ELSEVIER

Computers in Biology and Medicine 36 (2006) 339–362

Computers in Biology and Medicine

www.intl.elsevierhealth.com/journals/cobm

Automatic detection of unstained viable cells in bright field images using a support vector machine with an improved training procedure

Xi Long[a], W. Louis Cleveland[b,*], Y. Lawrence Yao[a]

[a] *Mechanical Engineering Department, Columbia University, 500 West 120th Street, 220 Mudd. MC 4703, New York, NY 10027, USA*

[b] *Department of Medicine at St. Luke's Roosevelt Hospital Center and Columbia University, New York, NY 10019, USA*

Received 7 September 2004; accepted 3 December 2004

Abstract

Detection of unstained viable cells in bright field images is an inherently difficult task due to the immense variability of cell appearance. Traditionally, it has required human observers. However, in high-throughput robotic systems, an automatic procedure is essential. In this paper, we formulate viable cell detection as a supervised, binary pattern recognition problem and show that a support vector machine (SVM) with an improved training algorithm provides highly effective cell identification. In the case of cell detection, the binary classification problem generates two classes, one of which is much larger than the other. In addition, the total number of samples is extremely large. This combination represents a difficult problem for SVMs. We solved this problem with an iterative training procedure ("Compensatory Iterative Sample Selection", CISS). This procedure, which was systematically studied under various class size ratios and overlap conditions, was found to outperform several commonly used methods, primarily owing to its ability to choose the most representative samples for the decision boundary. Its speed and accuracy are sufficient for use in a practical system.
© 2005 Elsevier Ltd. All rights reserved.

Keywords: Cell detection; Structural risk minimization; Kernel functions; Support vector machines

* Corresponding author. Tel.: +1 212 523 7302; fax: +1 212 523 5377.
E-mail address: wlc1@columbia.edu (W.L. Cleveland).

0010-4825/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compbiomed.2004.12.002

1. Introduction

Cell detection in bright field microscopy is an inherently difficult task due to the immense variability of cell appearance. Even more difficult is the recognition of the subtle differences in appearance that distinguish unstained viable from non-viable cells in bright field images. Although an experienced observer can sometimes recognize these differences, viability stains are commonly used for reliable determination of viability [1]. The requirement of a human observer represents a severe impediment to the development of high throughput robotic systems that require recognition of viable cells. Therefore, there is a great need for effective algorithms that automatically recognize viable cells.

Currently, a typical approach for cell detection is to use fluorescent probes that have a chemical specificity for cell organelles [2,3]. However, this approach can consume valuable fluorescence channels just for the purpose of cell identification. It is highly desirable to identify cells with a method that uses transmitted light illumination, thereby permitting all fluorescence channels to be used to obtain cellular and subcellular information for further cell analysis.

Classical image analysis approaches require end-users to have programming skills and require independent optimizations for different cell types [4,5]. An alternative is to use machine learning techniques, which avoid end-user programming since classifiers only need to be trained. For example, artificial neural networks (ANNs) have been successfully used to identify cells in bright field images [6,7]. These algorithms are able to capture complex, nonlinear, relationships in high-dimensional feature spaces. However, ANNs are based on the empirical risk minimization (ERM) principle [8]. Therefore, they are prone to false optimizations due to local minima in the optimization function and are susceptible to training problems such as "overfitting" [7]. This makes ANN-training a complex procedure that may be daunting for biologists and others who are not immersed in the complexities of ANNs. In a search for statistical learning methods that are easier to use and more accurate, we have explored support vector machines (SVMs).

In recent years, SVMs have been found to be remarkably effective in many real-world applications [8–12]. Unlike ANNs, SVMs follow the structural risk minimization (SRM) principle, which aims at minimizing an upper bound of the generalization error [8]. As a result, a SVM tends to perform well when applied to data outside the training set. SVMs also have many desirable properties such as flexibility in choice of kernel function and implicit mapping into high-dimensional feature spaces [8]. But what makes SVMs most attractive is that they avoid several major problems associated with ANNs. For example, SVMs control overfitting by restricting the capacity of the classifier. They also depend on the solution of a convex quadratic programming (QP) problem which has no local extrema. The unique optimal solution can therefore be efficiently obtained.

In this study, we have treated the detection of unstained viable cells in bright field images as a binary pattern classification problem. The two classes are referred to as "Viable-cell" (VC) and "Not-a-viable-cell" (NAVC). Data elements of these classes are derived from the "pixel patch" decomposition technique that we and others have used previously [2,3,7]. The "VC" class consists of pixel patches that enclose complete, viable cells. The "NAVC" class includes pixel patches that contain dead cells, fragments of cells and other non-cell objects.

We found that the "pixel patch" approach generates data sets that are problematic for SVMs. For example, in the 320×240 image shown in Fig. 1, only 43 patches were put into the "VC" class. The number of extracted patches in the "NAVC" class was 19,600 even after eliminating background patches

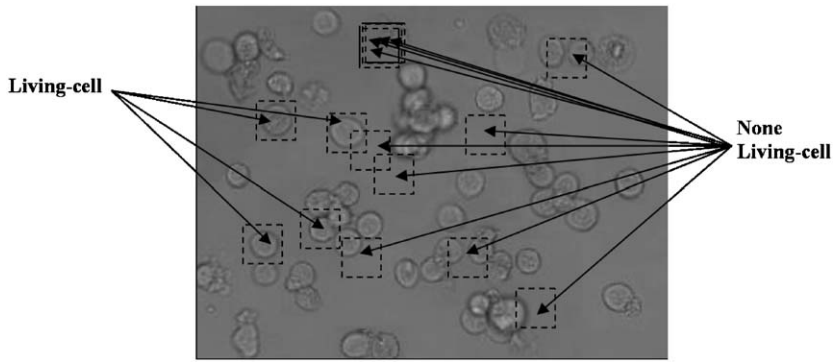

Fig. 1. Typical training patches in living cell detection. The sample numbers in the two classes are highly unbalanced. The total number of "VC" patches in this 320 × 240 image is 43 and that of "NAVC" patches is about 19,600 (after eliminating the background patches).

in the preprocessing step. This scenario is a difficult problem for SVMs because the classes are highly unbalanced and the total sample number is impractically large.

The large number of training samples required for cell detection is a formidable obstacle to many standard QP techniques, such as steepest ascent algorithms [13,14], which require that all data be held in memory and moreover require a memory space that grows quadratically with the sample size. This consideration has driven the design of alternative algorithms [9,15–20]. For example, the "chunking" method [15] maintains a variable-size subset of the training data called a "chunk" and iteratively updates it according to how the Karush–Kuhn–Tucker (KKT) conditions [26] are violated. Similar strategies are used in the decomposition algorithm proposed by Osuna et al. [9], in the *SVM light* application [19,20] and in the sequential minimal optimization (SMO) algorithm [17,18]. However, in these algorithms, only a fixed-size subset of the training set, called the "working set", is used. These iterative methods have been shown to be able to handle problems with several hundred thousand training points and over 100,000 support vectors (SVs) [16]. However, in all of these cases, balanced classes were considered. In our case, the training samples are also highly unbalanced in addition to being large in number.

Currently, a popular strategy to handle unbalanced training samples [9] is the one proposed by Osuna et al. who use different penalty parameters in the SVM formulation. This strategy produces useful results for some applications [11,21–23], but introduces new parameters, making parameter optimization more complex. For example, the number of parameters to be optimized increases from 2 (C and γ) to 3 (C_{+1}, C_{-1} and γ) for a binary classification task. The increased complexity can make many applications computationally impractical, especially those with large numbers of training samples.

In this study, an SVM training algorithm called "Compensatory Iterative Sample Selection" (CISS) has been developed and successfully applied to the challenging problem of identifying unstained viable cells in cultures. This algorithm, which is a modification of the strategy introduced in [9], exploits the geometrical characteristic of SVMs that samples closer to the boundary are more influential on shaping and defining the decision surface than others. Our experimental results suggest that this algorithm can reduce classification errors to the point where practical applications are readily possible.

2. Materials and experimental conditions

The cells used as testing materials in our study were A20.2J murine B-cell lymphoma cells (ATCC; CAT. no. HB-97) grown at 37.0 °C in BM + 1/2 TE1 + TE2 + 10% fetal calf serum (FCS) [24]. For microscope observation, cells in culture medium were dispensed into polystyrene 96-well microplates, which have glass bottoms that are 0.175 mm thick.

To obtain an accurate training and testing standard, the LIVE/DEAD viability reagent set for animal cells from molecular probes (CAT. no. L3224) was used to distinguish viable and dead cells. This reagent set produces an intense green fluorescence in viable cells (excitation/emission $\sim$ 495 nm/ $\sim$ 515 nm) and a bright red fluorescence in dead cells (ex/em $\sim$ 495 nm/ $\sim$ 635 nm). An ARCTURUS Pixcell II inverted microscope equipped with a 20× planachromat objective and a HITACHI model KP-D580-S1 CCD color camera was used to obtain digitized images. For each microscope field, three images were acquired (Fig. 2). One image was acquired with bright field illumination and was used for SVM training or testing. Two auxiliary fluorescence images were also acquired to distinguish viable and dead cells, one with a standard fluorescein bandpass filter and the other with a filter for propidium iodide.

Fifty nine sets of microscope images were acquired and used in our cell detection experiments. In each experiment, two subsets were extracted: one exclusively for training and another exclusively for testing. Ambiguous objects showing both red and green fluorescence were manually deleted. The deleted objects were a very small percentage of the total number of cells.

