are not scalable to the size of features: considering every possible combination of features is a combinatorial problem.
- They do not yield compact feature sets. For example, in Figure 1, although the third and fourth rules have the same correlation values, the fourth one ( $\{a, b, e\} \rightarrow \{d\}$ ) is a redundant rule of the third rule ( $\{a, b\} \rightarrow \{d\}$ ).

- Complementary feature sets that individually do not separate well the data are missed. Evaluating the discrimination power of an individual feature set does not take into account mutual information between feature sets.

## 2.2 Feature selection methods using SVM

Many feature ranking algorithm using correlation coefficients have been explored with many applications to bioinformatics [6, 8, 7]. Recently, feature selection methods using SVM have been researched to find a small subset of features that maximize the classification performance.

The recursive feature elimination (RFE) algorithm for gene selection for cancer classification was proposed by I. Guyon et al. [9]. The RFE algorithm normalizes each feature value into the same range, train a SVM model from the normalized data, and use the weight of each feature as the indication for the importance of the feature. When the feature values are normalized well, the weight of each feature derived from the SVM model implies the contribution of the feature for discriminating the target class. However, their method is limited to a linear kernel which is often too restrictive to express a good discriminative function for many biological data. Also linear kernels do not take into account feature combinations so it can rank only individual features. Our method is similar to RFE in the sense that we also utilize the weight information to find a set of most discriminative feature but we discover feature combinations using SVM polynomial kernels.

J. Weston et al. have recently proposed a feature selection method for nonlinear kernels. Feature selection for nonlinear kernels is fundamentally a combinatorial problem which requires searching over all possible subsets of features. They use gradient descent method to approximate the results. However, the gradient descent method introduces another parameter, and improper setting of the parameter could cause jig-jag effect or slow convergence. Even when the parameter is optimized well, the converged point is still a local minimum. When the kernel is complex nonlinear and the number of features are large, there could exist numerous local minima in which the performance of the gradient descent method is strongly dependent on the randomly-set starting point. In our experiments, compact feature sets generated by our method show higher classification accuracy than the feature sets found by the local minimum search method. Moreover, the local minimum search methods cannot discover the relations between the features. That is, they are for data classification not for data interpretation.

We discover the compact subsets of highly discriminative features or feature combinations by exploring SVM polynomial kernels. Our method has higher discrimination (or classification) power than the local minimum search methods, and also disclose the relations of the feature sets that is beneficial to data interpretation.

## 3 Behavior of SVM Polynomial Kernel

### 3.1 SVM Overview

In machine learning theory, the “optimal” class boundary function (or hypothesis)  $h(x)$  given a limited number of training data set  $\{(x, y)\}$  ( $y$  is the label of  $x$ ) is considered the one that gives the best *generalization* performance which denotes the performance on “unseen” examples rather than on the training data. The performance on the training data is not regarded as a good evaluation measure for a hypothesis because the hypothesis ends up *overfitting* when it tries to fit the training data too hard. When a problem is easy to classify and the boundary function is complicated more than it needs to be, the boundary is likely overfit. When a problem is hard and the classifier is not powerful enough, the boundary becomes underfit. (Figure 2 shows examples of overfitting and underfitting.) SVM is an excellent example of supervised learning that tries to maximize the generalization by maximizing the *margin* and also supports nonlinear separation using advanced

