Feature Reduction for Improved Recognition of Subcellular Location Patterns in Fluorescence Microscope Images

Kai Huang, Meel Velliste, and Robert F. Murphy*
Departments of Biological Sciences and Biomedical Engineering, Carnegie Mellon University,
4400 Fifth Avenue, Pittsburgh, PA, USA 15213

ABSTRACT

The central goal of proteomics is to clarify the mechanism by which each protein in a given cell type carries out its function. Automated protein subcellular location determination by fluorescence microscopy can play an important role in fulfilling this goal. The subcellular location of a protein is critical to understanding its function because each subcellular compartment has a unique biochemical environment. We have previously shown that neural network classifiers using sets of numerical features computed from fluorescence microscope images were able to recognize all major subcellular location patterns with reasonable accuracy. Current classifiers are limited by under-determined classification boundaries due to the limited number of available images compared to the number of features. In this paper, we compare various feature reduction methods that can address this problem. Specifically, principal component analysis, kernel principal component analysis, nonlinear principal component analysis, independent component analysis, classification trees, fractal dimensionality reduction, stepwise discriminant analysis, and genetic algorithms are used to select feature subsets that are evaluated using support vector machine classifiers. The best results were obtained using stepwise discriminant analysis and we found that as few as eight features can provide good classification accuracy for all major subcellular patterns in HeLa cells.
Keywords: proteomics, subcellular location pattern, subcellular location features, feature reduction, feature recombination, feature selection, support vector machines

1. INTRODUCTION

The central goal of proteomics is to understand the function and role that each protein plays in disease and development. Automated methods for determining a number of protein properties have been described. However, very little attention has been paid to the determination of protein subcellular location. Advances in fluorescent probe chemistry, protein chemistry, and imaging techniques have made fluorescence microscopy a valuable method for determining protein subcellular locations. To record the subcellular location of a target protein, a tag, which can be either a fluorescent probe or an antibody, is introduced into a cell and fluorescence microscope images are taken of the tagged protein. If such analyses can be converted into high throughput "location proteomics" assays, the resulting information could help us understand where a protein distributes in cells, what function a protein might have, whether several proteins might bind to each other, and how a protein changes its location in response to drugs, during diseases and during the cell cycle.

Automated image analysis will be critical to enabling location proteomics projects. As a start towards addressing this need, our group first introduced automated quantitative methods to describe protein subcellular location patterns in cultured cells.[1,2] In subsequent work we described a number of sets of **S**ubcellular **L**ocalization **F**eatures (SLF),[3-5] which can be computed from 2D and 3D fluorescence microscope images of single cells and used in training neural network classifiers to recognize location patterns in previously unseen images. Such classifiers were able to distinguish ten patterns covering all major subcellular organelles, including two Golgi proteins that can not be differentiated by human observers.[3-6] SLF features that describe protein subcellular location patterns in 2D fluorescence microscope images consist of several different types, including Zernike moment features, Haralick texture features, features from morphological and geometric image processing, and features that require a parallel reference image of a DNA probe.

Accuracy and efficiency are two factors often considered in pattern recognition. One potential advantage of having large

* murphy@cmu.edu; FAX 1 412 268 6571; murphylab.web.cmu.edu

sets of features is that they may contain sufficient information to describe protein subcellular location patterns. However, having many features does not necessarily guarantee high classification accuracy. There might be noisy features that only increase training time by wasting the computation power of a certain classifier. Thus in many cases, improved classification accuracy and/or efficiency can be achieved by reducing the number of features used. A small number of features is also desirable for use in content-based retrieval from image databases.

Previously, we have used one feature selection method, stepwise discriminant analysis (SDA),[7] and observed high classification accuracy after feature selection. For example, the SLF5 feature set containing 37 features was selected by SDA from feature set SLF4 that has 84 features (including six features that require a parallel DNA image).[3] When used with a neural network classifier containing one hidden layer of 20 nodes, SLF5 gives a 2% improvement in classification accuracy over SLF4 for 10 subcellular patterns.[3] The SLF8 set containing 32 features was selected from the most complete current 2D feature set, SLF7, that has 84 features and does not require a parallel DNA image. SLF8 gives an average 86% classification accuracy and is the best current feature set using the same neural network classifier.[5] In this paper, we compare eight feature reduction methods on the SLF7 feature set using a multi-class support vector machine classifier and the same 2D HeLa cells images used in the previous studies.[3,5]

There are two general approaches to feature reduction: feature recombination and feature selection. In the former case, a new smaller feature set is obtained by a weighted recombination of the original features. In this paper, four different methods were applied, including principal component analysis (PCA), nonlinear principal component analysis (NLPCA), kernel principal component analysis (KPCA), and independent component analysis (ICA). Although feature recombination can often provide a valuable, reduced representation of a dataset, the poor understandability of the recombined features can be a problem. For this reason, feature selection methods choose a subset of features from the original set. Brute-force feature selection is an NP-hard problem since exhaustive subset search in the huge (2^n, n is the number of features) feature space is impractical. Therefore, heuristic search, either sequential or randomized, is applied in practice to find the best feature subset, which may be globally suboptimal. There are three ways of doing sequential search: forward selection, which starts from an empty feature set and keeps adding the best feature found at each step until the evaluation function could not be further improved; backward elimination, which starts from a full feature set and keeps eliminating the worst feature found at each step until the evaluation function could not be further improved; and forward-backward search, a combination of forward and backward search that tries to either add back features that have been previously deleted or delete features that have been previously added until the evaluation function could not be further improved. Instead of deterministically adding or deleting features, randomized search employs sampling in its selection strategy such that all possible feature combinations have a chance to be considered. Both of these heuristic methods make use of an evaluation function that can be either a classification program or some global statistic computed from the training data set. The former is also referred to as the wrapper method, and the latter as the filter method.[8] In this paper, we also applied four different feature selection methods, classification trees, fractal dimensionality reduction (FDR), stepwise discriminant analysis (SDA), and genetic algorithms.

