Weighted Proximal Support Vector Machine for Dynamic Classification of Polyps in Virtual Colonoscopy

<table>
<thead>
<tr>
<th>Journal:</th>
<th>Transactions on Medical Imaging</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manuscript ID:</td>
<td>TMI-2006-0695</td>
</tr>
<tr>
<td>Manuscript Type:</td>
<td>Full Paper</td>
</tr>
<tr>
<td>Date Submitted by the Author:</td>
<td>06-Oct-2006</td>
</tr>
<tr>
<td>Complete List of Authors:</td>
<td>awad, mariette; IBM, Systems and Technology; UVM, EE Motai, Yuichi; UVM, EE Näppi, Janne; Massachusetts General Hospital, Radiology; Harvard Medical School, Radiology Yoshida, Hiroyuki; Harvard Medical School, Radiology</td>
</tr>
<tr>
<td>Keywords:</td>
<td>Computed Tomographic colonography, Weighted support vector machine, online learning, dynamic classification</td>
</tr>
</tbody>
</table>
Weighted Proximal Support Vector Machine for Dynamic Classification of Polyps in Virtual Colonoscopy

Mariette Awad, Yuichi Motai, Member, IEEE, Janne Nappi, and Hiroyuki Yoshida, Member, IEEE.

Abstract—This paper presents a novel dynamic learning method for classification of polyps in Computer Tomographic Colonography (CTC) using an adaptation of the Least Square Support Vector Machine (LS-SVM). The proposed technique, called Weighted Proximal Support Vector Machines (WP-SVM), extends the offline SVM capabilities to address practical CTC applications. Incremental data are incorporated in WP-SVM as a weighted vector space and the only storage required is for the hyper plane parameters. Evaluation of the performance based on clinical CTC cases showed that, when knowledge is applied in a chunk format, WP-SVM is, on average, 1.42% less accurate but 2.08% times faster than the retrain model with a feature space less than 200. When knowledge is applied sequentially, WP-SVM accuracy improves to reach 1.14% of the model retrain accuracy and be 1.5 times faster.

Index Terms—Computed tomographic colonography, weighted support vector machine, online learning, and dynamic multiclassification.

I. INTRODUCTION

Due to the advancement in Computed Tomography (CT) technology, and with more than 57,000 colon cancer deaths per year in the United States, CT Colonography (CTC) is becoming a promising tool for early diagnosis of colon cancer. CTC, also known as virtual colonoscopy, is a non-invasive technique that detects colorectal polyps and masses based on the CT scans of the distended colon [1].

One of the major obstacles for CTC to be an effective means for detection of polyps is that radiologists’ expertise is required for interpreting the CTC images, in particular, for detection of small polyps. Computer-aided detection (CAD) of polyps is attractive because it has the potential to overcome this difficulty, that is, it has the potential to improve radiologists’ detection performance and to reduce variability of detection accuracy among radiologists [1]. Such a CAD system typically employs a shape-based method for the initial detection of polyp candidates, followed by a machine learning (ML) method for classification of polyps from non-polyps (normal colonic structures) for the generation of the final list of polyps that are provided to the radiologist as a “second opinion” [2]. Typically, the input to a CAD system is a large number of CT images, ranging from 300 to 3000 images per patient. The large amount of data is one of the major obstacles for the ML method in any CAD system to be trained. Moreover, for updating the CAD system, the ML method needs to be retrained as the new CTC data of patients are available. Therefore, the need to scale up inductive learning algorithms in CAD systems is drastically increasing in order to extract valid and novel patterns from incremental data without a major ML retrain.

Dynamic, incremental or online learning refers, in this context, to the situation where the training dataset is not fully available at the beginning of the learning process. The data can arrive at different time intervals and need to be incorporated into the training data to preserve the class concept. Thus, constructing a ML capable of incremental classification as opposed to batch-mode learning is very attractive and will become a strategic necessity for CAD for CTC because of few reasons. First, the training period is the most significant time consuming and resource intensive element in ML. Second, the CTC data are a continuous and large data stream by nature.

Within the framework of dynamic learning, we present a novel method, called Weighted Proximal Support Vector Machine (WP-SVM), which extends traditional SVM beyond its existing static learning methodologies to handle dynamic and multiple classifications of polyps. The selection of SVM as a multi-classification technique is due to several of its main advantages: SVM has its roots in statistical learning theory which insures strong learning and generalization capabilities. It is computationally efficient in learning a hyper plane that correctly classifies the high dimensional feature space, and it is highly resistant to noisy data [3].