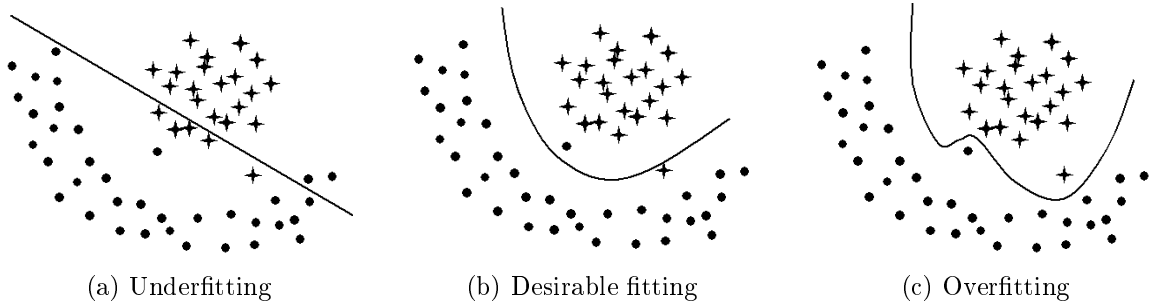


Figure 2: Example of boundaries overfitting and underfitting

kernels, by which SVM tries to avoid overfitting and underfitting [5, 12]. The *margin* in SVM denotes the distance from the boundary to the closest data in the feature space.

In SVM, the problem of computing a margin maximized boundary function is specified by the following quadratic programming (QP) problem:

$$\begin{aligned}
 \text{minimize : } \quad & W(\alpha) = - \sum_{i=1}^l \alpha_i + \frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l y_i y_j \alpha_i \alpha_j k(x_i, x_j) \\
 \text{subject to : } \quad & \sum_{i=1}^l y_i \alpha_i = 0 \\
 & \forall i : 0 \leq \alpha_i \leq C
 \end{aligned}$$

The number of training data is denoted by  $l$ ,  $\alpha$  is a vector of  $l$  variables, where each component  $\alpha_i$  corresponds to a training data  $(x_i, y_i)$ ;  $x_i$  is a vector representation of an example, and  $y_i$  is the class label of the example  $x_i$ .  $C$  is the soft margin parameter controlling the influence of the outliers (or noise) in training data.

The kernel  $k(x_i, x_j)$  for linear boundary function is  $x_i \cdot x_j$ , a scalar product of two data points. The nonlinear transformation of the feature space is performed by replacing  $k(x_i, x_j)$  with an advanced kernel, such as polynomial kernel  $(x^T x_i + 1)^d$ . The use of an advanced kernel is an attractive computational short-cut, which foregoes an expensive creation of a complicated feature space. An advanced kernel is a function that operates on the input data but has the effect of computing the scalar product of their images in a usually much higher-dimensional feature space (or even an infinite-dimensional space), which allows one to work implicitly with hyperplanes in such highly complex spaces.

### 3.2 Behavior of Polynomial Kernel

A polynomial kernel on feature set  $\mathcal{F}$  generates the same classification results as a linear kernel on feature set  $\mathcal{F}'$ —the combinations of each feature in  $\mathcal{F}$ —without an explicit creation of the feature set  $\mathcal{F}'$ . The polynomial kernel has a parameter  $d$  which denotes the degree of feature combinations. For instance, SVM with the polynomial kernel of  $d = 3$  computes the function  $f(x)$  formulated as:

$$f(x) = \sum_i^n w_i x_i + \sum_i^n \sum_j^n w_{ij} x_i x_j + \sum_i^n \sum_j^n \sum_k^n w_{ijk} x_i x_j x_k, \quad (1)$$

where  $x_i$  is the  $i$ th feature of an example  $x$  and  $w_i$  is the weight of the feature  $x_i$ . The classification function (1) takes into account the combinations of every feature up to the third degree. When  $d = 2$ , a two-dimensional feature space of feature set  $\{x_1, x_2\}$  will be transformed into a higher-dimensional space that includes also the second degree of each feature —  $\{x_1, x_2, x_1 x_2, x_1^2, x_2^2\}$ .

