ON FEATURE SELECTION FOR GENOMIC SIGNAL PROCESSING AND DATA MINING

S.Y. Kung

Princeton University, USA

ABSTRACT

An effective data mining system lies in the representation of pattern vectors. The most vital information to be represented is the characteristics embedded in the raw data most essential for the intended applications. In order to extract a useful high-level representation, it is desirable that a representation can provide concise, invariant, and/or intelligible information on input patterns.

The curse of dimensionality has traditionally been a serious concern in many genomic applications. For example, the feature dimension of gene expression data is often in the order of thousands. This motivates exploration into feature selection and representation, both aiming at reducing the feature dimensionality to facilitate the training and prediction of genomic data. The challenge lies in how to reduce feature dimension while conceding minimum sacrifice on accuracy.

For feature selection, both individual and group information are important, and each has its own pros and cons in measuring the truly relevant information. The individual quantification is simple as each of the M features can be represented by one single value. However, it cannot deal with the inter-feature redundancy, abounding specially in genomic data. In contrast, the group information can fully address the mutual redundancy, but it is often too complicated to process. (Note that there are 2^M possible groups.) Between the two extremes, fortunately, there is a convenient compromise: the pairwise kernel - which has a low complexity (M^2 pairs) and yet reveals the critical information regarding the m inter-feature redundancy. Indeed, it has been already found very useful for many genomic applications. Especially, we shall describe how pairwise-based feature selection may be successful applied to genomic subcellular localization. A special method (VIA-SVM) designed exclusively for pairwise scoring kernels is introduced. This is the first method that fully utilizes the reflexive property of the so-called self-supervised training data, arising uniquely available in multiple sequence alignment. Based on several subcellular localization experiments, the VIA-SVM when combined with some filter-type metrics appears to deliver a substantial dimension reduction (one-order of magnitude) with only little degradation on accuracy.

1. INTRODUCTION

Let N denote the number of training data samples, M the original feature dimension, the full raw feature can be expressed as a set of M-dimensional vectors:

$$\mathbf{x}(t) = [x_1(t), x_2(t), \ldots, x_M(t)]^T, \quad t = 1, \ldots, N.$$

The subset feature can be denoted as an m-dimensional vector process

$$\begin{aligned} \mathbf{y}(t) &= [y_1(t), y_2(t), \cdots, y_m(t)]^T & (1) \\ &= [x_{s_1}(t), x_{s_2}(t), \cdots, x_{s_m}(t)]^T & (2) \end{aligned}$$

where $m \leq M$ and s_i stands for index of a selected feature.

From the machine learning's perspective, one metric of special interest is the Sample-Feature-Ratio is $\frac{N}{M}$. For many multimedia applications, the sample-feature-ratios lie in a desirable range. For example, for speech data, the ratio can be as high as 10:1 or 100:1 in favor of training data size. For machine learning, such a favorable ratio plays a vital role in the training validation and statistical significance.

Unfortunately, for genomic data, this is often not the case. It is common that the number of samples is barely compatible with, and sometimes severely outnumbered by, the dimension of features. In such situation, it becomes imperative to remove the less relevant features, i.e., features with low SNR [7].

1.1. Reduction of Dimensionality (Biological Perspectives)

In genomic applications, each gene (or protein sequence) corresponds to a feature in gene profiling (or protein sequencing) applications. Feature selection/representation has its own special appeal from the genomic data mining's perspective. As an example, for gene expression profiles, the following factors necessitate an efficient gene selection. [2]

1. **Non-proportionate feature dimension w.r.t. number of training samples**: For most genomic applications, the feature dimension is excessively higher than the size of the training data set. Some examples of the Sample-Feature-Ratios $\frac{N}{M}$ are:

Protein Sequences	$\longrightarrow$	1:1
Microarray Data	$\longrightarrow$	1:10 or 1:100

1-4244-1566-7/07/$25.00 ©2007 IEEE.

Such an extremely high dimensionality has a serious and adverse effect on the performance. First, high dimensionality in feature spaces increases the computational cost in both the (1) learning phase and (2) prediction phase. In the prediction phase, the more features used the more the computation required and the lower the retrieval speed. Fortunately, the prediction time is often linearly proportional to the number of features selected. Unfortunately, in the learning phase, the computational demand may grow exponentially with the number of features.

2. **Plenty of irrelevant genes**: From the biological view point, only a small portion of genes are strongly indicative of a targeted disease. The remaining "housekeeping" genes would not contribute relevant information. Moreover, their participation in the training and prediction phases could adversely affect the classification performance.

3. **Presence of co-expressed genes**: The presence of co-expressed genes implies that there exists abundant redundancy among the genes. Such redundancy plays a vital role and has a great influence on how to select features as well as how many to select.

4. **Insight into biological networks**: A good feature selection is also essential for us to study the underlying biological process that lead to the type of genomic phenomenon observed. Feature selection can be instrumental for interpretation/tracking as well as visualization of a selective few of most critical genes for *in-vitro* and *in-vivo* gene profiling experiments. The selective genes closely relevant to a targeted disease are called bio-markers. Concentrating on such a compact subset of biomarkers would facilitate a better interpretation and understanding on the role of the relevant genes. For example, for *in-vivo* microarray data, the size of the subset must be carefully controlled in order to facilitate an effective tracking/interpretation of the underlying regulation behavior and inter-gene networking.

1.2. Reduction of Dimensionality (Computational Perspectives)

High dimensionality in feature spaces also increases uncertainty in classification. An excessive dimensionality could severely jeopardize the generalization capability due to overfitting and unpredictability of the numerical behavior. Thus, feature selection must consider a joint optimization and sometimes a delicate tradeoff, of the computational cost and prediction performance. From the computational perspectives, two major and serious adverse effects are elaborated below:

- **Data over-fitting.** Note that over-optimizing the training accuracy often results in overfitting the dataset which in turns degrades generalization and prediction ability.

  It is well known that data overfitting may happen when the vector dimension is relatively too large when compared with the size of training data. In other words, what matters most to classification/generalization is the sample-feature-ratio, the ratio between the size of the training data set and the feature dimension.

  Unfortunately, for many genomic applications, the feature dimension can be as high or much higher than the size of the training data set. For these applications, overtraining could significantly harm generalization and feature reduction is an effective way to alleviate the overtraining problem.

- **suboptimal search.** Relatively, the computational resources available for genomic processing are never sufficient, given the astronomical amounts of genomic data needing to be processed. High dimensionality in feature spaces increases uncertainty in the numerical behaviors. As a result, a computational process often converges to a solution far inferior to the true optimum, which may compromise the prediction accuracy.

To sum up, it is commonly acknowledged that the more features selected the more information is made available. For examples, $I[A] \leq I[A \cup B)] \leq \ldots$ where A and B represent two features, say x_i and x_j, respectively, and $I(X)$ denotes information of X. This results in a monotonic curve in Figure 1(a). However, when the feature size is too large, the degree of sub-optimality must reflect the performance degradation caused by data over-fitting and limiting computational resource. (See Figure 1(b).) This implies a non-monotonic property on achievable performance w.r.t. feature size, as shown in Figure 1(c). Accordingly, but not surprisingly, the best performance is often achieved by selecting an optimal subset of features.

Here, let us use a subcellular localization example as an evidence to support such a non-monotonic performance curve and highlight the importance of feature selection.

Example 1 Subcellular Localization. *Profile alignment SVMs [8] are applied to predict the subcellular location of proteins in an eukaryotic protein dataset provided by Reinhardt and Hubbard, 1998 [9]. The dataset comprises 2427 annotated sequences extracted from SWISSPORT 33.0, which amounts to 684 cytoplasm, 325 extracellular, 321 mitochondrial, and 1097 nuclear proteins. 5-Fold cross validation was used to obtain the prediction accuracy. The accuracy and testing time for different number of features selected by*

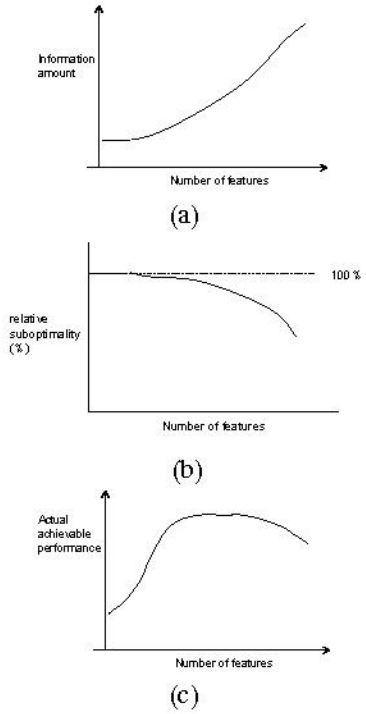


Figure 1: (a) Monotonic increasing property on the total information available. Upper Curve: Non-monotonic increasing property on the actual classification performance achievable. (b) Relative performance versus the feature size taking into consideration of data over-fitting and limited computational resources. (c) The best performance is often achieved by selecting an optimal size instead of the full set of the features available.

a Fisher-based method [12] are shown in Figure 2. This example offers an evidence of the non-monotonic performance property based on real genomic data.

□

2. FEATURE FORMATS: AXIS-BASED VERSUS VECTOR-BASED FORMAT

Feature selection and representation depends very much on the format in which features are mathematically described.

Note that for microarray data, there are very few samples and very large numbers of genes. Commonly, there are few samples[1] and several thousands to tens of thousands genes. (For example, in a typical yeast database, there are around 7000 genes with less than 20 samples.) For most

[1]Fewer than 100 samples available for training and testing altogether

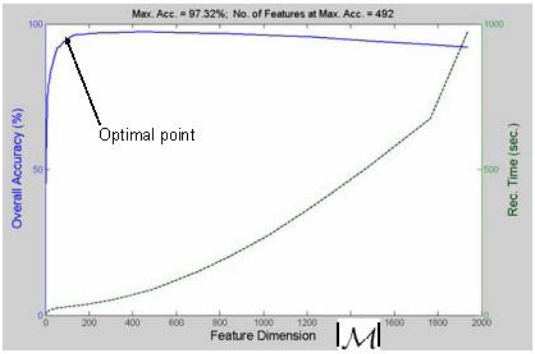


Figure 2: Real data supporting the Monotonic increasing property. Upper Curve: Performance reach a peak by selecting an optimal size instead of the full set of the features available. Lower curve: the computational time goes up (more than linear rate) as the number of features increases.

applications, the genes are the features of interest, i.e., one gene corresponds to one feature.

To illustrate this point, let us use a fictitious microarray matrix $M \in \Re^{3\times 2}$ with 3 genes and 2 conditions as displayed in Figure 3. In order to use a vector space to describe the data shown in the table, it is necessarily to first specify the bases in which the vector space is represented. When the data are pictorially displayed, such bases correspond to the axes in a coordinate system.

There are are two possible bases, each leading to its own mathematical format:

1. **Axis-Based Feature Format**

   In the first case, the bases corresponds to the features (genes).

   Figure 3(a), one feature is represented by one axis.