The computer programs were written in MATLAB and C++. Our algorithm was implemented with the LIBSVM version 2.5 [25], which was compiled as a dynamic link library for MATLAB. All experiments

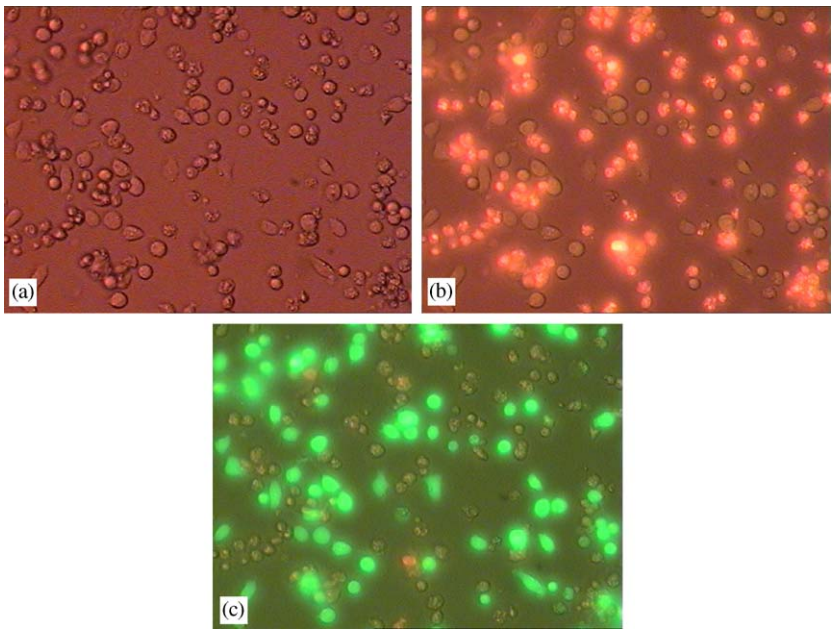

Fig. 2. Typical images used to construct a rigorously preclassified training set: (a) bright field image; (b) fluorescence image #1: red fluorescence stain; and (c) fluorescence image #2: green fluorescence stain.

were implemented in the environment of MATLAB version 6.5.0.180913a (R13) supplemented with image processing toolbox version 3.2. A standard PC equipped with an intel pentium 4/2.8 G processor and 256-MB RAM was used.

3. Compensatory iterative sample selection (CISS)

3.1. Important features of SVMs

Our development of the CISS algorithm requires a consideration of some of the fundamental aspects of SVMs. These are briefly described in this section. A detailed introduction to SVM can be found in [8,26].

SVMs represent an approximate implementation of the SRM principle and were first introduced by Vapnik et al. to solve pattern recognition and regression estimation problems [8]. In what follows, we denote the training data as $(x_i,\ y_i = f(x_i)) : i = 1, 2, \ldots, l,\ y_i \in \{-1, +1\}, x_i \in \mathbb{R}^n$.

In a linearly separable case shown in Fig. 3, the SVM classifier follows the intuitive choice and selects the hyperplane (among many that can separate the two classes) that maximizes the margin, where the margin is defined as the sum of the distances of the hyperplane to the closest points of the two classes.

If the two classes are non-separable, positive slack variables are introduced to allow some training samples to fall on the wrong side of the separating hyperplane. The SVM then finds the hyperplane that maximizes the margin and, at the same time, minimizes a quantity proportional to the number of classification errors. The trade-off between maximizing the margin and minimizing the error is controlled by a user-adjusted regularization parameter $C > 0$. A large C corresponds to a high penalty for classification errors.

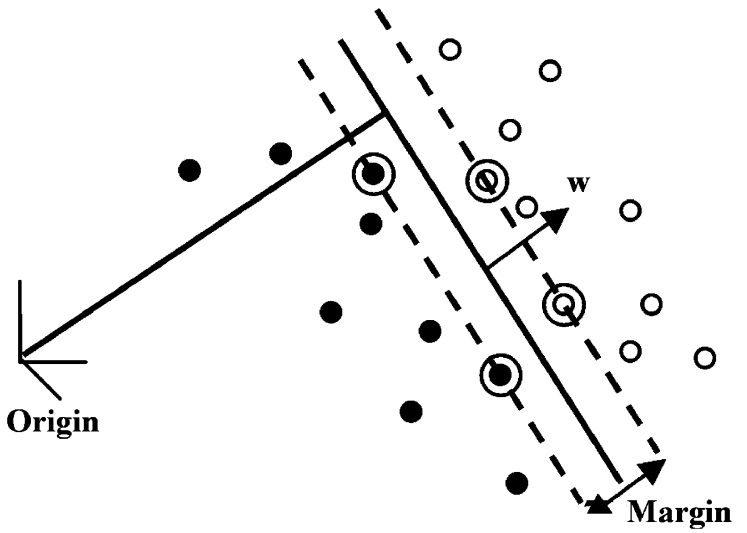

Fig. 3. Illustration of SVM classification (linearly separable case). The SVs are circled.

In many practical cases, however, nonlinear decision surfaces are needed. Nonlinear SVMs can be generalized from linear SVMs by using a nonlinear operator $\Phi(\cdot)$ to map the input pattern x into a higher (even infinite) dimensional Euclidean space H. It has the form:

$$f(x) = w^{\mathrm{T}}\Phi(x) + b. \tag{1}$$

Mathematically, it can be shown that the solution of the nonlinear case is

$$f(x) = \sum_{i=1}^{l} \alpha_i y_i \Phi^{\mathrm{T}}(x_i)\Phi(x) + b = \sum_{i=1}^{l} \alpha_i y_i K(x_i, x) + b, \tag{2}$$

where the coefficients α_i are the solution of the following convex QP problem:

$$\max L_{\mathrm{D}}(\alpha_i) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j), \tag{3}$$

$$\text{subject to } 0 \leqslant \alpha_i \leqslant C, \quad \text{and} \quad \sum_{i=1}^{l} \alpha_i y_i = 0, \tag{4}$$

where the function $K(\cdot,\cdot)$ is called kernel function and defined as

$$K(x, z) \equiv \Phi^{\mathrm{T}}(x)\Phi(z) \tag{5}$$

It turns out that, in a typical problem, only the coefficients α_i of a few training samples will be non-zero. These samples are called "SVs". Let $s_j,\ a_j^*,\ j = 1, 2, \ldots, m$, $(m < l)$ denote these SVs and their corresponding non-zero coefficients. The decision function in (2) can be rewritten in the "sparse" form of the SVs as

$$f(x) = \sum_{j=1}^{m} \alpha_j^* y_j \Phi^{\mathrm{T}}(s_i)\Phi(x) + b = \sum_{j=1}^{m} \alpha_j^* y_j K(s_j, x) + b. \tag{6}$$

This form shows that the decision functions are determined only by the SVs, which typically represent a very small fraction of the whole training set.

3.2. Improved SVM training with CISS

As described earlier, the impractically large size of the "NAVC" class and the extremely unbalanced size ratio of "NAVC" to "VC" class make our application a very difficult case. In many applications, random selections of training sets with a controllable size will typically be made. However, since only a very small portion of the total set is used, the randomly selected training sets may not be representative.

Osuna et al. proposed a "decomposition" algorithm [9] to make use of all the available training samples. The algorithm exploits the sparseness of SVM solution and iteratively selects the "most representative" examples for the classes. In the context of SVMs, "most representative" refers to samples that are close to

the boundary and are difficult to classify. We modified Osuna's algorithm and proposed a novel algorithm called "CISS" to handle the extremely unbalanced training sample set. The new algorithm includes four steps:

(1) Construct an initial working set S (with a user-selected size l) from the whole training set S_T, such that S consists of all samples from the "VC" class S_V, and a comparable number of samples randomly selected from the very large "NAVC" sample set S_N.
(2) Train a SVM classifier $f(x)$ with S and use $f(x)$ to classify the remainder of the preclassified training set S_T. Put misclassified "NAVC" samples into a set S_M.
(3) From S_M, a "replacement set" of size n is randomly selected and is used to replace an equal number of "NAVC" samples in the working set which were correctly classified in step 2.
(4) Repeat steps (2) and (3) until no further improvement occurs.

Our algorithm differs from that of Osuna et al. in the following ways:

(1) In Step 1, when generating the initial working set, Osuna et al. arbitrarily chose samples from the whole training set. Since only balanced classes were used in their case, the resulting working set was also approximately balanced. In our algorithm, since the training set is extremely unbalanced, arbitrary choices of samples from the whole training set would result in a working set that is likewise extremely unbalanced. To solve this problem, we constrained the initial working set to contain all the samples from the "VC" class and a comparable number of samples randomly selected from the very large "NAVC" class.
(2) In the strategy of Osuna et al. there is no deliberate control of the relative contributions of the two classes to the replacement set. This has been shown to work well for balanced cases. We have tested this approach with highly unbalanced classes and found it to be unusable (data not shown). In our algorithm, we deliberately constrain the relative contributions of the two classes to the replacement set. In the detailed studies described below, we make the contribution of the "VC" class zero and replace only samples from the much larger "NAVC" class.

A geometric interpretation of our algorithm is given in Fig. 4. As shown, CISS leads to a improved decision boundary by adding samples that violate the optimality conditions to the working set.

Using an argument similar to that used in [9], we show that CISS leads to a reduced classification error that converges to a stable value. Essentially, we show that substitution of correctly classified "NAVC" samples in the working set S with misclassified "NAVC" samples from S_N, an improvement of the function in (3) can be achieved.

Let $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_l, y_l)\}$ be the working set used to train the SVM classifier, and let $S_N = \{(x_{l+1}, y_{l+1}), (x_{l+2}, y_{l+2}), \ldots, (x_L, y_L)\}$ be the remainder of the training samples, where $y_i = -1,\ i = l+1, l+2, \ldots, L$, i.e. total training set $S_T = S \cup S_N$.

Let $\Lambda_1 = \{\alpha_1, \alpha_2, \ldots, \alpha_l\}$ be an optimal solution to (3) when training with the working set S. We can extend it to the entire training set as $\Lambda = \{\alpha_1, \alpha_2, \ldots, \alpha_l, \alpha_{l+1}, \alpha_{l+2}, \ldots, \alpha_L\}$ with $\alpha_i = 0,\ i = l+1, l+2, \ldots, L$. Note that although the solution of Λ is optimal over working set S, it may not be optimal over S_T. Since the SVM guarantees to find the optimal solution, what we need to prove here is that the solution Λ is not necessarily optimal when we replace a correctly classified sample in S with one that is misclassified in S_N.