---

Input:     - Training data set  $X = [(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)]$   
          - Discrimination (classification accuracy) threshold  $\theta$   
Output:    - A compact set of feature combinations  $F$

Algorithm:

```
/* Find  $d$  of the best performance */  
  
1. Initialize  $d = 1$   
2. Repeat  
  2.1.  $M = \text{train\_SVM}(X, d)$   
  2.2. estimate the performance of  $M$   
  2.3. if the performance is estimated lower than that of previous loop, then exit the loop  
  2.5. increase  $d$  by 1  
  
/* Extract a compact set of feature combinations whose classification accuracy is higher than  
the threshold  $\theta$  */  
  
3. Initialize  $F =$  the feature set  
4. Repeat  
  4.1.  $H = \text{compute\_weight}(M, d)$  /* See Figure 4 */  
  4.2.  $F' = \text{return\_top\_feature}(H, \text{len}(F) - \delta)$   
  4.3.  $X = \text{rebuild}(X, F')$   
  4.4.  $M = \text{train\_SVM}(X, d)$   
  4.5. if  $\text{test}(M)$  is lower than  $\theta$ , then exit the loop  
  4.6.  $F = F'$   
5. Return  $F$ 
```

Figure 3: Feature Combination Discovery (FCD) Algorithm

---

The boundary function becomes more powerful and complicated as  $d$  increases, and at some point the boundary function starts overfitting and degrading its generalization performance. Figure 2(a) shows an example of underfitting when  $d$  is too low. Figure 2(c) is a case of overfitting when  $d$  is too high. Overfitting due to high  $d$  corresponds to generating redundant feature sets in the correlation mining of Section 2.1. The  $d$  showing the peak performance is different depending on the data set.

## 4 Feature Combination Discovery (FCD) Algorithm

Considering all the combinations of features to discover a discriminative subset of feature combinations fundamentally takes an exponential time in terms of the number of features. However, SVM with a polynomial kernel is able to draw a classification function that considers  $d$  degree of feature combinations in linear time regardless of  $d$ . The key idea of our method is first to find the  $d$  of best generalization performance. (The best  $d$  is usually small in biological data sets. In all our experiments,  $d \leq 3$ .) Then we compute the weight of each feature combination up to  $d$  degree from the SVM model and thus avoid the expensive and fruitless search of higher degrees of combinations. The computed feature combinations will be compact and highly discriminative.

---

Input:     - Support vectors and corresponding coefficients  $S = [(x_1, c_1), (x_2, c_2), \dots, (x_n, c_n)]$   
           - Degree  $d$   
 Output:   - A hash table  $H$  of a ranked list of feature combinations

Algorithm:

1. Initialize a hash table  $H$ .  
    /\* each key is an attribute combination and the value is the weight of the attribute combination \*/
  2. Repeat for each support vector  $x_i = (a_1, a_2, \dots, a_m)$ 
    - 2.1. Repeat for each attribute combination  $A$  whose number of combination is equal to or less than  $d$ 
      - 2.1.1. Multiple the coefficient  $c_i$  of  $s_i$  to the value of  $A$  and accumulate the results into the hash table  $H[A]$
- /\* At this point, the hash table  $H$  includes the key-value pairs of every feature combination less than  $d$  degree and the corresponding weight of the attribute \*/
3. Sort the hash table  $H$  by its values
  4. Return  $H$

Figure 4: *compute\_weight(M, d)*

---

Figure 3 describes the FCD algorithm which extracts the most compact subset of feature combinations that generates higher classification accuracy than the user threshold  $\theta$ . The first part of the algorithm – Steps 1 and 2 – finds the degree  $d$  of feature combinations showing the peak performance by using the SVM polynomial kernel. As we discussed in Section 3, a polynomial kernel with  $d$  on a feature set  $\mathcal{F}$  generates the same effects as a linear kernel on  $d$  degree of feature combinations over  $\mathcal{F}$ . As  $d$  increases, the classification function starts overfitting and degrading the performance at some point, at which we stop the iteration. In our experiments in Section 5, the performance start degrading after  $d = 2$ .

There are several ways to estimate the generalization performance of SVM such as cross validation or estimation of leave-one-out errors. Cross validation techniques have been used for estimating the generalization error of a decision rule. However, it is computationally too expensive to perform iteratively in our framework. Estimating the SVM generalization performance has been researched actively based on a strong theoretical foundation. One of the most recent methods, estimation of leave-one-out errors [10], has shown to be efficient and fairly accurate. We used this method in our experiments.