   In general, the expression levels of M genes in N samples can be represented by N vectors an M-dim vector space. Here each vector (sample) is represented as an M-dim vector:
   $\mathbf{x}_n = [x_n(1), x_n(2), \cdots, x_n(M)]^T, n = 1, \cdots, N.$

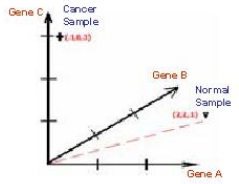
   In this case, feature selection becomes the question of axis selection, e.g., which of the three genes should be selected.

2. **Vector-Based Feature Format**

   In the above, the bases corresponds to the genes, which are regarded as features in this context. Alternatively, the bases corresponds to the conditions, cf. Figure 3(b).

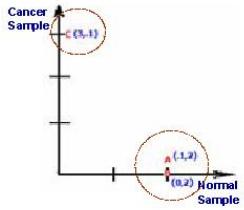
   In this case, the expression levels of M genes in N samples are represented by M vectors in an N-dim vector space. Here each feature is represented as an N-dim vector:

	Normal Sample	Cancer Sample
Gene A	2.0	0.1
Gene B	2.0	0.0
Gene C	0.1	3.0



(a)features=axes of the vector space

	Gene A	Gene B	Gene C
Normal Sample	2.0	2.0	0.1
Cancer Sample	0.1	0.0	3.0



(b)features=vectors in the vector space

Figure 3: (a) In the Table, each row stands for one feature which is represented by one axis. The feature selection becomes the question of axis selection, e.g., which of the three genes should be selected. The redundancy between the features can be determined by the similarity between the features. For example, features A and B have high similarity, due to high (pairwise)correlation or the high predictability of one from the other. (b) In the transposed Table, each row stands for each of the two conditions: normal versus cancer. Each condition is corresponding to one axis. In this case each data point represents one gene. Genes A and B again exhibit high similarity. However, now the similarity is being manifested by their close distance in the Cartesian coordinates. Consequently, they are represented by the same cluster if the data points are divided by the subclusters.

$\mathbf{x}(m) = [x_1(m), x_2(m), \cdots, x_N(m)]^T, m = 1, \cdots, M.$ Thus, the feature selection problems involves the selection of most representative vectors.

There is advantage to transpose the coordinate systems such that each patient is corresponding to one of the axes while each gene is displayed as a vector. This brings us to a more manageable situation where there are a lot of (thousands) data points on a low-dimensional (less than 100) vector space. Note that the vectors with lower dimension are much easier to handle computationally.

In this case, gene selection is equivalent to selection of vectors. For this, feature cluster is a powerful notion for feature selection and representation. It often useful to to first find clusters with best cluster separability and then determine the best representative feature(s) for each of the clusters formed. [2]

[2]Notion of feature cluster does not apply under the context of axis-based format.

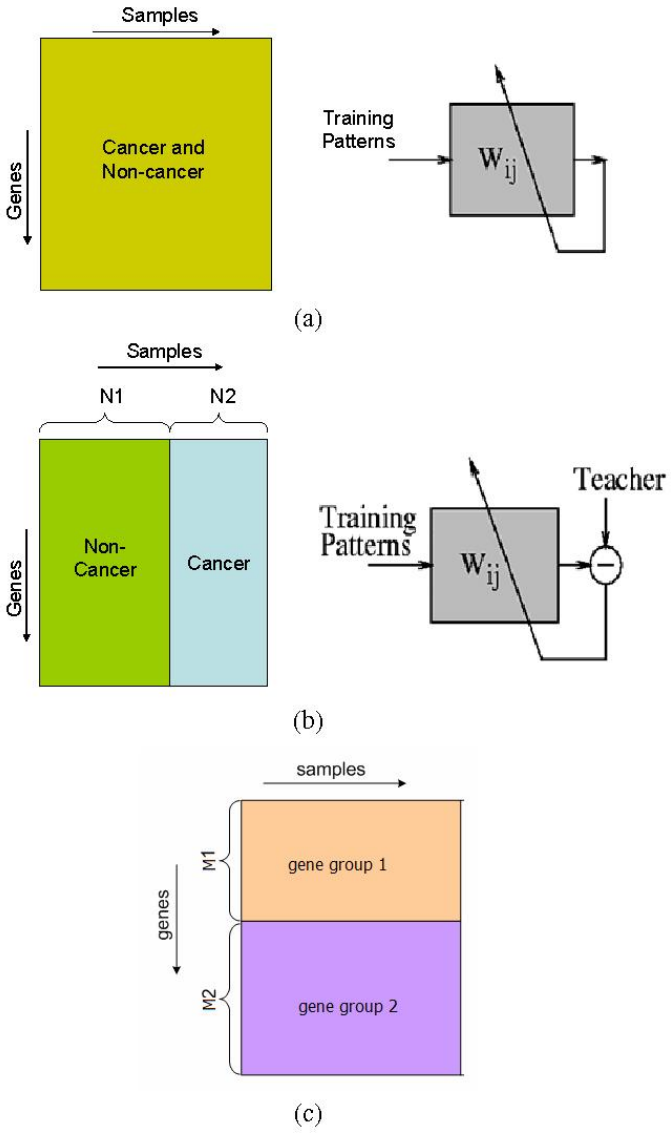


Figure 4: This figure illustrate the different prior knowledge of (a) unsupervised and (b) supervised, and (c) self-supervised training data. In the regular supervised training data, the class labels are assigned to the vectors. In the self-supervised situation. prior known group labels are assigned to the features, instead of the vectors.

3. UNSUPERVISED VERSUS SUPERVISED FEATURE SELECTION AND REPRESENTATION

The features selected serve very different objectives for unsupervised versus supervised learning scenarios, each inducing its own type of criterion.

3.1. Selection/Representation Criteria: Unsupervised Cases

In the unsupervised training data, the class labels are unknown a priori. See Figure 4(a). Under the *axis-based* format, there are two very different ways of designing the selection criteria.

- **Data fidelity perspective:**

  How well does the partial information reveal a full picture of the original information? The criterion is motivated by how much of the original information is retained (or lost) when the feature dimension is reduced. In terms of fidelity-driven metric, there are two major types:

  - One is based on the so-called mutual information: $I(\mathbf{x}|\mathbf{y})$.
  - Another is one which minimizes the reconstruction error:

$$\epsilon(\mathbf{x}|\mathbf{y}) \equiv \min_{\mathbf{y}\in\Re^m} \|\mathbf{x} - \hat{\mathbf{x}}_\mathbf{y}\|$$

    where $\hat{\mathbf{x}}_\mathbf{y}$ denotes the estimate of $\mathbf{x}$ based on $\mathbf{y}$.

- **cluster separability perspective:**

  How well does the partial information do in separating data into clusters each bears its own distinct biological significance. The criterion is motivated by how effectively can the selected features reveal the separability of the subclusters, which is crucial for the classification performance. A popular metric of this type is via a higher-order statistics, known as independent component analysis (ICA). For more discussion on this subject, see [10].

Under the *vector-based* format, many unsupervised clustering techniques, such as K-means and SOM, have been very popular. Moreover, these techniques may be used to cope with both the data fidelity and cluster separability perspectives.

3.2. Selection/Representation Criteria: Supervised Cases

In the regular supervised training data, the class labels are assigned to the vectors. See Figure 4(b). The state-of-the-arts feature selection methods have two types: open-loop (filter-type) and closed-loop (wrapper-type) approaches.

- **Filter Approach** A primitive supervised selection scheme is the so-called filter approach. Its processing simplicity makes it a promising and popular selection approach. For example, an SNR-type criterion based on the Fisher discriminant analysis is very appealing. [4] Such a feature selection approach entails computing

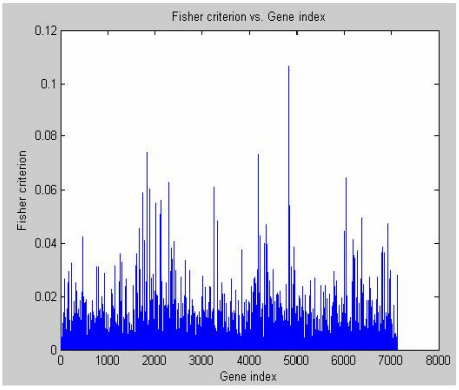


Figure 5: The Fisher discriminant ratio (FDR) of 7192 genes in the acute leukemia dataset [4]. The FDR cutoff point (threshold) and the corresponding number of remaining genes are shown in Table 1

Threshold	0.01	0.02	0.04	0.06
No. of Remaining Genes	596	142	19	8

Table 1: The FDR cutoff point (threshold) and the corresponding number of remaining genes.

Fisher's discriminant $J(x_i)$, $i = 1, \ldots, M$, which represents the ratio of inter-cluster distance to intra-cluster variance for each individual feature.

One variant of SNR is Fisher discriminant ratio (FDR) [11]:

$$\text{FDR}(j) = \frac{(\mu_j^+ - \mu_j^-)^2}{(\sigma_j^+)^2 + (\sigma_j^-)^2}. \tag{3}$$

As an illustrative example, Figure 5 shows the FDR of 7129 genes in an acute leukemia dataset [4] that contains two types of acute leukemia: ALL and AML. Evidently, only a small number of genes have large FDR, meaning that only a few genes are useful for differentiating the two types of acute leukemia. Table 1 shows the numbers of selected genes at different cutoff points.

- **Wrapper Approach** However, the ultimate objective for supervised cases lies in a high classification accuracies. Ideally speaking, if the classification information is known, denoted by C, a most direct measurement could be one based on the following type of mutual information: $I(C|\mathbf{x})$ versus $I(C|\mathbf{y})$. More exactly, it is desirable to have

$$I(C|\mathbf{y}) \to I(C|\mathbf{x})$$

while keeping the feature dimension m as small as possible. However, the above formulation is numeri-

cally difficult to achieve. The only practical solution makes use of the feedback from the actual classification result, which is computationally very demanding. The feedback-based method is related to the wrapper approach [14, 15], cf. Figure 4(b).

To strictly implement the exact formula given above will be computationally very demanding. Realistically speaking, motivated by computational consideration, a popular and simplifying assumption is to do feature selection is to make use of some kind of linear classification assumption. Let C_L denote linear classification. The goal is now focused on

$$w(C_L|\mathbf{y}) \rightarrow w(C_L|\mathbf{x})$$

so that the decision boundary based on the reduced feature set can best approximate that derived from the full feature. This forms the basis of the "wrapper approach", in which a large value of w_i implies that the $j-th$ feature is more important.

3.3. Self-Supervised Training Data

Assuming that the features are represented by the axes. In the regular supervised training data, the class labels are assigned to the vectors. In contrast, for the self-supervised situation, prior known group labels are assigned to the features, instead of the vectors. This is illustrated in Figure 4(c). [3] In genomic data mining applications, there arises such a peculiar type of "self-supervised" scenario. In Section 10, a subcellular localization example will be treated with great details.