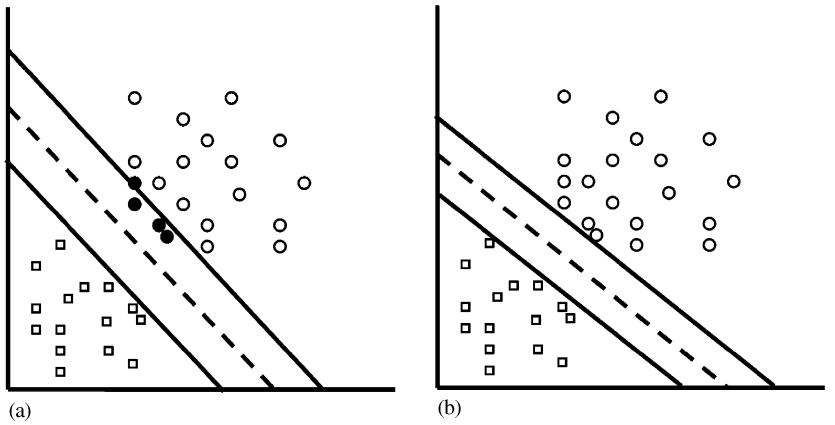

Fig. 4. A geometric interpretation of the way CISS allows the redefinition of the separating surface by adding samples that violate the optimality conditions: (a) a sub-optimal solution; and (b) a redefined decision boundary, where the solution is optimal.

We now replace a correctly classified (randomly chosen) sample $\alpha_i = 0,\ i \in [l, 1]$ (note that for SVMs, points which are correctly classified during the training will have a coefficient of $\alpha = 0$) with $a_m = 0, m \in [l + 1, L]$, $y_m f(x_m) < 1$. After replacement, the new subproblem is optimal if and only if $y_m f(x_m) \geqslant 1$.

Assume that there exists a margin support vector α_p, with corresponding label $y_p = -1$. We have $0 < \alpha_p < C$ since it is a support vector. (We can also pick out an error support vector [26]; the proof is analogous). Then there exists a positive constant δ such that $0 < \alpha_p - \delta < C$. Consider a vector $\Lambda' = \{\alpha'_1, \alpha'_2, \ldots, \alpha'_l, \alpha'_{l+1}, \alpha'_{l+2}, \ldots, \alpha'_L\}$ which has the elements : $\alpha'_m = \delta$, $\alpha'_p = \alpha_p - \delta$, and $\alpha'_i = \alpha_i$ for all other elements. Then we have:

$$L_{\mathrm{D}}(\Lambda') = \sum_{i=1}^{L} \alpha'_i - \frac{1}{2} \sum_{i=1}^{L} \sum_{j=1}^{L} \alpha'_i \alpha'_j y_i y_j K(x_i, x_j). \tag{7}$$

Note that: $\sum_{i=1}^{L} \alpha_i = \sum_{i=1}^{L} \alpha'_i$, $\alpha'_m = \delta = \delta + \alpha_m$ and $\alpha'_p = \alpha_p - \delta$. It can be shown (see appendix) that:

$$L_{\mathrm{D}}(\Lambda') = L_{\mathrm{D}}(\Lambda) - \delta[(y_m f(x_m) - 1] - \frac{\delta^2}{2}[K(x_p, x_p) - 2y_p y_m K(x_p, x_m) + K(x_m, x_m)], \tag{8}$$

if the δ is chosen small enough, the third term of above can be neglected. Then, above become:

$$L_{\mathrm{D}}(\Lambda') = L_{\mathrm{D}}(\Lambda) - \delta[(y_m f(x_m) - 1]. \tag{9}$$

Since $y_m f(x_m) < 1$, it can be easily shown from (9) that $L_{\mathrm{D}}(\Lambda') > L_{\mathrm{D}}(\Lambda)$.

4. Experiments with artificial 2D data

In our study, a series of simulation experiments using artificial data have been done to evaluate systematically the effectiveness of our algorithm. Different artificial 2D data sets were used in these simulation

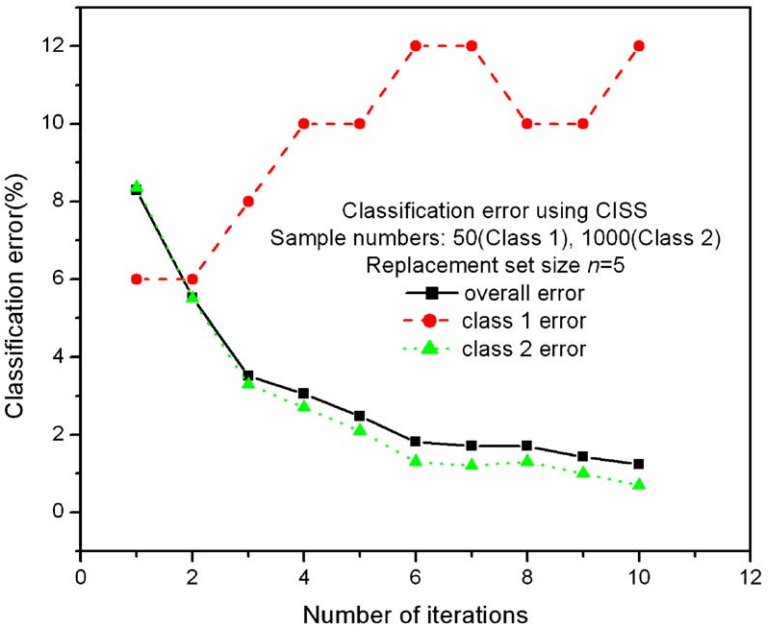

Fig. 5. The convergence of training process when trained with artificial 2D data. Sample numbers in the two classes are 50 for Class 1 and 1000 for Class 2. Replacement set size $n = 5$. A RBF kernel was used. The parameters of the SVM used are: $C = 100$, $\gamma = 0.05$.

experiments. The reason why simulation experiments were used is two fold. First, by using artificial 2D data, the training process and results can be clearly visualized. Second, with artificial data, we can create different scenarios to study the factors that affect classification accuracy.

4.1. Convergence

We first constructed a 2D data set which consists of two different multivariate normal distributions with different means and covariance. Class 1 consists of 50 random points with mean vector $\mu_1 = [-1 \ -1]^{\mathrm{T}}$ and covariance $\sigma_1 = \begin{bmatrix} 1.5 & 0.4 \\ 0.4 & 1.5 \end{bmatrix}$. Class 2 has 1000 random points with mean vector $\mu_2 = [2 \ 2]^{\mathrm{T}}$ and covariance $\sigma_2 = \begin{bmatrix} 3 & 0.1 \\ 0.1 & 4 \end{bmatrix}$. The samples in Class 1 are manually labeled $+1$ and those in Class 2 are labeled -1. An SVM classifier with a RBF kernel function was trained on this data set using CISS. The convergence of the training process is shown in Fig. 5. The parameters used in the training process were: $C = 100$, $\gamma = 0.05$, working set size $l = 150$ and replacement set size $n = 5$. The parameters C and γ were chosen by the "parameter optimization" procedure (see Section 5.2.3 for detailed description). Different values of l and n have been tried and similar results were produced. So, unless further indicated, the same working set size l and replacement set size n were used in other simulation experiments.

As one can see from Fig. 5, all classification errors level out at about the 10th iteration, which clearly indicates the convergence of our algorithm. Although the classification error of Class 1 gradually

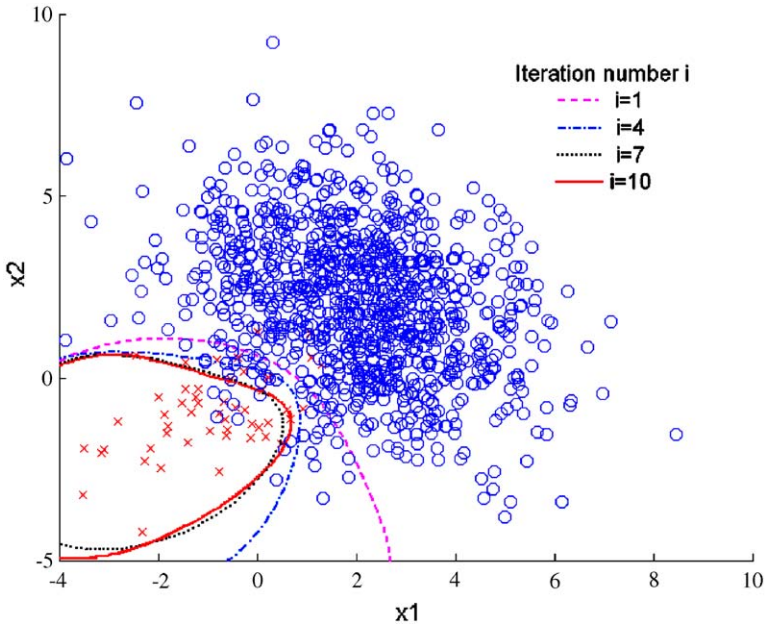

Fig. 6. The evolution of decision boundary as the number of iteration increases. Trained with artificial 2D data. The iteration number $i = 1, 4, 7, 10$, respectively. The sample numbers in the two classes are 50 for Class 1 and 1000 for class 2. Replacement set size $n = 5$. A RBF kernel was used. The parameters of the SVM used are: $C = 100$, $\gamma = 0.05$.

increases, the overall classification error decreases dramatically with the iteration number due to the reduction of classification error in the dominant Class 2. The error rate decreases from 8.29% at the 1st iteration (this is equivalent to standard, non-iterative SVM training procedures) to 1.24% at the 10th iteration.

The evolution of decision boundary in this experiment is also shown in Fig. 6, where the sample points and the decision boundary at iteration numbers: $i = 1, 4, 7$ and 10 are plotted. It is clearly shown that, as the iteration number increases, the decision boundary between the two classes is gradually readjusted to reduce the training error. As a result, the decision boundary gets more and more accurate until it converges to a stable value, i.e. as the iteration number increases, the change of the decision boundary gets smaller and smaller until there is no further significant change.

