INCREMENTAL LEARNING WITH LARGE DATASETS

Balathasan Giritharan

Dissertation Prepared for the Degree of

DOCTOR OF PHILOSOPHY

UNIVERSITY OF NORTH TEXAS

May 2012

APPROVED:

Xiaohui Yuan, Major Professor
Armin Mikler, Committee Member
Jianguo Liu, Committee Member
Parthasarathy Guturu, Committee Member
Barrett Bryant, Chair of the Department of Computer Science and Engineering
Costas Tsatsoulis, Dean of the College of Engineering
James D. Meernik, Acting Dean of the Toulouse Graduate School

Giritharan, Balathasan. Learning with Large Datasets. Doctor of Philosophy (Computer Science and Engineering), May 2012, 69 pp., 9 tables, references, 86 titles.

This dissertation focuses on the novel learning strategy based on geometric support vector machines to address the difficulties of processing immense data set.

Support vector machines find the hyper-plane that maximizes the margin between two classes, and the decision boundary is represented with a few training samples it becomes a favorable choice for incremental learning. The dissertation presents a novel method Geometric Incremental Support Vector Machines (GISVMs) to address both efficiency and accuracy issues in handling massive data sets. In GISVM, skin of convex hulls is defined and an efficient method is designed to find the best skin approximation given available examples. The set of extreme points are found by recursively searching along the direction defined by a pair of known extreme points. By identifying the skin of the convex hulls, the incremental learning will only employ a much smaller number of samples with comparable or even better accuracy. When additional samples are provided, they will be used together with the skin of the convex hull constructed from previous dataset. This results in a small number of instances used in incremental steps of the training process.

Based on the experimental results with synthetic data sets, public benchmark data sets from UCI and endoscopy videos, it is evident that the GISVM achieved satisfactory classifiers that closely model the underlying data distribution. GISVM improves the performance in sensitivity in the incremental steps, significantly reduced the demand for memory space, and demonstrates the ability of recovery from temporary performance degradation.

Copyright 2012

by

Balathasan Giritharan

TABLE OF CONTENTS

Page

CHAPTER 1

INTRODUCTION

With the advancement in data acquisition, storage technologies, and the proliferation of the world wide web, large amounts of multimedia data in form of text, images and videos has grown tremendously. The applications with these data pose new challenges to classification algorithms as they do not always scale up to large datasets, and new or modification to existing machine learning algorithms are sought to handle them.

This dissertation is motivated by the need in processing of capsule endoscopy (CE) videos, which shares many characteristics of large databases. CE a miniature device, to visualize the gastrointestinal tract and the only method to view the entire small intestine. It is now a widely adopted procedure for diagnosis of diseases including obscure-bleeding, Crohn's disease, gastric ulcers, and colon cancer. CEs are miniature capsules and are produced by the manufacturers from around the world including Given Imaging, Olympus, Sayka IntoMedic, Jinshan Science, under the brand names PillCam, Endo Capsule, Sayka, MicroCam, OMOM-Pill. The CE videos used in this research are produced with Pillcam by Given Imaging. The CE is used to examine the entire small intestine non-invasively [35, 36, 38, 49, 5, 2], and has been used mainly to diagnose lesions beyond the reach of conventional push enteroscopy and colonoscopy. Its clinical importance has been confirmed in inflammatory bowel diseases [6, 20, 27, 34, 42, 73] and gastrointestinal (GI) bleeding. The results have shown great improvement in diagnostic yield for bleeding sources in patients with obscure GI bleeding, and in diagnosing and localizing the source of blood loss [50, 78, 77, 79].

Since the implementation of the CE, research and technology improvements have added additional values to the CE and its video interpretation. Efforts have been devoted to many aspects, such as video enhancement [52], video segmentation [58, 57, 31], and anomaly detection. Among all the efforts, anomaly detection has been studied the most, primarily due to its importance and the significance in CE. The Suspected Blood Indicator (SBI) feature provided by Given Imaging, a CE manufacturer, claims it has the capability of detecting blood in the video frames. However, in a study by Liangpunsakul et al. [53] the overall sensitivity and accuracy of SBI were only 25%

and 34.8%, respectively. SBI exhibited better performance in the setting of active bleeding lesions in the small bowel with a reported sensitivity and accuracy of 81.2% and 83.3%.

The Pillcam device captures two images per second for approximately 8 hours, during which the device captures about 55,000 color images of 256×256 pixels in size. Reviewing CE videos is a tedious and time consuming task for medical specialists. It usually takes more than one hour to annotate a full length video by a medical specialist. Computer algorithms are sought to reduce the review time by identifying suspicious frames for verification.

With improved optical sensors the size of each image captured is expected to increase. In addition, to obtain better generalization performance and avoid curse of dimensionality, we expect to train classification algorithms using videos of multiple human subjects. For a dataset with n samples (each containing m features) the memory requirement would be as high as $O(n^2)$ (e.g. support vector machines) or $O(mn)$ (e.g. decision trees). The implementation of classification algorithms that require all training data to be present in memory would make learning task extremely challenging. Existing classification algorithms are facing difficulties in handling large number of training samples.

Incremental learning has great potential to handle the aforementioned challenges. At each incremental step, only the samples that are useful to build future classifiers along with model information is retained. When new data becomes available, they are integrated with the retained samples, and the model is updated and/or trained to build a new classifier.

On the other hand with the large amount of digital data it is not possible to annotate all of them, or due to privacy concerns it is not possible to collect for research purpose. In such cases it would be practical to build a classifier based on initial data with a decent performance and have the classifier learn as new samples arrive. Applications for which all the training data are not available initially and become available overtime, incremental learning naturally fits in.

An incremental learning algorithm is said to be exact with respect to some criterion of interest e.g. the learned hypothesis's expected generalization accuracy: if it is guaranteed to achieve the same result as that obtained in the batch learning setting where the entire dataset is available to the learning algorithm[14]. In real world applications it would not be possible to learn the class boundaries, due to the underlying topology.

In many real world problems involving sufficiently expressive concept classes, exact incremental or distributed learning may not be possible even in principle. In other instances although possible in principle it may not be feasible in practice for computational reasons.

A key question of incremental SVMs is how to retain knowledge from the training examples in each iteration to maximize the unbiased representation of underlying data distribution while maintain a concise subset. Intuitively, SVs from the current SVM are kept. This works well in cases where the existing examples closely represent the topography of the classes. However, if a new instance dramatically changes the topography of the distributions and hence the decision hyper-plane, a previously removed non-support vector instance could become the margin mover.

The goal of this dissertation is to advance the learning strategy for classification of large datasets. The scientific contribution includes the verification of the following hypothesis and an advanced technology for automatic capsule endoscopy screening.

Hypothesis : With the identification of the skin of the convex hull, maximum margin classification handles large amount of data incrementally without performance degradation.

Goal : Enabling learning with large databases more practical.

Approach : Finding an efficient method to identify and retain samples and model information such that when new examples become available an optimal classifier could be built without revisiting all data available, with generalization accuracies which are exact or comparable to those obtained in batch learning fashion.

In the rest of this dissertation, Chapter 2, I provide a literature survey on Incremental Support Vector Machines and highlighting the open issues. Next Chapter 3 starts with an introduction to Support Vector Machines, then relates to SVMs to Geometrical Support Vector Machines, and the remaining of the chapter is dedicated to the Geometrical Incremental Support Vector

Machines(GISVM). Chapter 4 discusses the experiments on synthetic and benchmark datasets to compare GISVM with libSVM. Next, Chapter 5 provides two applications of GISVM to enable computer aided diagnosis for capsule endoscopy videos. Finally, Chapter 6, summaries the dissertation contributions.

CHAPTER 2

INCREMENTAL LEARNING FOR SUPPORT VECTOR MACHINES

Conventional SVM algorithms require that the entire dataset is available at the time of training a classifier [81]. While a large number of training examples helps to reduce the generalization error, the learning process can become computationally expensive. In addition, data acquired may not represent the underlying distribution. For instance, medical data are collected and stored over a long period. The cumulative datasets become overly large. Out of the data collected, only a small fraction is positive. In such situations, data is to be processed in parts, and the results are combined to make it feasible to use with the available computational resources.

Incremental learning algorithms are usually applied to problems where the complete training set becomes available over time [28, 25]. It is also applied when the complete training sets are larger in size [26, 84]. and makes the learning process is possible when limited computation resource is available.

Initially, incremental learning was a method to handle large dataset. Later it was applied to address on-line learning as well. When developing classifiers using learning methods, a large number of training data can help reduce the generalization error. However, the learning process itself can get computationally intractable [25], in which case data can be processed in parts, and the results are combined to fit the limited memory available. Efficient and scalable approaches are required that can modify knowledge structures in an incremental fashion without having to revisit previously processed data. Three challenges faced in developing incremental learning methods are as follows:

- How to retain knowledge from the training examples in each iteration to maximize the unbiased representation of underlying data distribution, while maintaining a concise subset
- How to adapt to the changing trends of the data distribution over time

2.1. Modeling the Representative Set

Conventional SVMs [81, 13] classify instances by assigning them to one of the two disjoint half-spaces. Fung and Mangasarian [29] proposes a proximal SVM to reduce the time complexity of

finding the separating hyper-plane. In proximal SVMs the binary classification problem bounded by the soft margin planes are replaced by two planes with data clustered around them [59]. Proximal SVMs were shown to perform at the similar level of accuracy as conventional SVMs, however it takes less time searching the separating hyper-plane. Later work of Fung and Mangasarian [30] showed that the proximal SVM can also be applied to incremental learning.

Attempts of incremental SVM starts with retention of the SVs from each training iteration [76, 63, 43]. The method in [76] keeps only the SVs at each incremental step, which are aggregated into the next training iteration. The model obtained via this strategy will be the same or highly similar to what would have been obtained by using all data samples to train. Unless kept as SVs, examples are discarded after training. Mitra et al. use an error-driven technique in the incremental SVMs [63]. In addition to the SVs, the method keeps a number of non-SV instances. Given a trained SVM_t at iteration t, new instances are classified using SVM_t. The SVs of SVM_t together with a certain number of correctly classified and misclassified instances are used to train the new model SVM_{t+1}. Alternatively, work of [25] keeps only the misclassified data. Given the model at time t, new data are loaded into memory and classified using SVM_t. If the data is misclassified it is kept, otherwise discarded. Once a given number of misclassified data is collected, the update takes place. The SVs of the last trained SVM, together with the misclassified instances, are used as training data to obtain the new model.

The assumption of minimum change of the hyper-plane serves the foundation of the above methods, which primarily retains the SVs. These methods could delete candidates, which have the potential to be SVs during future iterations. In actuality, new examples emerge in time and the SVMs need to be updated to a significantly changed hyper-plane.

Katagiri and Abe [43] proposed one-class SVMs to select candidate SVs, which reduce the possibility of SVs being deleted when hyper-plane is rotated. A hyper-sphere is generated for each class and only the instances lying close to the boundary of the hyper-sphere are retained as candidate SVs for future iterations. Although this method handles the rotation of the decision boundary, the assumption of hyper-sphere to model data distribution becomes unrealistic in many real-world applications. When a class is not inherently spherical, a small reduction on the radius

would leave out a significant number of samples, which is not recoverable in the later iterations of training.

2.2. Space Complexity and Size of the Representative Set

Hernandez et al. present an incremental procedure for growing support vector classifiers (GSVC), which avoids pruning mechanism after SVM training [69]. GSVC employs a multi-resolution approach to design of a multi-kernel support Vector Classifiers. The learning using Gaussian kernels starts with initial large spreading value for σ which is a coarser level of detail, and progressively decreases this value in newly added kernels, such that finer solutions are obtained as the machine grows. Agarwal et al. [3] show that the concept of the span of SVs can be used to build a classifier that performs reasonably well while satisfying given space and time constraints, thus making it potentially suitable for such online situations. Imbalance handled through [83], presents a new method for imbalanced data classification. This method is based on SVM classifiers and backward pruning technique to reduce the complexity of the learnt classifier.

Mitra, Murthy and Pal present probabilistic SVMs, where training set is refined by actively querying for the labels of pixels from a pool of unlabeled data. The label of the most interesting or ambiguous unlabeled point is queried at each step [64]. In this method, active learning is exploited to minimize the number of labeled data used by the SVM classifier by several orders. Orabona et al. propose an on-line algorithm, called On-line Independent SVMs (OISVMs) which approximately converges to the standard SVM solution each time new observations are added; the approximation is controlled via a user-defined parameter. The method uses a set of linearly independent observations and tries to project every new observation onto the set obtained so far, reducing time and space requirements at the of a negligible loss in accuracy. As opposed to similar algorithms, authors claim that the size of the solution obtained by OISVMs is always bounded, also implying a bounded testing time [67].

Proximal SVM employs a greedy search across the training data to select the basis vectors of the classifier, and tunes parameters automatically using the simultaneous perturbation stochastic approximation after incremental additions are made [72]

Instead of selecting training samples randomly Chen Yin et al., divide training set into groups

and k-means clustering algorithm applied to collect the initial set of training samples [19]. In active query, a weight is assigned to each sample according to its confidence factor and its distance to the separating hyper-plane. The confidence factor is calculated from the error upper bound of the SVM to indicate the closeness of the current hyper-plane to the optimal hyper-plane [19].

LocalSVM is a classification approach that combines instance-based learning and statistical machine learning. It builds an SVM on the feature space neighborhood of the query point in the training set and uses it to predict its class. LocalSVM can improve over SVM, in terms of classification accuracy, but it is computationally feasible only on small datasets. FastLSVM, is based on LocalSVM that decreases the number of SVMs that must be built for large datasets. FastLSVM pre-computes a set of local SVMs in the training set and assigns to each model all the points lying in the central neighborhood of the k points on which it is trained. The prediction is performed applying to the query point the model corresponding to its nearest neighbor in the training set [75]. An algorithm for data condensation using SVMs extracts data points lying close to the class boundaries, which form a much reduced but critical set for classification. The problem of large memory requirements for training SVM in batch mode is circumvented by adopting an active incremental learning algorithm [63]. Cauwenberghs and Poggio present on-line recursive algorithm for training SVMs, by adding one vector at a time, in adiabatic increments retain the Kuhn-Tucker conditions on all previously seen training data. Here, The incremental procedure is reversible, and decremental unlearning offers an efficient method to exactly evaluate leave-one-out generalization performance [15].