4. FEATURE EVALUATION: RELEVANCE AND REDUNDANCY

Optimal feature selection/representation hinges upon the following two criteria:

1. **Relevance and Signal Strength:** *Individual Feature Criterion.* Only vital representations or features need to be extracted. Some features are much more relevant than others. Some genes exhibit very distinct responses between two different types of samples, but others have nearly random behavior. This makes one gene more relevant than the other. More exactly, under supervised framework, some feature have higher SNRs, as they exhibit distinct responses between two classes of samples, so they are the preferred features.

2. **Redundancy:** *Inter-Feature Criterion.*

[3] Another variant of self-supervision (not shown) is that, in addition, class labels are also assigned to the vectors.

Table 2: Some examples on how to make use of the features in different formats.

	axis-based format	vector-based format
Relevance:	SNR	length (rare[4])
Redundancy:	correlation	distance

There are also many features which are closely related and therefore carrying duplicated information. If such redundancy is well identified and obviated, it could allow a substantial reduction of gene dimension without missing much total information.

Recall that there are two possible ways to specify the coordinates: axis-based and vector-based formats. Accordingly, the relevance and redundancy information have distinct role for each of the two formats. For examples, under supervised scenario and the axis-based format, then the strength of a feature may be best reflected by its SNR [4]. Under unsupervised scenario and the axis-based format, then the inter-feature redundancy (between any two features) can best be represented by their correlation. A high correlation means a good predictability of one feature from another. Indeed, the greater the inter-feature redundancy, the higher the correlation, and then the better the predictability. On the other hand, assuming the vector-based format, i.e. features are represented by vectors in the vector space, then two features with close distance (or close similarity) are likely to bear high inter-feature redundancy. These examples are summarized in Table 2.

5. INDIVIDUAL VERSUS GROUP INFORMATION

A representative feature is the one that can represent a group of similar features. Denote S as a feature subset, i.e., $S \equiv \{y_i\}, i = 1, \cdots, m$. In addition to the general case, what of most interest is either a single individual feature $m = 1$ or a pair of features $m = 2$. A generic term $I(S)$ will be used temporarily to denote the information pertaining to S, as the exact form of it has to depends on the application scenarios.

There are two contrasting types of quantitative measurements of the feature information:

1. *Individual* information: The quantification cost is in the order of $O(M)$.

2. *Group* (three or more) information.

   There are totally $O(2^M)$ possible groups.

Both individual and group representations have their own pros and cons in measuring the truly relevant information.

The individual quantification is simple as each of the M features can be represented by one single value. However, it cannot deal with the inter-feature redundancy, abounding specially in genomic data. In contrast, the group information can fully address the mutual redundancy, but it is often too complicated to process.

5.1. Individual Feature Information

Given a single feature x_i, its information is denoted as $I(x_i)$. Such a measure is often the most effective when the features are statistically independent. This leads to the individual ranking scheme, which is suboptimal but the most straightforward and fast. In this scheme, each individual feature is independently and simultaneously evaluated. Therefore, ranking only considers the information and/or discriminative ability of individual features.

In unsupervised scenario, the signal strength can be adopted as the selection criterion. However, more effective applications arise for feature selection in supervised learning. Under supervised framework, some feature have higher SNRs, as they exhibit distinct responses between two classes of samples, so they are the preferred features.

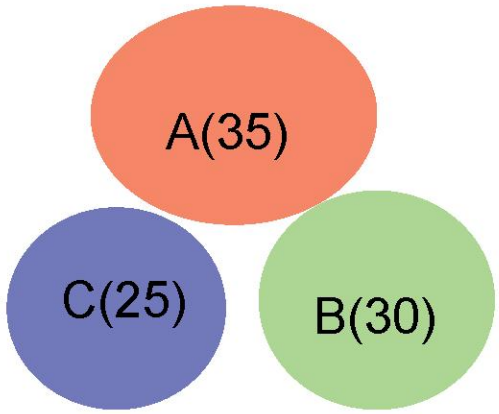
Let us use a hypothetical example to illustrate the individual ranking scheme.

Example 2 3-Party Problem – Without Inter-feature Redundancy.

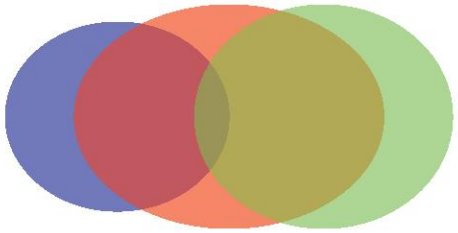
The individual ranking method works the best when the redundancy plays no or minimal role in affecting the final ranking. In this example, each area in Figure 6(a) and (b) represents one feature. The size of the area indicates the information or discriminativeness pertaining to a feature. In Figure 6 (b), 'overlapping" between areas reflect the degree of inter-feature redundancy. In the special case, such as Figure 6(a), no "overlapping" symbolizes no mutual redundancy. In this case, the combined information of any two features is simply the sum of two individual amount. For example: $I(A \cup B) = I(A) + I(B) = 35 + 30 = 65$. *When all the features are statistically independent, it corresponds to the fact that there is no overlap pictorially. All the selection schemes lead to the same and correct result. This is shown in Table 3(a).*

□

The downside of considering the feature individually is that it does not fully account for the redundancy among the features. For example, it is very possible that two highest-rank individual features share a great degree of similarity. As a result, the selection of both features would amount to a waste of resource. In fact, one needs to take the inter-feature relationship (such as mutual similarity/redundancy) into account. This problem can be fully resolved (possibly overdone) by adopting the group information approach discussed next.



3-Party Problem: (a) Without inter-feature redundancy



3-Party Problem: (b) With inter-feature redundancy

Figure 6: (a) 3-party problem without redundancy. No "overlapping" between elements symbolizes the fact that no mutual redundancy exists between the features. (b) 3-party problem with redundancy, where $A(35)$, $B(30)$, $C(25)$ exhibit a peculiar overlapping pattern. "Overlapping" between elements symbolizes the fact that mutual redundancy exists between the features.

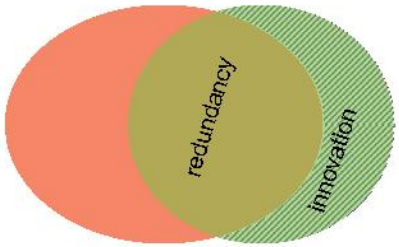
5.2. Group Feature Information

The individual information represent is computationally most simplistic. Full information to represent group (three or more features) will be highly involved and costly from the computational perspective.

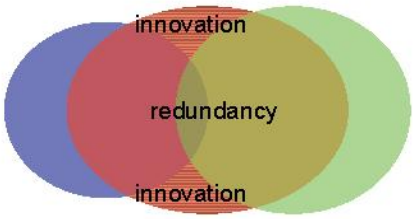
In order to optimally evaluate the total information contained in a subset of multiple-features, the most prudent approach is to have the entire group's information content, evaluated collectively as an undivided entity.

Example 3 3-Party Problem – With Inter-feature Redundancy.

The scenario is illustrated in Figure 6(b). If only one feature is to be selected, since $I(A) > I(B) > I(C)$, *the*



(a) Forward innovation: B w.r.t. A



(b) Backward innovation: A w.r.t. ABC

Figure 7: Innovations for 3-party problem with redundancy. (a) *Forward innovation*: Suppose that A is already chosen. We now search for a new member to admit to the subset. The redundancy is shown as the overlapping region of A and B. The innovation of B (w.r.t. A) is the remaining region after removal of its redundancy with A. Note that the innovation of B (w.r.t. A) is larger than C (w.r.t. A), therefore, B will be chosen as the new member. (b) *Backward innovation* of A w.r.t. ABC is equivalent to the net information loss when A is eliminated from the full set.

correct solution is A. If more than one feature are to be selected, then the redundancy among the features selected plays a role. For example, it is often the case that a gene A is very discriminative on its own, and so is gene B. But the information revealed by gene A overlaps significantly with that of gene B.

The inter-feature redundancy implies that

$$I(A \cup B) \leq I(A) + I(B) \tag{4}$$

In Figure 6(b), such inter-feature redundancy, is pictorially display as "overlapping" of 2 corresponding elliptic areas. The size of overlap indicates the the amount of "information discount" in Eq. 4.

In this example, there is a large overlap between A and B, which is, visually speaking, about half of the size of B ($\approx \frac{30}{2} = 15$). Therefore,

$$I(A \cup B) \approx 50 \leq 55 = I(A) + I(B)$$

Selection Method	Select 1 Feature	Select 2 out of 3 Features
Independent Ranking	A	AB
Forward Selection	A	AB
Backward Elimination	A	AB
Correct Solution	A	AB

3-Party Solution: (a) Without inter-feature redundancy

Selection Method	Select 1 Feature	Select 2 out of 3 Features
Independent Ranking	A	AB
Forward Selection	A	AB
Backward Elimination	B	BC
Correct Solution	A	BC

3-Party Solution: (b) With inter-feature redundancy

Table 3: (a)-(b) Tables illustrating search results of different strategies.

In contrast, there is virtually no overlap between B and C.

$$I(B \cup C) \approx 55 = I(A) + I(B)$$

Therefore, $I(A \cup B) \leq I(B \cup C)$. If a two-feature subset is to be selected, the optimal and correct solution is B and C. Note that the individual ranking would select A and B, because they have the two highest scores, which is an incorrect selection. See Table 3(b).

□

The importance of **group information** is evident:

- Unlike the individual ranking approach, the performance of group evaluation can take inter-feature redundancy into account. In fact, this is what theoretically required if the truly optimal decision is to be reached.
- Unlike the consecutive ranking (to be discussed momentarily), its solution is relatively fair, because whether a feature is evaluated earlier or later would not affect its selection or elimination.

The price of group ranking is, however, its excessively high computational cost. In order to find the best combination of features, an exhaustive search would consider every of 2^M possible combinations.[5] Such a formidable computation cost would render the group evaluation approach impractical in most, if not all, applications.

[5] Even if only those subsets of size m or lower are searched, it still amounts to $C_m^M \times 2^m$ possible combinations. The search space can be substantially reduced to only C_m^M groups, if the size of subsets is predetermined to be exactly m.

5.3. Tradeoff Between Information and Computation

It is evident that the individual information misses the inter-feature redundancy, while the group information is too computationally too costly to implement. Some remedies must be found.

- One viable solution is based on a consecutive search strategy in which features are added or dropped on a one-by-one basis (simplifying the computation by doing so) and at the same time taking into account the relevance and inter-feature redundancy. This is the subject of Section 6.
- Another option is based on the *pairwise* relationship. Computationally, its quantification cost amounts to only $O(M^2)$ pairs. Moreover, this approach can cope well with inter-feature information while incurring a very affordable computational cost. For more details, see Section 7.

6. INNOVATION AND ITS APPLICATION TO CONSECUTIVE SEARCH OF FEATURES

The consecutive ranking is an evaluation on a one-by-one basis. It provides a reasonable compromise between (1) accuracy and (2) cost.