4.2. Effect of class size ratio on the classification error

To study the effect of class size ratio on the classification error, five data sets, each with 50 samples in Class 1 and different numbers of samples in Class 2, were constructed using the same means and covariances described above. Each data set was used to train an SVM classifier with CISS. The parameters of the SVMs were optimized individually. The classification errors are plotted against iteration number in Fig. 7. The data clearly show that although the low ratio sets give lower classification error after the first iteration, further improvement as iteration number increases is quite limited. On the other hand, even

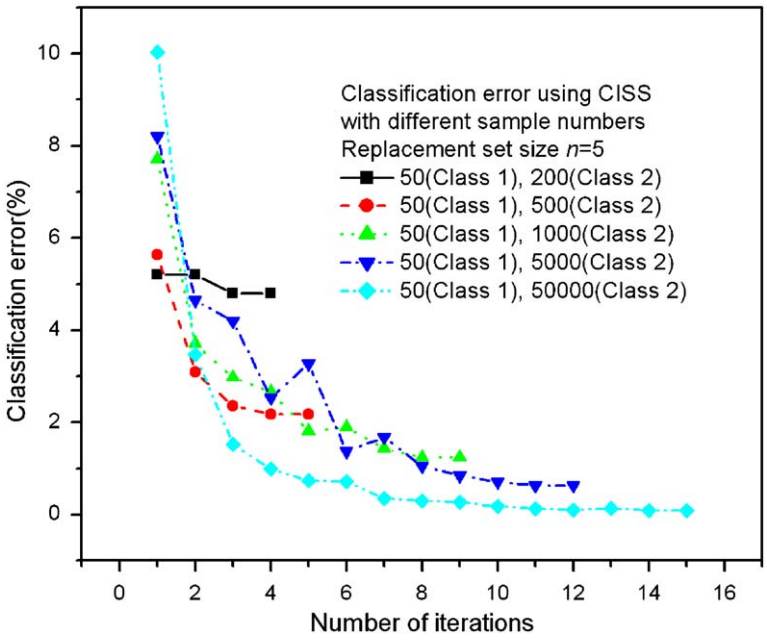

Fig. 7. Classification error using CISS with different synthetic sample sets which have different sample numbers (different sample number ratio between the two classes). Replacement set size $n = 5$. A RBF kernel was used.

though the high ratio sets do not perform very well after the first iteration, they can be greatly improved with multiple iterations. For example, after the 10th iteration, a much lower error rate, even lower than the error rate in the low ratio case, is achieved. This is reasonable since many more samples were used in the high ratio cases.

4.3. Geometric interpretation

The following experiment with 2D data was designed to verify the geometric interpretation of CISS given in Section 3.2 and to reveal the primary reason why CISS can improve classification accuracy. We constructed three different data sets, each with the same covariance and sample numbers, but different mean vectors. The data sets are plotted in Fig. 8(a)–(c), respectively. Sample set 1 represents a simple scenario, since the two classes are well separated. Sample set 3 represents a very difficult scenario, since there is a large overlap between the two classes. When CISS was applied, one can see from Fig. 8(d) that Sample set 3 shows a great improvement, whereas Sample set 1 is only slightly improved. These results show that CISS iteratively chooses the "most representative" training samples from Class 2 while keeping the total number of training sample small and balanced. It should be noted that in the context of SVMs, "most representative" refers to samples that are close to the boundary between the two classes and are difficult to classify. Such a scheme can help to make the decision

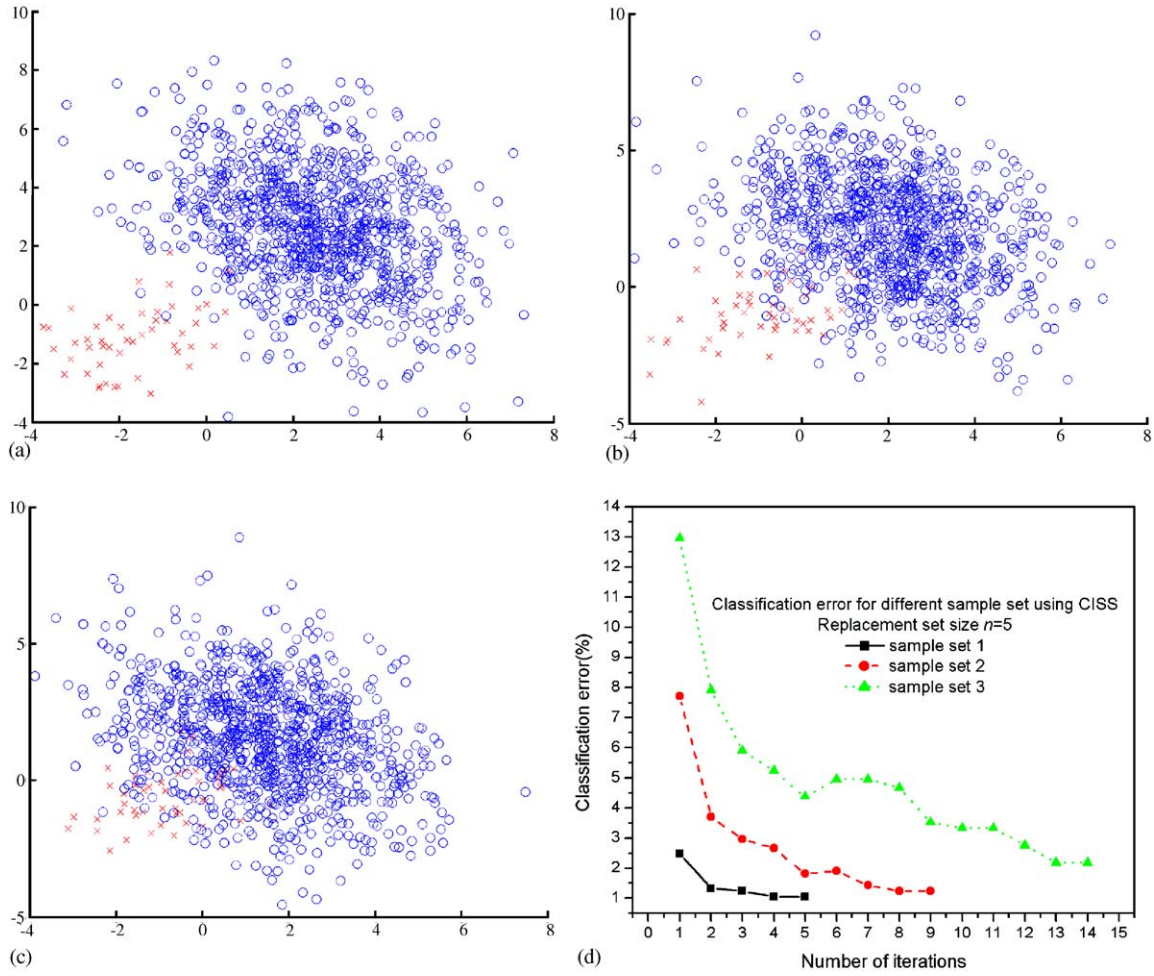

Fig. 8. Classification errors using CISS with different artificial sample sets: (a) sample set 1: with small overlapping; (b) sample set 2: with median overlapping; (c) sample set 3: with large overlapping; and (d) classification error rate changes with number of iterations. Replacement set size $n = 5$. A RBF kernel was used.

boundary more accurate than standard, non-iterative SVM training, especially when applied to difficult scenarios.

5. Automatic detection of unstained viable cells

In this section, a supervised SVM learning framework for automatic viable cell detection is presented and quantitatively evaluated under practical conditions. We use pixel patches as the primary input data elements. Essentially, the software is taught to classify pixel patches into two classes. One class contains desired objects (e.g. viable cells) and is usually much smaller than the other class, which contains all undesired objects (e.g. non-viable cells, fragments of cells, trash).

5.1. Feature extraction

Since individual cells typically occupy only a small percentage of total image area, it is advantageous to decompose an image using pixel patches that are just large enough to contain the largest cells in the image. In actual experiments, a 31×31 pixel patch was extracted at each location in a microscope image, except for those in the 15-pixel margin around the edges. Our experiments indicate that performance is not very sensitive to small variations in patch size, e.g. a patch size of 33×33 produced similar results. Since many locations in the image are uniform background, a "mask" was created to exclude these patches. Essentially, the "mask" eliminated all pixel patches whose average pixel intensities were below a user-chosen threshold.

5.2. SVM training

5.2.1. Construction of preclassified training set

A training set was created with the aid of an interactive program that displays the digitized microscope images and allows a user to select the locations of viable cell centers with a mouse cursor after manual comparison of bright field and fluorescence images. The pixel patches extracted from the selected viable cell locations were preprocessed by principal component analysis (PCA) [2,3,7,27] and used as "VC" class (labeled $y = +1$) sample vectors. The sample vectors in the "NAVC" class (labeled $y = -1$) were then generated automatically by extracting all the pixel patches whose centers were $r \geqslant 5$ pixels away from any of the selected viable cell locations. As for "VC" class vectors, PCA preprocessing was used to reduce dimensionality to 10. The value of r was empirically chosen in relation to the sizes of cells and pixel patches. Finally each attribute of the PCA-preprocessed vectors was linearly scaled to the range $[-1, +1]$. The main advantage of scaling is to avoid computational difficulties and to avoid the dominance of attributes with greater numeric ranges over those with smaller numeric ranges [25].

5.2.2. Selection of kernel function

In order to train a SVM, the following parameters have to be chosen: the kernel function type, the regularization parameter C and γ, the parameter associated with the kernel function. These parameters are often optimized by v-fold crossvalidation [25]. In v-fold crossvalidation, the training set is first divided into v subsets of equal size. Sequentially, one subset is tested using the classifier trained on the remaining $v - 1$ subsets. Thus, each instance of the whole training set is predicted once and the crossvalidation accuracy is the percentage of data which are correctly classified.

The performance of an SVM depends critically on its kernel function. In this study, we considered three commonly used kernel types: linear ($K(x, y) = xy + 1$), polynomial ($K(x, y) = (xy + 1)^{\mathrm{P}}$) and Gaussian radial basis function (RBF, $K(x, y) = \mathrm{e}^{-\gamma\|x-y\|^2}$).

A training set containing 496 "VC" samples was created from 15 randomly picked microscope images. We also randomly selected 1000 samples from the very large "NAVC" set. A five-fold crossvalidation procedure was performed using the 1496 samples for training and testing the SVM classifier under various kernels and parameters. The results are shown in Figs. 9 and 10, the linear kernel yields an accuracy of only about 80%. The polynomial kernel (third order) gives an accuracy of 90.64%. However, neither are as good as the RBF kernel function, which yields an accuracy of 92% when a coarse grid search is used. Therefore, we chose the RBF kernel function for further studies.