This paper is organized as follows: Section II presents an overview of Least Square SVM (LS-SVM) principles. Section III and IV covers our proposed multiclassification SVM and WP-SVM technique. Section V validates the effectiveness of the proposed approach.
of WP-SVM in terms of performance, storage and computational requirements. Finally, Section VI contains concluding remarks and outlines our plans for follow-on work.

II. LS-SVM

Originally designed for binary classification, the SVM techniques were invented by Boser, Guyon and Vapnik, and were introduced during the Computational Learning Theory (COLT) conference of 1992 [4]. Traditionally, SVM was considered for unsupervised off-line batch computation, binary classifications, regressions and structural risk minimization (SRM) [4]. Adaptations of SVM were applied to density estimation (Vapnik and Mukherjee in [5]), Bayes point estimation (Herbrich et al. in [6]) and transduction [4] problems. Researchers also extended the SVM concepts to address error margin (Platt in [7]), efficiency (Suykens and Vandewalle in [8]), multi-classification [9,10] and incremental learning (Shilton et al. [11], Diehl and Cauwenberghs [12], Cauwenberghs and Poggio [13], Ralaivola and d’Alche’-Buc [14]).

From a structural or representational capacity, SVM behaves like a neural network form, but it differs in the learning technique and always insures a global solution [15]. SVM solutions are obtained from solving quadratic programming (QP) problems. LS-SVM instead uses linear system instead of QP to solve the problem of minimizing the confidence interval, and keeping the training error fixed while maximizing the distance between the calculated hyper plane and the nearest data points known as support vectors [8].

Given a binary classification task, let a training set with attributes or features sets \( \{ f_1, f_2, \ldots, f_n \} \) be defined as \( \{ x_i, y_i \}_{i=1}^{N} \), where \( x_i \in R^n \) represents the \( n^{th} \) input image and \( y_i \) the output class. SVM aims at finding the optimal plane for the classifier by minimizing Eq. (1) subject to the constraint in Eq.(2).

Objective function

\[
\frac{1}{2} w^T w + \frac{1}{2} \sum_{i=1}^{N} (e_i^2)
\]

(1)

Constraint:

\( y_i (w^T x_i + b) \geq 1 \)

(2)

Any new data point is then classified by the decision function in Eq. (3).

Decision function:

\( f(x) = \text{sign}(w^T x + b) \)

(3)

LS-SVM classifiers as introduced in [8] are obtained by optimizing the Lagrangian as defined in Eq. (4):

\[
L_w(w,b,e,\alpha) = \frac{1}{2} w^T w + \frac{1}{2} \sum_{i=1}^{N} (e_i^2) - \sum_{i=1}^{N} \alpha_i (y_i (w^T x_i + b) - 1 + e_i)
\]

(4)

with \( \alpha_i \) being the Lagrange multipliers which can be either positive or negative. These parameters are derived from Karuch Kuhn-Tucker conditions which are valid as long as the objective function and conditions are convex [15]. The gradient of the inequality restrain solutions to the interior of the acceptable region and the optimal solution is the saddle point that satisfies Eq. (5)

\[
\frac{\partial L}{\partial w} = 0 \rightarrow w = \sum_{i=1}^{N} \alpha_i y_i x_i
\]

\[
\frac{\partial L}{\partial b} = 0 \rightarrow \sum_{i=1}^{N} \alpha_i y_i = 0
\]

\[
\frac{\partial L}{\partial \alpha_i} = 0 \rightarrow \alpha_i = \lambda e_i
\]

\[
\frac{\partial L}{\partial e_i} = 0 \rightarrow y_i (w^T x_i + b) - 1 + e_i = 0
\]

Eliminating \( w \) and \( e \), the system of linear equations in Eq.(5) can be written more concisely in a matrix form as:

\[
\begin{bmatrix}
0 \\
Y^T \\
YZ^T + \lambda^{-1} I
\end{bmatrix}
\begin{bmatrix}
b \\
\alpha
\end{bmatrix}
= \begin{bmatrix}
0 \\
I
\end{bmatrix}
\]

(6)

where

\[ Z = [x_1^T y_1; \ldots; x_N^T y_N] \]

\[ Y = [y_1; \ldots; y_N] \]

\[ I = [1; \ldots; 1] \]

\[ e = [e_1; \ldots; e_N] \]

\[ \alpha = [\alpha_1; \ldots; \alpha_N] \]

with Mercer conditions applied to \( ZZ^T \) [15].