1. The advantages are two folds. (1) accuracy: Relevance and inter-feature are both taken into account. See Figure 7. (2) It enjoys a substantial computational saving when compared with the comprehensive and exhaustive evaluation of the group scores.
2. The downside is clear too. The order of feature selection can significantly affect the inter-feature redundancy revealed, which in turns affect the final outcome of the selection. In other words, any feature's selection/elimiantion will depend on whether it is evaluated earlier or later.

There are two ways to conduct an consecutive search:

1. **Forward Search: Most Innovative First Admitted.** Such a search usually begins at an empty feature set, and then augments the membership by a most-innovative-first-admitted strategy. In the recursive search scheme, one feature is added to the existing chosen subset at each step. The *importance* of the candidate features is judged by by how much extra added-value it brings to the existing subset, instead of its individual merit or strength. In other words, the selection a feature depends exclusively on how well can it complement the current subset. [6] The search continues until either a preset number of features is reached or a pre-specified performance is achieved.

   To incrementally augment the current subset S (with m features) to S^+ (with $m+1$ features) involves a very manageable search space.

   At the m-th stage, There are $M-m$ candidate features to select from, thus there are exactly $M-m$ *forward innovations* need to be computed.[7] Pictorially, an example of the forward innovation is illustrated in Figure 7(a).

   (a) Computationally, it should be feasible to evaluate each of the $M-m$ groups and decide the best feature to admit, i.e. via a group ranking strategy.

   (b) However, this is not very computationally effective as it fails to utilize the fact all the $M-m$ groups are different by only one feature. This motivates to adopt a notion of *innovation component*, popular in the linear estimation literature.

2. **Backward Elimination: Most-Dispensable-First-Eliminated.**

   This is a recursive search scheme, usually starting at the full set. In this scheme, at each step, features are removed from the current subset based on a one by one basis. [8] In this case, it is more meaningful and direct to evaluate the information loss (with respect to the full set x) due to the absence of the dropped features from the set. This is denoted by $I(\mathrm{x})-I(\mathrm{y})$. The objective is to reduce the feature dimension as much as possible but giving up a minimum loss of performance (if any). Therefore, the feature which incurs the least loss of information is deemed the most dispensable, thus becoming a natural target to eliminate.

   To remove one feature from the current subset S (with m features) to obtain a new subset S^- (with $m-1$ features), we need to deal with a very manageable search space, because there are only m candidate features to select from. The incurred loss of information:

$$\text{Loss} = I(S) - I(S^-)$$

   will serve as the basis of backward innovation, which is pictorially illustrated in Figure 7(b).

Example 4 3-Party Problem: Forward Selection and Backward Elimination

[6] Once a feature is admitted, it will not be dropped anymore. In an iterative procedure, though, an opportunity is provided to have the relevance of all features get reevaluated again.

[7] In total, it amounts to $M+(M-1)+\cdots+1 = O(M^2)$ innovations.

[8] Once a feature is eliminated, it will not be reconsidered anymore.

Let's continue the example illustrated in Figure 6(b). In the forward search, A will be chosen in the first round. In the second round of selection, B will be a better choice to be teamed up with A. So the best two features would be A and B, again an incorrect selection. See the Table 3(b).

As already discussed just now,

As shown in Figure 7(b), feature A is deemed to be most dispensable since it causes the least loss of information. Therefore, even though A has the most information (i.e. the highest strength) by itself, it is nevertheless the first to be eliminated if reduction is necessary.

The good new is the the best two-feature subset contains B and C. This is correct (optimal) solution as it matches with the group evaluation. Unfortunately, with feature A already eliminated in the first round, this also spells a bad news. If only one feature is needed in the final selection, then the backward approach has to find a solution from the two remaining features B or C. But both are incorrect selection, because the correct solution A is already eliminated. Let us complete the backward search. In the second round, feature B would beat feature C and get selected. The search results are summarized in Table 3(b).

□

7. PAIRWISE FEATURE INFORMATION

7.1. Why Pairwise Information?

Both individual and group representations have their own pros and cons in measuring the truly relevant information. The individual quantification is simple as each of the M features can be represented by one single value. However, it cannot deal with the inter-feature redundancy, abounding specially in genomic data. In contrast, the group information can fully address the mutual redundancy, but it is often too complicated to process. (Note that there are 2^M possible groups.)

Between the two extremes, fortunately, there is a convenient compromise: the pairwise kernel. To represent pairwise information, it incurs a very affordable quantification cost is $O(M^2)$. Yet, the pairwise information reveals plenty of inter-feature information (such as similarity and redundancy). Moreover, the application of use of pairwise information for genomic processing has already a long history.

7.2. Types of Pairwise Matrix

The pairwise information appear in many forms, including:

1. Pearson correlation coefficient (for axis-based format)
2. Distance (for vector-based format)
3. Similarity (for sequence-based case)
4. Kernel function (needs verification of Mercer condition for some pairwise matrix).

7.3. When/How to Use Pairwise Information?

Given that pairwise quantification involves $O(M^2)$ scores, each very simple to compute, and that group-wise quantification amounts up to 2^M scores, each very costly to derive. This represents a very drastic computational contrast. The next important issue is how adequate can pairwise information be used to evaluate group information. A comprehensive answer would have to depend on for what kind of applications the information is intended. [9]

The pairwise information can be as effective as group-wise quantification in at least three important application scenarios:

1. Feature selection based on prediction (i.e. reconstruction) and mutual information. This should not be too surprising as the (pairwise) correlation and covariance coefficients are for long known to be effective for linear prediction. The details as to how to use pairwise information are to be deferred to Section 7.

2. Cluster discovery.

   Section 8 will show that the pairwise information is suitable - and in fact effective - for unsupervised cluster discovery, including k-means, hierarchical clustering, and SOM.

3. Self-supervised training sequence data.

   For pairwise and/or multiple sequence alignments, we aim to classify a very peculiar type of "self-supervised" training sequence data. Section 10 will show that how the pairwise information, in combination with SVM, can lead a novel and effective feature selection approach.

8. CASE STUDY ON PAIRWISE INFORMATION: FEATURE SELECTION BASED ON MUTUAL INFORMATION

This section will focus on the unsupervised learning from the axis-based perspective. Here, each feature is represented as an axis of the coordinate system, say x_i. To simplify the illustration, for all the unsupervised cases, we shall assume for the time being that all the feature vectors are zero-mean Gaussian random process. [10] The (pairwise) covariance

[9] As a counterexample, while pairwise information may be useful for problems with second-order statistics, such as predictability, it may not be too useful for ICA type processing, which involves a higher-order statistics.

[10] Thus x follows the distribution

$$p(x) = \frac{1}{\sqrt{(2\pi)^n |R_x|}} \exp\{-\frac{1}{2}(x^T R_x^{-1} x)\}$$

matrix is estimated as $R = \{r_{ij}\}$ where $r = E[x_i^T x_i] \approx \frac{1}{N}\sum_{t=1}^{N} x_i(t)x_j(t)$.

This section will purposely treat feature selection and feature representation side by side. While the feature selection involves the selection of most useful axes, the feature representation involves finding an optimal linear combinations of them as most effective representations for data processing. Obviously, the former is only a special case of the latter. In feature selection, a selected feature can be expressed as a special linear combination, $y = [w_1 \dots w_m]^T x$, where one and only one of $\{w_i\}$ can be nonzero. If m features are selected, then they form a subset feature vector, denoted as an m-dimensional vector process $\mathbf{y}(t) = [x_{s_1}(t), x_{s_2}(t), \cdots, x_{s_m}(t)]^T$, where s_i stands for index of a selected feature.

8.1. Feature Selection Criteria

To determine the best features, two popular criteria from the fidelity perspective are as follows:

- **Reconstruction Error criterion:**

  Due to its mathematical simplicity, the prediction error has been a popular criterion for feature selection and dimension reduction. The goal is to the minimize least-squared-error (LSE) of reconstruction:

  $$\epsilon(\mathbf{x}|\mathbf{y}) \equiv \min_{\mathbf{y}\in\Re^m} \|\mathbf{x} - \hat{\mathbf{x}}_\mathbf{y}\|$$

  where $\mathbf{x}$ is the original vector and the best estimate of x given y is:

  $$\hat{\mathbf{x}}_\mathbf{y} = E(\mathbf{x}|\mathbf{y}) = R_{xy}[R_y]^{-1}y.$$

  Thus LSE is

  $$||x - \hat{\mathbf{x}}_\mathbf{y}||^2 = ||x - R_{xy}[R_y]^{-1}y||^2.$$

  While the prediction error is useful for data compression, it's suitability for genomic information processing needs to be carefully analyzed.

- **Mutual Information Criterion:** The "mutual information" is denoted by $I(\mathbf{x}|\mathbf{y})$:

  $$\begin{aligned} I(\mathbf{x}|\mathbf{y}) &= I(y|x) = H(y) - H(y|x) \\ &= H(y) = \frac{1}{2}\log_2((2\pi)^m|R_y|) + \frac{m}{2}\log_2 e \end{aligned}$$

Note that LSE and MIC do not yield the same selection in general. This can be demonstrated by a simple example:

where a data vector are represented by an M-dimensional zero-mean wide-sense stationary vector process $\mathbf{x}(t) = [x_1(t), x_2(t), \cdots, x_M(t)]^T$.

Example 5 Selection of Best Feature – Dependent Case.
Given 3 features with covariance matrix

$$R = \begin{bmatrix} 1 & 0.8 & 0 \\ 0.8 & 1 & 0 \\ 0 & 0 & \delta = 1.2 \end{bmatrix}$$

- *For LSE criterion, the choice is* x_1.
- *For MIC criterion, the best feature for the MIC is* x_3.

Since the two solutions are different, which one would be preferred and why? Two major reasons may be used to argue in favor of the use of MIC criterion.

- **application perspective:** The problem of reconstruction error criterion is that it does not consider the strong inter-feature redundancy between x_1 and x_2. As such it may lead to a misleading selection for, say, classification purpose. On the other hand, MIC is likely to be more suitable for genomic classification applications, because it truly reflects the net information gained or lost.
- **computational perspective.** For the LSE criterion, the selection depends on the inter-feature relationship among all the M features. In contrast, the MIC criterion depends only on the m individual feature(s) concerned. More precisely, the LSE criterion is a function of R_x and R_y while the MIC criterion is only a function of $|R_y|$. (See also Eq. 5.) Therefore, the rest of section will place its focus only on the MIC.

8.2. Innovation for Efficient Search

The consecutive search approach offers a good compromise between accuracy and cost. Once some features are already selected, then we should search a feature which could best complement the existing selection.

It implies that the *innovation component* associated with such a feature should have the highest strength. Therefore, our consecutive search strategy is built upon an efficient computation of the "innovation" component. Moreover, This such an approach can substantially expedite the consecutive search process.