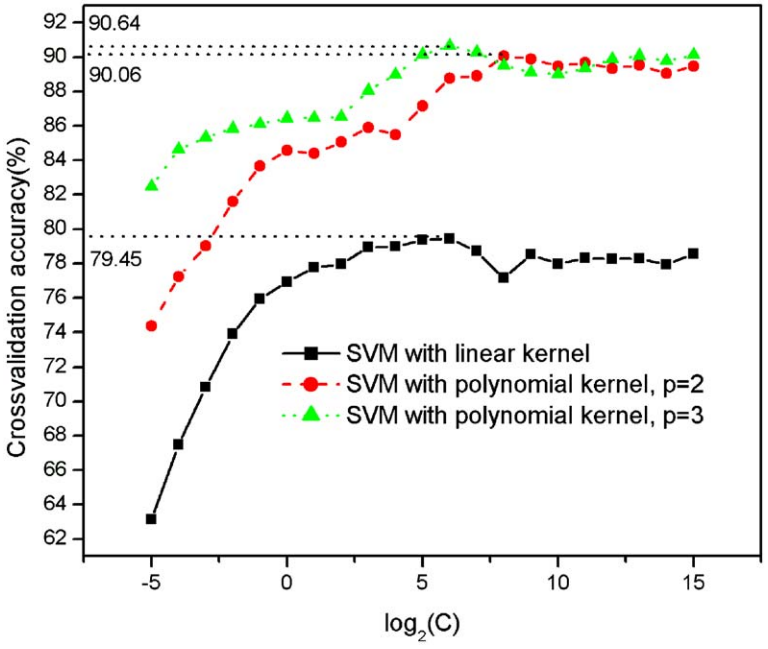

Fig. 9. Kernel function selection: evaluation of the classification accuracy of SVMs with linear and polynomial kernels using five-fold cross validation.

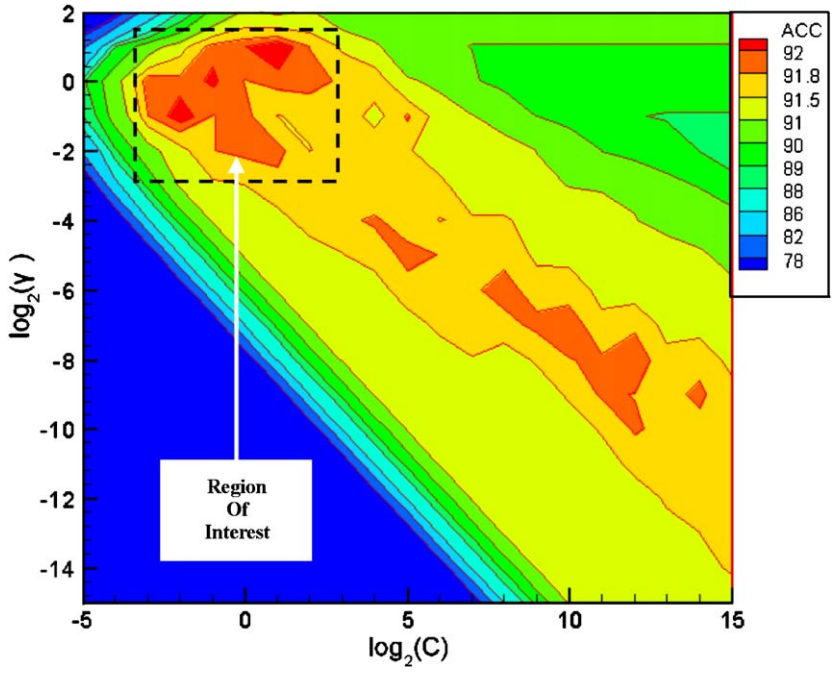

Fig. 10. A coarse grid-search for C and γ with grid size = 1. The five-fold cross validation accuracy (ACC in figure) of an SVM with a RBF kernel is plotted for different combinations of C and γ.

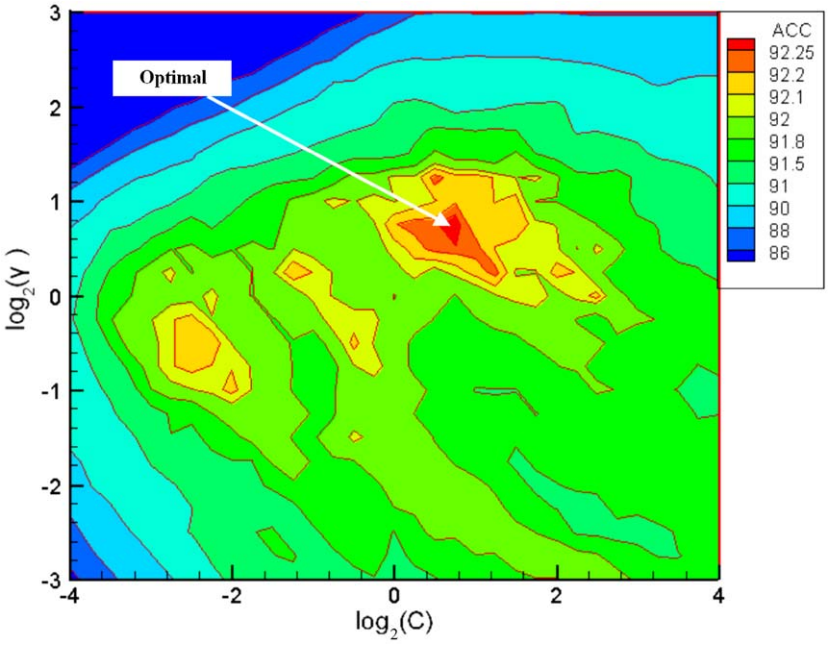

Fig. 11. A fine grid-search with grid size = 0.25 on the region that was identified in the coarse grid-search. The five-fold cross validation accuracy (ACC in figure) of an SVM with a RBF kernel is plotted for different combinations of C and γ.

5.2.3. *Optimization for C and γ*

The goal for parameter optimization is to identify values for C and γ that optimize classification accuracy with samples not in the training set. A two-step "grid-search" method [25] on C and γ was used in this study. In the first step, a coarse grid-search with a grid size of 1 was used to localize a region of interest (ROI) containing the optimal values (shown in Fig. 10). Since the cost function of an SVM is convex, we do not need to worry about the SVM being trapped in local extrema. Therefore, in the second step, a fine grid-search over the ROI with a grid size of 0.25 was used to give more precise values for C and γ. The result is shown in Fig. 11. We picked the optimized parameters $C = 1.6818$, $\gamma = 1.6818$ and used them for further studies.

5.2.4. *Training with CISS*

In the above SVM training procedures, 1496 samples were used: 496 "VC" and 1000 "NAVC". To explore the use of CISS with a much larger training set, we added 200,000 randomly selected "NAVC" samples to the initial training set. It should be noted that this number of samples is much larger than the number that can be handled with a standard PC [9] in conventional non-iterative training procedures. In the CISS training process, a "working set" with size $l = 1496$ was used. In each iteration, a "replacement set" consisting of $n = 25$ misclassified "NAVC" samples was selected and substituted into the "working set". We used two schemes for selecting the "replacement set": the "random" scheme, in which 25 misclassified samples are randomly selected from the total; the "greedy" scheme, in which the 25 most wrongfully classified samples are chosen. The degree of misclassification is measured in relation to the prediction value $f(x)$.

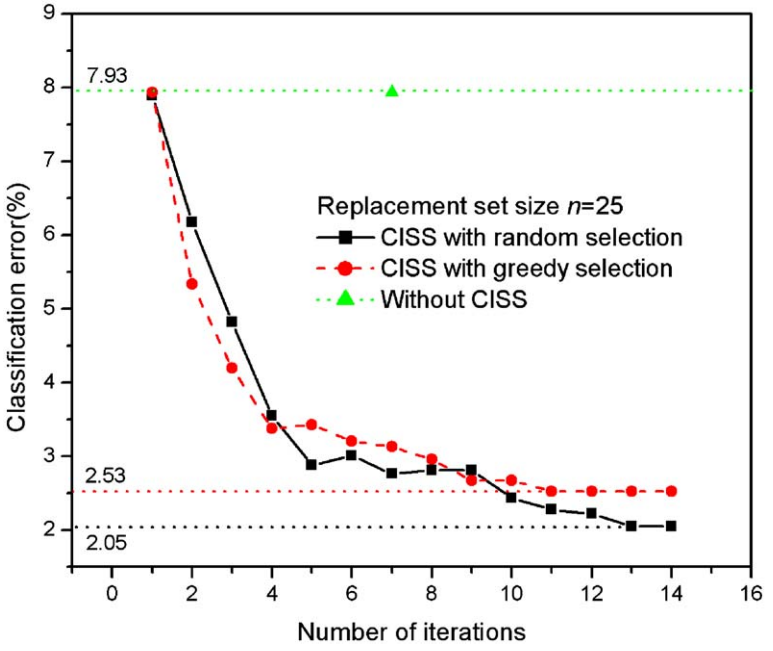

Fig. 12. Plot of classification error rate of the CISS-trained SVM versus the number of iterations. Replacement set size $n = 25$. The parameters of the SVM used are: $C = 1.6818$, $\gamma = 1.6818$.

We also evaluated the effect of replacement set size (data not shown). It was found that if the size is too small, convergence requires a larger number of iterations. If the size is too large, fewer iterations are needed, but oscillations are observed. Generally, a choice from 10 to 50 samples has been found suitable in our cell recognition experiments.

To test the classification accuracy of our trained SVM classifier, a testing set with 500 "VC" and 1500 "NAVC" samples was constructed. None of these samples were in the training set. At each iteration, the trained SVM classifier was applied on the testing set to determine the classification error. In Fig. 12, classification error is plotted against iteration number. Two interesting points are revealed. First, as expected, both the "random" and "greedy" schemes for replacement set selection dramatically reduce the classification error, which converges to a stable value. In addition, the "random" scheme achieves a lower classification error than the "greedy" scheme.