2.3. Adaptability to Changing Trends

Cauwenberghs and Poggio [15] present an incremental and decremental SVM learning method. This proposed method optimizes the Karush-Kuhn-Tucker (KKT) conditions and partitions the training data and the corresponding coefficients of the hyper-plane, denoted with $\{\alpha_i, b\}$, into three categories: *the margin SVs* which are strictly on the margin, *the error SVs* which violate the margin but are not necessarily misclassified, and *ignored vectors* which are within the margin. When a new instance c is misclassified, the SVM is updated by finding new values for the hyper-plane parameters $\{\alpha_i, b\}$. Bookkeeping is required to assign samples to one of the three categories. The

bookkeeping effort increases the computational expense, which could be as high as $O(n^3)$ for each incremental example included. This reflects a trade-off between memory and time. A later work of Diehl and Cauwenberghs [24] eliminates the computationally intensive tuning of hyper parameters and the objective functions, using leave-one-out error estimation. Again, the methods assume that the hyper-plane does not change significantly.

Klinkenberg and Joachims proposed a method to handle drift in SVMs [45]. The drift represents changes of underlying distribution of the data collected over an extended period for learning tasks. The method addresses the problem by maintaining a window on the training data stream and adjusting the window size so that the estimated generalization error is minimized. Ensemble methods are also explored in an incremental way to classify imbalanced datasets [65]. For each classifier, a conditional weight is used, which is the ratio of the number of instances from a particular class used for training that classifier to the number of instances from that class used for training all classifiers thus far in the ensemble. The voting weights are determined as individual training performances of the classifiers, adjusted by the class conditional weight factors [65].

Shilton et al. address the of sequentially arriving data and fast constraint parameter variation. This method uses a warm-start algorithm for the training of SVMs, which allows to take advantage of the natural incremental properties of the standard active set approach to linearly constrained optimization problems. This Incremental training allows quickly retraining a SVM after adding a small number of additional training vectors to the training set of an existing (trained) SVM [1].

Boubacar et al. employ an online clustering algorithm that is developed to learn continuously evolving clusters from non-stationary data. This algorithm is based on SVM methods with kernel trick in reproducing Hilbert space, and uses a fast incremental learning procedure to take into account model changes over time. Dedicated to online clustering in multi-class environment, the algorithm is based on an unsupervised learning process with self-adaptive abilities [4].

CHAPTER 3

METHODOLOGY

3.1. Support Vector Machine

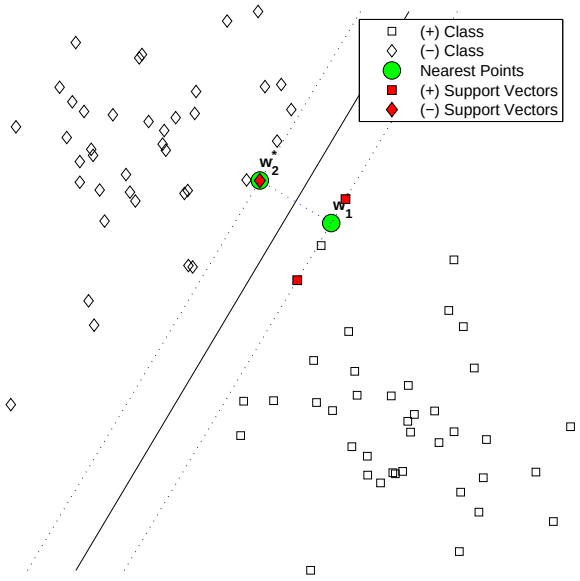


FIGURE 3.1. The nearest points w_+^*, w_-^* between the two classes found using a SVM. The support-vectors of w_+^*, w_-^*, are denoted using shaded diamond and square respectively. Decision boundary that is perpendicular to the vector to connect the nearest points is denoted with a solid line.

SVMs find the hyper-plane that maximizes the margin between two classes. The margin is defined by a few training instances of the two classes, i.e. the SVs. The decision boundary in Fig 3.1 separating the two classes could be represented by using the SVs denoted in red and a weight b. Removing the rest of instances from the dataset will not alter the classification performance of SVM. This property makes the SVM a favorable choice for incremental learning, which spawns many research on incremental SVMs [76, 85, 43, 63, 15, 80].

SVMs [18, 13, 16] originated from the statistical learning theory [81], is based on structural risk minimization (SRM). The learning task is defined as $f : \mathbb{R}^N \rightarrow \{-1, +1\}$, using input-output training dataset,

$$(x_1, y_1), \ldots, (x_n, y_n), \text{ where } x_i \in \mathbb{R}^N, y_i \in \{-1, +1\}$$

where n is the number of training samples, and N is the number of features of the samples. Let $\{x_i\}$ be a set of patterns having the labels $\{y_i\}$, $y_i \in \{-1, 1\}$, where $X_+ \in \{x_i : y_i = 1\}$, $X_- \in \{x_i : y_i = -1\}$. SVMs find a decision boundary that maximizes margin between the two classes.

The decision boundary is represented as $w \cdot x + b = 0$, where w is the weight vector, which is perpendicular to the hyper-plane. The training data satisfies the constraints $x_i \cdot w + b \geq +1$ for X_+ and $x_i \cdot w + b \leq -1$ for X_-. Combining both functions generates a unified decision function $y_i(x_i \cdot w + b) \geq 1$ for $i = 1, \cdots, n$. The optimization problem to maximize the margin is hence, formulated as follows:

$$\min_{w,b} ||w||, \text{ s.t. } y_i(x_i \cdot w + b) \geq 1, \forall i \tag{1}$$

Eq (1) consists of the square root, which makes it difficult to solve. Eq (1) can be replaced by,

$$\min_{w,b} \frac{1}{2}||w||^2, \text{ s.t. } y_i(x_i \cdot w + b) \geq 1, \forall i \tag{2}$$

Eq (2) can be transformed to its dual form by incorporating positive Lagrange multipliers α_i $i = 1, \cdots, n$, for each of the equality constraints. Switching to the Lagrangian formulation makes it easier to convert into a quadratic equation, and the dual formulation is based only on dot products of the training data and can be applied to nonlinear case. The Lagrangian dual problem of Eq (2) is

$$\min L = \frac{1}{2}||w||^2 - \sum_{i=1}^{m} \alpha_i \left[y_i(x_i \cdot w + b) + 1\right] \tag{3}$$

By setting the gradient of L with respect to w, b we have,

$$w = \sum_{i=1}^{l} \alpha_i y_i x_i$$

$$\sum_{i=1}^{l} \alpha_i y_i = 0$$

Since the optimization problems are convex, any local minimum found can be identified as global minimum. Only the data points that satisfy $y_i(x_i \cdot w + b) = 1$ would have $\alpha_i > 0$, which are known as SVs and lie closest to the decision boundary. The Fig 3.1 illustrates the decision boundary between the two classes and the SVs are denoted using shaded diamonds and squares.

3.1.1. Linearly Non-separable Classes

For real-world applications, it is possible that the two classes are non-separable using a linear hyper-plane. In such situations the Eq (2),(3) could not be solved. The non-separable case can be solved by including error penalty, by adding a slack variable $\xi_i \geq 0$ to the constraints. This leads to a new objective function:

$$\min_{w,b,\xi} \frac{1}{2}||w||^2 + C\sum_{i=1}^{m} \xi_i,$$

$$\xi_i \geq 0,\ y_i(x_i \cdot w + b) \geq 1, \forall i \tag{4}$$

If a training example lies on the wrong side of the hyper-plane, the corresponding $\xi_i \geq 1$. C allows the trading off between training error and model complexity. Using Lagrangian multiplier, we rewrite Eq (4) as follows:

$$L = \frac{1}{2}||w||^2 - \sum_{i=1}^{m}\sum_{j=1}^{m} i\alpha_i\alpha_j y_i y_j (x_i \cdot x_j)$$

subject to

$$\sum_{i} \alpha_i y_i = 0, 0 \leq \alpha_i \leq C \tag{5}$$

(6)

And the solution of Eq (6) would be given by $w = \sum_i \alpha_i y_i x_i$.

Time complexity of solving such a quadratic programming problem lies between quadratic and cubic of the number of training samples used [74].

Kernel trick [11] enables a non-linear classifier, for the maximum margin classifiers following the linear model. This transforms the original algorithm by replacing the dot products with a kernel functions, and allows finding the maximum margin separating plane in higher dimensional feature space.

Linearly separable. For linearly separating hyper-planes as computed by a perception the classification without training errors are,

$$y_i((w \cdot x_i) + b) \geq 1 \text{ for } i = 1, \ldots, n \tag{7}$$

Here the goal is to find $w \in \mathbb{R}^N$ and b such that expected risk is minimized. Since the expected risk cannot be quantified, the upper bound on the expected risk is minimized. The bound consists of empirical risk and the VC confidence term. One approach is to keep be empirical risk zero by constraining w and b to a linearly separable case, and minimizing the VC confidence term. For linear classifiers the VC dimension h is bounded by $h \leq ||w||^2 R^2 + 1$ where R is the smallest ball round the training data which is fixed for a given training data. Therefore the complexity term could be minimized by minimizing the $||w||^2$.This can be represented as a quadratic optimization problem as,

$$\min_{w,b} \frac{1}{2}||w||^2 \tag{8}$$

The optimization problem represented by (8) could be solved by forming the dual optimization problem. Introducing Lagrange multipliers $\alpha_i > 0$ for $i = 1, \ldots, n$ we get

$$L(w, b, \alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{n} \alpha_i (y_i((w \cdot x_i) + b) - 1) \tag{9}$$

The optimization problem now it to minimize (9) with respect to w and b and to maximize

respect to α_i. At the optimal point we solve for the following saddle point equations.

$$\frac{\partial L}{\partial b} = 0$$
$$\frac{\partial L}{\partial w} = 0$$

Linearly Non Separate Case. In the linearly separable case we considered the empirical error to be zero. For linearly non-separable case this would not be so, since it could push towards over-fitting. A trade off is achieved by including a slack variable to relax the hard margin constraints.

$$y_i((w \cdot x_i) + b) \geq 1 - \xi_i \; \xi_i \geq 0 \text{ for } i = 1, \ldots, n \tag{10}$$

The SVM solution can be found by keeping the upper-bound on VS small and by minimizing upper-bound on $\sum i = 1^n \xi_i$ on empirical risk. Thus the new optimization problem would be,

$$\min_{w,b} \frac{1}{2}||w||^2 + C \sum i = 1^n \xi_i \tag{11}$$

where C is a regularization constant tradeoff between the two components. And (11)'s dual is solved similar to previous.

3.2. Geometrical SVMs

Geometric interpretations of the optimization problem SVM has been presented in the past [81, 44, 10], both geometric and the quadratic optimization view of SVM have been shown to be equivalent [18, 10, 22, 86]. The dual formulation of the optimization problem of SVM can be interpreted as finding the shortest distance between the Convex-Hulls.

Geometric approach to SVM represents the two classes as convex hulls and finds the minimum distance between the two [44]. The convex hull based SVM is not efficient for all non-separable classes. To overcome this, Reduced Convex Hull (RCH) [18, 22] has been used for Geometric SVM [62, 61].

3.2.1. Convex Hull and Extensions

Caratheodory's theorem on convex sets states that if a point $x \in R^d$ lies in the convex hull of a set P, there is a subset $\acute{P}$ of P consisting of no more than $d+1$ points such that x lies in the convex hull of $\acute{P}$.

Let x be a point in the convex hull of P. Then, x is a convex combination of a finite number of points in P. In other words, x lies in a d-simplex with vertices's in P.

Let x be a point in the convex hull P. Then x is a convex combination of a finite number of points in P.

$$x = \sum_{j=1}^{k} \lambda_j x_j$$

where every x_j in every λ_j is positive, and $\sum_{j=1}^{k} \lambda_j = 1$

Suppose $k > d+1$, then $x_2 - x_1, x_3 - x_1, \ldots, x_k - x_1$ are linearly independent. Therefore there exists scalars $\mu_2, \ldots, \mu_k$ not all zero such that,

$$\sum_{j=2}^{k} \mu_j (x_j - x_1) = 0$$

Let us define $\mu_1 = -\sum_{j=2}^{k} \mu_j$, then

$$\sum_{j=1}^{k} \mu_j x_j = 0$$

$$\sum_{j=1}^{k} \mu_j = 0$$

and not all of the μ_j are equal to zero, therefore at least one $\mu_j > 0$. then,

$$x = \sum_{j=1}^{k} \lambda_j x_j - \alpha \sum_{j=1}^{k} \mu_j x_j$$

$$= \sum_{j=1}^{k} (\lambda_j - \alpha \mu_j) x_j$$

for any $\alpha \in R$.

Note that $\alpha > 0$ for all $j = 1, \ldots, k$

$$\lambda_j - \alpha\mu_j >= 0$$

In particular, $\lambda_i - \alpha\mu_i = 0$, by definition of α.

$$x = \sum_{j=1}^{k} (\lambda_j - \alpha\mu_j) x_j$$

where every $\lambda_j - \alpha\mu_j$, is nonnegative, their sum is one. Furthermore, $\lambda_i - \alpha\mu_i = 0$. Therefore, x represented as a convex combination of at most $k-1$ points of P. This process can be repeated until x is represented as a convex combination of at most $d+1$ points in P.

Convex Hull - Representation. Given a set X, convex hull is a linear combination of all the elements of X

$$C(X) = \left\{ \sum_{i=1}^{k} \alpha_i x_i;\ x_i \in X,\ 0 \leq \alpha_i \leq 1,\ \sum_{i=1}^{k} \alpha_i = 1 \right\}$$

Reduced Convex Hull (RCH) [10] also known as Soft Convex Hull [22] is the set of all convex combinations of elements of the X, denoted by $R(X, \mu)$, $\mu < 1$, is

$$R(X, \mu) = \left\{ \sum_{i=1}^{k} \alpha_i x_i;\ x_i \in X,\ 0 \leq \alpha_i \leq \mu,\ \sum_{i=1}^{k} \alpha_i = 1 \right\}$$

The difference between a convex hull and RCH, is that the α_i of convex hull are bounded by μ in a RCH. By finding a suitable μ for each class, it is possible to make the RCH of two classes linearly-separable [61]. For example Fig 3.4-(a) is the convex hull of the two classes are linearly non-separable. By using $R(X_i, 0.08)$, for $i = \{+, -\}$ it is possible to make the two class linearly separable as illustrated by outermost dashed lines in Fig 3.4-(b).

RCH has been used to handle SVM classification using data with non-separable classes [22, 10, 61]. The RCH does not preserve the shape of convex hull and does not provide the sufficient conditions to find the extreme points. To overcome this, compressed convex hulls [70] has been proposed, however it makes explicit assumptions on the kernel, therefore not well suited for SVM.