Let y denote the feature already selected. It can be written in a linear representation, $y = W^T x$, where W is a $M \times m$ matrix, then the (forward) innovation component is

$$\vec{\nu} \equiv x - \hat{\mathbf{x}}_\mathbf{y} = x - R_x\, W[W^T\, R_x\, W]^{-1}y$$

As illustrated in Figure 7(a), the innovation efficiently filters out the inter-feature redundancy, so its remaining component fully reflect the "new" information which complements the existing selection. Let y denote the feature al-

	MIC-FS	MIC-BE
1-feature	x_1 (optimal)	x_2 (incorrect)
2-features	x_1, x_3 (incorrect)	x_2, x_3 (optimal)

Table 4: The selections of Optimal (group ranking) selection, forward selection, and backward elimination for the case in Example 6.

ready selected and $\vec{\nu}$ its innovation vector. For MIC criterion[11], the next feature to select is

$$\arg\max_i E\{|\nu_i|^2\} \tag{5}$$

Naturally, this procedure can be carried out recursively until we select enough number of features.

The backward innovation component can also be derived in a similar fashion. (Omitted here.) Generally speaking, the forward search and backward elimination will yield suboptimal, and different, solutions. This is easily verified by the following example.

Example 6 Forward Search(FS) vs. Backward Elimination (BE)

Given 3 features with covariance matrix

$$R = \begin{bmatrix} 1 & .9 & .4 \\ .9 & .9 & 0 \\ .4 & 0 & .8 \end{bmatrix}$$

To make selection by MIC, the ranking depends on the determinant: $|R_y|$. Note that x_1 is the top choice by FS, because $1 > 0.9 > 0.8$. However, x_1 is also the first get eliminated by BE. [12]

In summary,

- *The FS's one-feature choice x_1, the correct selection, and for 2 features: x_1 and x_3, which is incorrect.*
- *The BE's one-feature choice is x_2 - an incorrect solution; and for 2 features: x_2 and x_3, which is correct.*

It is interesting to note that x_1 is the first to be chosen by FS, but it is also the first to get eliminated by BE. On the other hand, x_2 is the finalist by BE, but it will miss the selection by FS if two features are selected. See Table 6.

□

[11]**MIC Criterion** Since y is independent of ν_i, $I(x|y, y_{new}) = I(x|y) + I(x|\nu_{new})$. Since y is already fixed (in the FS scheme), so we focus on the optimization of $I(x|\nu_i) = H(\nu_i)$ which is a function of $E\{|\nu_i|^2\}$. This confirms Eq. 5.

[12]Removal of x_1 results in a least reduction of information:

$$det\begin{bmatrix} 1 & .9 \\ .9 & .9 \end{bmatrix} (= 0.09) < det\begin{bmatrix} 1 & .4 \\ .4 & .8 \end{bmatrix} (= 0.64) < det\begin{bmatrix} .9 & 0 \\ 0 & .8 \end{bmatrix}$$

8.3. Feature Representation

Given any selected feature (or more generally any linear representation), $\mathbf{y}$, it can be written as $\mathbf{y} = W\mathbf{x}$. Since a linear combination of Gaussian variables are also Gaussian, hence $\mathbf{y}$ has a probability density function of the form

$$p(\mathbf{y}) = \frac{1}{\sqrt{(2\pi)^m |R_\mathbf{y}|}} \exp\{-\frac{1}{2}(\mathbf{y}^T R_\mathbf{y}^{-1}\mathbf{y})\}$$

where $R_\mathbf{y} = W R_\mathbf{x} W^T$ is the covariance matrix of $\mathbf{y}$.

In statistics, a popular approach of extracting representation with maximum information is the *principal component analysis* (PCA). To retain the maximum amount of relevant information, the principal components (1) extract the most significant features that can best manifest the original patterns, and at the same time (2) avoid duplication or redundancy among the representations used.

Mathematically, the PCA is to find a matrix $\mathbf{W}$ such that an optimal estimate $\hat{\mathbf{x}}(t)$ of $\mathbf{x}(t)$ can be reconstructed from $\mathbf{y}(t)$ in terms of the mean-squared error. The solution $\mathbf{W}$ is formed by the first m eigenvector $\mathbf{e}_i, i = 1, 2, \ldots, m$. In other words, the first m principal components are the eigen-components corresponding to the largest m eigenvalues: λ_i, $i = 1, 2, \ldots, m$.

Two compatible questions may be raised:

- From the prediction error's perspective: *"What is the optimal m-dimensional linear representation, $\mathbf{y}$, which minimizes the following LSE: $\|\mathbf{x} - \hat{\mathbf{x}}_\mathbf{y}\|$?*
- From the information theoretical perspective: *"What is the optimal m-dimensional linear representation, $\mathbf{y}$, which maximizes the mutual information $I(\mathbf{x}, \mathbf{y})$?*

The solution to both questions converge to one and the same. The answer is the standard PCA, i.e.

$$W_{opt} = M[w_1 \ldots w_m]^T$$

where $M \in \Re^{n \times n}$ is any invertible matrix. Therefore, as a sort of unification, PCA serves as the optimal solution for both the prediction error and mutual information criteria.

9. CASE STUDY ON PAIRWISE INFORMATION: CLUSTER DISCOVERY

There exist numerous clustering techniques for multidimensional vector space, most of them are "friendly" to the use of pairwise information. For examples, with some modification, K-means and EM algorithms can be applied to the pairwise information. In the following, it will be shown that each step of the learning and classification algorithms can be made based on the use of pairwise information.

9.1. Cluster Discovery on (New) M-Dimensional Vector Space

Let us use Eq. 8 to illustrate how K-means (or EM) can be obtained via the pairwise formulation. For simplicity and without loss of generality, let us assume a linear pairwise kernel:

$$K(x_i, x_j) = x_i^T x_j \tag{6}$$

Thus the $j-th$ column of the pairwise matrix may be expressed as $z_j = \mathbf{T}x_j$, where

$$\mathbf{T} = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_M^T \end{bmatrix} \tag{7}$$

Recall that the K-means or EM, when applied to the original vector space, amounts to minimization of the following

$$E(h, X) = \sum_t \sum_{j=1}^{K} h^{(j)}(x_t) \|x_t - \mu^{(j)}\|^2. \tag{8}$$

Given the pairwise information, the same clustering algorithms may again be applied to the new vector space of $z = \mathbf{T}x$. This amount to the optimization of a new criterion - which is a simple transformation of Eq. 8:

$$E'(h, Z) = \sum_t \sum_{j=1}^{K} h^{(j)}(x_t) \|\mathbf{T}(x_t - \mu^{(j)})\|^2. \tag{9}$$

Note also that the clustering is now performed on the M-dimensional space, instead of the original N-dimensional space.

Since the "distance-metric" is now changed, therefore it will in general yield a cluster discovery result different from the that obtained according to the original vector space.

The same derivation can be extended in a straightforward manner to any nonlinear pairwise kernel function. Strictly speaking, it can lead to a very different cluster result. Nevertheless, with the potential promise of choosing a suitable nonlinear kernel, a good or improved clustering performance is rather realistic.

The same idea can be extended to other more structured cluster discovery techniques such as hierarchical clustering, and SOM. [6] [3] [5]

9.2. Nonlinear Pairwise Kernel Functions

The M-dimensional cluster discovery is readily applicable to various forms of nonlinear kernel functions. Three typical examples are:

$$\text{Polynomial Kernel} \quad : \quad K(\mathbf{x}, \mathbf{x}_i) = \left(1 + \frac{\mathbf{x} \cdot \mathbf{x}_i}{\sigma^2}\right)^p, \quad p > 0$$

$$\text{RBF Kernel} \quad : \quad K(\mathbf{x}, \mathbf{x}_i) = \exp\left\{-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma^2}\right\}$$

$$\text{Sigmoidal Kernel} \quad : \quad K(\mathbf{x}, \mathbf{x}_i) = \frac{1}{1 + e^{-\frac{\mathbf{x} \cdot \mathbf{x}_i + b}{\sigma^2}}}$$

9.3. Similarity of Two Sequences: Pairwise Scoring Kernels

The notion of similarity can be further extended to patterns beyond a vector space, i.e. non-vector data. It is especially convenient for genomic data mining, one constantly encounters sequences or time-series, for which vector space no longer serves a suitable mathematical platform. Nevertheless, the treatment via pairwise similarity score matrix remains largely unchanged.

Note that the comparison of two temporal sequences is often hampered by the fact that the two sequences often have different lengths whether or not they belong to the same family. To overcome this problem, pairwise comparison between a sequence and a set of known sequences has been a popular scheme for creating fixed-size feature vectors from variable-length sequences [8, 16, 17]. This process is referred to as *vectorization*. Suppose that we have M sequences, then each sequence is converted into a (column) vector of dimension M with entries representing the pairwise similarity between that sequence with all of the M sequences. In total, there will be M such M-dimensional (column) vectors. For example, in Figure 8, three sequences $S^{(1)}$, $S^{(2)}$, and $S^{(3)}$ are converted to three 3-dimensional column vectors. Together these vectors form an $M \times M$ matrix, named the *kernel matrix*.

Denote $\mathcal{D} = \{S^{(1)}, \ldots, S^{(T)}\}$ as a training set containing T protein sequences. Let us further denote the operation of PSI-BLAST[13] search given the query sequence $S^{(i)}$ as

$$\phi^{(i)} \equiv \phi(S^{(i)}) : S^{(i)} \longrightarrow \{\mathbf{P}^{(i)}, \mathbf{Q}^{(i)}\},$$

where $\mathbf{P}^{(i)}$ and $\mathbf{Q}^{(i)}$ are the PSSM and PSFM of $S^{(i)}$, respectively.[14] Because these matrices are based on the information of a large number of sequences that are similar to the query sequence, they contain rich information about the remote homolog of the query sequence, which may help

[13] To efficiently produce the profile of a protein sequence (called query sequence), the sequence is used as a seed to search and align homologous sequences from protein databases such as Swissprot [18] using the PSI-BLAST program [19].

[14] The homolog information pertaining to the aligned sequences is represented by two matrices (profiles): position-specific scoring matrix (PSSM) and position-specific frequency matrix (PSFM). Both PSSM and PSFM have 20 rows and L columns, where L is the number of amino acids in the query sequence.

improves the prediction of subcellular locations and protein functions. Given the profiles of two sequences $S^{(i)}$ and $S^{(j)}$, we can apply the Smith-Waterman algorithm [20] and its affine gap extension [21] to align $\mathrm{P}^{(i)}$, $\mathrm{Q}^{(i)}$, $\mathrm{P}^{(j)}$, and $\mathrm{Q}^{(j)}$ to obtain the normalized profile-alignment score $\zeta(\phi^{(i)}, \phi^{(j)})$.[15]

The scores $\{\zeta(\phi^{(i)}, \phi^{(j)})\}_{i,j=1}^{T}$ constitute a symmetric matrix $\mathbf{Z}$ whose columns can be considered as T-dimensional vectors:

$$\boldsymbol{\zeta}^{(j)} = [\zeta(\phi^{(1)}, \phi^{(j)}) \ \ldots \ \zeta(\phi^{(T)}, \phi^{(j)})]^{\mathsf{T}} \quad j = 1, \ldots, T. \tag{10}$$

An M-class protein prediction problem can now be solved by M one-vs-rest SVMs:

$$f_m(S) = \sum\nolimits_{j \in \mathcal{S}_m} y_{m,j} \alpha_{m,j} K(\phi(S), \phi(S^{(j)})) + b_m \tag{11}$$

where S is an unknown sequence, $m = 1, \ldots, M$, $y_{m,j} \in \{+1, -1\}$, $\mathcal{S}_m$ contains the indexes of support vectors, $\alpha_{m,j}$ are Lagrange multipliers, and

$$K(\phi(S)), \phi(S^{(j)}) = g(\boldsymbol{\zeta}, \boldsymbol{\zeta}^{(j)})$$

is a kernel function.