III. PROPOSED MULTICLASSIFICATION SVM

We extend the standard SVM to use it for multi-classification tasks. Thus, the objective function as shown in Eq. (1) becomes:

\[
\frac{1}{2} \sum_{m=1}^{M} \sum_{n=1}^{M} (w_m^T w_n + b_m b_n) + \frac{\lambda}{2} \sum_{i=1}^{N} \sum_{j=n}^{m} (e_{im}^m)^2
\]

(7)

We added the plane intercept term \( b \) to the objective function to uniquely define the hyper plane by its slope \( w \) as well as its intercept \( b \). The penalty parameter \( \lambda \) represents a cost to the error term \( e \) and it enables its regulation for behavior classification during the training phase. Hsu and Lin [16] showed that SVM accuracy rates were influenced by the selection of \( \lambda \) which varies in ranges depending on the problem under investigation. The selection of \( \lambda \) can be found heuristically or by a grid search – as we show in Section V. Typically large \( \lambda \) values tend to favor less smooth solutions that drive large \( w \) values.

As proposed in the proximal binary classification [17,18], we carry the optimization step with an equality constraint dropping the Lagrange multipliers. The problem solution now becomes equal to the rate of change in the value of the objective function. In this approach, we do not solve the equation for the support vectors that correspond to the nonzero Lagrange multipliers. Instead the classification of data points will be performed by assigning them to the closest parallel planes.

Selecting the multi-classification objective function, we use the constraint function shown in Eq. (8)

\[
(w^T x_i) + b = (w^T x_j) + b + 2 - e_i^m
\]

(8)
Since it is a multiclassification problem, a data point is assigned to a specific class after being tested against all existing classes. The final class label is the one that has the majority vote (or the largest value) for Eq. (9)

\[ f(x) = \arg \max((w_i^T x) + b_m), m = 1...c \]

(9)

Substituting Eq. (8) into Eq. (7), we obtain:

\[ L(w, b) = \frac{1}{2} \sum_{i=1}^{c} (w_i^T w_i + b_m b_m) + \frac{\lambda}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} (w_i^T - w_j^T) x_i (b_j - b_m) - 2 \]

Taking partial derivatives of \( L(w, b) \) with respect to both \( w \) and \( b \) gives

\[ \frac{\partial L(w, b)}{\partial w} = 0, \quad \frac{\partial L(w, b)}{\partial b} = 0. \]

(10)

Defining \( a_i = \begin{cases} 1 & y_i = n, \\ 0 & y_i \neq n, \end{cases} \)

Eq. (10) becomes:

\[
\begin{align*}
\frac{w_i}{\lambda} + \sum_{j=1}^{N} [(-x_i^T x_j^T) (w_i^T - w_j^T) x_i (b_j - b_m) - 2x_i (1 - a_j)] + \\
\sum_{i=1}^{N} (x_i^T x_j^T) (w_i^T - w_j^T) + x_i (b_j - b_m) + 2x_i a_j] &= 0
\end{align*}
\]

(11)

\[
\begin{align*}
\frac{b_i}{\lambda} + \sum_{j=1}^{N} [(-x_i^T x_j^T) (w_i^T - w_j^T) (b_j - b_m) + 2(1 - a_j)] + \\
\sum_{i=1}^{N} (x_i^T x_j^T) (w_i^T - w_j^T) + (b_i - b_m) + 2a_j] &= 0
\end{align*}
\]

Let us define:

\[
S_w := \sum_{i=1}^{N} [(-x_i^T x_j^T) (w_i^T - w_j^T) x_i (1 - a_j)] + \sum_{i=1}^{N} (w_i^T - w_j^T) x_i^2 a_j]
\]

\[ S_w = -\sum_{i=1}^{N} (w_i^T - w_j^T) x_i^2 + \sum_{i=1}^{N} a_i \sum_{j=1}^{N} (w_i^T - w_j^T)
\]

A similar argument shows that:

\[
S_b := \sum_{i=1}^{N} [(-x_i^T x_j^T) (b_j - b_m) x_i (1 - a_j)] + \sum_{i=1}^{N} (b_i - b_m) x_i a_j]
\]

\[ S_b = -\sum_{i=1}^{N} (b_i - b_m) x_i + \sum_{i=1}^{N} x_i \sum_{j=1}^{N} (b_j - b_m)
\]

and \( S_a := \sum_{i=1}^{N} [2x_i (1 - a_j)] + \sum_{i=1}^{N} 2x_i a_j]\n
\[ S_a = \sum_{i=1}^{N} 2x_i - \sum_{i=1}^{N} \sum_{j=1}^{N} 2x_i = \sum_{i=1}^{N} x_i - c \sum_{i=1}^{N} x_i
\]