Given the generated SVM model of  $d$  degree polynomial kernel, Steps 3 to 5 extracts the most compact subsets of feature combinations that generates higher classification accuracy than the user threshold  $\theta$  as follows:

- Step 4.1. computes the weight of each feature combination. Figure 4 describes how to extract the weights from a SVM model of polynomial kernel with degree  $d$ . First a hash table  $H$  is initialized whose key will be an attribute combination and the value will be the weight of the attribute combination. The key of the combination of more than  $d$  degree will be excluded. An SVM model is a collection of  $(x_i, c_i)$ , where  $c_i$  is the non-zero coefficient of an example  $x_i$ , which corresponds to  $y_i \alpha_i$  in the QP formula. Only support vectors will have non-zero  $c_i$ . The

weight of a feature combination  $A$  will be determined by the support vectors that includes the elements of  $A$ . For instance, if  $A = a_1a_2$ , the weight of  $A$  is the sum of the coefficients of the support vectors that have both  $a_1$  and  $a_2$ .

- Step 4.2. removes the least discriminative  $\delta$  number of features from the previous feature set  $F$  by taking the highest ranked  $len(F) - \delta$  number of features. This removing step is equal to that of recursive feature elimination in [9]. For computational reasons, it may be more efficient to remove several features at a time at the expense of possible classification performance degradation. With the lower *delta*, it is likely to generate the more compact set [9]. Our experiments in Section 5 also show that a more compact set generated 100% accuracy with  $\delta = 1or5$  than with *delta* = 10.
- Steps 4.3. and 4.4. re-construct the training set with the reduced feature set and train another SVM model from the training set.
- As iterating Step 4., the classification performance may start decreasing at some point, and we return the feature set before the performance becomes lower than the user threshold. Our experiments show that the classification performance stays stable until the point that the performance starts decreasing rapidly (See Figure 5).

We discuss the computational complexity of our algorithm at the end of Section 5.

## 5 Experimental Evaluations

In this section, we experimentally evaluate our method (the FCD algorithm). To evaluate the discrimination quality and the compactness of the feature combinations derived by the FCD algorithm, we compare the classification performance of the derived feature set with the state-of-the-art feature selection method [13]. The relations within the subset are disclosed as a by-product in the FCD algorithm.

We use a dataset of 294 chemical compounds. A total of 264 features are collected. Each feature is a numerical descriptor that represents a property of the compound structure, such as mass, diameter, etc. The target classification is on whether the compound is an active drug.

We used LIBSVM version 2.36<sup>1</sup> for the SVM implementation. We fixed the soft margin parameter; the classification accuracy is insensitive to the parameter when it is set in a reasonable range and the data set does not have a significant amount of noise.

The SVM with linear kernel showed less than 30% accuracy on the this data set, which implies the data set is not linearly separable. However, when we use SVM with polynomial kernel, it showed 100% accuracy with  $d \geq 2$ . Thus we run the recursive elimination with the polynomial kernel with  $d = 2$ , and it generated the results shown in Figure 5.

As we discussed in Section 2.2, the classification performance of the local minimum search method is unstable because the performance depends on the starting point and there exists numerous local minimum points when the SVM involves a large number of features with nonlinear kernels. In Figure 5, FCD with  $\delta = 1$  or 5 shows 100% accuracy with 50 features whereas the local minimum search method requires at least 80 features to achieve 100% accuracy. (We report the average accuracy of five runs of their method.) The feature sets generated by the local minimum search show lower classification accuracy than those generated by FCD at each number of features.

Table 5 shows the 50 features that the FCD (with  $\delta = 5$ ) extracts, and Table 5 shows within the 50 features the top 10 positive and negative feature combinations respectively sorted by their absolute weights. It is obvious that the best descriptor combinations are not necessarily composed of the most important individual descriptors.