3.2.2. Geometric SVM

The optimization problem Eq (2) for the separable classes is equivalent to finding the minimum distance between the two classes, for $w_+ \in C(X_+), w_- \in C(X_-)$ can be written as

$$\begin{aligned} &\min_{\alpha} \|w_+ - w_-\|, \\ &= \min_{\alpha} \left\| \sum_{x_i \in X_+} \alpha_i x_i - \sum_{x_j \in X_-} \alpha_j x_j \right\| \\ &\text{subject to } \alpha_i \geq 0, \sum_{i \in X_+} \alpha_i = 1, \sum_{i \in X_-} \alpha_i = 1, \end{aligned} \tag{12}$$

And the optimization problem Eq (4) for the non-separable classes can be written in terms of, RCH as follows:

$$\begin{aligned} &\min_{\alpha} \|w_+ - w_-\| \\ &= \min_{\alpha} \left\| \sum_{x_i \in X_+} \alpha_i x_i - \sum_{x_j \in X_-} \alpha_j x_j \right\| \\ &\text{subject to } 0 < \alpha_i \leq \mu, \sum_{i \in X_+} \alpha_i = 1, \sum_{i \in X_-} \alpha_i = 1, \end{aligned} \tag{13}$$

The nearest points w_+^*, w_-^*, are linear combinations of samples with $\alpha_i > 0$, which are the SVs. The decision boundary is perpendicular to $w = w_+^* - w_-^*$, and the bias b would be the offset of the hyper-plane from the origin.

For linearly non-separable classes, the SVM finds the optimal separating hyper-plane between two classes of training samples in the feature space. The SVM works on the feature space, which is a Reproducing Kernel Hilbert Space, where the mapped patterns lie $\phi : X \to \mathcal{H}$. It is not necessary to know the mapping function ϕ, but only the kernel is needed. The kernel is represented as the inner products of the mappings of all the samples $k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$ for all $x_i, x_j \in R^n$. Using kernels, nonlinear classification problems are converted into linear classification problem in a higher dimensional space $\mathcal{H}$.

Geometric SVM represents two classes as convex hulls and solves the problem by finding the

minimum distance between the two [44]. Given a set $X = \{x_1, x_2, \ldots, x_n\}$, the function ϕ maps each instance into features space, $\phi(x_i)$. For simplicity, we use ϕ_i to denote $\phi(x_i)$ and the mapped points forms a feature set $\Phi = \{\phi_1, \phi_2, \ldots, \phi_n\}$. The convex hull, $C(\Phi)$, is a linear combination of all the instances in X:

$$C(\Phi) = \left\{ \sum_{i=1}^{k} \alpha_i \phi_i |\ \phi_i \in \Phi,\ 0 \leq \alpha_i \leq 1,\ \sum_{i=1}^{k} \alpha_i = 1 \right\} \tag{14}$$

In the case of linearly non-separable classes, soft convex hulls [22] or Reduced Convex Hull (RCH) [62, 61] has been proposed. The RCH, $R(\Phi, \mu)$, is the set of convex combinations of instances in Φ with α_i bounded by a μ, $\mu \leq 1$:

$$R(\Phi, \mu) = \left\{ \sum_{i=1}^{k} \alpha_i \phi_i |\ \phi_i \in \Phi,\ 0 \leq \alpha_i \leq \mu,\ \sum_{i=1}^{k} \alpha_i = 1 \right\} \tag{15}$$

By selecting appropriate μ for each class, the linearly non-separable problem can be reduced to linearly-separable case [61]. The decision boundary is then perpendicular to the nearest points between the two RCHs derived from training samples.

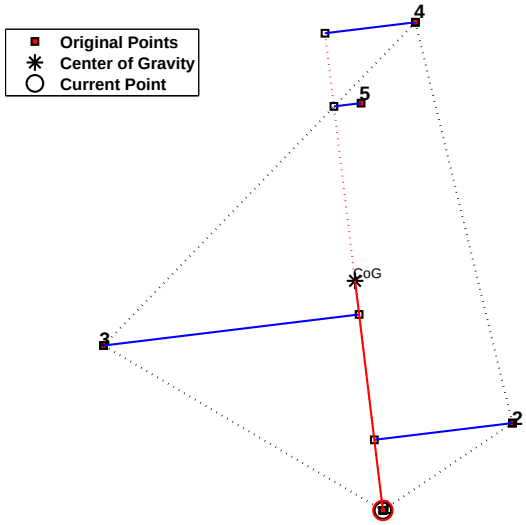
Our method uses the RCHs and defines the approximate skin segments of convex hulls. The intuition is that only the samples within the skin are retained in training. When additional samples are provided, they will be used together with the skin of the convex hull constructed from previous dataset. Therefore, much less number of instances is used in training process. However, with a superset of the possible SVs retained, missing SVs due to significant change of data distribution by adding new examples can be circumvented.

3.3. Geometric Incremental Support Vector Machine

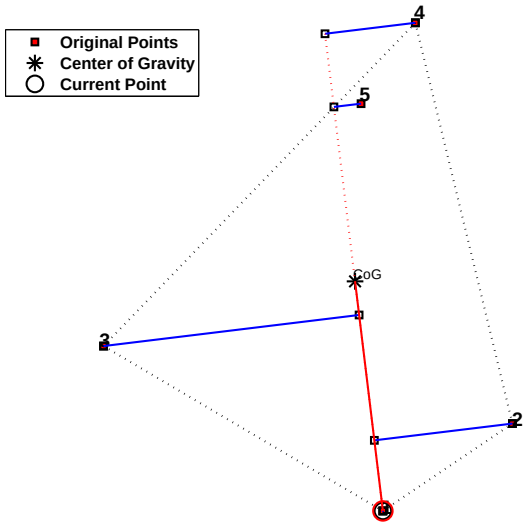
SVMs find the decision plane as a hyper-plane and is represented in terms of samples which are support vectors. The problem of finding the samples to retain in Geometric Incremental Support Vector Machine(GISVM) is manageable only if the chosen method is able to make all calculation using the kernel function instead of mapping the points explicitly in the feature space. The Kernel functions can be used by writing all formulations in terms of inner products of data points, which

can be later replaced by kernel functions evaluations to obtain final feature space formulations [68].

3.3.1. Skin of Convex Hull



(a)



(b)

FIGURE 3.2. Finding seed points using center of gravity

The skin of a convex hull consists of the out-most vertices (or samples). Given two bounds μ_u and μ_l, $0 < \mu_l < \mu_u \leq 1$, the skin, $S(\Phi, \mu_l, \mu_u)$, of the convex hull $C(\Phi)$ is the set of instances between two RCHs and can be expressed as follows:

$$S(\Phi, \mu_l, \mu_u) = \{\phi_i | \phi_i \in \{R(\Phi, \mu_u) - R(\Phi, \mu_l)\}. \tag{16}$$

Finding the skin of convex hull is nontrivial due to the lack of knowledge of the data distribution. Katagiri and Abe used hyper-sphere to approximate the convex hull and selected boundary samples for incremental learning [43]. We propose a recursive method that finds the vertices (i.e., extreme points) of a convex hull, which are used to represents the skin.

Given a set of instances Φ, $\phi_j \in \Phi$ is an extreme point of $C(\Phi)$ if there exists a direction $d = \phi_b - \phi_a$, $\phi_a, \phi_b \in C(\Phi)$ such that

$$\phi_j = \arg\max_{\phi_k \in \Phi} P(\phi_k, d), \tag{17}$$

where $P(\phi_k, d)$ denotes the projection of ϕ_k to d. The projection of a vector ϕ_k to a direction $d = \phi_b - \phi_a$ is defined as the inner product of the two difference vectors with respect to ϕ_a (the reference vector):

$$P(\phi_k, d) = \langle \phi_k - \phi_a, d \rangle. \tag{18}$$

Extreme points are found in two steps: 1) a set of seed points are identified based on the center of gravity; and 2) the set of extreme points are then found via recursively searching along the direction defined by a pair of extreme points.

The seed points identified in the first step are the extreme points of the convex hull. For a set of feature vectors Φ, the gravity center, $\bar{\Phi}$, is approximated with the average of the samples in Φ, i.e., $\bar{\Phi} = \sum_{i=1}^{n} \alpha_i \phi_i$ and $\alpha_i = \frac{1}{n}$. The initial set of extreme points, $E \subseteq \Phi$, are identified by projecting each point $\phi_j \in \Phi$ to the direction $d(\phi_m) = \phi_m - \bar{\Phi}$ and selecting the ones that give the maximum projection:

$$E_{\text{Seed}}(X) = \{\phi_n | \arg\max_{\phi_n} P(\phi_n, d(\phi_m)) \forall \phi_m, \phi_n \in \Phi\}. \tag{19}$$

where $P(\phi_n, d(\phi_m))$ denotes the projection of ϕ_n to $d(\phi_m)$.

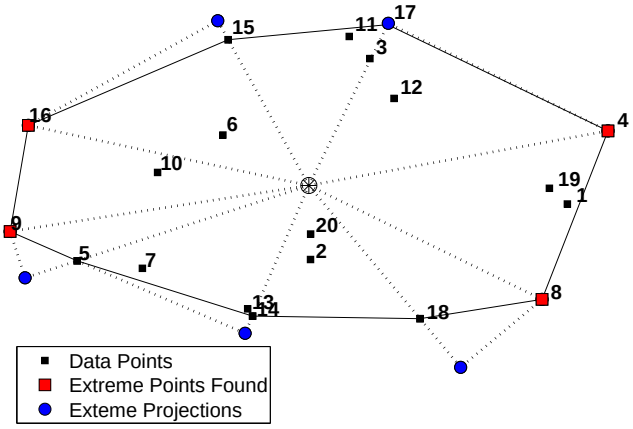
Knowing the explicit expression of the feature vectors ϕ_i is unnecessary to compute the extreme points in the above procedure. The projection $P(\phi_n, d(\phi_m))$ in the feature space can be achieved with the kernel operation in the input space as follows. Given two feature vectors ϕ_a and ϕ_b in Φ, the projection of vector ϕ_c is $P(\phi_c, d(\phi_a, \phi_b))$:

$$\begin{aligned}
P(\phi_c, d(\phi_a, \phi_b)) &= \langle \phi_b - \phi_a, \phi_c - \phi_a \rangle \\
&= \langle \phi_b, \phi_c \rangle - \langle \phi_b, \phi_a \rangle - \langle \phi_a, \phi_c \rangle + \langle \phi_a, \phi_a \rangle \\
&= \sum_i b_i \phi_i \cdot \sum_j c_j \phi_j - \sum_i b_i \phi_i \cdot \sum_j a_j \phi_j \\
&\quad - \sum_i a_i \phi_i \cdot \sum_j c_j \phi_j - \sum_i a_i \phi_i \cdot \sum_i a_i \phi_i \\
&= \sum_i \sum_j b_i c_j K(x_b, x_c) - \sum_i \sum_j b_i a_j K(x_b, x_a) \\
&\quad - \sum_i \sum_j a_i c_j K(x_a, x_c) - \sum_i \sum_i a_i a_i K(x_a, x_a)
\end{aligned}$$

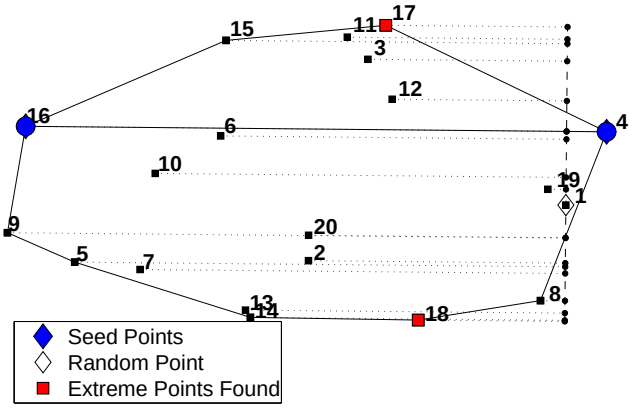
where $\sum_i a_i \phi_i$, $\sum_i b_i \phi_i$, and $\sum_i c_i \phi_i$ are convex representations of feature vectors ϕ_a, ϕ_b, and ϕ_c, respectively. For an vector $\phi_j \in \Phi$, its coefficient vector equals $[0, 0, \ldots, 1, \ldots, 0, 0]'$, which the index of value 1 is j. For a vector $\phi_k \notin \Phi$ but $\phi_k \in C(\Phi)$, the values in its coefficient vector are in the range of $[0,\ 1)$, e.g., coefficient vector of the gravity center $\bar{\Phi}$ is $[\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n}]'$

When the sample data are dense enough and evenly distributed in the region, the geometric center can be used to find the extreme points of the convex hull. However, this is usually not the case in real-world applications. Due to the lack of knowledge of data distribution, the above procedure could miss less prominent extreme points and find a subset of of the extreme points.

An example is illustrated in Figure 3.3(a). The solid squares denote the data samples and the gravity center is marked with a large circle. The projected vectors are marked with solid dots. Using our method, three extreme points are identified and highlighted with solid squares. For example, point 16 is identified as an extreme point since it gives the greatest projection to $d(x_{16}, \bar{X})$ (as well as $d(x_{15}, \bar{X})$). However, instances 14, 15, 17 and 18 are the extreme points but are missed by the process.



(a)



(b)

FIGURE 3.3. Finding extreme points recursively.

The primary cause of missing extreme points is the insufficient number of examples, which could be exaggerated in high dimensional cases. If data points in feature space are known, classical algorithms such as QuickHull [8] and Gift Wrapping [71] can be used to complete the searching. The idea of our algorithm is to recursively search along the perpendicular directions of the convex hull boundaries, which is lists in Algorithms 1 and 2.

Our algorithm randomly selects two extreme points $\phi_p, \phi_q \in E$ and another instance $\phi_m \in \Phi$. The perpendicular searching direction d^* can then be determined as follows:

Algorithm 1 Search for extreme points

Require: Φ and E
1: Randomly select $\phi_p, \phi_q \in E$
2: Randomly select $\phi_m \in \Phi$ and $m \neq p, m \neq q$
3: Identify probing direction d^* using Eq. (20)
4: $\Phi_- \leftarrow \{\phi_i | P(\phi_i, d^*) < 0\}$
5: $\Phi_+ \leftarrow \{\phi_i | P(\phi_i, d^*) > 0\}$
6: $E \leftarrow E \bigcup \text{Probing}(\Phi_+, d^*, \phi_p, \phi_q)$
7: $E \leftarrow E \bigcup \text{Probing}(\Phi_-, -d^*, \phi_p, \phi_q)$
8: **return** E

$$d^* = \phi_m - \phi_p - P(\phi_m, d(\phi_p, \phi_q)) \frac{\phi_q - \phi_p}{||\phi_q - \phi_p||} \tag{20}$$

Hyper-plane through $\phi_q - \phi_p$, and perpendicular to d splits the space into two halves. The projections of instances, i.e., $P(\phi_i, d^*)$, that are on the same sides as ϕ_m are positive, denoted with Φ_+; whereas the projections of the rest instances are negative, denoted with Φ_-. Hence, the further searching for extreme points is divided into two parts, as shown in Algorithm 1.