2. METHODOLOGY

All image features are normalized to have zero mean and unit variance across all classes. Throughout this paper, X represents each feature vector across all images and x stands for each image represented by multiple features. Assuming each image is represented by m features and there are n images in total, the feature matrix D is n by m, in which X is the column vector and x is the row vector respectively. The total number of classes is represented by q.

2.1 Principal Component Analysis (PCA)

Among all feature recombination methods, principal component analysis (PCA) is best known for capturing linear relationships among features and recombining features in a way that most of the total variance of the original data set is preserved.[9] Principal components are defined as:

$$PC_i = w_{i1}X_1 + w_{i2}X_2 + \cdots + w_{im}X_m \quad (1)$$

where PC_i represents the ith principal component, X_j $(j=1,...,m)$ represents the jth feature vector and w_{ij} $(j=1,...,m)$ are weights associated with each feature.

Principal components can be extracted using eigenvalue decomposition. The covariance matrix of the dataset is computed first,

$$A = \frac{1}{n}\sum_{j=1}^{n} x_j \, x_j^{T} \tag{2}$$

where x_j $(j=1,...,n)$ represents the jth image data. Eigenvalues and eigenvectors are then derived from the covariance matrix A and are sorted in decreasing order of eigenvalue. A linear transformation matrix is formed by taking the k eigenvectors corresponding to k largest eigenvalues. Applying this linear transformation on the original feature space, we will get a reduced k-dimensional feature space, whose features, also called principal components, are the coordinate values of the original data in the new orthogonal-transformed coordinate system. We used the PCA function in the Matlab classification toolbox developed by Drs. Stork and Yom-Tov (http://tiger.technion.ac.il/~eladyt/classification/index.htm).

2.2 Nonlinear Principal Component Analysis (NLPCA)

PCA only seeks linear relationships among original features with least squared-errors. Therefore, it might become a poor representation when original features have nonlinear relationships. In this case, nonlinear principal component analysis can be used to find a nonlinear mapping of the original features with least squared-error.

A five-layer neural network can be employed to conduct NLPCA.[10] The original features serve as both the input and output layers of the neural network. The network has one linear middle layer and two nonlinear layers on either side of the middle layer. The data propagated from the input nodes are nonlinearly activated and combined at the second layer followed by linear processing in the middle layer. The neural network is trained under the least squared-errors criterion and only the first three layers of the trained network are kept as a nonlinear-component extractor. In other words, the k linear nodes in the middle layer represent the nonlinear principal components extracted by the network. Due to the variability of neural networks, namely that different starting points might converge to different local minimums, a weight-penalty improved neural network NLPCA toolbox NeuMATSA[11] was used for extracting nonlinear principal components. The algorithm used in this toolbox extracts one nonlinear principal component at a time, meaning that the middle layer of the network it uses has only one node.

2.3 Kernel Principal Component Analysis (KPCA)

A variant of PCA, kernel principal component analysis (KPCA), uses a nonlinear kernel function to first transform the original feature space to a very high dimensional space and then applies normal PCA in this new higher feature space.[12] By employing a nonlinear kernel function, nonlinear relationships among features can be found.

The covariance matrix (eq. 2) can be regarded as taking dot product of each pair of centered data points in the original feature space. The computation of dot products can be generalized in any dimensional space F. We define a mapping Φ from our original feature space R^m to F such that the mapping Φ could be any nonlinear mapping and the space F could have any arbitrary dimensionality.[12] A dot product matrix K can be defined whose element is the dot product of any two mapped data points:[12]

$$K(i,j) = \Phi(x_i) \bullet \Phi(x_j) \qquad i,j \in 1,2,\ldots,n \tag{3}$$

where x_i represents the ith data point in the original feature space, and $\Phi(x_i)$ is the mapped data point in the new feature space. Each x_i has m dimensions, and there are n data points. Eigenvectors of K are then computed and normalized in F. Kernel principal components can be obtained by projecting the original data points onto the eigenvectors.[12] The dot product in eq. 3 can be computed indirectly through a kernel function:[12]

$$k\left(x_i, x_j\right) = \Phi(x_i) \bullet \Phi(x_j) \tag{4}$$

where the kernel k could be a polynomial function, multilayer perceptron, or a radial basis function such as:

$$k\left(x_i, x_j\right) = \exp\left(-\frac{\left\|x_i - x_j\right\|^2}{2\sigma^2}\right) \tag{5}$$

where σ is a supplied standard deviation for the function. We used a radial basis kernel function with various values for σ using a Matlab program modified from Dr. Bernhard Schoelkopf's kpca code.[12] Note that KPCA can also be used as a feature expansion method in that the number of features extracted by KPCA can range up to the number of data points. Strictly speaking, KPCA is therefore a feature extraction method, not only a feature reduction method.