Applying similar reasoning for \( b \), we can re-arrange Eq. (11) to obtain:

\[
\left[ \frac{L}{\lambda} + \sum_{i=1}^{N} x_i^T x_j^T + \sum_{i=1}^{N} \sum_{j=1}^{N} x_i x_j \right] w_i + b_i \sum_{j=1}^{N} x_j + c \sum_{j=1}^{N} w_j + 2 \sum_{j=1}^{N} a_i = 0
\]

(12)

To rewrite Eq. (12) in a matrix form, we use the series of definitions as mentioned in Table I.

<table>
<thead>
<tr>
<th>TABLE I</th>
<th>MATRIX NOTATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matrix</td>
<td>Symbol</td>
</tr>
<tr>
<td>( C )</td>
<td>Diagonal matrix of size ((f \times c)) by ((f \times c)), the diagonal elements are composed of the square matrix ( c_n ) which is of size ( f ):</td>
</tr>
<tr>
<td>( D )</td>
<td>Diagonal matrix of size ((f \times c)) by ( c ), the diagonal elements are the column vector ( d_n ) of length ( f ):</td>
</tr>
<tr>
<td>( E )</td>
<td>Column vector of size ( c ) made from</td>
</tr>
<tr>
<td>( H )</td>
<td>Matrix of size ((f \times c)) by ( c ). The row vector is ( h_n ) of length ( c ) and of the form:</td>
</tr>
<tr>
<td>( G )</td>
<td>Square matrix of size ((f \times c)) by ((f \times c)), composed of matrix ( g_n ) of size ( f ) by ( c ) such that</td>
</tr>
<tr>
<td>( Q )</td>
<td>Square matrix of size ( c ), made from the row vector ( q_n ) of length ( c )</td>
</tr>
<tr>
<td>( U )</td>
<td>Column vector of size ( c ), made from ( u_n )</td>
</tr>
<tr>
<td>( R )</td>
<td>Square diagonal matrix of size ( c ), the diagonal elements ( r_n ) are as follows</td>
</tr>
</tbody>
</table>

\( f \) denotes the dimensions of feature space and \( q(n) \) the size of class \( n \)
The above definitions allow us to manipulate Eq. (12) and rewrite it as

\[
\begin{align*}
(C - G)W + (D - H)B &= E \\
(D - H)^T W + (R - Q)B &= U
\end{align*}
\]

Solving these equations for \(W\) and \(B\), we obtain

\[
\begin{align*}
W &= \left[ (C - G) (D - H) \right]^{-1} E \\
B &= \left[ (D - H)^T (R - Q) \right]^{-1} U
\end{align*}
\]

We define matrix \(A\) to be:

\[
A = \left[ (C - G) (D - H) \right]
\]

and \(L\) to be:

\[
L = \begin{bmatrix} E \\ U \end{bmatrix}
\]

These definitions allow us to rewrite Eq. (12) in a very compact form:

\[
\begin{bmatrix} W \\ B \end{bmatrix} = A^+ L
\] (16)

Eq. (16) provides the separating hyperplane slopes and intercepts values for the different \(c\) classes. The hyperplane is uniquely defined based on matrix \(A\) and \(L\), and it does not depend on the support vectors or the Lagrange multipliers.

IV. PROPOSED WP-SVM

Once the hyperplane slopes are defined incorporation of recently acquired data into a traditional SVM or LS-SVM model necessitates a full scale retraining for the system in order to calculate the new parameters.

\[
\begin{align*}
W_n &= \left[ (C_{new} - G_{new}) (D_{new} - H_{new}) \right]^{-1} E_{new} \\
B_n &= \left[ (D_{new} - H_{new})^T (R_{new} - Q_{new}) \right]^{-1} U_{new}
\end{align*}
\]