Searching in each half space is achieved recursively using a pair of identified extreme points, ϕ_p and ϕ_q. Let Φ' denote the instances in the half space. With a random instance ϕ_m in Φ', a probing direction, d^*, can be determined using Eq (20). d^* points toward the outside of the convex hull; Otherwise, change its direction. Hence, an extreme point is identified in Φ' following Eq. 17. ϕ_m is paired with ϕ_p and ϕ_q to split the feature space for further probing. The process stops when no additional points exist in Φ'.

Fig. 3.3(b) illustrates an example of probing in a half space. The two extreme points are 17 and 18, which determines the probing direction (d and $-d$ in Algorithm 1). The dotted lines depict the projections of the instances. In two iterations, extreme points 1 and 12 are found.
In the algorithm the length of vectors $\phi^{(2)} - \phi^{(1)}$ is found using,

$$||\phi^{(2)} - \phi^{(1)}|| = \left[\langle \phi^{(1)}, \phi^{(1)} \rangle + \langle \phi^{(2)}, \phi^{(2)} \rangle - 2\langle \phi^{(2)}, \phi^{(1)} \rangle\right]^{1/2}$$

$$= \left[\sum_i \sum_j \beta_i^{(1)} \beta_j^{(1)} K(x_i, x_j) + \sum_i \sum_j \beta_i^{(2)} \beta_j^{(2)} K(x_i, x_j) - 2 \sum_i \sum_j \beta_i^{(1)} \beta_j^{(2)} K(x_i, x_j)\right]^{1/2}$$

Algorithm 2 Recursively probing and search for the extreme points. Probing$(\Phi', d, \phi_p, \phi_q)$

Require: $\Phi' \subseteq \Phi$, d, ϕ_p, and ϕ_q
$F \leftarrow \emptyset$
Randomly select $\phi_m \in \Phi'$ and $m \neq p, m \neq q$
if $\Phi' \neq \emptyset$ **then**
  Identify probing direction d^* using Eq. (20)
  if $\langle d^*, d \rangle < 0$ **then**
    $d^* \leftarrow -d^*$
  end if
  $d^* \leftarrow \frac{d'^*}{||d^*||}$
  $F \leftarrow F \bigcup \{\phi_e | \phi_e = \arg\max_{\phi_k \in \Phi'} P(\phi_k, d^*)\}$
  for all $\phi_i \in \Phi'$ **do**
    if $P(\phi_i, d^*) > 0$ **then**
      $\Phi'' \leftarrow \Phi'' \bigcup x_i$
    end if
  end for
  $F \leftarrow F \bigcup \text{Probing}(\Phi'', d, \phi_p, \phi_e)$
  $F \leftarrow F \bigcup \text{Probing}(\Phi'', d, \phi_q, \phi_e)$
end if
return F

Range of the projections : $R_{X,d}$ of a set of samples X in a given directions $d = x^{(1)} - x^{(2)}$, $x^{(j)} = \sum_i x_i \beta_i^{(j)}$ for $j = 1, 2$ would be

$$R_{X,d} = \left[\min_{x_i \in X} \{P_{x_i,d}\}, \max_{x_i \in X} \{P_{x_i,d}\} \right]$$

Skin Segment : Given a set $X = \{x_1, \cdots, x_n\}$, the skin segment with an angle θ around $d = x^{(1)} - x^{(2)}$, is

$$SS_{X,d} = \left\{ x_i | x_i \in E(X), \cos^{-1} \frac{\langle d, x^{(2)} - x_j \rangle}{||d|| \cdot ||x_i^{(2)} - x_j||} \leq \theta \right\}$$

The angle θ_j between vectors $x_j - x^{(2)}$ and $d = x^{(1)} - x^{(2)}$ is,

$$\theta_j$$

$$\theta_j = \cos^{-1} \frac{\langle d, x^{(2)} - x_j \rangle}{||d|| \cdot ||x_i^{(2)} - x_j||}$$

The angle θ_j between vectors $x_j \in X_i$ and center of gravity $x_i^{(g)}$ and $w = x_+^* - x_-^*$ can be found

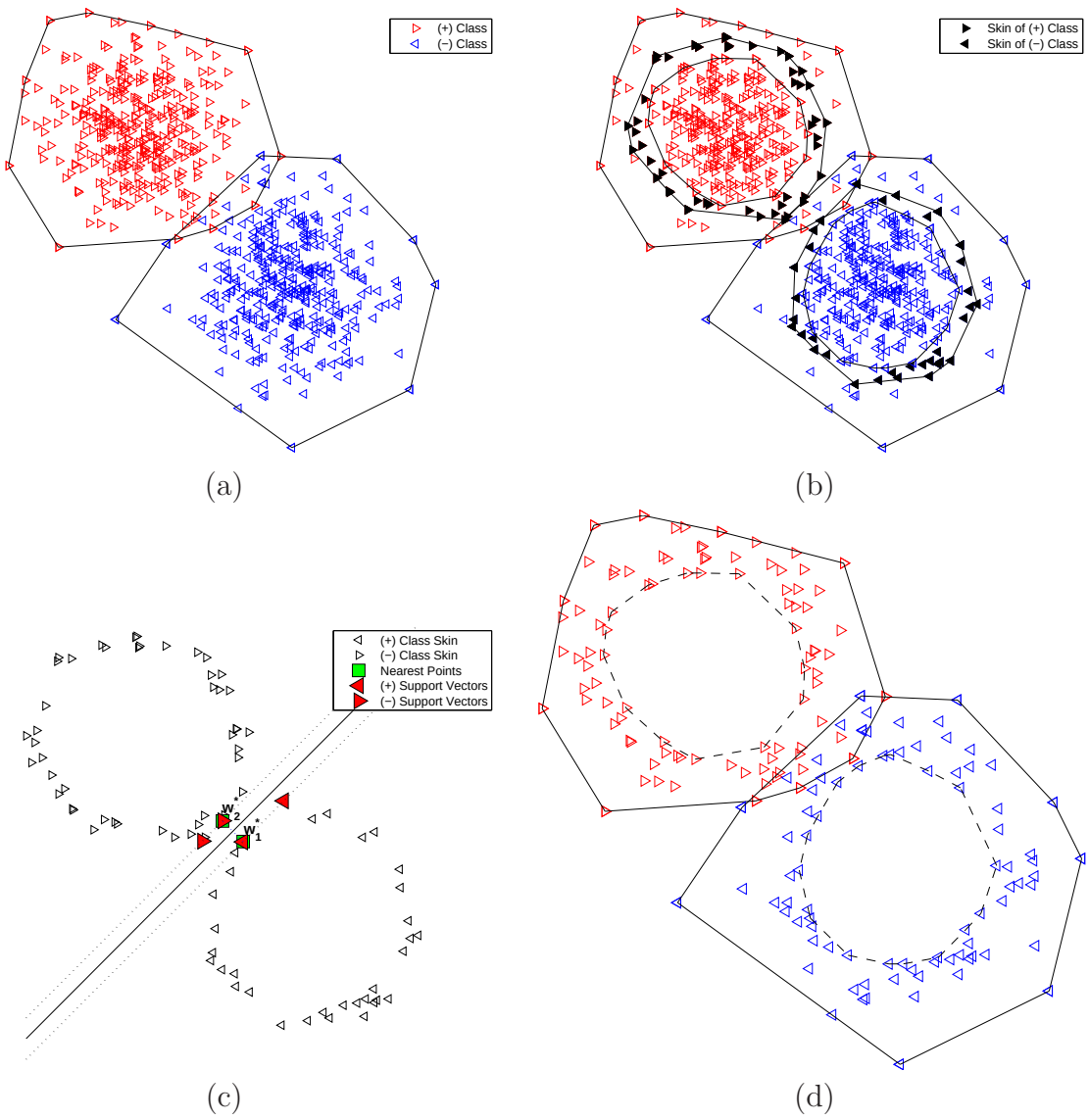


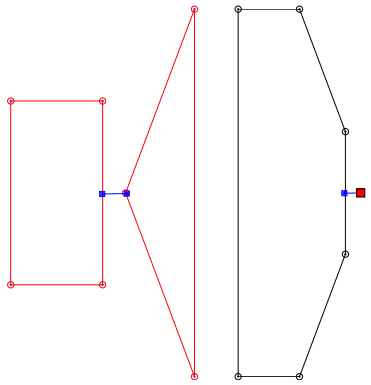
FIGURE 3.4. (a) Two linearly non-separable classes consisting of 700 data points (b) The classes are linearly separable using $R(\mu = 0.1)$ (c) Decision boundary is found by finding the nearest points between the $R(\mu = 0.1)$ of both classes (d) The $S(x_i, 0.1, 1)$ is retained for future iterations

using,

$$\langle w, x_i^{(g)-x_j} \rangle = ||w|| \cdot ||x_i^{(g)} - x_j|| \cos\theta$$

$$\theta = \cos^{-1} \frac{\langle w, x_i^{(g)} - x_j \rangle}{||w|| \cdot ||x_i^{(g)} - x_j||}$$

The decision boundary is perpendicular to $w_2^* - w_1^*$, where w_1^*, w_2^* are the nearest points



(a) (b)

FIGURE 3.5. Minkowski Set

between the RCHs. Decision boundary would be represented by w_0, w, where midpoint of w_1

3.3.2. Determining separability of two classes.

The nearest point algorithms find the nearest points between two hulls that are non-intersecting. Formally if two classes X_+, X_- are linearly separable if there exists a direction $d = (\hat{x}_+ - \hat{x}_-)$, where $\hat{x}_+ \in X_+$, $\hat{x}_- \in X_-$ such that,

$$\left[\frac{d}{|d|} \cdot (\hat{x}_+ - x_+)\right] d \cap \left[\frac{d}{|d|} \cdot (\hat{x}_- - x_-)\right] d = \emptyset$$

$$\forall x_+ \in X_+, x_- \in X_-$$

In other words there exists a direction d such that the projection of all the points of both classes are linearly separable. One possible approach would be to check in the pairs of all data points of the classes, for each direction project all points and find separability of the two classes. This would result in $O(n^3)$ projections and comparisons.

3.3.3. Constructing SVM by Finding the Nearest Points

For simplicity, the two classes are denoted with $\Phi_+ = \{\phi_i : y_i = 1\}$ for the positive class, and $\Phi_- = \{\Phi_i : y_i = -1\}$ for the negative class. The learning task maps input instances into one of the classes according to the training set $\{(x_1, y_1), \cdots (x_n, y_n)\}$:

$$f : \mathbb{R}^N \to \{-1, +1\}$$

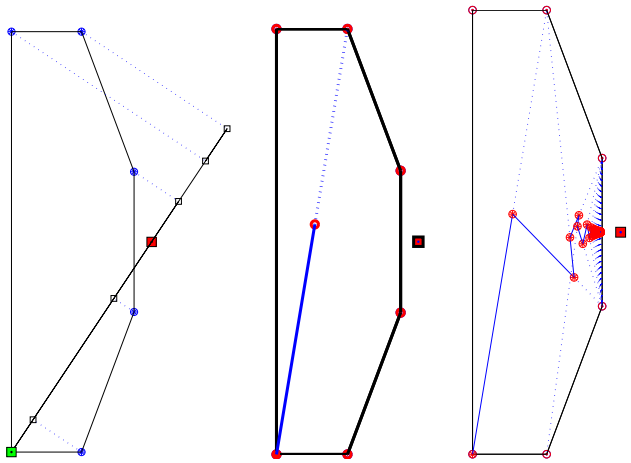


FIGURE 3.6. (a) Illustrates the Line 5 of Gilbert's algorithm, z^* is denoted using solid green square, and all points of convex hull are projected on to z^* (b) Illustrates the Line 6 of Gilbert's algorithm of finding the nearest point to origin on segment $[z^* z]$ (c) The solid blue line denotes the path of z^* of Gilbert's algorithm as, z^* converges towards the nearest point to origin.

Algorithm 3 Finding the nearest points of two convex hulls [32]

1: $Z \leftarrow \{\phi_+ - \phi_- | \phi_+ \in \Phi_+, \phi_- \in \Phi_-\}$
2: Randomly select $z^* \in C(Z)$
3: **repeat**
4: $\quad z^*_{old} \leftarrow z^*$
5: $\quad z \leftarrow \arg\min_{z_i \in Z} P(z_i, z^*)$
6: $\quad z^* \leftarrow \arg\min_{z'} ||z'||$ where $z' \in [z^*_{old}\ z]$
7: **until** ($||z^* - z^*_{old}|| \approx 0$)

where $x_i \in \mathbb{R}^N$, $y_i \in \{-1, +1\}$.

The quadratic and geometric solutions of SVM have been shown to be equivalent [18, 10, 22, 86, 44]. In geometric approach to SVM, the two classes are represented by their convex hulls and the nearest points are found. The SVM is the hyper-plane that perpendicular to the connection of the nearest pair of points, denoted with (ϕ^*_+, ϕ^*_-), in the two convex hulls, i.e.,

$$(\phi^*_+, \phi^*_-) = \arg \min_{\phi_+ \in \Phi_+, \phi_- \in \Phi_-} (||\phi_+ - \phi_-||) \tag{21}$$

where $\phi^*_+ \in \Phi_+, \phi^*_- \in \Phi_-$, are found using the Gilbert's algorithm [32]. The nearest point problem is equivalent to finding the minimum norm problem (or the closest to the origin) of the Minkowski

difference set. The solution of SVM is hence becomes finding w^*:

$$w^* = \arg\min_{z \in Z}(||z||) \tag{22}$$

where $Z = \{\phi_+ - \phi_- | \phi_+ \in \Phi_+, \phi_- \in \Phi_-\}$. It is assumed that the convex hulls are disjoint.