2.4 Independent Component Analysis (ICA)

The reduced feature set from PCA, KPCA, or NLPCA might still contain dependency among features. Independent component analysis (ICA) is designed to capture those features that are most independent from each other.[10] Suppose we have d independent source signals s such that our data matrix can be represented as:

$$D = sB \tag{6}$$

where D is a $n \times m$ feature matrix containing n data points and m features, s is a $n \times d$ source matrix containing d independent source signals, and B is a $d \times m$ transformation matrix. In order to solve for s from D, we assume that s is a linear combination of the features in D followed by a nonlinear transformation f:[10]

$$s = f\left(WD + w_0\right) \tag{7}$$

where W and w_0 are weights and the central goal of ICA is to solve for these weights such that the source signals in s are as independent from each other as possible.

To maximize the independence in s, one can use a criterion function of s, such as nongaussianity. The rational behind nongaussianity is that any orthogonal transformation of Gaussian distributed variables gives the same Gaussian distribution.[13] Therefore, one can only recover the hidden independent variables up to an orthogonal transformation if they are Gaussian distributed. To carry out ICA, we used the FastICA Matlab toolbox,[13] which employs a fixed-point scheme to maximize the nongaussianity of s.

2.5 Classification Trees

Feature selection using the classification tree method is conducted by ranking all features according to their information gain ratios. Suppose our data set D has m features, the information gain ratio of feature X_i is defined as:[14]

$$Gain(D, X_i) = \frac{Entropy(D) - \sum_{v \in V_i} \frac{|D_v|}{|D|} Entropy(D_v)}{-\sum_{v \in V_i} \frac{|D_v|}{|D|} \log \frac{|D_v|}{|D|}} \qquad i = 1,2,\ldots,m \tag{8}$$

where V_i is the set of all possible values that X_i could have and D_v represents the data subset in which X_i has the value of v. Note that the numerator is the information gain of X_i, a measure of the goodness of an attribute in separating the training data, and the denominator is called the split information of feature X_i.[14] Those features that are different at almost every data point are favored by the information gain but not by information gain ratio because they are penalized by the split information. We used the C4.5 system[15] to compute the information gain ratio of every feature.

2.6 Fractal Dimensionality Reduction (FDR)

A highly redundant data set should have an intrinsic dimensionality much smaller than the actual dimensionality of the feature space m. In other words, many features might have no contribution to the intrinsic dimensionality characterizing the data. If one could measure the intrinsic dimensionality of the data directly, features that do not affect the intrinsic dimensionality could be dropped. The fractal dimensionality of a data set, which describes how self-similar the data points are, is a good approximation of the intrinsic dimensionality of the data (it is often represented as correlation fractal dimension[16]). This is the basis for the FDR[16] algorithm, which uses a backward elimination scheme to select important features. First, the correlation fractal dimensionality of the whole dataset is computed. For each feature in the feature space, partial fractal dimensionality is then computed using all features except the current feature. Features are sorted by their partial fractal dimensionalities and those that lead to minimum decrease from the original fractal dimensionality are dropped. The whole process continues until no feature could be dropped, namely dropping any feature in the final feature set would change the previous partial fractal dimensionality less than a fixed threshold.[16] Two advantages of FDR are that it gives an approximation of the number of features we should keep after reduction, namely the fractal dimensionality of the whole data set, and it does not require class separation of the original dataset. We used the FracDim package[16] with various values of the retention threshold (the min_gain parameter).

2.7 Stepwise Discriminant Analysis (SDA)

In a classification problem, good features should have the natural characteristic of separating different classes from one another while at the same time keeping each cluster as tightly packed as possible. As reviewed previously,[17] this property can be measured using Wilks' Λ, defined as:

$$\Lambda(m) = \frac{|W(X)|}{|T(X)|}, X = [X_1, X_2, \ldots, X_m] \tag{9}$$

where X represents the m features currently used and the within-group covariance matrix W and the among-group covariance matrix T are defined as:

$$W(i,j) = \sum_{g=1}^{q} \sum_{t=1}^{n_g} (X_{igt} - \overline{X}_{ig})(X_{jgt} - \overline{X}_{jg}), \quad i,j \in 1,2,\ldots,m \tag{10}$$

$$T(i,j) = \sum_{g=1}^{q} \sum_{t=1}^{n_g} (X_{igt} - \overline{X}_{i})(X_{jgt} - \overline{X}_{j}), \quad i,j \in 1,2,\ldots,m \tag{11}$$

where i and j represent the ith and jth features, X_{igt} is the ith feature value of the data point t in the class g, $\overline{X}_{ig}$ is the mean value of the ith feature in the class g, and $\overline{X}_{i}$ is the mean value of the ith feature in all classes, q is the total number of classes, and n_g is the number of data points in the class g. The partial Wilks' Λ is defined by adding an additional feature X_{m+1} to X in (9):

$$\Lambda(m+1) = \frac{\Lambda([X_1, X_2, \ldots, X_m, X_{m+1}])}{\Lambda(m)}, \quad X_{m+1} \text{ is the } (m+1)\text{th feature} \tag{12}$$

The corresponding F-statistic, also called F-to-enter, decides whether the feature m+1 should be added into the current feature subset and is defined as:

$$F = \left(\frac{n-q-m}{q-1}\right)\left(\frac{1-\Lambda(m+1)}{\Lambda(m+1)}\right) \tag{13}$$

where n is the total number of data points, q is the number of classes, and m is the number of features currently considered. F-to-remove can be defined similarly by taking a feature out of the current feature subset.