For large datasets, retraining is not efficient. It is expensive in terms of memory requirement and computation time. To maintain an acceptable balance between storage, accuracy and computational time, we propose a WP-SVM, a dynamic Weighted Proximal SVM. Whenever the model needs to be updated, each incremental sequence is expected to alter matrices \(C, G, D, H, E, R, Q\) and \(U\) in Eqs. (13) and (14) by an amount of \(\Delta C, \Delta G, \Delta D, \Delta H, \Delta E, \Delta R, \Delta Q\) and \(\Delta U\) respectively. For illustrative purposes, let us consider a recently acquired patient CT data \(x_{N+1}\) belonging to class \(t\). Eq. (17) then becomes:

\[
\begin{align*}
W_n &= \left[ (C_{new} - G_{new}) - (G + \Delta G) (D_{new} - H_{new}) - (D + \Delta D) (H + \Delta H) \right]^{-1} E + \Delta E \\
B_n &= \left[ (D_{new} - H_{new})^T - (H + \Delta H) (R + \Delta R) - (R + \Delta R) (Q + \Delta Q) \right]^{-1} U + \Delta U
\end{align*}
\]

If we were to linguistically describe the impact of the newly acquired sequences on the hyperplane orientations \(W\) and \(B\), we would distinguish 3 main types of impact: severe, limited and negligible. The category class is determined by the weight elements (\(\Psi\)) that are calculated as described as follows.

\[
\Psi_{fc} = \left( \frac{v \log \frac{N}{v}}{\sqrt{2 \sigma_{fc}^2}} \right)
\]

where \(v\) is the frequency of the incremental data sequence acquired, \(N\) the total number of sequence data used to determine the model hyperplane parameters, \(N_{fc}^2\) is the Euclidian distance between the incremental data feature \(f\) and the hyperplane parameters for class \(c\).

A. Dynamic Processing for Sequential Data

Sequential data refers to incremental data being acquired and processed serially as they are acquired. To assist in the mathematical manipulation, we define the following matrices:

\[
\begin{align*}
I &= \begin{bmatrix} 0 & 0 & 0 & \ldots & 0 \end{bmatrix} \\
I_c &= \begin{bmatrix} 0 & 0 & 0 & \ldots & 1 \end{bmatrix}
\end{align*}
\]

We can then rewrite the incremental change as follows:

\[
\begin{align*}
\Delta C &= \Psi_{x_{N+1}x_{N+1}^T}(I_c - I) \\
\Delta G &= \Psi_{x_{N+1}x_{N+1}^T}(I_c - I) \\
\Delta D &= \Psi_{x_{N+1}x_{N+1}^T} + \Delta H = \Psi_{x_{N+1}x_{N+1}^T} \\
\Delta E &= -2\Psi_{x_{N+1}x_{N+1}^T} \\
\Delta Q &= I_c \\
\Delta U &= -2I_c
\end{align*}
\]

The dynamic model parameters now become:

\[
\begin{align*}
\begin{bmatrix} W_n \\ B_n \end{bmatrix} &= A^+ \begin{bmatrix} \Psi_{x_{N+1}x_{N+1}^T}(I_c - I) \\ \Psi_{x_{N+1}x_{N+1}^T}(I_c - I) \end{bmatrix} + \begin{bmatrix} -2\Psi_{x_{N+1}x_{N+1}^T} \end{bmatrix}
\end{align*}
\]

Let

\[
\Delta A = \begin{bmatrix} \Psi_{x_{N+1}x_{N+1}^T}(I_c - I) \\ \Psi_{x_{N+1}x_{N+1}^T}(I_c - I) \end{bmatrix}
\]

and \(\Delta L = \begin{bmatrix} -2\Psi_{x_{N+1}x_{N+1}^T} \end{bmatrix}
\]

We thus can re-write Eq. (16) to reflect incremental learning:

\[
\begin{bmatrix} W_n \\ B_n \end{bmatrix} = (A + \Delta A)^{-1}(L + \Delta L)
\] (18)

Eq. (18) shows that the separating hyper-planes slopes and intercepts for the different \(c\) classes of Eq. (16) can be efficiently updated by using the old model parameters. The incremental change introduced by the recently acquired data stream is incorporated as a weighted ‘perturbation’ to the initially developed system parameters. This insures that points, which have low probability of occurrence, but are key with respect to the hyperplanes position are not outnumbered and neglected in the dynamic model update process.