The algorithm starts by randomly selecting an initial point $z^* \in Z$. Next the point $z \in Z$ is found from all $z_i \in Z$, where z makes the minimum projection on z^*. The current value of z^* is stored in z^*_{old}, the new value for nearest point z^* is updated with the closest point to origin on the segment $[z^*_{old}\ z]$. Nearest point between the origin and segment $[z^*_{old}\ z]$, can be found using,

$$[z^*_{old}, z] = \left\{ \begin{array}{ll} z^*_{old} & \text{if } -z^*_{old}.(z - z^*_{old}) \leq 0 \\ z & \text{if } ||z - z^*_{old}||^2 \leq -z^*_{old} \cdot (z - z^*_{old}) \\ z^*_{old} + \frac{-z^*_{old}.(z - z^*_{old})}{||z - z^*_{old}||^2}(z - z^*_{old}) & otherwise \end{array} \right\}$$

By choosing z to have the minimum projection, allows z^* to move to closer to the origin, while at the same time for z^* to remain within the enclosing convex hull. Finding z the minimum projection onto z^*, and moving z^* along segment $[z^*_{old}\ z]$ is repeated until z^* does not change.

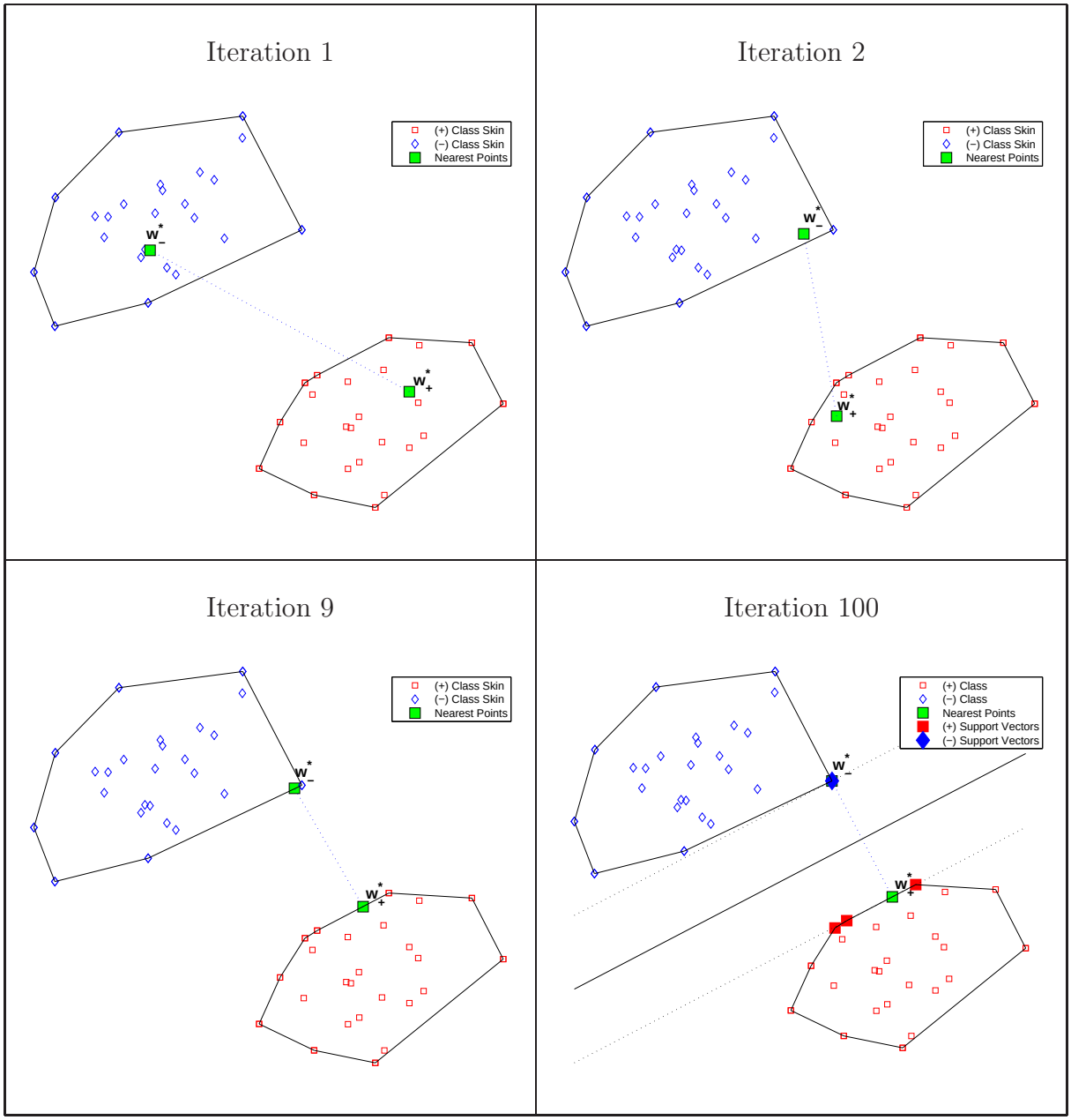


FIGURE 3.7. Steps of Gilbert algorithm, and the decision boundaries between the two classes found after 100 iterations.

CHAPTER 4

EXPERIMENTAL RESULTS

In our experiments, both synthetic and real-world benchmark datasets are used to evaluate the proposed method. Synthetic data are computer generated in the two-dimensional domain, and benchmark data are from the UCI [7] repository.

The proposed Geometric Incremental Support Vector Machine(GISVM) was compared with libSVM [16]. The GISVM is implemented in Matlab, a C++ implementation of libSVM was downloaded from the authors site. The experiments were conducted on Intel Dual Core PC with 4GB memory running Ubuntu operating system. The parameters for the SVMs including the kernel, for GISVM was handpicked, and the libSVM parameters were determined by optimization of libSVM.

Training vectors x_i are mapped into a higher dimensional feature space by the function ϕ. The decision boundaries are found in feature space where the classes are linearly separable. SVM finds a linear separating hyper-plane with the maximal margin. $C > 0$ is the penalty parameter of the error term.

$K(x_i, x + j) = \phi(x_i)\phi(x_j)$ is called the kernel function.

The following four kernels were used in the experiments reported:

- Linear Kernel : $x_i \cdot x_j$
- Polynomial Kernel : $(x_i \cdot x_j + 1)^\lambda$
- Radial Basis Function (RBF) Kernel : $e^{\gamma|x_i - x_j|^2}$
- Sigmoid : $\tanh(\gamma x_i \cdot x_j + \text{coefficient})$

The RBF is one of the most popular choice of kernel types used in SVMs. This is mainly because of their localized and finite responses across the entire range of the real x-axis [66]. In addition, the Sigmoid kernel behaves like RBF for certain parameters[54]. Further, the number of hyper-parameters which influences the complexity of model selection, whereas the polynomial kernel has more hyper-parameters than the RBF kernel.

4.1. Evaluation Metrics

For quantitative evaluation, we adopted measures of sensitivity and specificity are as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + TN + FP}$$
$$\text{Specificity} = \frac{TN}{TN + FP}$$
$$\text{Specificity} = \frac{TN}{TN + FP}$$

where

- TP - Number of positive examples that are correctly labeled,
- TN - Number of negative examples that are correctly labeled,
- FP - Number of negative examples that are labeled as positive,
- FN - Number of positive examples that are labeled as negative.

Incremental Learning methods should be robust and reliable [76] this can be identified using stability, improvement, and Recoverability.

Stability: of the prediction accuracy; this should not significantly vary over incremental steps.

Improvement: of the prediction accuracy as the training progresses; when more training examples are seen.

Recoverability: the learning method should be able to recover from its errors. Even if accuracy drops at a certain step, should be able to recover to previous states.

4.2. Synthetic Dataset

The synthetic datasets are generated to visualize the decision boundaries and to compare the underlying distribution. The block and the disk datasets are linearly separable, and the Gaussian data sets are non-linearly separable in kernel space as well (Fig 4.1). The block and Gaussian datasets are created in two fashions 1) Simple 1×2 - where the block datasets are linearly separable in the feature space, however the Gaussian is not linearly separable due to overlap. 2) XOR - The

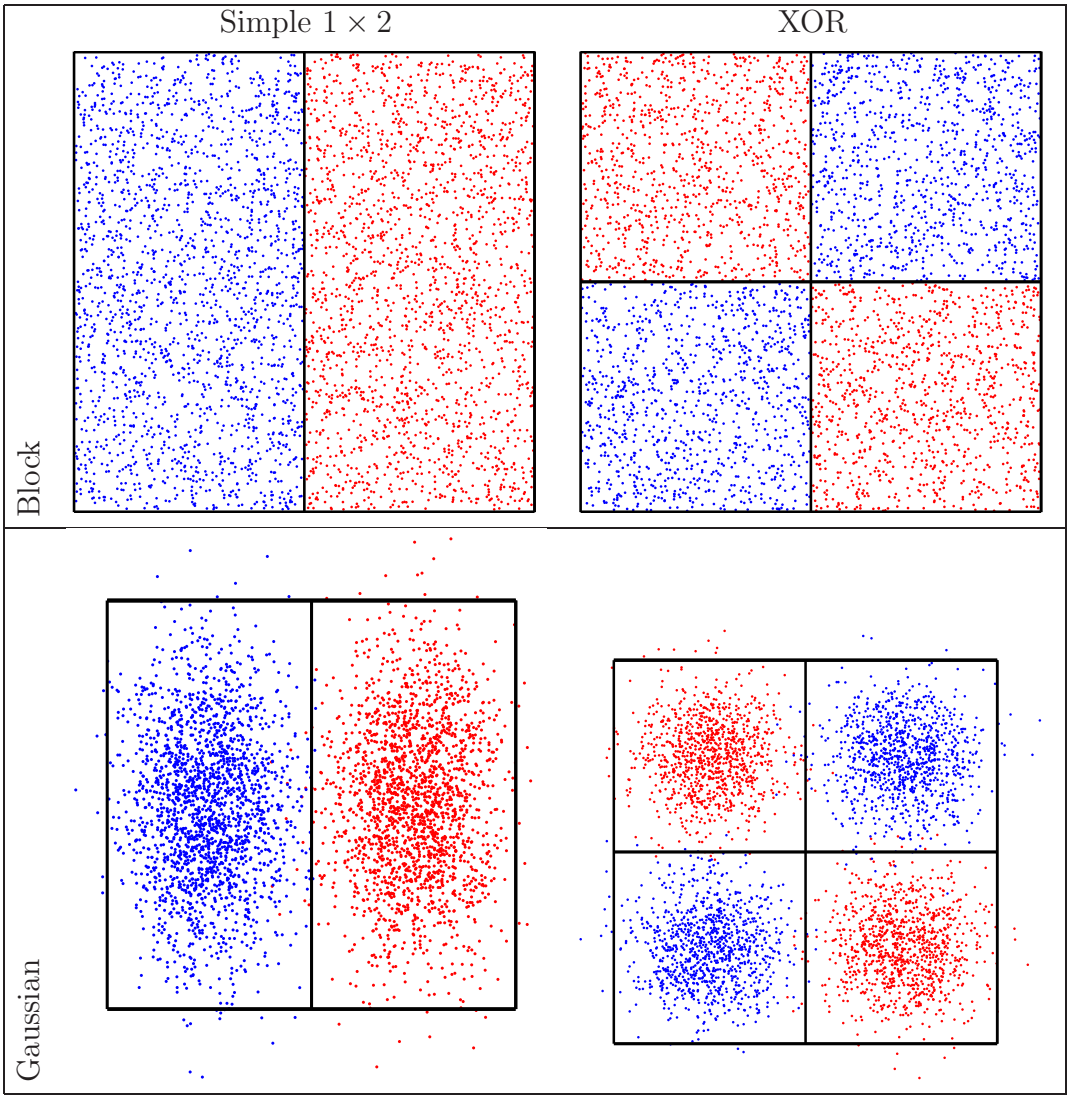


FIGURE 4.1. Synthetic Datasets

two datasets are created as the classic XOR problem. Where the block XOR dataset can be linearly separable in a feature space, but not the Gaussian dataset due overlap.

Each of the synthetic dataset was created using 200 randomly generated instances for each class. Each attribute ranged from [0 1], and equally divided among the cells in the attribute, after leaving a 0.01 separation between the cells for the non-overlapping classes. For the block dataset the number of samples for each cell in XOR was set as equal. The Gaussian datasets were created by randomly sampling Gaussian functions, with the underlying distributions having overlap by 1%. Success rate was estimated by generating ten sets of samples were randomly generated using each model, and testing on a test set of 200 instances uniformly sampled in a grid fashion.

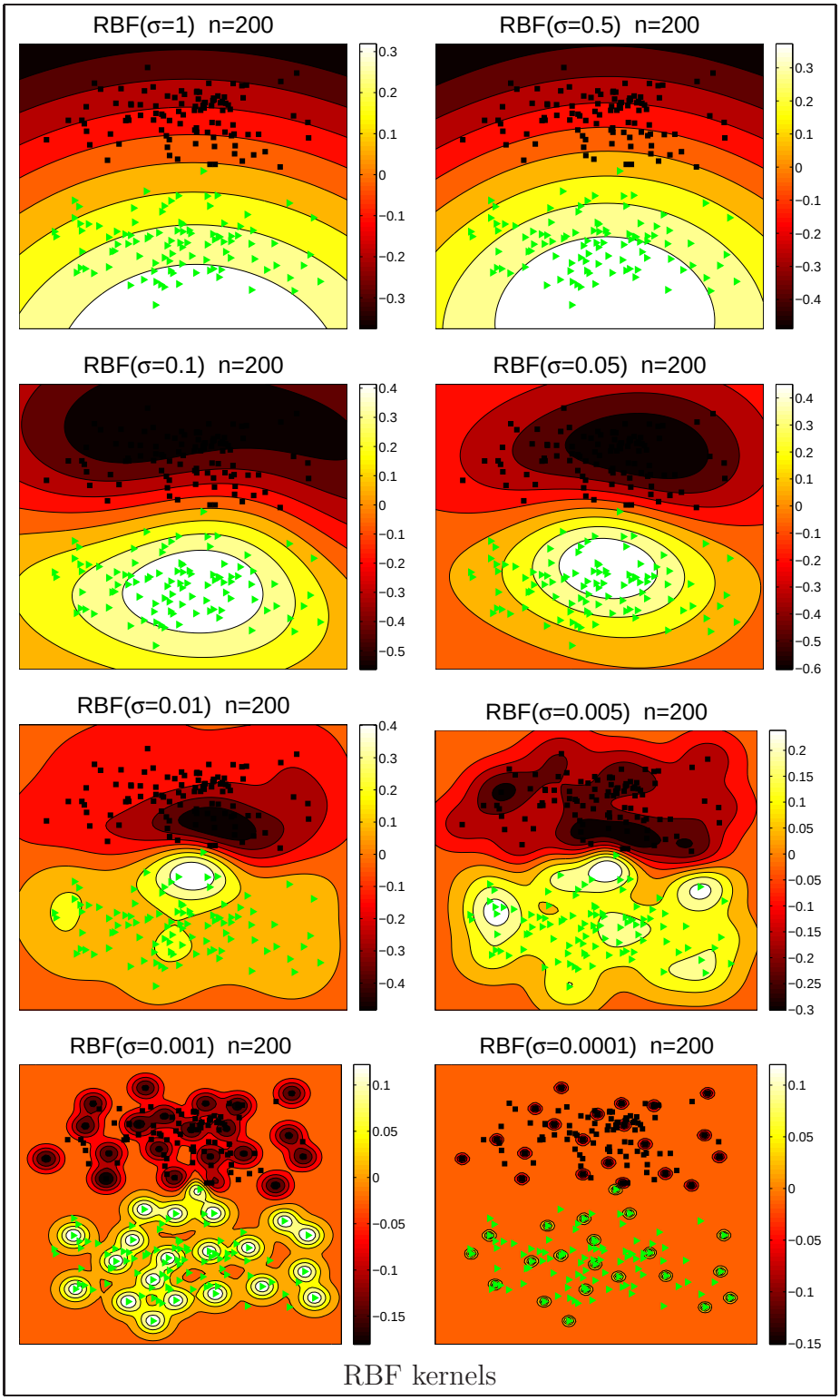


FIGURE 4.2. Decision boundaries of the classifiers for the 1×2 Gaussian datasets using RBF Kernal.