We used the STEPDISC procedure of SAS (SAS Institute, Cary, NC, USA) to select a ranked feature subset. Specifically, stepwise discriminant analysis[7] starts with the full set of all features, calculates the F-to-remove of each

feature, and removes the feature with lowest F value under a significance level. The W and T matrices are then updated by column-wise sweeping and F-to-enter is calculated for each feature that is not currently in the set. The feature that corresponds to the highest F value under a significance level is added to the current set. The whole process goes back and forth until no features can be added or deleted.

2.8 Genetic Algorithm

The search path that sequential search methods explore in both 2.6 and 2.7 is directed by the distribution of the computed statistic. We could, alternatively, use a method that goes through as many random combinations of features as possible to find the one giving the best classification accuracy. Feature selection employing genetic algorithms does a random search in the whole feature subset space.[10] It can also take into account multiple criterions in selecting features such as classification accuracy, cost, and risk.[18] Genetic algorithms are a general feature selection method regardless of the application domain.

Several factors make up a genetic-algorithm-based feature selection scheme[18] (Figure 1). First, we need to represent each feature subset in a way easy to manipulate by genetic operators. A common solution is to use a bit string. Second, we generate an initial pool of candidate feature subsets on which genetic operators such as crossover and mutation are applied. For each resulting feature subset, a fitness function is employed to evaluate its goodness. The feature subsets are further processed by a selection strategy to choose individuals with highest goodness. A stop criterion is applied after each iteration to examine whether we should stop and output selected feature subsets or proceed to the next round of genetic manipulation. A stop criterion could be a maximum number of iterations or a minimum improvement of fitness over the previous iteration.

We use the matlab toolbox GAOT[19] in our experiments. Each feature subset is represented as a bit string, in which one stands for inclusion of the corresponding feature. A multi-class support vector machine (SVM) along with 10-fold cross validation is used as the fitness function. Genetic operators, mutation and crossover, and a ranking selection scheme based on normalized geometric distribution are adapted from the toolbox GAOT. The initial population size was 50, and the probabilities of crossover and mutation were 0.8 and 0.1 respectively. The maximum number of iteration was 100. These parameters were determined after several preliminary trials.

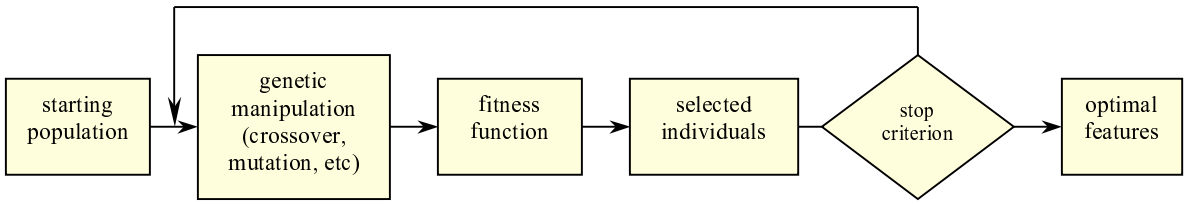

Figure 1. Flow chart of feature selection using genetic algorithm

2.9 Classifier

Support vector machines (SVM)[20] have been extensively used in many machine learning domains with comparable performance to neural network.[21] We used an SVM matlab toolbox (http://theoval.sys.uea.ac.uk/~gcc/svm/ toolbox/) implementing a tree-based multi-class SVM, *dagsvm*,[22] with a Gaussian kernel ($\sigma^2 = 50$, C=20) along with 10-fold cross validation to evaluate the resulting feature subsets from all eight feature reduction methods. It was also used as the fitness function in feature selection using genetic algorithm. The choice of kernel type and parameter setting of *dagsvm* was based on preliminary cross validation trials on predefined feature subsets (data not shown).

3. DATA

The 2D HeLa image set described previously[3] was used for all analyses. It contains 862 fluorescence microscope images representing ten typical subcellular organelles and structures. This data set was generated by applying

fluorescent probes against characteristic molecules in specific subcellular locations: DNA in the nucleus, actin in microfilaments, an endoplasmic reticulum protein, transferrin receptor in endosomes, LAMP2 in lysosomes, Giantin and gpp130 in the Golgi apparatus, nucleolin in nucleoli, an outer membrane protein in mitochondria, and tubulin in microtubules. Each image in the set has a resolution of 382 × 512 pixels, each 0.23 x 0.23 microns.