Now we may consider the columns of a pairwise scoring matrix as high-dimensional vectors. This means that there are T feature vectors with dimension equal to the training set size. The T T-dimensional column vectors can be used to train M SVMs. Because of the high dimensionality, linear SVM is a preferred choice, i.e., $g(\boldsymbol{\zeta}, \boldsymbol{\zeta}^{(j)}) =< \boldsymbol{\zeta}, \boldsymbol{\zeta}^{(j)} >$. The class of S can then be obtained by $y(S) = \arg\max_{m=1}^{M} f_m(S)$, where M is the number of classes.

In conclusion, the pairwise kernel approach has been very popular for this circumstance. Indeed, it has been successful applied to genomic sequencing applications. [16] One such example will be discussed subsequently.

10. FEATURE SELECTION FOR SUBCELLULAR LOCALIZATION

In the previous sections, we have constantly used an axis to represent a feature, and therefore, feature selection/elimination means axis selection/elimination. Note that under the pairwise kernel representation, we have as many features as vectors. Then does a feature correspond to an axis or a data point? The somewhat surprising answer is: **Both**. Because the symmetrical kernel matrix exhibits a useful reflexive property, which is best illustrated by an example shown in Figure 9. It can be advantageous to harness such a symmetry property, which motivates the discussion of this section.

[15] See http://www.eie.polyu.edu.hk/~mwmak/BSIG/PairProSVM.htm.

[16] Furthermore, pairwise sequence alignments can be effectively extended to derive multiple sequence alignments.

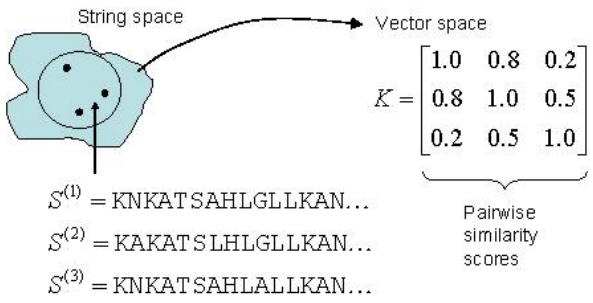
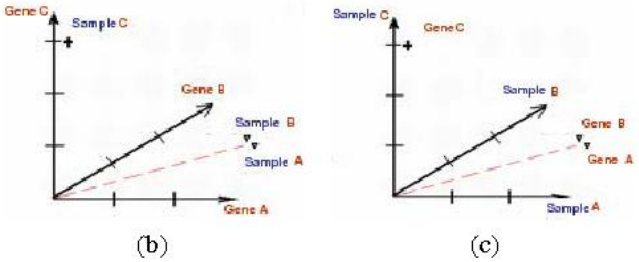


Figure 8: Vectorizing Sequences. Because strings are composed of alphabets and have different lengths, they need to be converted to vectors of the same dimension for classification via kernel methods.

	Sample A	Sample B	Sample C
Gene A	2.1	2.0	0.1
Gene B	2.0	2.1	0.1
Gene C	0.1	0.1	3.0

(a)



(b) (c)

Figure 9: Symmetry (reflexive) property of pairwise data. (a) Table showing the feature/sample data. (b) Features are displayed as axes and samples as data vectors. (c) Features are displayed as data vectors and samples as axes.

In [1], a method was proposed that makes use of the symmetric property of pairwise scoring matrices to select relevant features. The method considers the columns of a pairwise scoring matrix as high-dimensional vectors and uses the column vectors to train a linear SVM [13]. Because of the symmetric property of the score matrix, the row vectors with row indexes equal to the support vector indexes are identical to the support vectors. Also, because the support vectors define the decision boundary and margins of the SVM, they are critical for classification performance. Therefore, the support vector indexes are good candidates for selecting features for the column vectors, i.e., only the rows corresponding to the support vectors are retained. The column vectors with reduced dimension are then used to

train another SVM for classification. Because the indexes of support vectors are used for selecting features, we referred this method to as vector-index-adaptive SVM, or simply VIA-SVM.

10.1. VIA-SVM Approach to Pairwise Scoring Kernels

To design a feature selection algorithm for pairwise scoring vectors, we need to exploit the reflexive property of pairwise scoring matrices. The idea is based on the notion that support vectors are important for classification and pairwise scoring matrices are symmetric. (Namely, the elements of the i-th column of $\mathbf{Z}$ are identical to those in the i-th row.) This suggests a possible hypothesis:

> **Hypothesis.** *The support vector indexes are good candidates for selecting features for the column vectors, i.e., only the rows corresponding to the support vectors are retained.*

10.1.1. Why Consider Only Support Vectors?

In VIA-SVM [1],, the support vector indices are reused as feature selection indices. The use of support vectors to select relevant features is intuitively appealing because they are "critical" for establishing the decision boundary of SVM classifiers. Because of the symmetrical property of kernel matrices, the elements of the i-th column of $\mathbf{Z}$ are identical to those in the i-th row. If the i-th column of $\mathbf{Z}$ happens to be a support vector, the corresponding feature dimension (the i-th row of $\mathbf{Z}$) will also be critical for classification. On the other hand, non-support vectors are irrelevant for classification, so as their corresponding feature dimensions.

The above interpretation of VIA-SVM is consistent with how SVM-RFE selects features in that, in both methods, indexes with large weight will be chosen first. Moreover, they both prune the vectors/features corresponding to zero α_i. However, there is also an important difference, which lies in the treatment of the vectors/features corresponding to non-zero α_i. More exactly, in VIA-SVM, different types of support vectors receive different level of preference.

10.1.2. Differential Treatments of Support Vectors

Because the SVM-RFE takes the overall weight vector $\mathbf{w}$ into account, it only considers the Lagrange multipliers α_i but not the slack variables ξ_i. In contrast, the VIA-SVM considers both α_i (related to SV) and ξ_i (indicates safety margin or, sometimes, outlier). In this sense, the VIA-SVM offers a more comprehensive coverage of all the critical factors made available by the SVM classifier.

It is important to recognize the fact that *not all SVs are created equal*. Therefore, in the VIA-SVM, support vectors are differentially treated. In fact, they are divided into

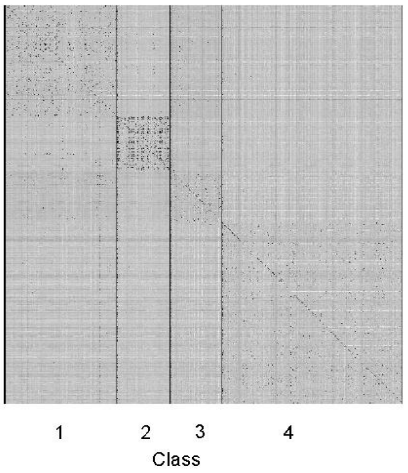


Figure 10: Profile alignment score matrix $\mathbf{Z} = \{\zeta(\phi^{(i)}, \phi^{(j)})\}_{i,j=1}^{T}$. The training vectors have been prearranged such that the vectors belonging to the same class are all grouped together, i.e., they are consecutively indexed. The three vertical lines were artificially added to divide the column vectors into four classes.

four levels of preferences as specified by the four regions in Figure 11:

Level 1 **Most-preferred**: The SV is on the margin, i.e., $0 < \alpha_i < C$ and $\xi_i = 0$, where C is the penalty factor in SVM training.

Level 2 **Preferred**: The SV is in the fuzzy region and on the correct side of the decision boundary, i.e., $\alpha_i = C$ and $0 < \xi_i < 1$.

Level 3 **Marginally-preferred**: The SV is in the fuzzy region but on the wrong side of the decision boundary, i.e., $\alpha_i = C$ and $1 \leq \xi_i < 2$.

Level 4 **Non-preferred**: The SV is regarded as an outlier, i.e., $\alpha_i = C$ and $\xi_i \geq 2$.

The reason of ruling out the outlier SVs is self-explanatory. The decision to have the marginal support vectors assigned the highest preference level can be justified on the basis that they offer relatively higher confidence than the fuzzy SVs.

Although it appears that two parameters (α_i and ξ_i) are required to define the preferences, ξ_i alone can already provide sufficient information to determine the preference level of an SV. In fact, the preference decreases with increasing ξ_i. More exactly, $\xi_i = 0$, $0 < \xi_i < 1$, $1 \leq \xi_i < 2$, and $2 \leq \xi_i < \infty$ define Level 1 to Level 4, respectively (cf. Figure 11).

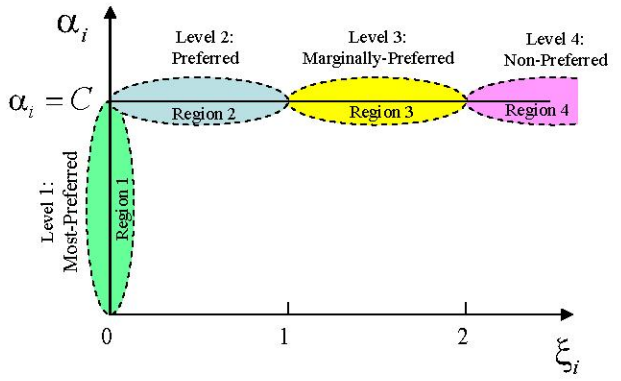


Figure 11: The four levels of preferences for the support vectors in VIA-SVM. Regions 1-4 correspond to preference levels 1-4, respectively.

10.2. The VIA-SVM Algorithm

Feature selection in VIA-SVM is divided into two steps:

Step 1 The score matrix $\mathbf{Z} = \{\zeta(\phi^{(i)}, \phi^{(j)})\}$ is used to train M SVMs (Eq. 11) from which M sets of support vector indexes $\mathcal{S}_m$ are determined. This results in a set of support vectors $\boldsymbol{\zeta}^{(j)} = [\zeta(\phi^{(1)}, \phi^{(j)}) \cdots \zeta(\phi^{(T)}, \phi^{(j)})]^{\mathsf{T}}$ for each class, where $j \in \mathcal{S}_m$.

Step 2 For the m-th class, the indexes in $\mathcal{S}_m$ are used to select the feature dimensions (rows of $\mathbf{Z}$) of the column vectors to obtain vectors $\boldsymbol{\zeta}^{'(j)}$ of reduced dimension, where $j = 1, \ldots, T$. These vectors are then used to train another SVM for classification. This process is repeated for all classes.