<sup>1</sup><http://www.csie.ntu.edu.tw/~cjlin/libsvm>

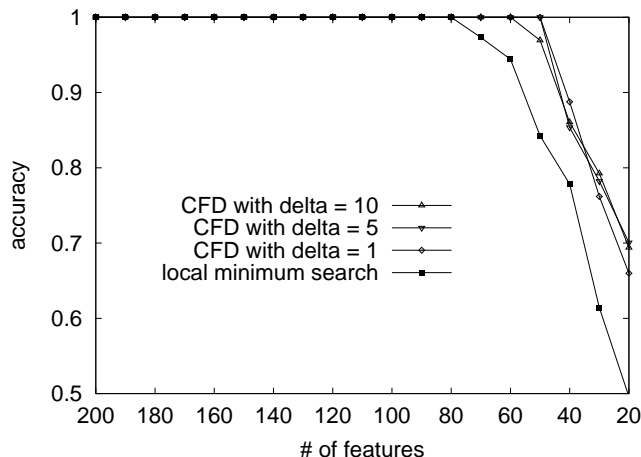


Figure 5: Classification performance ( $d = 2$ )

ZM1V, SNar, Xt, RHyDp, Jhetp, X0, X5A, X0v, X1v, X0Av, X2Av, X3Av, PW3, PW4, PW5, PJI2, CSI, ECC, MDDD, UNIP, IDE, IDM, IDET, HV<sub>cpx</sub>, Uindex, SIC0, BIC0, SIC1, BIC1, IC2, SIC2, BIC2, IC3, SIC3, BIC3, IC4, BIC4, IC5, BIC5, LP1, SEigv, SEige, SEigp, AEigm, VRA1, VRZ2, VRm2, D\_Dr07, D\_Dr11, T\_N\_S-

Table 1: The 50 Discriminative Feature Derived by FCD

Using higher degree of polynomial kernel could allow to extract even more compact features – less than 50 features – which generates 100% accuracy. For instance, in our experiments, SVM with  $d = 3$  reduced into 30 the number of features needed to generate 100% accuracy. (See Figure 6.) However, as we discussed in Section 3, the generalization performance of the function would degrade. The 50 features we extracted here with  $d = 2$  can be viewed as the most compact set that generates the same classification and generalization performance as the original data set which also generates the best performance with  $d = 2$ .

### Computational considerations

The FCD algorithm runs SVM iteratively whose training time is quadratic to the number of data set ( $n$ ) and linear to the number of features ( $m$ ). Our experiments indicated that better features are obtained with smaller  $\delta$  in the iteration of FCD. However, there are only significant differences for the smaller subset of features (e.g., less than 100), which suggests that, without trading accuracy for speed, one can remove chunks of features at the beginning of the iterations and remove smaller number of features as it iterates. Then, we can approximate the number of total iteration into  $\log(m)$ . Extracting the weights from a polynomial kernel with  $d$  degree takes  $m^d$  time. Thus the complexity of FCD becomes  $O(n^2 * m^{d+1} * \log(m))$  if the complexity for training a SVM is  $O(n^2 * m)$ .

In our experiments on the 294 data records of 264 features, using the LIBSVM, training a SVM with polynomial kernel with  $d = 2$  takes less than two seconds in a Pentium III 800Mhz machine. Running the FCD on the data set took about two hours in the same environment. However, it is quite acceptable in bioinformatics that the data analysis takes a few hours because the data collection and preparation usually take several much longer, e.g., several months or years.