4. RESULTS

To provide a baseline for comparison, we determined that the average classification accuracy obtained with SVM was 85.2% using the SLF7 feature set and was 86.7% using the SLF8 feature set. SLF8 was previously optimized for neural network classifiers from a subset of SLF7 selected by SDA, providing an average accuracy of 86%.[5]

4.1 Feature Recombination

To obtain initial insight into the degree of redundancy in SLF7, we generated a histogram depicting all pair-wise feature correlation coefficients (Fig. 2). The results show that extensive correlation exists between some of the features. PCA was then carried out to examine whether a few linearly recombined features that capture most data variance could give high classification accuracy. Fig. 3a displays the percentage of total variance represented by sorted principal components. From Fig. 3a and 3b, we can see that the highest accuracy achieved using PCA is 83.4% with 41 principal components that represent 97.3% of the total data variance.

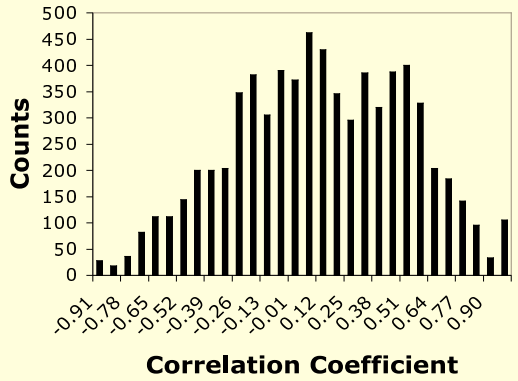

Figure 2. Histogram of all pair-wise feature correlation coefficients in SLF7.

Since the remaining 43 principal components account for less than 3% of the total data variance, they could be regarded as noise. Inclusion of noisy features can decrease classification accuracy, and this was observed above 41 principal components in Fig. 3b.

While the best performance with PCA was comparable to that for all SLF7 features, the highest accuracy achieved using NLPCA was 75.3% with 64 features (Fig. 3b). The fact that PCA performs better than NLPCA implies that the SLF7 features are more likely to have linear relationships and that the additional overhead of trying to fit nonlinear relationships degrades performance.

KPCA was then conducted using a radial basis kernel with different variance levels (expressed as the denominator, $2\sigma^2$,

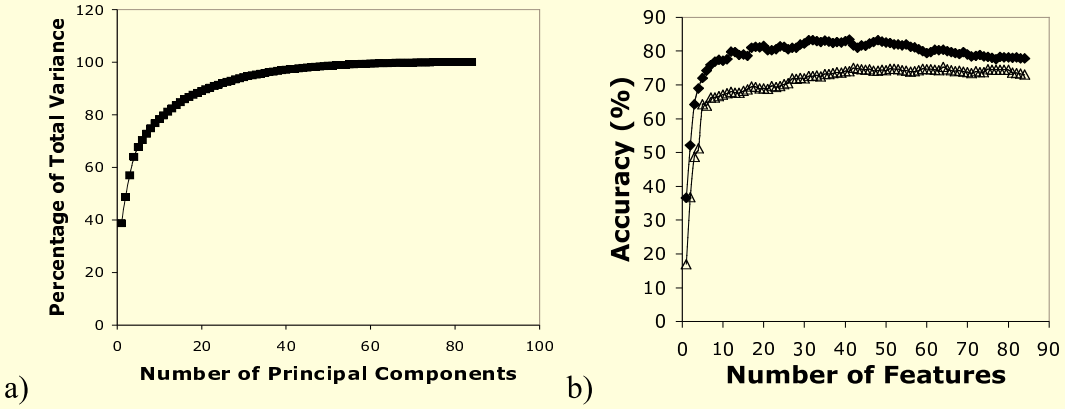

Figure 3. Feature reduction using PCA and NLPCA. (a) Variance captured by increasing numbers of sorted principal components. (b) Classification accuracy using SVM after feature reduction by PCA (◆) or NLPCA (Δ).

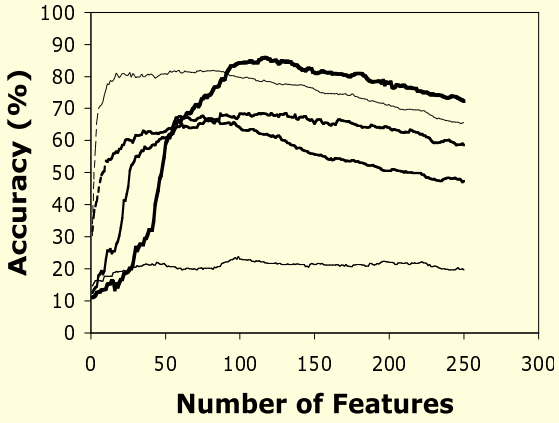

Figure 4. Feature recombination using KPCA with different kernel variances: 100 (thin dashed line), 10 (medium dashed line), 1 (thin solid line), 0.1 (medium solid line), 0.01 (thick solid line).