For the first point, we believe it is due to the fact that more and more "representative" samples (i.e., samples that are close to the boundary and hard to classify) were iteratively selected by our algorithm and used for training. To verify this hypothesis, we compared iteration-1 pixel patches for "NAVC" SVs with corresponding iteration-14 patches. Both groups were randomly selected from the "random" scheme and are shown in Fig. 13(a) and (b). To a human observer, it is clear that iteration-14 patches contain objects that more closely resemble viable cells than objects in iteration-1 patches. This suggests that the decision boundary at the 14th iteration is supported by vectors that are more difficult for the SVM to classify. For the second point, a possible explanation could be that the most wrongfully classified samples selected by the "greedy" scheme can be too far from the boundary to generate an optimal decision surface and therefore are not necessarily the best vectors to define the decision boundary.

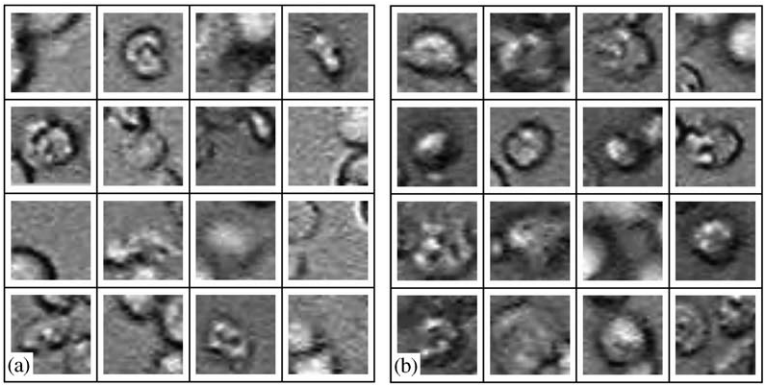

Fig. 13. Typical samples of SVs in the "NAVC" class after applying CISS: (a) SVs after the 1st iteration; and (b) SVs after the 14th iteration. Replacement set size $n = 25$. The parameters of the SVM used are: $C = 1.6818$, $\gamma = 1.6818$. Samples were randomly selected using matlab.

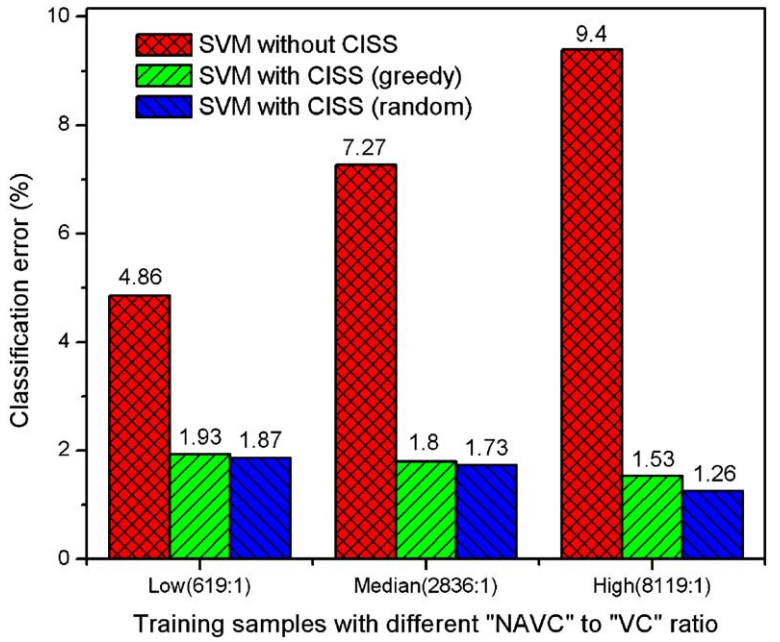

Fig. 14. Effect of different training images on our algorithm. Patches were extracted from three image groups that have different "None living-cell" to "Living cell" ratio: Low (619 to 1), Median (2836 to 1) and High (8119 to 1). Replacement set size $n = 25$. The parameters of the SVM used are: $C = 1.6818$, $\gamma = 1.6818$.

We also tested our algorithm with training images that have different levels of imbalance. Three different image groups were created: low, median and high, each with different "NAVC" to "VC" ratios. In the "Low" group, the images are dominated by viable cells. The "High" group, on the other hand, has much more trash, dead cells and debris. The "Median" group is intermediate. Therefore, after eliminating the

background patches, 112,711 "NAVC" and 182 "VC" patches could be extracted from the "Low" group. These samples were used to construct a training set with the "NAVC" to "VC" ratio of 619:1. For the "Median" group, the "NAVC" to "VC" ratio was 513,274 to 181 (i.e. 2836:1). For the "High" group, the ratio was 1,363,990 to 168 (i.e. 8119:1).

Each of the three training sets was used with CISS to train a SVM, which was then tested on the testing set described above. The results are compared in Fig. 14 with results obtained by SVMs trained with a non-iterative procedure using a training set similar to the initial working set used with CISS. One can see that these results closely parallel those obtained in the simulation experiments with artificial 2D data as shown in Fig. 7. The improvement obtained with CISS on the "Low" group is quite limited, whereas, a substantial and valuable improvement was obtained for the "High" group (classification error decreased from 9.4% to 1.26%).

5.3. Identification and localization of unstained viable cells in bright field images

To generate a more practical tool, we developed a method that combines an automatic "pixel patch" decomposition technique [2,3,7] with CISS-trained SVMs. This method permits not only the identification of the desired objects but also gives their locations relative to the pixel coordinates of the primary image.

The essential aspects of this method have been described in a previous publication [7]. Basically, for each pixel p in the image (excluding pixels in the margin around the edges), a pixel patch (e.g. 31×31) centered at that pixel is extracted and mapped to a "confidence value" $C[p] \in [-1, 1]$ by a CISS-trained SVM. After all the pixels in the primary image are processed, a new image called a "confidence map" is created. Pixels in the confidence map are the confidence values of their corresponding patches in the original image and form "mountains" with large peaks representing a high probability of cell presence. Cell positions can then be easily found by identifying local maxima in these "mountains". To increase speed, patches that clearly indicate the background were not used.

In order to examine the effect of our algorithm on images with different levels of complexity, three testing groups, each consisting 10 images, were created. Scenario 1 is the simplest case where the cell density is low; and there is little cell aggregation. Trash and debris are also at a very low level. Scenario 2 is more complex since many cells are aggregated and there are higher levels of trash and debris. Scenario 3 represents the most complex case where the cell density and aggregation are high; as are trash, debris, and the percentage of dead cells. Typical images from these three scenarios are shown in Fig. 15.

Fig. 16(a) shows the confidence map for Fig. 15(c). The confidence value range of the SVM ($[-1, 1]$) has been linearly scaled to [0, 255] for grayscale representation in the confidence map. In Fig. 16(b), the viable cell positions detected are denoted by "plus" signs in the image.

Statistical results for viable cell detection in Scenarios 1, 2 and 3 are summarized in Figs. 17–19, respectively. We employed a "Free-response Receiver Operating Characteristics" method (FROC) [28] with the average number of false positives (FPs) in each image and the sensitivity (i.e., the percentage of viable cells that are identified by the classifier) as performance indexes. As described above, the cell positions are identified as "peaks" of the "mountains" in the confidence map. This requires a user-defined threshold for the definition of "peak". The FROC curve plots the relationship of false positives and sensitivity as a function of the threshold (not explicitly represented in the plot). In a practical application, a suitable threshold can then be selected to achieve the required behavior. Note that a total of four methods were used in the experiment. All the SVM classifiers followed the procedures described above. A detailed description of the ANN method used here can be found in [7].

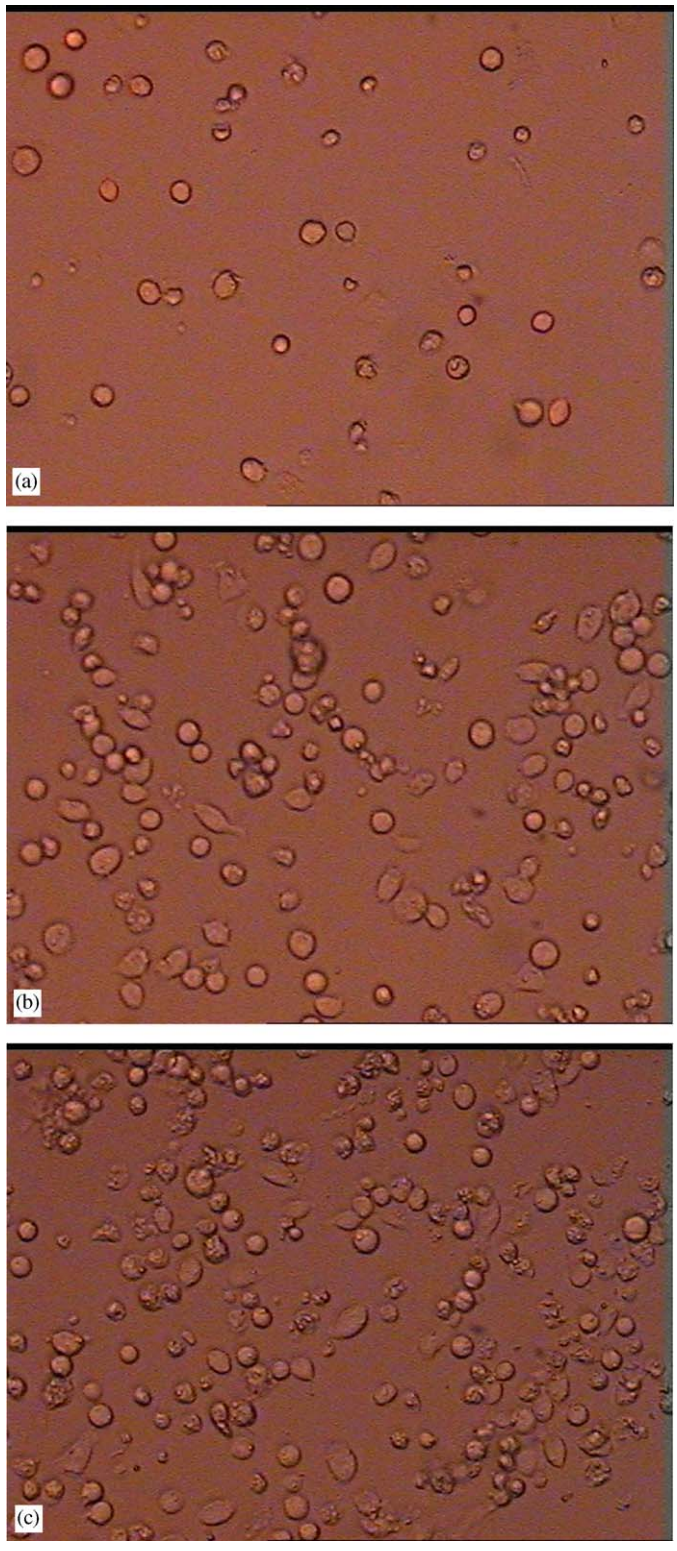

Fig. 15. Typical testing images for viable cell experiments: (a) Scenario 1; (b) Scenario 2; and (c) Scenario 3.