B. Dynamic Processing for Chunk Data

For incremental chunk processing, the data is still acquired incrementally, but it is stored in a buffer awaiting batch processing. After capturing \(k\) sequences and if the model needs
to be updated, the recently acquired data are processed and the model is updated as described in Eq. (18). Alternatively, we can use the Sherman-Morrison-Woodbury [19] generalization formula described in Eq. (19) to account for the perturbation introduced by matrices $M$ and $L$ defined such that $(I+M^T A^{-1} L)^{-1}$ exists.

$$(A+LM^T)^{-1} = A^{-1} - A^{-1}L(I+M^T A^{-1} L)^{-1}M^T A^{-1}$$

(19)

where $M = \psi \left[ x_{N+1}(I - I_c) \right]$; $L = \psi \left[ x_{N+1}^T \right]$;

Using Eqs. (16) and (19), the new model can represent the incrementally acquired sequences according to Eq. (20).

$$\begin{bmatrix} W \\ B \end{bmatrix}_{old} = \begin{bmatrix} W \\ B \end{bmatrix}_{old} + \begin{bmatrix} \Delta E \\ \Delta U \end{bmatrix}$$

$$\begin{bmatrix} \Delta E \\ \Delta U \end{bmatrix} \left[ \begin{bmatrix} W \\ B \end{bmatrix}_{old} \right]^{-1} \left[ I - A^{-1}M(I+M^T A^{-1} L)^{-1}M^T A^{-1} \right]$$

(20)

Eq. (20) shows the influence of the incremental data on calculating the new separating hyper-plane slopes and intercept values for the different $c$ classes.

C. Performance Measurement for WP-SVM Classifier

ML algorithms have a tradeoff between classification accuracy on the training data and generalization accuracy on novel data. To demonstrate the validity of our proposed dynamic WP-SVM, we opt to use the balanced error rate (BER) and the confusion rate (CR) for performance measurement. CR is a performance measure derived from the entries $S_{ij}$ of the confusion matrix $CM$.

$$CM = \begin{bmatrix} S_{11} & \ldots & S_{1c} \\ S_{21} & \ldots & S_{2c} \\ \vdots & \ddots & \vdots \\ S_{c1} & \ldots & S_{cc} \end{bmatrix}$$

and $CR = \sum_{i=1}^{c} S_{ij}$

Index $i$ represents the correct class and index $j$ the predicted on. Thus, $S_{ij}$ represents the number of data belonging to class $i$ whereas SVM classifier recognized them as being class $j$.

BER is defined as the average statistic of the misclassification rate in each class.

$$BER = \frac{1}{c} \left[ \sum_{i \neq j}^{c} S_{ij} + \sum_{i \neq j}^{c} S_{ij} + \ldots \right]$$

BER coincides with the misclassification error rate (Mis_Err)

when: $\sum_{j=1}^{c} S_{ij} = \sum_{j=1}^{c} S_{2j} = \ldots = \sum_{j=1}^{c} S_{cj}$

For performance evaluation, we use different sets for model training and model validation because any criterion that has been optimized during the model training phase is likely to result in a biased optimistic estimate about the model generalization performance. Table 1 depicts the workflow of our proposed WP-SVM classifier.

**Table II: WP-SVM Algorithm Flow**

<table>
<thead>
<tr>
<th>Step #</th>
<th>Algorithm</th>
</tr>
</thead>
</table>
| **Step 1** | Train initial model using TrainSet which consists of $N$ patient data each having $f$ features. $\begin{bmatrix} W \\ B \end{bmatrix} = A^{-1} L$; $\begin{bmatrix} (C - G) \\ (D - H)^T \end{bmatrix}$; $L = \begin{bmatrix} E \\ U \end{bmatrix}$.

Store only $W$ and $B$ as Initial_Model. Discard TrainSet. |
| **Step 2** | Acquire incremental data IncSet. |
| **Step 3** | Test the generalization performance using TestSet with Initial_Model |

Case 1: If Mis_Err $<=$ Acceptable Rate, Initial_Model still valid.
- If Mis_Err is statistically increasing and or Count=$ Limit$, initiate a model retrain to insure learning and delete the incrementally acquired video sequence stored in the buffer. New Model= Retrain_Model - Go to Step 2.

Case 2: If Mis_Err $>$ Acceptable Rate, use WP-SVM. Inc_Model becomes.

1- For dynamic sequential processing: $\begin{bmatrix} W \\ B \end{bmatrix}_{new} = (A + \Delta A)^{-1}(L + \Delta L)$

$$\Delta A = \begin{bmatrix} \psi x_{N+1}^T (I_c - I_j) & \psi x_{N+1}^T (I_c - I_j) \\ \psi x_{N+1}^T (I_c - I_j) & \psi x_{N+1}^T (I_c - I_j) \end{bmatrix}$$

$$\Delta L = \begin{bmatrix} -2\psi x_{N+1}^T I_c \\ -2I_c \end{bmatrix}$$

2- For dynamic batch processing: $\begin{bmatrix} W \\ B \end{bmatrix}_{new} = \begin{bmatrix} W \\ B \end{bmatrix}_{old} + \begin{bmatrix} \Delta E \\ \Delta U \end{bmatrix}$

$$\begin{bmatrix} \Delta E \\ \Delta U \end{bmatrix} \left[ \begin{bmatrix} W \\ B \end{bmatrix}_{old} \right]^{-1} \left[ I - A^{-1}M(I+M^T A^{-1} L)^{-1}M^T A^{-1} \right]$$