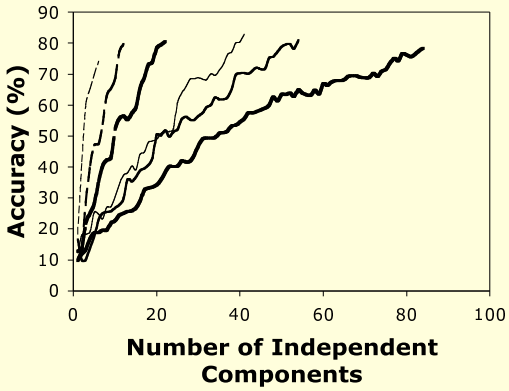

Figure 5. Feature reduction using ICA with different preprocessing levels as described by the number of principal components and the corresponding percentage of variance: 6PC-70% (thin dashed line), 12PC-80% (medium dashed line), 22PC-90% (thick dashed line), 41PC-97% (thin solid line), 54PC-99% (medium solid line), 84PC-100% (thick solid line).

in eq. 5). Since KPCA maps the original feature space to a very large nonlinear space, the maximum number of features that could be retrieved from the new space equals the number of data points (862). KPCA works as a feature extraction method and does not necessarily reduce the dimensionality of the original feature space. The first 250 of all possible features that are related to the first 250 eigenvectors of the matrix K in eq. 3 were extracted and classification accuracy was determined. As Fig. 4 shows, the highest accuracy achieved was 86.0% using a kernel variance of 0.01 and 117 features. The slightly better accuracy achieved by KPCA compared to PCA comes at the penalty of using 33 more features than originally present in SLF7. As observed with PCA, adding more features beyond this point decreases overall classification accuracy. With increasing kernel variance, KPCA behaves more and more like PCA, as can be seen by comparing the PCA curve in Fig. 3b and KPCA curve for a variance of 100 in Fig. 4.

ICA was used as the last feature reduction method. Before performing ICA, preprocessing on the original feature space was carried out to test the performance of ICA under different noise levels. We compared the results of directly applying ICA without any noise smoothing to those of different noise-smoothing settings, namely removing noise by keeping the first 6, 12, 22, 41, and 54 principal components that represent 70%, 80%, 90%, 97%, and 99% of total data

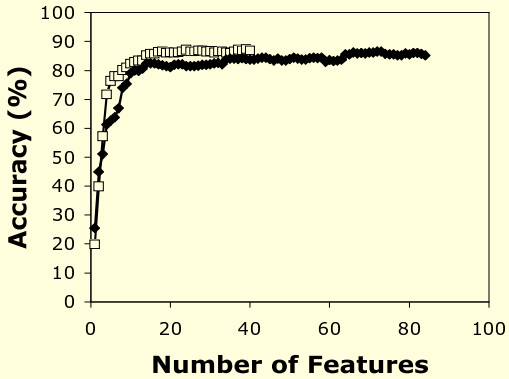

Figure 6. Feature selection using SDA (□) and Classification Tree information gain ratio (◆).

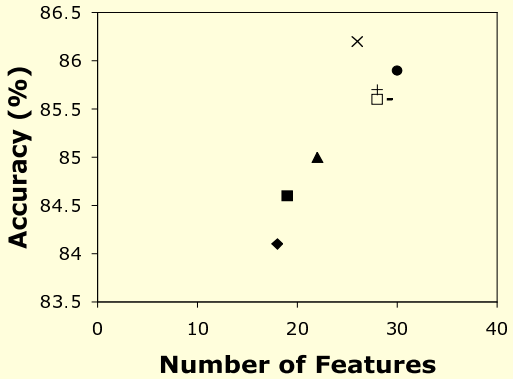

Figure 7. Feature selection using FDR with different threshold parameters: 0.1(◆), 0.08(■), 0.04(▲), 0.02(×), 0.01(□), 0.008(+), 0.004(–), 0.001(●).

variance respectively (Fig. 5). ICA with noise-smoothing preprocessing performs much better than without preprocessing. The highest accuracy achieved was 82.9% by keeping 41 principal components in the preprocessing step, consistent with the best PCA result.

True Class	Output of Classifier									
	DNA	ER	Gia	Gpp	LA	Mit	Nuc	Act	TfR	Tub
DNA	99	1	0	0	0	0	0	0	0	0
ER	1	88	0	0	4	3	0	0	3	1
Gia	1	0	74	21	0	0	0	0	4	0
Gpp	0	0	22	75	0	0	1	0	2	0
LA	0	1	0	0	87	1	0	0	11	0
Mit	4	7	0	0	1	81	0	0	4	3
Nuc	1	0	1	2	0	0	96	0	0	0
Act	0	0	0	0	0	0	0	100	0	0
TfR	1	0	0	0	9	2	0	3	82	3
Tub	0	4	0	0	0	0	0	2	2	92

Table 1. Confusion matrix for SVM classifiers using 39 features selected by SDA. The average accuracy was 87.4%.

4.2 Feature Selection

Classification tree information gain ratio and SDA were next compared as two feature selection methods that give a consecutive number of selected features (Fig. 6). SDA performs better than the classification tree information gain ratio with higher classification accuracy achieved by fewer features. The highest accuracy achieved by SDA is 87.4% with 39 features (Fig. 6 and Table 1) while that using information gain ratios is 86.6% with 72 features. The redundancy of SLF7 can be noticed by the fact that no significant increase in the classification accuracy could be observed with more features beyond the first 35 features selected using both methods. We noticed that there is a small feature subset containing 8 features selected by SDA that could give an average classification accuracy of 80.1% (Table 2). This subset was defined as SLF12 and Table 3 shows the descriptions of its features. Comparing Table 1 and 2, we noticed that the largest reduction in accuracy with SLF12 is for the two most confusing patterns, Giantin and Gpp130, that are both located in the Golgi apparatus. SLF12 may be able to serve as a feature set applicable for coarse recognition of major subcellular structures with high accuracy, as may be seen if we combine the two Golgi patterns (and perhaps those for endosomes and lysosomes) together.