Experiments were performed to visualize the 1. decision boundaries and compare with the underling distribution from which data was generated 2) compare performance with batch learning of libSVM.

Using our 2-D synthetic datasets, we plot the final classifiers using our method in Fig. 4.2-4.4. The data instances are marked with squares and triangles in the plots. The shade indicates the distance to the decision boundary. With the shades at 0 indicate, equal distance to both classes. The training started with randomly selected 10 samples and, at each iteration, 10 new samples from the training set were randomly selected and used.

Fig- 4.2,4.4 represent the behavior of kernels and their parameters on on the decision boundary found. In the experiments with both datasets, as we decrease the σ of the RBF kernel, the final classifier appears over fitting. Among all the kernels tested, RBF kernels with σ around 0.1 resulted in better decision boundaries than others. With appropriate kernels, it is evident that our method achieves satisfactory classifiers that closely model the underlying data distribution.

Fig. 4.6(a) and (c) illustrate the maximum number of samples retained (i.e., the samples in the skin of the RCHs) from both classes as the function of σ using the RBF kernels. The total training examples are 200 in both Gaussian and XOR datasets. With the decrease of σ value, the number of samples retained significantly increased. In both cases, when σ reaches 0.01, the number of samples retained is approximately the total number of samples used in the training. This indicates possible over-fitting of the model, which is also depicted in Fig. 4.3,4.2,4.4. The number of SVs changes much slightly. The good choice for σs for the synthetic datasets is 0.1, which provides a good description of the datasets and retains much less number of samples in the training iterations.

Fig. 4.6(b) and (d) show the number of samples retained from both classes during the training iterations. With properly chosen σs, the number of samples retained is small, which implies the stability of the incremental learning process. With all training examples exposed to the learner, the number of retained samples is only a fraction of the total samples used. That is, a much smaller amount of memory was used to complete the training. In case of batch learning, all examples are loaded in to memory at one time for training. Hence, it is evident that our incremental learning

method is memory efficient.

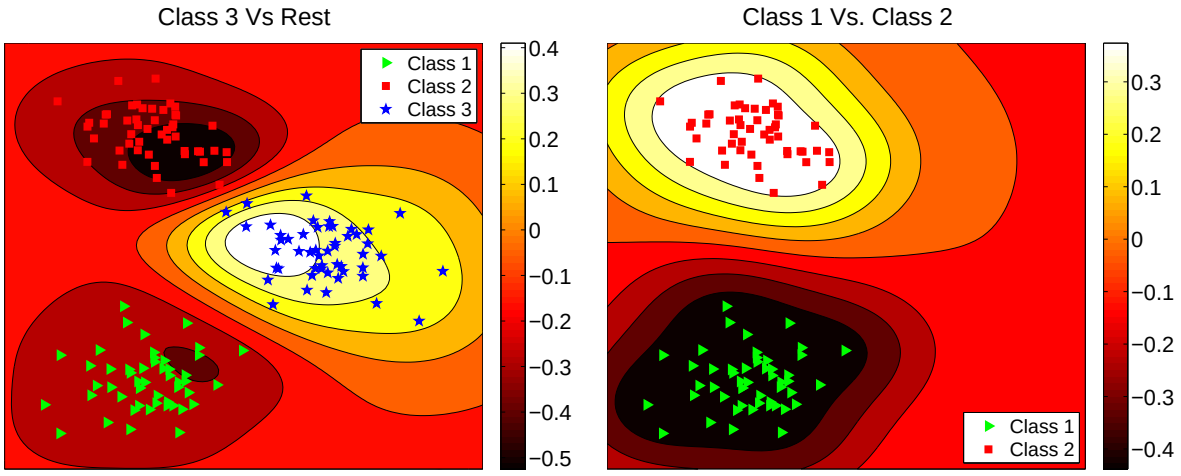
Table 4.2 lists the number of seconds it takes by our method and batch learning-based SVM to complete training and updating. Ten repetitions were conducted and the average times are reported. Because random samples were used, training time varies and the standard deviation is listed in the parenthesis. The incremental learning simulates the case that new examples, when they become available, are used to update the classifier. Assuming equal number of examples are used to update classifiers, which is referred to as step size Δ. The time reported for incremental learning is the average iteration time to complete the learning. If batch-learning strategy is used, the new examples are used together with the existing ones to retrain the classifier. The times reported for batch learning are the ones it took to learning from all examples.

Because of the small number of examples available in some datasets, we were unable to conduct experiments with larger step size. Based on the two incremental learning cases ($\Delta = 10$ and $\Delta = 20$), we can see that our incremental learning handles data efficiently and can update the classifier in much shorter time. The average time cost for our method to complete is approximately 13.4% of the time cost for batch learning SVM.

It is an interesting observation that larger step size does not necessarily result in longer training time. This is probably due to the fact that the much less number of examples (i.e., examples lie in the skin of the RCHs) were carried to the next round of training. That is, the training complexity is mostly determined by a subset of training examples.

In our experiments, we use hierarchical SVM to perform multi-class classification. In the first level one-against-all is for all classes is performed to find the class which discriminates from the rest. The first level classifier is used to classify the class identified. In the next level the one-against-all for remaining classes is repeated to find the class which discriminates the rest of the remaining classes. This is repeated until all classes can be classified.

Figure 5.6 shows results or classifying three classes. Since there are three classes there would be two levels of classifications. The decision boundary in Figure 5.6 is denoted by the zero level contour. Given a test example the first level classifier would be used to test for class 3, and the second level would be used to classify class 1, 2.



(a) (b)

FIGURE 4.7. Results of hierarchical multi class classification. Contours of decision boundaries of (a) Class 3 Vs Class(1&2) (b) Class(1)

4.3. Benchmark Dataset

Five real-world datasets were obtained from UCI machine learning repository [7], which contain no missing values. In addition, a mammography[17] dataset is used. For the multi-class datasets, one class is selected as the positive class and the rest are grouped into the negative class. Each feature in a dataset was normalized with the mean value and the range of that feature. Hence, the instances are in the range of [0, 1]. Table 4.1 lists the properties of the datasets used in our experiments.

To evaluate the performance of our method, sensitivity and specificity are used as metrics as well as accuracy. Fig. 4.8-4.12 illustrates the classifier performance based on these three metrics during the incremental learning iterations. Ten repetitions were conducted with random initial training set. For each dataset, 50% of the data were randomly selected and used for training. The remaining data were used as the test set.

In each case, a SVM classifier was created using all the training data. The best parameters were selected based on their generalization performance with the testing dataset. Table 4.3 lists the selected kernels and parameters that gave the best performance measures. The results from these classifiers are used as reference and are depicted as horizontal lines in Fig. 4.8-4.12.

In our incremental learning process, 10 samples were randomly selected from each class of

TABLE 4.1. Properties of the benchmark datasets used in the experiments.

Name	Dim.	Classes	Sample Size		
		(+) Class	Total	(+) Class	(-) Class
IRIS	4	Setosa	150	50	100
SPECT	22	1	267	212	55
PIMA	8	1	768	268	500
YEAST	8	CYT	1,484	463	1,021
IONOSPHERE	34	b	351	126	225
MAMMOGRAPHY(ISM)	6	2	11,183	260	10,923

TABLE 4.2. Training time (in second) used in our incremental and the conventional batch learning. The results show the average over five repetitions. The standard deviation is in the parenthesis.

	Batch Learning SVM			
Datasets	Training Example	Time		
Gaussian	Training Example	2.9 (0.4)		
XOR	Training Example	5.2 (0.4)		
iris	Training Example	0.65 (0.1)		
SPECT	Training Example	2.7 (0.3)		
yeast	Training Example	97.2 (12.5)		
pima	Training Example	71.4 (7.9)		
ionosphere	Training Example	3.8 (0.5)		
Mammography	Training Example	2261(478)		
	Our Incremental Learning Method			
	$\Delta = 10$		$\Delta = 20$	
	Time	Iter.	Time	Iter.
Gaussian	0.2 (0.04)	19	0.21 (0.06)	9
XOR	0.34 (0.03)	19	0.54 (0.08)	9
iris	0.15 (0.02)	5	0.18 (0.03)	3
SPECT	0.35 (0.05)	11	0.25 (0.02)	6
yeast	3.59 (0.25)	73	5.16 (0.42)	37
pima	21.14 (0.25)	36	13.39 (1.34)	19
ionosphere	0.3 (0.04)	15	0.43 (0.05)	8
Mammography	2859(541)	445	1540 (338)	223

the training set and a SVM is trained. In each incremental step, randomly selected 10 samples from the remaining training dataset were used to update the classifier. The intermediate classifiers were evaluated with the test dataset. For each dataset, 10 repetitions were conducted and the average performance is plotted with solid line in Fig. 4.8-4.12. The shaded area depicts the range of performance in each training iteration.

TABLE 4.3. Parameters used for the experiments

Datasets	GISVM Kenal	μ_u	μ_l
IRIS	Linear	1	1
SPECT	RBF σ=0.05	0.9	0.6
PIMA	RBF σ=0.10	0.9	0.6
YEAST	RBF σ=0.15	0.5	0.3
IONOSPHERE	RBF σ=0.01	0.5	0.3
MAMMOGRAPHY	RBF σ=0.01	0.8	0.4

Name	Dim.	(+) Class	Sample Size Total	(+)	(-)	Average Memory Usage
IRIS	4	Setosa	150	50	100	36.4%
SPECT	22	1	267	212	55	44.8%
PIMA	8	1	768	268	500	39.5%
YEAST	8	CYT	1,484	463	1,021	26.4%
IONOS	34	b	351	126	225	28.1%
ISM	6	2	11,183	260	10,923	21.8%

TABLE 4.4. Memory Usage of for UCI datasets by GISVM

With more examples included in the training process, the classifier trained with our method improves its performance. It is evident in the cases of Yeast, SPECT, Pima, and Ionosphere. In the cases of Iris and Ism, the performance at the very beginning is already superior and there is not much of improvement space. Hence, the change of performance in the following iterations is minimum. However, improvement in sensitivity can still be observed in the training using Ism dataset and by the end of iterations, classifier outperformed the batch learning by a small margin.

Despite a slightly drop of specificity of the SPECT dataset, the SVMs trained with our method achieved the same performance or even outperformed the batch learning method. As listed in Table 4.1, the SPECT dataset contains more positive examples than negative examples. The ratio is approximately 4:1 between positive and negative classes. Hence, the improvement of sensitivity leverages the under-performance in terms of specificity and the overall accuracy is close to the batch training result. It is interesting that in five cases (except Iris), the intermediate classifier had a degradation in early iterations, but the training process was able to recover to the benchmark performance asymptotically as additional data instances are in-cooperated.

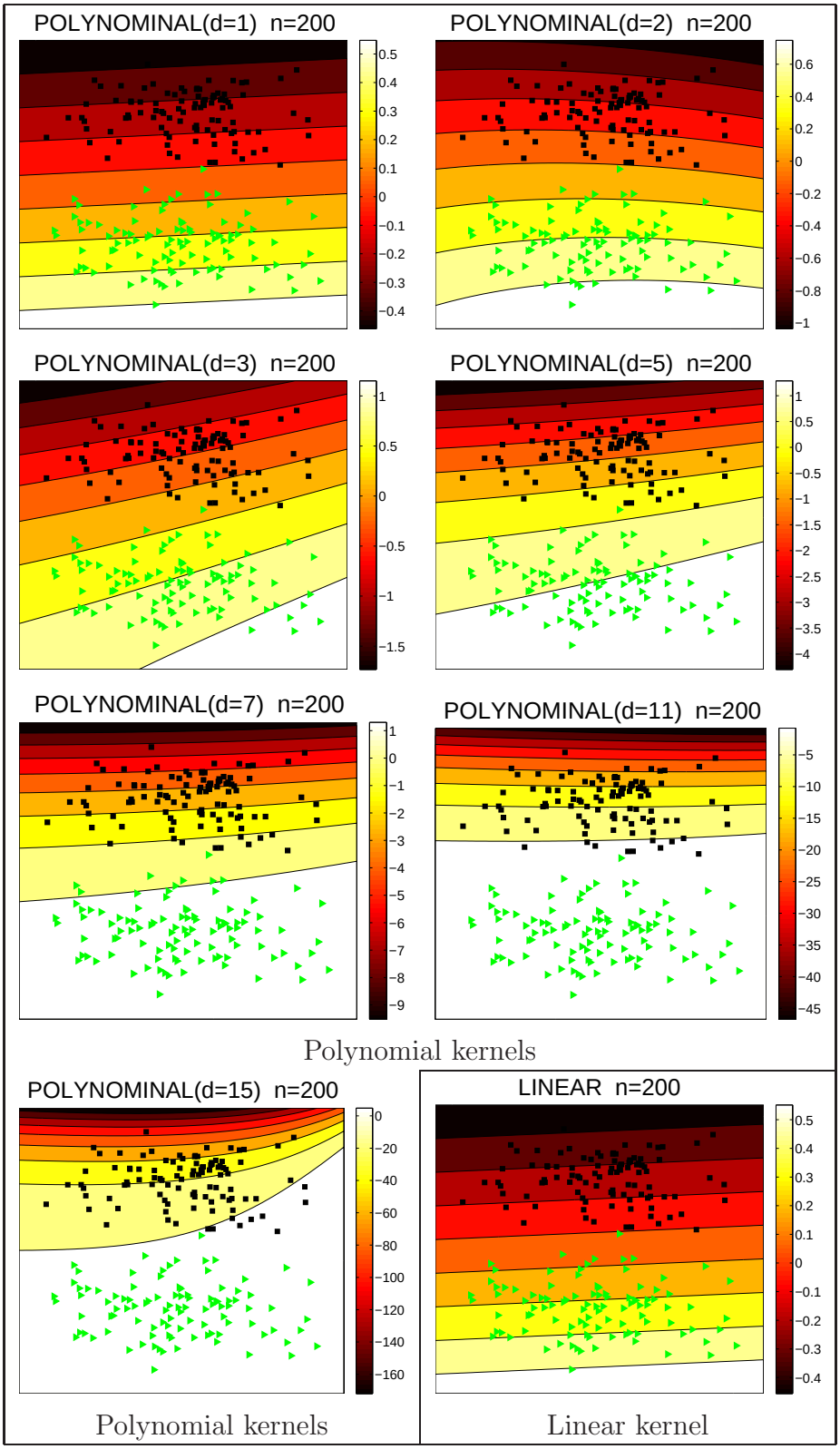


FIGURE 4.3. Decision boundaries of the classifiers for the 1×2 Gaussian datasets using Polynomial and linear Kernal.