Test the generalization performance of Inc_Model using TestSet - Go to Step 2.
V. EXPERIMENTAL RESULTS

A. Data Set Details

For validating our technique, we used labeled CTC data of true polyps (TP) and false positives (FP) that were detected by our CAD system [2]. These CTC data were acquired as follows: Helical single-slice and multi-slice CT scanners (GE HiSpeed CTi, LightSpeed QX/I, and LightSpeed Ultra; GE Medical Systems, Milwaukee, WI) were used in scanning the colon of a patient in supine and prone positions with collimations of 1.25 - 5.0 mm, reconstruction intervals of 1.0 – 5.0 mm, X-ray tube currents of 50 – 260 mA with 120 – 140 kVp, in-plane voxel sizes of 0.51–0.94 mm, and a CT image matrix size of 512 x 512. A total of 146 patients were scanned, which resulted in 108 normal cases (216 data sets in supine and prone views) and 38 abnormal cases (76 data sets). There were a total of 61 colonoscopy-confirmed polyps ≥6 mm. Twenty-eight polyps were 6-9 mm and 33 polyps were ≥10 mm (including 7 lesions ≥30 mm) in size.

For each dataset, the axial CT images were interpolated for generation of an isotropic CTC volume which was subjected to our CAD system for segmentation of polyp candidates that consist of actual polyps (i.e., TPs) and non-polyps (i.e., FPs) as determined by polypectomy followed by pathologic examination. We then added the artificially created data sequences, false calibration (FC), to the polyp candidates to reflect calibration issues with the experimental setup. A set of volumes of interest (VOIs), whose centers correspond to the centers of mass of the TP, FP, and FC, was extracted from the CTC volume. The whole database of VOIs was divided into two databases, Database 1 (DB1) and Database 2 (DB2), as shown in Table I. DB1 was used for pilot studies whereas DB2 was used for testing of WP-SVM.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Name</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB1</td>
<td>Database 1</td>
<td>188FP, 40 TP, 40 FC, VOI=12<em>12</em>12=1728 Features</td>
</tr>
<tr>
<td>DB2</td>
<td>Database 2</td>
<td>8008FP, 131TP, 131 FC, VOI=16<em>16</em>16=4096 Features</td>
</tr>
</tbody>
</table>

TABLE III: DATABASE PROPERTIES

FP: False Positive; TP: True Positive; FC: False Calibration, VOI: Volume of Interest

Because the main objective is to substantiate the effectiveness of WP-SVM in adequately binning incremental data to the appropriate TP, FP, or FC class, we decided not to use any special kernel or novel image preprocessing techniques to reduce the dimension of the feature space. Kernel parameters often heavily influence LS-SVM model generalization performance; however, appropriate kernel selection requires extensive a priori knowledge of the data being investigated, and thus it is not the focus of this paper. Instead, we ran multiple heuristic experiments to determine the best sampling rate for the feature space.

B. Parameter Tuning and Classifier Stability

To assess the effect of the scaling factor $\lambda$ in Eq. (7) on Mis_Err and to find a range for acceptable Mis_Err, we performed a pilot study on DB1. Using all of the 1728 features, we ran 21 different experiments with variable scaling factor values. We found that Mis_Err is directly proportional to the scaling factor $\lambda$. The SVM classifier accuracy rates increased with decreasing $\lambda$ values. We decided to use $\lambda = 10^4$ for the remaining of our analysis.

Using DB1, we evaluated the performance of the standard SVM routine (RBF kernel) available in the Matlab Bioinformatics Toolbox (The MathWorks, Inc., Natick, MA). For 5 different training and testing sample sizes, we obtained a misclassification rate of 14.6%, which was our acceptable rate for any calculated Mis_Err using WP-SVM.