When the data set needs a high degree of combinations, our current method could become extremely slow because, as we see from the complexity, the time for extracting the weights from a

Positively Discriminative Combination	Negatively Discriminative Combination
IDET · SIC1 (15.99)	UNIP · IDET (-11.88)
ECC · HV <sub>cpx</sub> (14.08)	X0 <sub>v</sub> · X2A <sub>v</sub> (-10.98)
PW4 · HV <sub>cpx</sub> (12.41)	ECC · SEigp (-10.79)
ECC · ECC (10.81)	RHyDp · ECC (-10.64)
PJ12 · ECC (10.59)	IDET · BIC0 (-10.38)
ECC · IDM (10.34)	IDE · IDE (-10.34)
UNIP · LP1 (9.86)	ECC · AEigm (-9.96)
IDET · BIC1 (9.84)	ECC · Uindex (-9.91)
CSI · IC3 (9.79)	CSI · ECC (-9.89)
Xt · CSI (9.63)	X0 <sub>v</sub> · X3A <sub>v</sub> (-9.74)
...	...

Table 2: Positively and Negatively Discriminative Feature Combinations and Their Weights

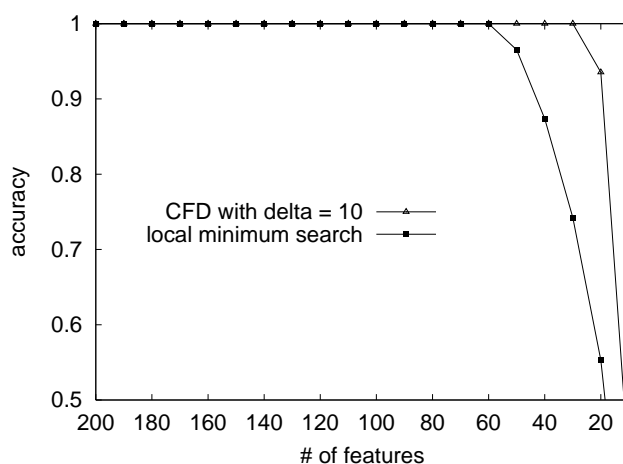


Figure 6: Classification performance ( $d = 3$ )

SVM model has an exponential dependency to the degree of the combinations. Approximating the weights for fast extraction could be an interesting further research for this feature selection problem.

## 6 Conclusions

In this paper, we present a new method that uses SVM weight information to extract discriminative feature combinations from a rich feature space. We have shown that this method delivers a promising result on the drug activity data that has a large number of features and a relatively small number of instances. It successfully identifies a much smaller set of features that produce a better classification quality than the (larger) set of features selected by other methods. As one direction of our future work, we shall evaluate the trend of the classification quality as the number of features increases and develop a more systematic solution to determine the form of kernel functions. We also plan to deploy the proposed method on other biological data such as the gene expression data and the protein subcellular location data.

## References

- [1] C. C. Aggarwal and P. S. Yu. A new framework for itemset generation. In *Proc. 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 18–24, Seattle, Washington, 1998.
- [2] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., 1993.
- [3] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499, Santiago de Chile, Chile, 1994.
- [4] S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: generalizing association rules to correlations. In *Proc. ACM SIGMOD International Conference on Management of Data*, pages 265–276, Tucson, Arizona, 1997.
- [5] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.
- [6] R. Duda and P. Hart. *Pattern classification and scene analysis*, 1973.
- [7] T. S. Furey, N. Christianini, N. Duffy, D. W. Bednarski, M. Schummer, and D. Haussler. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10):906–914, 2000.
- [8] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, pages 531–537, 1999.
- [9] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.
- [10] T. Joachims. Estimating the generalization performance of a SVM efficiently. In P. Langley, editor, *Proceedings of ICML-00, 17th International Conference on Machine Learning*, pages 431–438, Stanford, US, 2000. Morgan Kaufmann Publishers, San Francisco, US.
- [11] S. Morishita and J. Sese. Traversing itemset lattice with statistical metric pruning. In *Proc. of the 19th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 226–236, Dallas, Texas, 2000.
- [12] V. N. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, 1998.
- [13] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In *Advances in Neural Information Processing Systems (NIPS)*, pages 668–674, Vancouver, Canada, 2000.