This two steps are iterated N times (5 in this work). Specifically, the features selected at the n-th iteration are used to train a new SVM in the $(n+1)$-th iteration, whose support vectors are subsequently used for determining the feature set in the $(n+2)$-th iteration, and so on. The classification accuracy on the training data at each iteration is recorded. At the end of the N-th iteration, the support vectors of the SVM with the highest training accuracy are used for selecting the final set of features. The column vectors with reduced dimension are then used to train another SVM for classification. Figure 12 shows the pseudo-code of VIA-SVM.

10.2.1. Level-Dependent VIA-SVM Selection Strategies

For the actual implementation, it is worth noting that NOT all SVs are created equal. Here, we propose four level-dependent VIA-SVM selection strategies.

Strategy 1 (*Level 1 only*): Select the most-preferred SVs, i.e., select only the "pure" marginal SVs ($\alpha_i < C$) while excluding those fuzzy and outlier SVs ($\alpha_i = C$).

Strategy 2 (*Levels 1 and 2*): Select the correctly classified SVs only, i.e., select the "pure" marginal SVs ($\alpha_i < C$) and the correctly classified SVs that are falling on the fuzzy region ($\alpha_i = C$ and $0 < \xi_i < 1$).

Strategy 3 (*Levels 1–3*): Remove the non-preferred, outlier SVs, i.e., only keep those SVs with $\xi_i < 2$.

Strategy 4 (*Levels 1–4*): Select ALL SVs, i.e., select all marginal and fuzzy SVs with $0 < \alpha_i \leq C$.

Experiments on subcellular localization support that Strategies 2 and 3 appear to produce the best performance.

Algorithm VIA-SVM

Input: $\mathbf{X} = [\mathbf{x}_1 \quad \mathbf{x}_2 \quad \cdots \quad \mathbf{x}_T], \mathbf{y} = [y_1 \quad y_2 \quad \cdots \quad y_T]$,
and $C \in \Re$, where $\mathbf{x}_t \in \Re^T$, $y \in \{+1, -1\}$

Initialization: $\mathbf{X}' = \mathbf{X}$;

for k = 1 to N
do

// Train an SVM to obtain the indexes to the support vectors in **i**
[**α**, *b*, **i**] = SVM_train(**X**′, **y**, *C*);

// Find the support vector indexes **j** *such that* $0 < \alpha_j \leq C$ and $0 \leq \xi_j < 2$
// where $\xi_j = 1 - \alpha_j(\mathbf{x}_j \cdot \mathbf{w} + b)$ *and* $\mathbf{w} = \sum_{i \in \mathbf{i}} \alpha_i y_i \mathbf{x}_i$
j{k} = find_sv_index(**X**′, **y**, *C*, **α**, *b*, **i**);

// Select a feature subset based on the selected support vector indexes
X′ = **X**(**j**{k}, :);

// Train another SVM based on the selected features
[**α**, *b*, **i**] = SVM_train(**X**′, **y**, *C*);

// Computing classification accuracy on training data
a(k) = SVM_test(**X**′, **y**, **α**, *b*, **i**);

end

Output: $\mathbf{j}\{k'\}$ where $k' = \arg\max_k a(k)$

Figure 12: Pseudo code of VIA-SVM.

10.2.2. Comparing VIA-SVM and SVM-RFE

Although both VIA-SVM and SVM-RFE are based on the support vector machines, SVM-RFE uses the weight vector $\mathbf{w}$, whereas VIA-SVM uses the Langrange multipliers α_i and slack variables ξ_i.

10.3. Combined With Other Selection Criteria

10.3.1. Redundance Removal for VIA-SVM

A common weakness of both VIA and RFE approaches is that they do not explicitly consider the redundancy factor.

This will result in wasteful selection because some of the selected features can be either repetition of each other or highly redundant. To minimize feature redundancy, we propose two pruning methods for VIA-SVM.

1. **Euclidean distance**: For those support vectors in Region 2 or 3, their pairwise Euclidean distances in the reduced kernel space (defined by the selected feature dimensions) are examined. If two features (with similar ξ_i) are close to each other, one of them may be removed without affecting the decision boundary.

2. **K-means**: Support vectors that have similar ξ_i in Region 2 or 3 are clustered by K-means in the reduced kernel space. The SVs closest to the centers are retained and all remaining SVs in Regions 2 and 3 are removed.

If the allowable number of features is very small, the redundance-reduced feature set can be further pruned by a filter method such as symmetric divergence. Results of cascading VIA-SVM, redundance removal, and feature pruning will be shown in Section 10.4.

Note that while SVM-RFE can also apply a post-processing redundancy removal selection process (such as applying K-means or double-checking the similarity score in the kernel matrix), it does not enjoy the numeric information provided by ξ_i.

10.3.2. Fusion of Selection Criteria

It is natural to combine/consider both the filter and wrapper methods in order to reach an optimal strategy. Specifically, the features and/or criterion functions obtained from the filter and wrapper methods can be combined via a cascaded fusion scheme shown in Figure 13.

It is desirable to seek an optimal compromise between accuracy and cost. A possibility is via a comprehensive selection procedures comprising two consecutive phases. Here, we refer to these procedures as Overselect-and-Prune strategy. The strategy has two steps.

Step 1: **Over-selection Phase.** This is a stage in which only those obviously irrelevant features are quickly weeded out. Preferably, it involves a quick and coarse (suboptimal) evaluation, e.g., individual ranking.

Step 2: **Pruning Phase.** The second stage may serve as a fine tuning process. The goal is to remove redundant features with minimum information loss. If major redundancy exists between A and B, then one of the two may be pruned without incurring much loss of information.

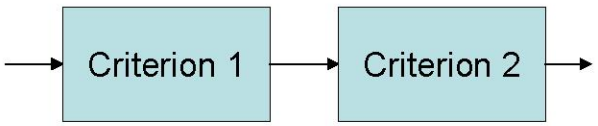


Figure 13: In the cascade fusion architecture, Criterion 1 is meant for coarse and fast over-selection and Criterion 2 represents pruning. The threshold for Criterion 1 can be more relaxed (i.e., its value does not have to be very close to the optimal number of features) to include more candidate features for the second stage.

10.4. Experiments on Subcellular Localization

Two datasets were used for evaluating the performance of VIA-SVM and for comparing it against other feature selection algorithms. The first dataset is provided by Reinhardt and Hubbard [9]. It comprises 2427 amino acid sequences extracted from SWISSPROT 3.3, with each protein annotated with one of the four subcellular locations: cytoplasm, extracellular, mitochondrial, and nuclear. The second dataset was provided by Huang and Li [22]. It was created by selecting all eukaryotic proteins with annotated subcellular locations from SWISSPROT 41.0 and by setting the identity cutoff to 50%. The dataset comprises 3572 proteins (622 cytoplasm, 1188 nuclear, 424 mitochondria, 915 extracellular, 26 golgi apparatus, 225 chloroplast, 45 endoplasmic reticulum, 7 cytoskeleton, 29 vacuole, 47 peroxisome, and 44 lysosome). We used 5-fold cross validation for performance evaluation so that every sequence in the datasets will be tested.

10.4.1. Performance of VIA-SVM

Now let us discuss the case study results based on the four selection strategies mentioned in Section 10.2.1. Because Strategy 1 includes only very few SVs, the features are extremely under-selected. In contrast, because Strategy 4 includes all SVs regardless of their types, it is likely to cause over selection, particularly when the penalty factor C is small. In Strategy 2, all the SVs that are incorrectly misclassified will be excluded. However, this may lead to under selection as there are some useful SVs falling on the fuzzy regions. Strategy 3 is a compromise between the over selection in Strategy 2 and the under selection in Strategy 4. More exactly, Strategy 3 excludes all the outlier SVs. (The SVs lying beyond the margin of the opposite class are deemed to be outliers.) In this strategy misclassified SVs that lie within the margin of separation will still be selected, leading to over selection, especially when the penalty factor C is very small.

Figure 14 shows the performance of Strategies 1–4 when

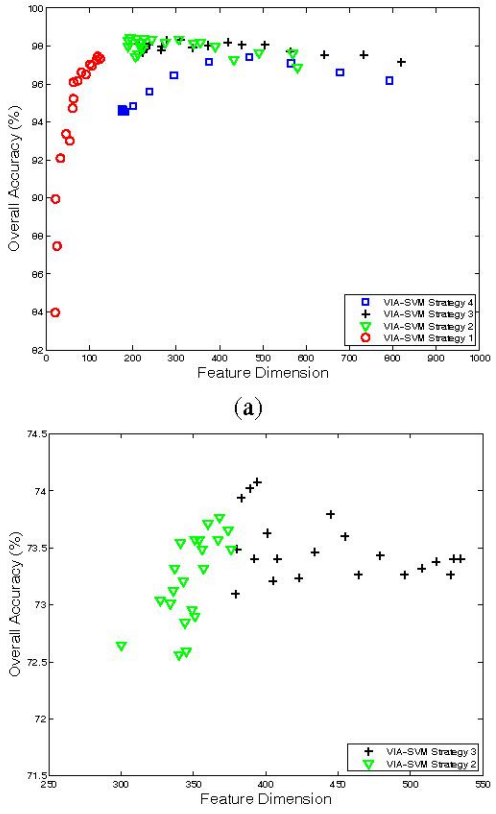


Figure 14: Prediction performance of different strategies of VIA-SVM on (a) Reinhardt and Hubbard's dataset and (b) Huang and Li's dataset when the penalty factor C varies from 0.004 to 4,096. See Section 10.4 for details of the strategies.

the penalty factor C varies from 0.004 to 4,096. Note that the number of selected features (feature dimension) is automatically determined by the SVMs. For each strategy, a smaller penalty factor C will generally lead to a larger number of features, and vice versa for a larger C. Therefore, markers on the right region of the figure correspond mainly to small C's. Notwithstanding the larger number of features obtained by Strategy 4, the maximum accuracy attained by it is still poorer than the other strategies. This confirms our earlier hypothesis that including all SVs will lead to over selection. Results also show that Strategy 1 will lead to under selection when the penalty factor C becomes large. These case studies suggest that Strategies 2 and 3, which exclude either the non-preferred SVs or both the marginally preferred and non-preferred SVs (cf. Figure 11), seem to be the least sensitive to the penalty factor, because they can keep the number of features within a small range and maintain the accuracy at a constant level for a wide range of C.

10.4.2. *Comparison between VIA-SVM and SVM-RFE*

We compared the proposed VIA-SVM (Strategy 2) with SVM-RFE [15] in the subcellular localization benchmarks mentioned earlier.[17] Note that SVM-RFE does not make use of the symmetric property of the pairwise scoring matrices in the selection process, because it is primarily designed for gene selections in microarray data where expression matrices are neither square nor symmetric.

Figures 15 shows the performance of SVM-RFE (blue –□– and VIA-SVM (red ○). Evidently, VIA-SVM is superior to SVM-RFE in two aspects: (1) It outperforms SVM-RFE at almost all feature dimension, particularly at low feature dimensions and (2) it automatically bounds the number of selected features within a small range. A drawback of SVM-RFE is that it requires a cutoff point for stopping the selection. On the other hand, VIA-SVM is insensitive to the penalty factor in SVM training and can avoid the need to set a cutoff point for stopping the feature selection process.