For preliminary screening, we ran 25 experiments on DB2 while varying the initial training, incremental, and testing pattern sizes. Typically, in a predicting benchmark, the size of a testing set (hereafter, TestSet) is set to be ten times larger than that of the training set (hereafter, TrainSet) with the ratio of TP and FP examples closely matched. However, because both DB1 and DB2 are not balanced with respect to TP and FP, and because BER is sensitive to the error in the minority class, we opted to use 40% to 60% of each class as TrainSet and equally divided the remaining data points between TestSet and IncSet. In each experiment, we heuristically varied the feature space patterns up to 70 times in order to provide a complete coverage of the whole space, calculated Mis_Err, and selected the features that returned a Mis_Err ≤ 15%. Fig. 1 shows a histogram for these selected features.

\[ \text{Mis_Err} \leq 15\% \]

As Fig. 1 indicates, the sampling rate and classifier accuracy are inversely related; as the former increases, the number of features included in the initially trained model (hereafter, Initial_Model) decreases and so does the classifier's accuracy.
in classification. However, it is worth mentioning that large numbers of features in a classification task provide sufficient degrees of freedom to the model. The classifier could then become sensitive to the specific data set which can lead to the over-fitting to the data when feature variances are not low. To achieve acceptable model accuracy and generalization, we kept the number of feature space contained by the upper bound of 730.

C. WP-SVM Performance in Processing Chunk Data

We evaluated BER for TP for the training model with respect to selected increased features sampling rates and initial sample sizes.

Fig. 2 shows that BER degrades as sampling rates increase, which results in fewer features being considered in the model. However, as sample size increases, BER becomes less susceptible to the high sampling rates.

Using the feature space identified earlier, we ran 20 different experiments and we varied the numbers of the initial, dynamic, and test vectors that were used for developing the classification model. We compared the performance of the classifier when WP-SVM was applied (hereafter, Inc_Model) versus the case where the classifier was retrained (hereafter, Retrain_Model).

Fig. 3 represents the confusion rate observed in both models.

The same TestSet was used to calculate the CRs for the Retrain_Model and the Inc_Model. On average, the Inc_Model showed a CR that is 1.42 times higher than that of the Retrain_Model. However the CR for the Initial_Model was two times higher than that of the Retrain_Model. The CR for the Inc_Model is thus deemed acceptable because it is dramatically confined by the generalization capabilities of the Initial_Model generalization and accuracy capabilities, when data is being processed in batch mode.

D. WP-SVM Performance in Processing Sequential Data

To validate processing of incremental knowledge, we applied the sequential WP-SVM to the same datasets used in subsection C above.

Fig. 5 shows the normalized CRs for the Retrain_Model and
the Inc Model. In sequential processing, the CRs for WP-SVM were within 1.1% of the misclassification errors for the Retrain Model. This represents 30% improvement over the chunk data processing scenario.

Fig. 6 shows the marginal degradation in normalized CPU usage time. The Retrain Model is now 1.52% slower that the Inc_model, which means that the improved performance of the sequential classifier degraded CPU usage time by 10% with respect to batch processing. However this is a reasonable price to incur for performance enhancement.

Finally Table IV compares the storage requirements for the Retrain Model and the Inc_model.

<table>
<thead>
<tr>
<th>Classifier Type</th>
<th>Data Structure Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrain Model</td>
<td>1- f by c for classifier parameters</td>
</tr>
<tr>
<td></td>
<td>2- permanent storage of size (N+incnum)*f that is always increasing.</td>
</tr>
<tr>
<td>Inc Model</td>
<td>1- f by c for classifier parameters</td>
</tr>
<tr>
<td></td>
<td>2- temporary memory of size incnum*f for dynamic data if classifier is not updated.</td>
</tr>
</tbody>
</table>

incnum = number of dynamic data acquired

VI. CONCLUSION

We presented a novel extension to LS-SVM to dynamically classify CTC. The incremental WP-SVM model accuracy is more constrained by the initial model accuracy when chunk learning instead of iterative learning is applied. In the latter, model confusion rate proved to be within 1.1% of the retrain model confusion rate when iterative learning is applied. Storage requirement is drastically reduced in WP-SVM because basically only the hyper plane parameters are kept in memory for system update. Computation time for WP-SVM, as a worst case scenario when feature space is greater than 250, was found to be 1.64 times faster than the Retrain Model computation time. When a smaller feature set was used, WP-SVM showed to be twice as fast. Follow on work plans on including kernel to WP-SVM so that we compare it to benchmark data. We will also investigate applying an adaptation of SVM as an image preprocessing technique for feature extraction in addition to the dynamic learning task presented in this paper.

ACKNOWLEDGMENT

This work was supported in part by USPHS Grant CA095279. The authors would like to thank IBM’s Systems and Technology Group of Essex Junction VT for their support to this study. The authors would like to acknowledge J. B. Zurn for her comments in improving this manuscript.

REFERENCES