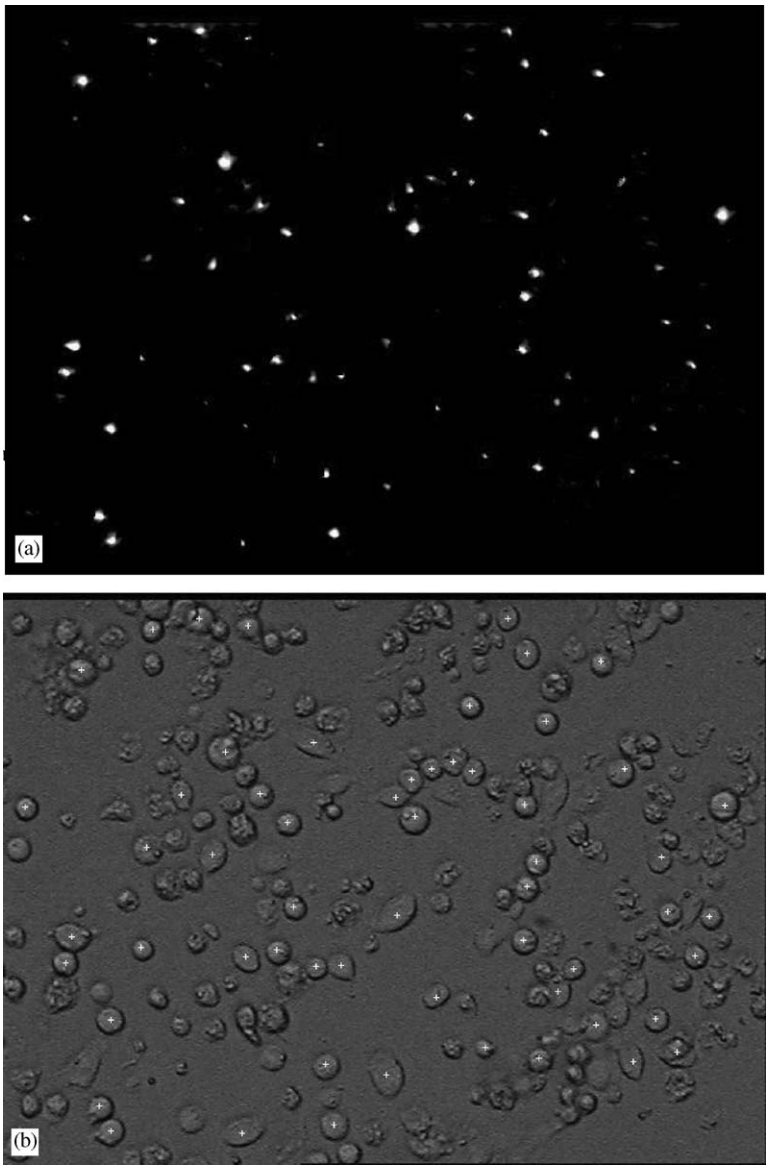

Fig. 16. CISS-trained SVM results for image Fig. 15(c): (a) *Confidence map*: The confidence values are linearly scaled to 0–255 for display and (b) superposition of primary image and calculated cell positions ("plus" signs), which correspond to confidence maxima in (a). Replacement set size $n = 25$. SVM parameters are: $C = 1.6818$, $\gamma = 1.6818$.

The results show that for Scenario 1, a very easy case, all methods produce very good results. For Scenario 2, where the images are more complex, our algorithm starts to show some advantage. A much greater advantage is seen in the very difficult case represented by Scenario 3. For example, if the false positive acceptance number in each image is set at 3, CISS-trained SVM (random) achieves a sensitivity of 94%, which is 1 percentage point greater than that of CISS-trained SVM (greedy), 3 points greater than that of SVM without using CISS and 7 points greater than the ANN approach in [7].

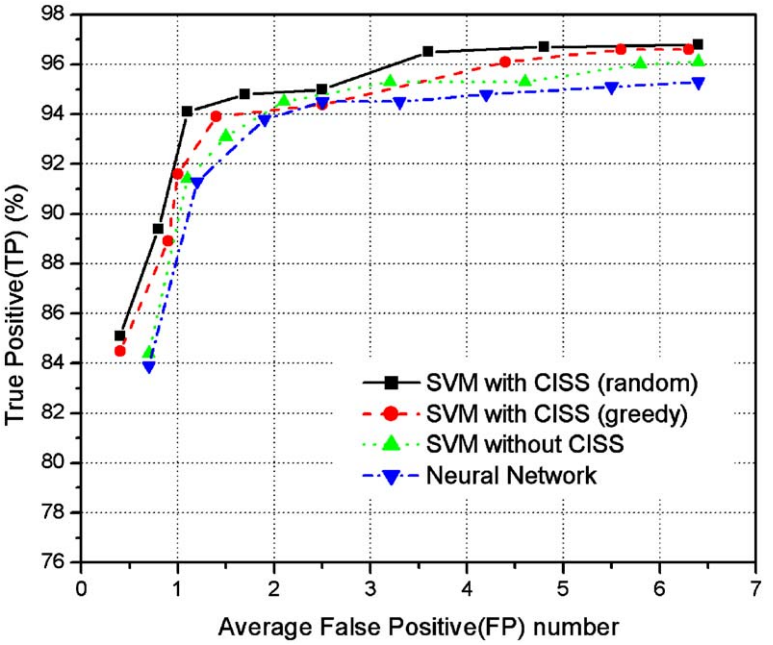

Fig. 17. FROC plots of different candidate methods when applied to Scenario 1: (a) ANN from [7]; (b) SVM trained without CISS; (c) SVM trained with CISS (greedy); and (d) SVM trained with CISS (random), replacement set size $n = 25$. A higher FROC curve indicates better performance. The parameters of the SVM used are: $C = 1.6818$, $\gamma = 1.6818$.

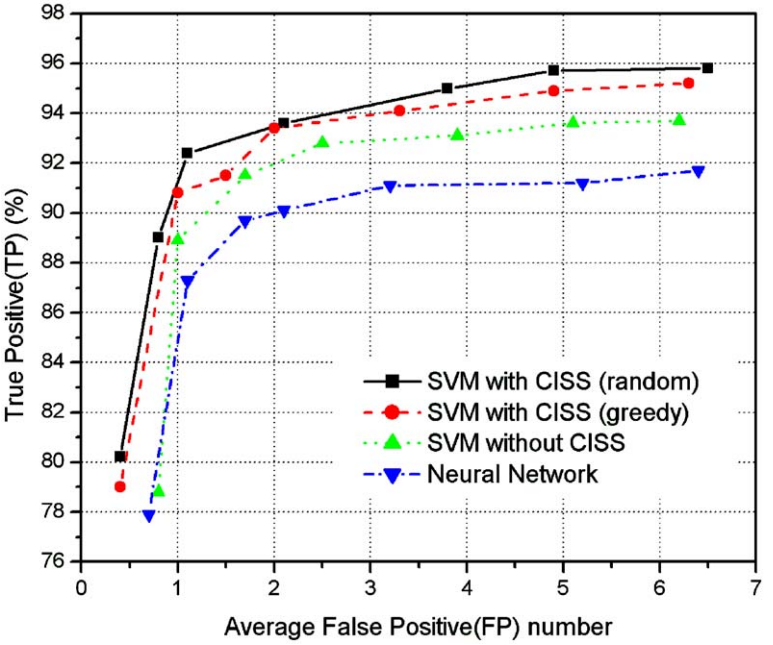

Fig. 18. FROC plots of different candidate methods when applied to Scenario 2: (a) ANN from [7]; (b) SVM trained without CISS; (c) SVM trained with CISS (greedy); and (d) SVM trained with CISS (random), replacement set size $n = 25$. A higher FROC curve indicates better performance. The parameters of the SVM used are: $C = 1.6818$, $\gamma = 1.6818$.

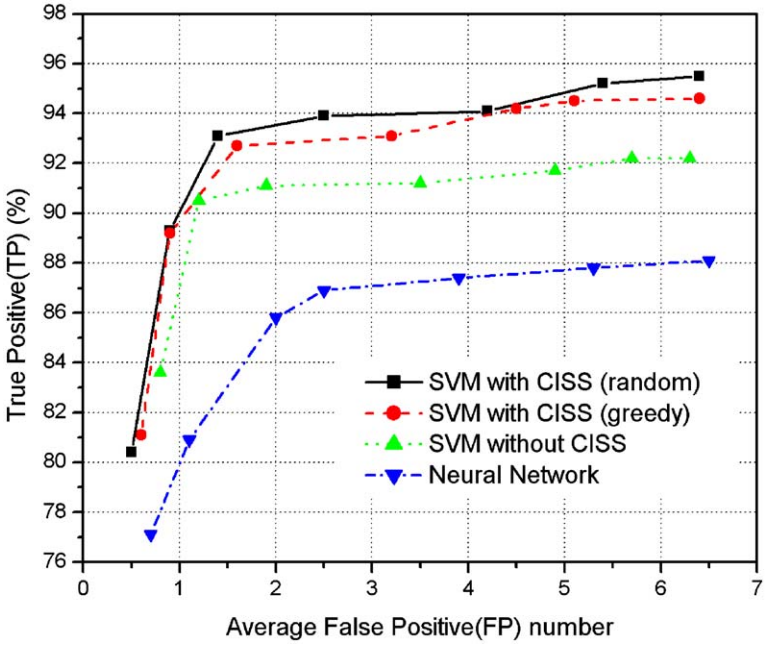

Fig. 19. FROC plots of different candidate methods when applied to Scenario 3: (a) ANN from [7]; (b) SVM trained without CISS; (c) SVM trained with CISS (greedy); and (d) SVM trained with CISS (random), replacement set size $n = 25$. A higher FROC curve indicates better performance. The parameters of the SVM used are: $C = 1.6818$, $\gamma = 1.6818$.