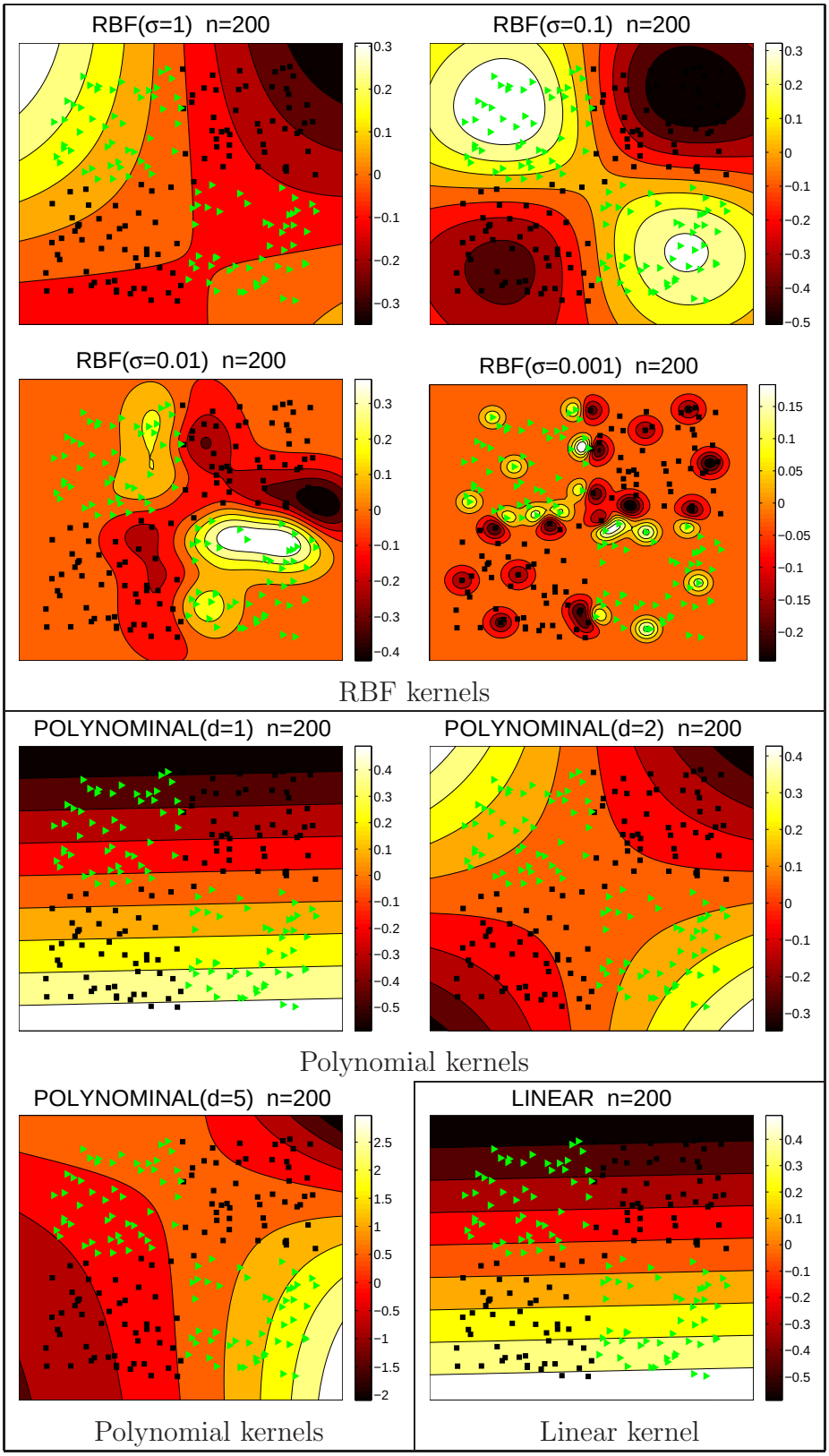


FIGURE 4.4. Decision boundaries of the classifiers for the XOR block datasets using RBF Kernal, Polynomial and linear kernels.

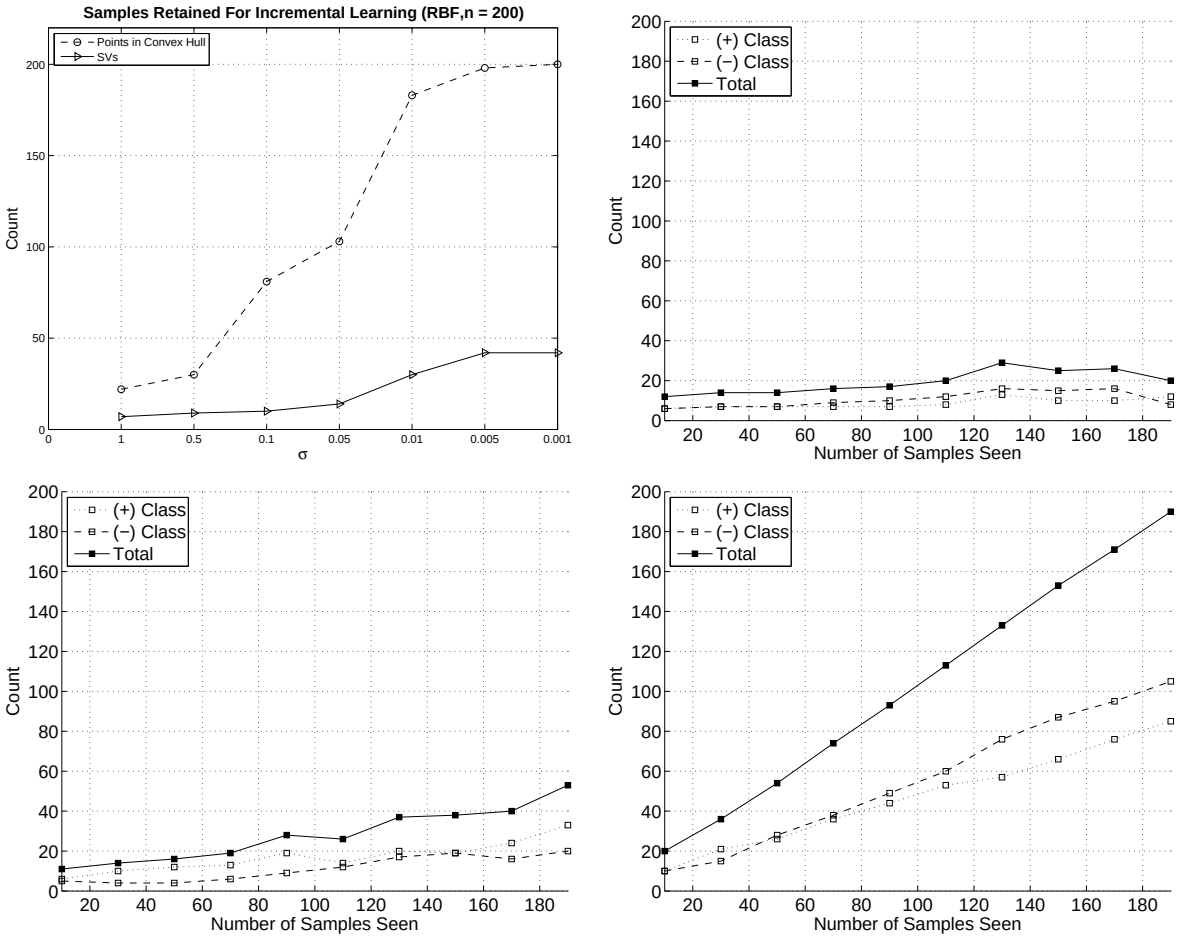


FIGURE 4.5. The number of extreme points identified in the incremental learning progresses using RBF kernels. (a) and (c) show the number of retained samples and SVs as a function of *sigma* using Gaussian dataset and XOR dataset, respectively. (b) and (d) show the number of retained samples using RBF kernel with $\sigma = 1.0$, $\sigma = 0.1$, and $\sigma = 0.01$ (from left to right)

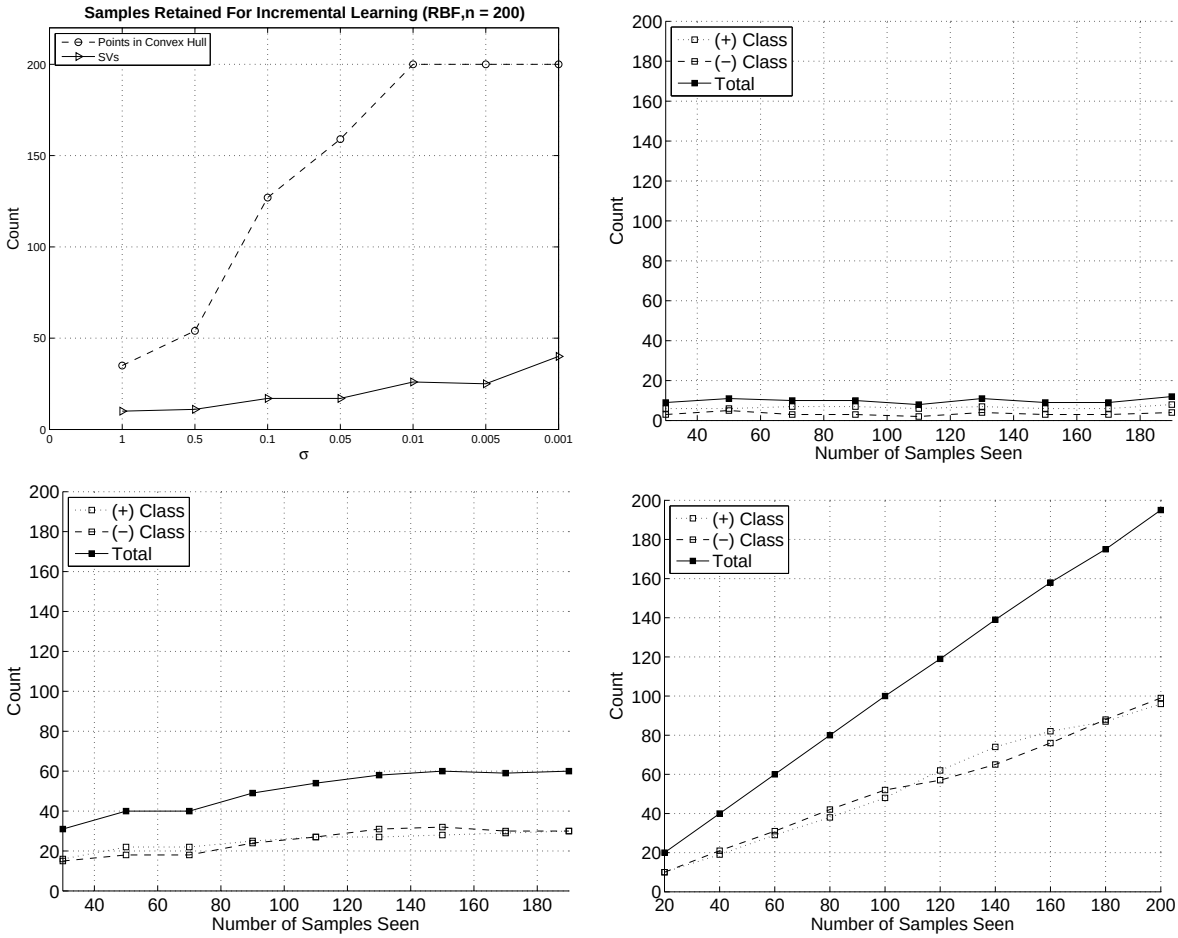


FIGURE 4.6. The number of extreme points identified in the incremental learning progresses using RBF kernels. (a) and (c) show the number of retained samples and SVs as a function of *sigma* using Gaussian dataset and XOR dataset, respectively. (b) and (d) show the number of retained samples using RBF kernel with $\sigma = 1.0$, $\sigma = 0.1$, and $\sigma = 0.01$ (from left to right)