Feature selection using fractal dimensionality reduction was conducted with different stop thresholds (Fig. 7). With decreasing threshold, the number of retrieved features increases until the threshold approaches 0.001, after which the number of resulting features is constant at 30 (data not shown). The highest accuracy achieved is 86.2% with 26 features for a threshold value of 0.02. This result shows that there is equilibrium in the heuristic search of FDR such that small threshold values can result in swamping the system with noisy features and large threshold values can increase the risk of losing good features.

Lastly, we carried out feature selection using a genetic algorithm. Such algorithms have dependence on where in the feature space we start the search. We tried three starting choices (Fig. 8): random subsets of SLF7; the SLF8 feature set; and the SLF12 feature set. A random start performs better than the other two locally good starts with fewer steps to converge and higher classification accuracy achieved. A good start might trap the program in some local minimum and prevent the system from converging to an optimal feature subset. For instance, SLF8 as a very good start subset traps the system to a suboptimal state after two iterations, while a random start jumps out from the same suboptimal to a better suboptimal state after one more iteration. The highest accuracy achieved is 87.5% with 43 features and random starting feature subsets, while 87.1% with 34 features and 85.9% with 11 features are achieved using SLF8 and SLF12 as the starting subset respectively.

True Class	Output of Classifier									
	DNA	ER	Gia	Gpp	LA	Mit	Nuc	Act	TfR	Tub
DNA	99	1	0	0	0	0	0	0	0	0
ER	0	83	0	0	6	0	0	0	0	11
Gia	2	0	55	38	1	0	0	0	4	0
Gpp	1	0	28	64	3	0	2	0	2	0
LA	0	3	0	6	83	0	1	0	7	0
Mit	0	11	0	0	3	79	0	0	0	7
Nuc	2	0	1	0	1	0	96	0	0	0
Act	0	0	0	0	0	1	0	99	0	0
TfR	2	0	0	1	17	0	0	1	55	24
Tub	0	8	0	0	0	1	0	0	3	88

Table 2. Confusion matrix for SVM classifiers using the 8 features of SLF12. The average accuracy was 80.1%.

5. CONCLUSION

The choice of feature reduction method is often a big concern in machine learning applications. Our experiments comparing eight feature reduction methods in the protein subcellular location domain shows that feature selection methods not only give clearer description of reduced feature subset, but perform generally better than feature recombination methods. Table 4 summarizes the results from all eight methods. Stepwise Discriminant Analysis (SDA) and the Genetic Algorithm give the highest accuracy. SDA is also the most efficient algorithm, which generates the smallest current feature set, SLF12, for an average 80% classification accuracy in 10 major subcellular organelles and structures. As a relatively old statistical method, SDA deserves more attention since it surprisingly outperforms recent advances, such as KPCA and ICA. The reason might be that the simple first order statistics employed in SDA captures the ground truth in machine learning: the ability of each feature to separate different classes while keeping every single class as tight as possible determines the discrepancy among classes.

The 84-dimensional SLF7 feature set could be reduced to a subset containing 39 features selected by SDA that gives higher accuracy by using *dagsvm* than that of SLF8 with a one-hidden-layer neural network. In addition, *dagsvm* trained by the SMO method performs much faster than neural network both in training and testing. Therefore, it is desirable to use *dagsvm* in any wrapper-based feature selection system where the training and testing time of the wrapper classifier determines the efficiency of the system. The smaller SLF12 contains three texture features, two moment features, and three morphological features, which demonstrates the importance of diversity in choosing feature sources for protein subcellular location recognition via fluorescence microscope images. The smaller size of the reduced feature subset can reduce computation load in real-time applications such as online image classification and could serve as multidimensional indexing basis for content-based image database applications.

	Feature Description
SLF1.3	The average number of above-threshold pixels per object
SLF7.74	Haralick texture features: Entropy
SLF3.19	Zernike moment feature Z_2_0 with polynomial degree 2 and angular dependence 0
SLF7.79	The fraction of cellular fluorescence not included in objects
SLF7.71	Haralick texture features: Sum Average
SLF7.76	Haralick texture features: Difference Entropy
SLF3.23	Zernike moment feature Z_4_0 with polynomial degree 4 and angular dependence 0
SLF7.9	The fraction of the non-zero pixels in a cell that are along an edge

Table 3. The features in SLF12, which comprise the first 8 features selected by SDA from SLF7.

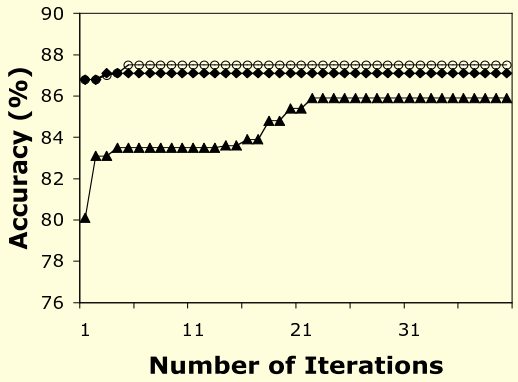

Figure 8. Feature selection using Genetic Algorithms with random feature subsets(❍), SLF8 subset(◆), and SLF10 subset(▲) as the starting populations.