As noted previously, our results with the simulation shown in Fig. 8 suggest that our algorithm iteratively chooses the "most representative" samples for training, and therefore can produce a decision boundary that is more accurate than other methods. The experiments with viable cells described in this section add further support to this claim.

When our current method is used with a 31×31 pixel patch, a 640×480 image requires a processing time of 30–120 s, depending on the number of cells present. This speed in conjunction with the high classification accuracy makes CISS-trained SVM highly useful for many practical applications.

6. Conclusions

An effective algorithm for SVM training on large and highly unbalanced data sets has been proposed and successfully applied to identify and localize unstained viable cells in bright field images. The CISS algorithm has been shown to be superior to SVMs with non-iterative training, especially when the ratios are high. We suggest that CISS is effective because it iteratively chooses the most representative training samples, i.e., the samples that are close to the boundary and are more difficult to classify. This scheme can make the decision boundary more accurate, especially when applied to difficult scenarios. The speed and accuracy of CISS-trained SVMs suggest that it can be very useful in high throughput robotic systems that require automatic cell recognition and localization.

Acknowledgements

This work is supported by NIH Grant CA89841.

Appendix

Note that: $\sum_{i=1}^{L}\alpha_i = \sum_{i=1}^{L}\alpha'_i$, $\alpha'_m = \delta = \delta + \alpha_m$ and $\alpha'_p = \alpha_p - \delta$.

$$
\begin{aligned}
L_{\mathrm{D}}(\Lambda') &= \sum_{i=1}^{L}\alpha_i - \frac{1}{2}\sum_{i=1}^{L}\sum_{j=1}^{L}\alpha_i\alpha_j y_i y_j K(x_i, x_j) - \frac{1}{2}\sum_{i=1}^{L}\sum_{j=1}^{L}(\alpha'_i\alpha'_j - \alpha_i\alpha_j) y_i y_j K(x_i, x_j) \\
&= L_{\mathrm{D}}(\Lambda) - \frac{1}{2}\sum_{i=1}^{L}\sum_{j=1}^{L}(\alpha'_i\alpha'_j - \alpha_i\alpha_j) y_i y_j K(x_i, x_j) \\
&= L_{\mathrm{D}}(\Lambda) - \frac{1}{2}\left[\sum_{i\neq m}\sum_{j\neq m}(\alpha'_i\alpha'_j - \alpha_i\alpha_j) y_i y_j K(x_i, x_j)\right. \\
&\quad \left. +2\sum_{i=m}\sum_{j\neq m}(\alpha'_i\alpha'_j - \alpha_i\alpha_j) y_i y_j K(x_i, x_j) + \sum_{i=m}\sum_{j=m}(\alpha'_i\alpha'_j - \alpha_i\alpha_j) y_i y_j K(x_i, x_j)\right] \\
&= L_{\mathrm{D}}(\Lambda) - \tfrac{1}{2}\{2\delta[(y_m f(x_m) - 1] + \delta^2 K(x_p, x_p) - 2\delta^2 y_p y_m K(x_p, x_m) + \delta^2 K(x_m, x_m)\} \\
&= L_{\mathrm{D}}(\Lambda) - \delta[(y_m f(x_m) - 1] - \frac{\delta^2}{2}[K(x_p, x_p) - 2y_p y_m K(x_p, x_m) + K(x_m, x_m)].
\end{aligned}
$$

References

[1] B.B. Mishell, S.M. Shigi, C. Henry, et al., Preparation of mouse cell suspensions, in: B.B. Mishell, S.M. Shigi (Eds.), Selected Methods in Cellular Immunology, W.H. Freeman and Co., New York, 1980, pp. 16–22.
[2] T.W. Nattkemper, T. Twellmann, H. Ritter, W. Schubert, Human vs. machine: evaluation of fluorescence micrographs, Comput. Biol. Med. 33 (2003) 31–43.
[3] T.W. Nattkemper, H. Ritter, W. Schubert, A neural classifier enabling high-throughput topological analysis of lymphocytes in tissue sections, IEEE Trans Inform. Techol. Biomed. 5 (2) (2001) 138–149.
[4] Shiotani, Shigetoshi, et al., Cell recognition by image processing. (Recognition of dead or living plant cells by neural network), JSME Int. J. Ser. C: Dyn. Control Robot. Des. Manuf. 37 (1) (1994) 202–208.
[5] C.G. Loukas, G.D. Wilson, B. Vojnovic, A. Linney, An image analysis-based approach for automated counting of cancer cell nuclei in tissue sections, Cytometry Part A 55A (2003) 30–42.
[6] P.J. Sjöström, B.R. Frydel, L.U. Wahlberg, Artificial neural network-aided image analysis system for cell counting, Cytometry 36 (1999) 18–26.
[7] Xi Long, W.L. Cleveland, Y.L. Yao, A new preprocessing approach for cell recognition, IEEE Trans. Inform. Technol. Biomed., accepted for publication.
[8] V. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.
[9] E. Osuna, R. Freund, F. Girosi, Support vector machines: training and applications, A.I. Memo 1602, MIT A.I. Lab., 1997.
[10] P. Papageorgiou, T. Poggio, A trainable object detection system: car detection in static images, A.I. Memo 1673, MIT A.I. Lab., 1999.

[11] K. Veropoulos, N. Cristianini, C. Campbell, The application of support vector machines to medical decision support: a case study, ACAI 99, Workshop on Support Vector Machines Theory and Applications, Crete, Greece, July 14, 1999.
[12] M.P.S. Brown, W.N. Grundy, et al., Knowledge-based analysis of microarray gene expression data by using support vector machines, PNAS 97 (1) (2000) 262–267.
[13] T. Friess, N. Cristianini, C. Cambell, The kernel-Adaton: a fast and simple training procedure for support vector machines, in: J. Shavlik (Ed.), Machine Learning: Proceedings of the 15th International Conference, Morgan Kaufmann, Los Altos, CA, 1998.
[14] C. Campbell, N. Cristianini, Simple training algorithms for support vector machines, Technical Report CIG-TR-KA, University of Bristol, Engineering Mathematics, Computational Intelligence Group, 1999.
[15] B.E. Boser, I.M. Guyon, V. Vapnik, A training algorithm for optimal margin classifiers, in: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, Pittsburgh, ACM, New York, 1992.
[16] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines (and other Kernel-based Learning Methods), Cambridge University Press, Cambridge, 2000.
[17] J. Platt, Fast training of support vector machines using sequential minimal optimization, in: B. Schölkopf, C. Burges, A. Smola (Eds.), Advances in Kernel Methods—Support Vector Learning, MIT Press, Cambridge, MA, 1998.
[18] J. Platt, Sequential minimal optimization: a fast algorithm for training support vector machines, Microsoft Research Technical Report MSR-TR-98-14, 1998.
[19] T. Joachims, in: Making large-scale SVM learning practical, B. Schölkopf, C. Burges, A. Smola (Eds.), Advances in Kernel Methods—Support Vector Learning, MIT Press, Cambridge, MA, 1999 (Chapter. 11).
[20] T. Joachims, Learning to Classify Text Using Support Vector Machines, Dissertation, Kluwer Academic Publishers, Dordrecht, 2002.
[21] Peng Xu, A.K. Chan, Support vector machines for multi-class signal classification with unbalanced samples, IJCNN 2003, 2003, pp. 1116–1119.
[22] K. Morik, P. Brockhausen, T. Joachims, Combining statistical learning with a knowledge-based approach, a case study in intensive care monitoring, in: Proceedings of the 16th International Conference on Machine Learning, Bled, Slovenia, 1999.
[23] J. Drish, Obtaining calibrated probability estimates from support vector machines, Technique Report, Department of Computer Science and Engineering, University of California, San Diego, CA, 2001.
[24] W.L. Cleveland, I. Wood, B.F. Erlanger, Routine large-scale production of monoclonal antibodies in a protein-free culture medium, J. Immunol. Methods 56 (1983) 221–234.
[25] http://www.csie.ntu.edu.tw/∼cjlin/libsvm/.
[26] C. Burges, A tutorial on support vector machines for pattern recognition, Data Min. Knowledge Discovery 2 (1998) 122–167.
[27] M. Turk, A.P. Pentland, Eigenfaces for recognition, J. Cognitive Neurosci. 3 (1) (1991) 71–96.
[28] D.P. Chakraborty, Maximum likelihood analysis of free-response receiver operating characteristic (FROC) data, Med. Phys. 16 (1989) 561–568.

Xi Long is pursuing his Ph.D. degree in the Department of Mechanical Engineering at Columbia University, USA. He got his M.S. and B.S. degree in Mechanical Engineering in 2001 and 1998, respectively, both from Tsinghua University, PR China. His research interests are in medical image processing, artificial intelligence, and pattern recognition.

W. Louis Cleveland is currently Director of the Laboratory for Molecular Mechanisms in Human Diseases at St. Luke's-Roosevelt Hospital Center and is a Research Scientist in Medicine at Columbia University. He received a Batchelor of Science degree in Physics from Columbia University and a Ph.D. in Chemistry from Rutgers University. His postdoctoral training in basic immunology was in the Department of Microbiology, Columbia University. Dr. Cleveland's current research interests include molecular mechanisms in cancer and schizophrenia and the development of technology to study these mechanisms.

Y. Lawrence Yao is a Professor in the Department of Mechanical Engineering at Columbia University. He received his Ph.D. from the University of Wisconsin-Madison in 1988. He is interested in multidisciplinary research in manufacturing and design, non-traditional manufacturing processes, laser materials processing, industrial manipulators, and monitoring and fault diagnosis of manufacturing processes and machinery.