10.4.3. *Fusion of VIA-SVM and SD*

Given a particular axis, which corresponds to one feature, two factors can be considered: VIA-SVM parameters (C and ξ_i) and symmetric divergence of feature i (SD_i). We adopt the Over-Select-and-Prune Cascaded Fusion Architecture, as proposed in Section 10.3.2, to combine VIA-SVM and SD. Based on this cascade fusion strategy, the selection process is divided into two stages.

Stage 1: Use VIA-SVM (Strategy 2 or 3) to select all-but-outlier SVs, i.e., only keep those with $\xi_i < 2$.

Stage 2: Use SD to sort the features found in Stage 1 and keep the most relevant $x\%$.

In this work, we set x to 70. Figures 15 shows the fusion results (blue ×), which suggest that fusion can produce more compact feature subsets without significant reduction in prediction accuracy. We also note that although VIA-SVM is inferior to SVM-RFE for large feature-set size, the combination of SD and VIA-SVM performs better at small feature-set size.

10.4.4. *Redundance Removal*

The Euclidean-distance (eDist) and K-means based methods mentioned in Section 10.3.1 were applied to reduce the redundance among the features selected by VIA-SVM. For the former, the constant η was set to 0.3; for the latter, the number of centers in K-means was set to 300 for Huang and Li's dataset. The setting of these values was based on the

[17] Because the performance of SVM-RFE, R-SVM, and symmetric divergence are comparable in the two benchmarks, we only report the results of SVM-RFE for clarity of presentation.

observation from Figure 14 that the accuracy drops rapidly when the number of features is smaller than these values. The results (green $+$ and pink $\diamond$) shown in Figures 15 suggest that both methods can reduce the feature size without scarifying accuracy. This once again demonstrates the merit of using the slack variables ξ_i in selecting features.

To further reduce the feature size, we applied SD to pruned the features after redundance removal. Figures 15 (black $\triangledown$ and red $\star$) shows that the resulting feature sets achieve a significant higher accuracy when compared with SVM-RFE.

We have also compared Strategy 2 and Strategy 3 in terms of the mean accuracy and mean feature dimension obtained by VIA-SVM and VIA-SVM cascaded with various pruning methods. Based on the Table 5, three observations can be made:

1. For each pruning method, there is no significant difference between the accuracy obtained by Strategies 2 and 3;

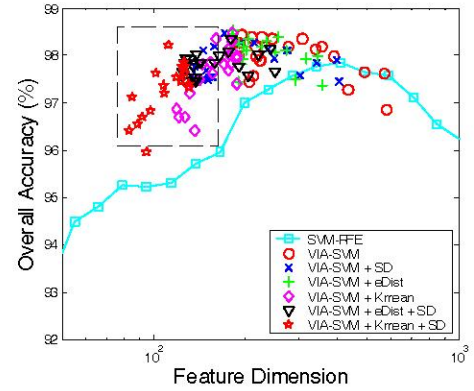
2. Strategy 2 generally leads to a smaller number of features than Strategy 3 with almost the same performance statistically;

3. SVM-RFE not only gives lower average accuracies but also leads to a larger variation in both accuracy and feature dimension;

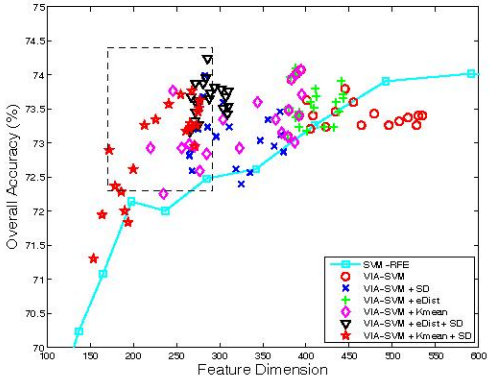
These observations suggest that Strategy 2 is a winner because it can keep the number of features to a minimum without scarifying accuracy. Strategy 2 is also a better choice for applications where feature dimension (and hence recognition time) should be kept to a minimum. On the other hand, for applications where accuracy is of primary importance, Strategy 3 is a better option.

10.4.5. Range of Desirable Feature Dimension

In most pattern recognition problems, a smaller feature size can result in faster recognition speed, but at the expense of lower classification accuracy, and vice versa. Usually, it should be possible set a range of desirable feature dimension for a particular problem. For example, in bioinformatic applications where accuracy is far more important than speed, we may prefer using the upper limit that gives high accuracy but not to the point that causes the curse of dimensionality. Based on the performance observed, the desirable range of feature dimension is highlighted by the dashed rectangles in Figures 15. Evidently, the redundance removal and the cascade fusion of VIA-SVM and SD enable us to find the feature sizes falling on the desirable range. In particular, when recognition speed is a concern, we can over-select features by using VIA-SVM (Strategies 2 or 3) and then remove feature redundance by using K-means, and finally, we can further prune the features by using SD ('VIA-SVM + Kmean + SD'). On the other hand, if accuracy is more important, we may skip the final pruning stage, i.e., using 'VIA-SVM + Kmean', or do not apply pruning at all.



(a) Strategy 2



(b) Strategy 3

Figure 15: Prediction performance of SVM-RFE, VIA-SVM, and VIA-SVM cascaded with various pruning methods on Huang and Li's dataset. (a) The VIA-SVM uses Strategy 2 for feature selection. (b) Strategy 3 was used. (c) The means and standard derivations (in parentheses) of the classification accuracies and feature dimensions for 21 penalty factors ranging from 0.004 to 4096; for SSVM-RFE in (c), the means and standard derivations are based on 9 points in (a) and (b) whose feature dimensions range from 114 to 492. Pruning was applied according to the order indicated in the legend; for example, 'VIA-SVM + eDist + SD' means that the features selected by VIA-SVM was pruned by Euclidean distance-based method followed by the symmetric divergence-based method.

Method	Accuracy (%)		Feature Dimension	
	Strategy 2	Strategy 3	Strategy 2	Strategy 3
VIA-SVM	73.48 (0.27)	73.23 (0.38)	445 (55.77)	348 (17.22)
VIA-SVM+SD	73.12 (0.41)	72.66 (0.51)	312 (39.01)	244 (12.07)
VIA-SVM+eDist	73.56 (0.29)	73.23 (0.42)	408 (22.69)	332 (27.77)
VIA-SVM+Kmean	73.26 (0.48)	73.09 (0.47)	327 (61.84)	296 (65.29)
VIA-SVM+eDist+SD	73.63 (0.27)	73.11 (0.47)	286 (15.83)	232 (19.35)
VIA-SVM+Kmean+SD	72.93 (0.71)	72.73 (0.78)	229 (43.39)	207 (45.76)
SVM-RFE	71.90 (1.45)		264 (129.10)	

Table 5: This table summarize the performance (accuracy versus feature dimension) corresponds to Figures 15.

11. CONCLUSION

An effective data mining system lies in the representation of pattern vectors. The curse of dimensionality, has traditionally been a serious concern in many genomic applications. That is why this paper places its emphasis on feature selection. Unfortunately, the page limit does not allow a full or more comprehensive treatment to do justice to such an emerging field. In addition, most research issues remain very much open in terms of novel and efficient algorithms for feature selection. In addition, a lot more investigation will be needed in order to more convincingly demonstrate how the machine learning tools may be applied to real genomic application, including (sequence-based) genomic sequencing and (matrix-based) gene expression profile analysis.

Acknowledgments

The author wish to acknowledge the technical contributions of Dr. M. W. Mak, Hong Kong Polytechnic University, who was coauthor of many of previous publications which form the basic body of this manuscript. Especially, Section 10 basically follows [1].

12. REFERENCES

[1] S. Y. Kung and M. W. Mak, "*Feature Selection for Pairwise Scoring Kernels with Applications to Protein Subcellular Localization*", Proceedings, IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), April 2007.

[2] "DNA Microarrays and Gene Expression", P. Baldi and G.W. Hatfield, Cambridge Press, 2002.

[3] M. B. Eisen, P.T. Spellman, P.O. Brown, and D. Botstein, "*Genetics Cluster analysis and display of genome-wide expression patterns*", Proc. Natl. Acad. Sci. USA Vol. 95, pp. 1486314868, December 1998.

[4] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. 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*, vol. 286, pp. 531–537, 1999.

[5] P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E. S. Lander, and T. R. Golub,"Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation", Proc. Natl. Acad. Sci. USA Vol. 96, pp. 29072912, March 1999.

[6] T. Kohonen, "Automatic formation of topologicalmaps of patternsin a self-organizingsystem", Proceedings of 2SCIA, Scan. Conference on Image Analysis, pp. 214-220, 1981.

[7] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[8] J. Guo, M. W. Mak, and S. Y. Kung, "Eukaryotic protein subcellular localization based on local pairwise profile alignment SVM," in *2006 IEEE International Workshop on Machine Learning for Signal Processing (MLSP'06)*, pp. 391–396, 2006.

[9] A. Reinhardt and T. Hubbard, "Using neural networks for prediction of the subcellular location of proteins," *Nucleic Acids Res.*, vol. 26, pp. 2230–2236, 1998.

[10] A. Hyvarinen and E. Oja, "Independent component analysis: Algorithms and applications," *Neural Networks*, vol. 13, pp. 411–430, 2000.

[11] P. Pavlidis, J. Weston, J. Cai, and W. N. Grundy, "Gene functional classification from heterogeneous data," in *Int. Conf. on Computational Biology*, (Pittsburgh, PA), pp. 249–255, 2001.

[12] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, pp. 179–188, 1936.

[13] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[14] R. Kohavi and G. H. John, "Wrappers for feature selection," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.

[15] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46, p. 389422, 2002.

[16] L. Liao and W. S. Noble, "Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships," *J. Comput. Biol.*, vol. 10, no. 6, pp. 857–868, 2003.

[17] J. K. Kim, G. P. S. Raghava, S. Y. Bang, and S. Choi, "Prediction of subcellular localization of proteins using pairwise sequence alignment and support vector machine," *Pattern Recog. Lett.*, vol. 27, no. 9, pp. 996–1001, 2006.

[18] http://www.expasy.org/sprot.

[19] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, "Gapped BLAST and PSI-BLAST: A new generation of protein database search programs," *Nucleic Acids Res.*, vol. 25, pp. 3389–3402, 1997.

[20] T. F. Smith and M. S. Waterman, "Comparison of biosequences," *Adv. Appl. Math.*, vol. 2, pp. 482–489, 1981.

[21] O. Gotoh, "An improved algorithm for matching biological sequences," *J. Mol. Biol.*, vol. 162, pp. 705–708, 1982.

[22] Y. Huang and Y. D. Li, "Prediction of protein subcellular locations using fuzzy K-NN method," *Bioinformatics*, vol. 20, no. 1, pp. 21–28, 2004.