Method	Minimum Number of Features for Over 80% Accuracy	Highest Accuracy (Number of Features)
None	N/N	85.2%(all 84)
PCA	17	83.4% (41)
NLPCA	N/N	75.3% (64)
KPCA	17	86.0% (117)
ICA	22	82.9% (41)
Classification Tree	11	86.6% (72)
SDA	8	87.4% (39)
FDR	18	86.2% (26)
Genetic Algorithm	N/N	87.5% (43)

Table 4. Summary of results from all feature reduction methods.

6. ACKNOWLEDGMENTS

We thank Christos Faloutsos and Leejay Wu for helpful discussions on the FDR feature selection method. This work was supported in part by NIH grant R33 CA83219 and by a research grant from the Commonwealth of Pennsylvania Tobacco Settlement Fund. K.H. was supported by a Graduate Fellowship from the Merck Computational Biology and Chemistry Program at Carnegie Mellon University through a grant from the Merck Company Foundation.

7. REFERENCES

1. M.V. Boland, M.K. Markey, and R.F. Murphy, "Classification of Protein Localization Patterns Obtained via Fluorescence Light Microscopy," *19th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 594-597, Chicago, IL, USA, 1997.
2. M.V. Boland, M.K. Markey, and R.F. Murphy, "Automated recognition of patterns characteristic of subcellular structures in fluorescence microscopy images," *Cytometry*, **33**, 366-375, 1998.
3. M.V. Boland and R.F. Murphy, "A Neural Network Classifier Capable of Recognizing the Patterns of all Major Subcellular Structures in Fluorescence Microscope Images of HeLa Cells," *Bioinformatics*, **17**, 1213-1223, 2001.
4. M. Velliste and R.F. Murphy, "Automated Determination of Protein Subcellular Locations from 3D Fluorescence Microscope Images," *Proceedings of the 2002 IEEE International Symposium on Biomedical Imaging (ISBI-2002)*, pp. 867-870, Bethesda, MD, USA, 2002.
5. R.F. Murphy, M. Velliste, and G. Porreca, "Robust Classification of Subcellular Location Patterns in Fluorescence Microscope Images," *Proceedings of the 2002 IEEE International Workshop on Neural Networks for Signal Processing (NNSP 12)*, pp. 67-76, 2002.
6. R.F. Murphy, M.V. Boland, and M. Velliste, "Towards a Systematics for Protein Subcellular Location: Quantitative Description of Protein Localization Patterns and Automated Analysis of Fluorescence Microscope Images," *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pp. 251-259, San Diego, 2000.
7. R.I. Jennrich, "Stepwise discriminant analysis," *Statistical Methods for Digital Computers*, K Enslein, A Ralston, and HS Wilf, Editors, pp. 77-95, John Wiley & Sons, New York, 1977.
8. G. John, R. Kohavi, and K. Pfleger, "Irrelevant features and the subset selection problem," *Machine Learning: Proceedings of the Eleventh International Conference*, pp. 121-129, Morgan Kaufmann, 1994.
9. W.R. Dillon and M. Goldstein, *Multivariate Analysis: Methods and Applications*, John Wiley and Sons, Inc., New York, 1984.
10. R.O. Duda, P.E. Hart, and D.G. Stork, *Pattern Classification*, 2nd ed, John Wiley & Sons, New York, 2000.

11. W.W. Hsieh, "Nonlinear principal component analysis by neural networks," *Tellus*, **53A**, 599-615, 2001.
12. B. Scholkopf, A. Smola, and K.-R. Muller, "Nonlinear Component Analysis as a Kernel Eigenvalue Problem," *Neural Computation*, **10**, 1299-1319, 1998.
13. A. Hyvärinen, "Fast and Robust Fixed-Point Algorithms for Independent Component Analysis," *IEEE Transactions on Neural Networks*, **10**, 626-634, 1999.
14. T.M. Mitchell, *Machine Learning*, WCB/McGraw-Hill, 1997.
15. J.R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, CA, 1993.
16. C. Traina, A. Traina, L. Wu, and C. Faloutsos, "Fast feature selection using the fractal dimension.," *XV Brazilian Symposium on Databases (SBBD)*, Paraiba, Brazil, 2000.
17. M.V. Boland, *Quantitative Description and Automated Classification of Cellular Protein Localization Patterns in Fluorescence Microscope Images of Mammalian Cells*, PhD thesis, Biomedical Engineering, Carnegie Mellon University, Pittsburgh, PA, U.S.A., 1999.
18. J. Yang and V. Honavar, "Feature subset selection using a genetic algorithm," *IEEE Intelligent Systems*, **13**, 44-49, 1998.
19. C. Houck, J. Joines, and M. Kay, *A Genetic Algorithm for Function Optimization: A Matlab Implementation*, NCSU-IE TR, 95-09, 1995.
20. C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, **20**, 1-25, 1995.
21. S.R. Gunn, *Support Vector Machines for Classification and Regression*, University of Southampton TR, 10, 1998.
22. J. Platt, N. Cristianini, and J. Shawe-Taylor, "Large Margin DAGs for Multiclass Classification," *Advances in Neural Information Processing Systems*, **12**, 547-553, 2000.