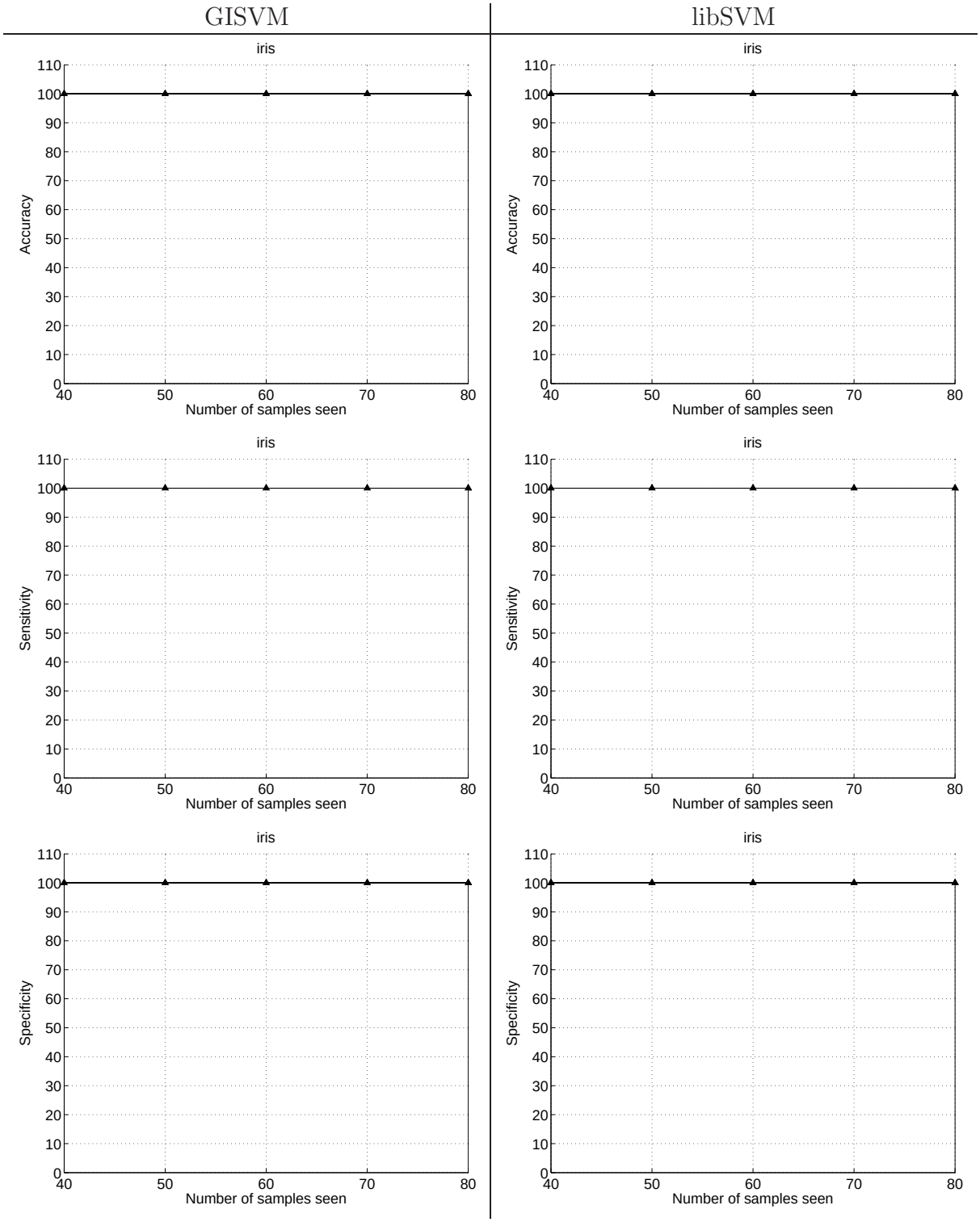


FIGURE 4.8. Results IRIS dataset. The rows top to bottom accuracy, sensitivity, and specificity.

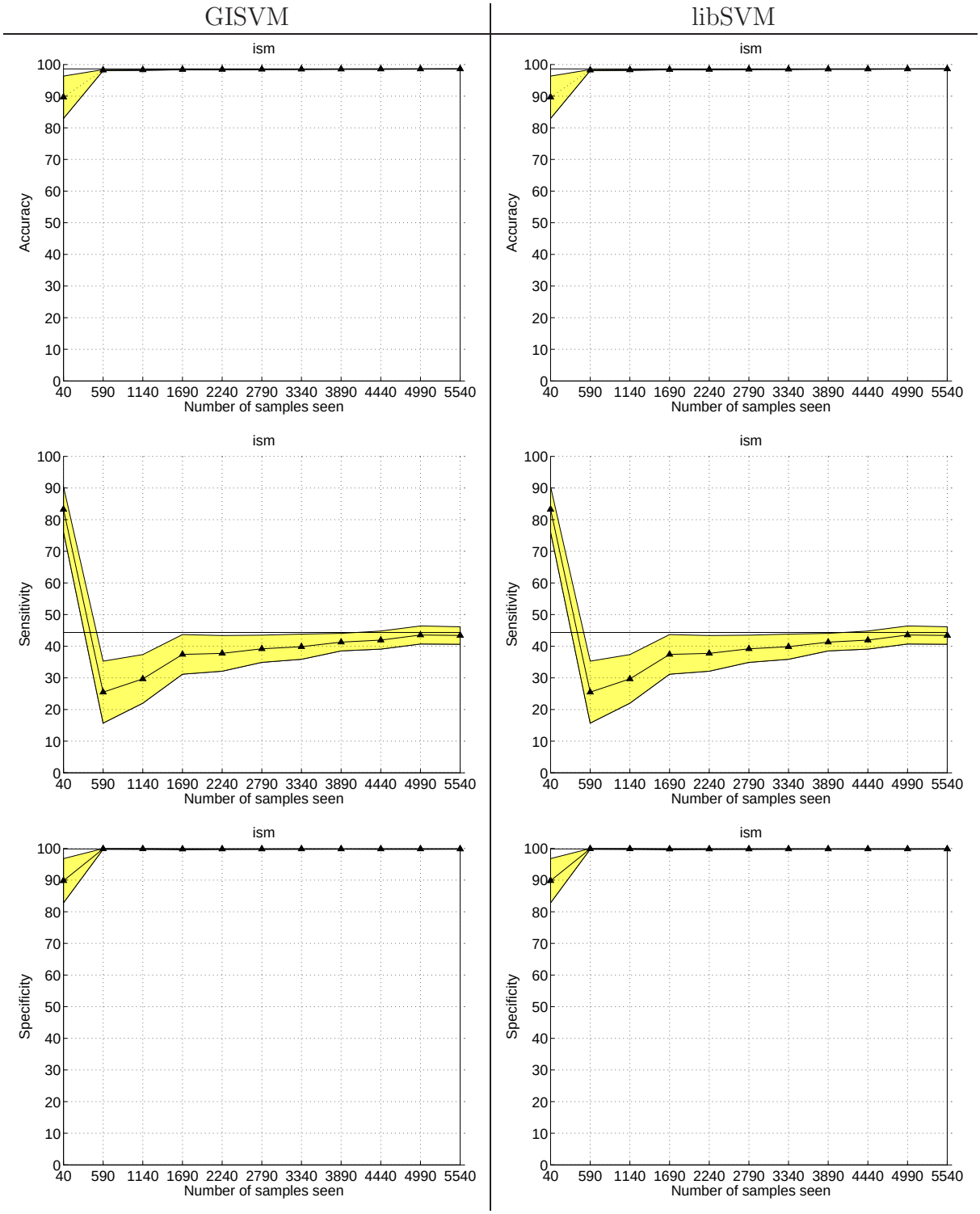


FIGURE 4.9. Results ISM(Mammography)[17] dataset. The rows top to bottom accuracy, sensitivity, and specificity.

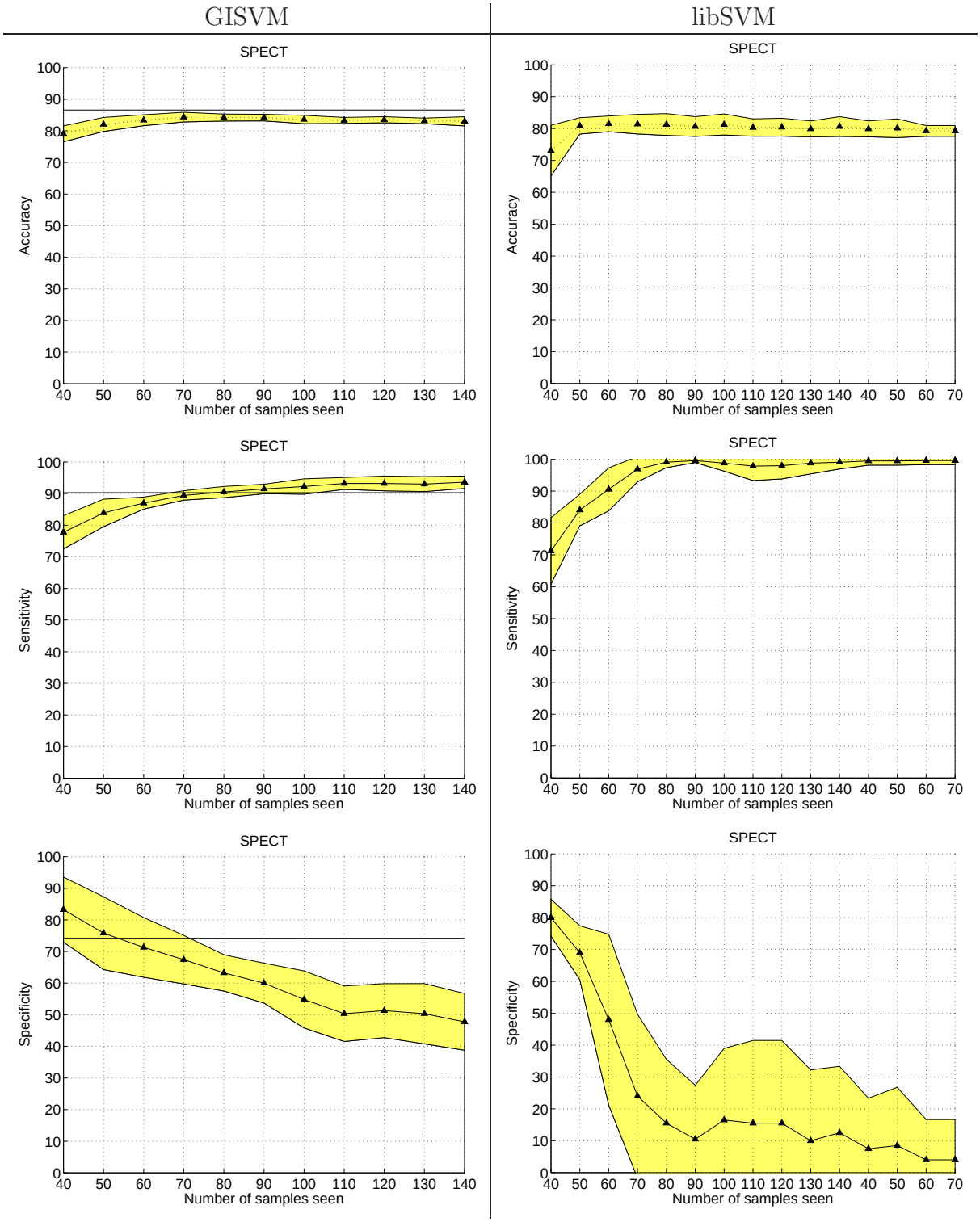


FIGURE 4.10. Results SPECT dataset. The rows top to bottom accuracy, sensitivity, and specificity.

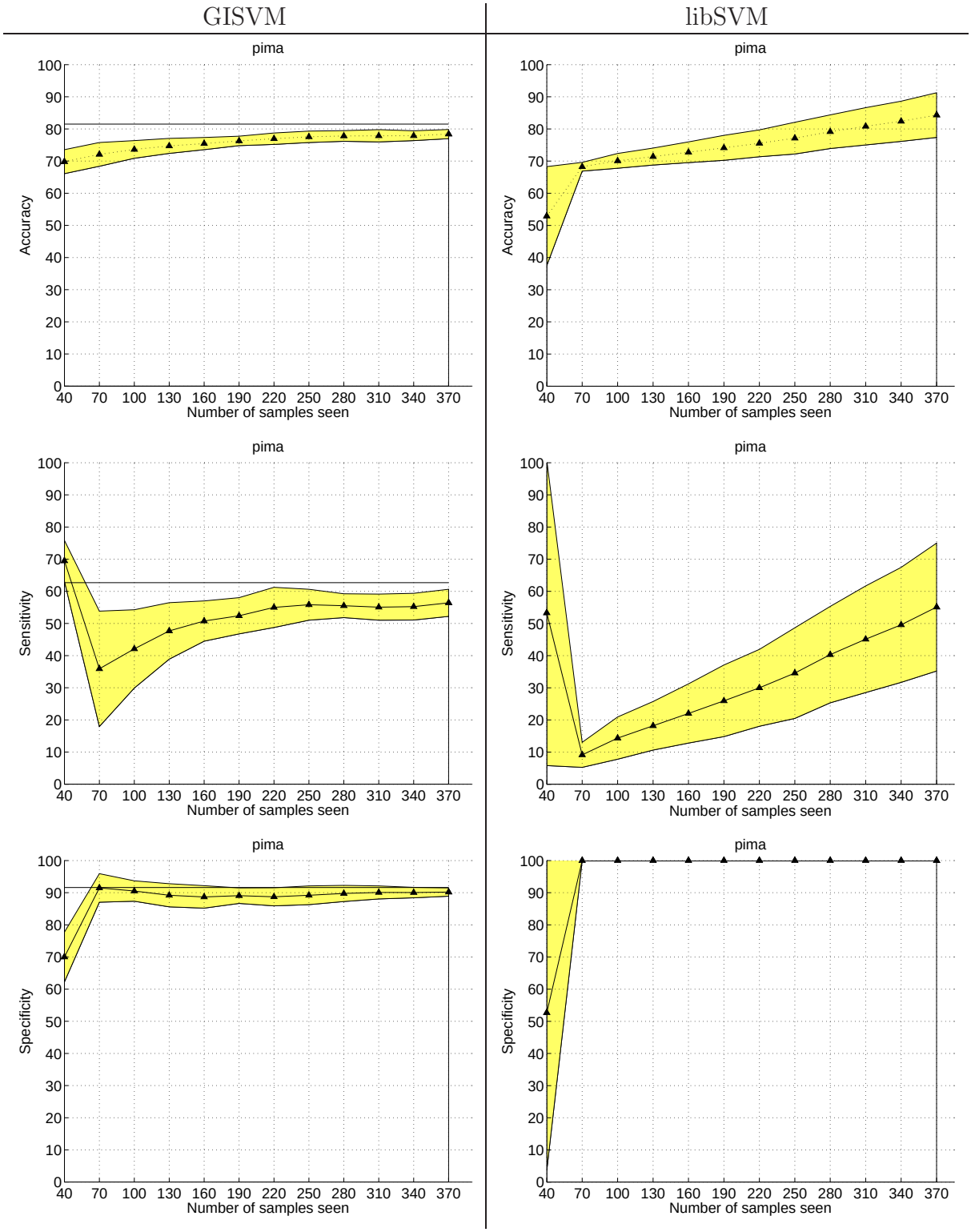


FIGURE 4.11. Results PIMA dataset. The rows top to bottom accuracy, sensitivity, and specificity.

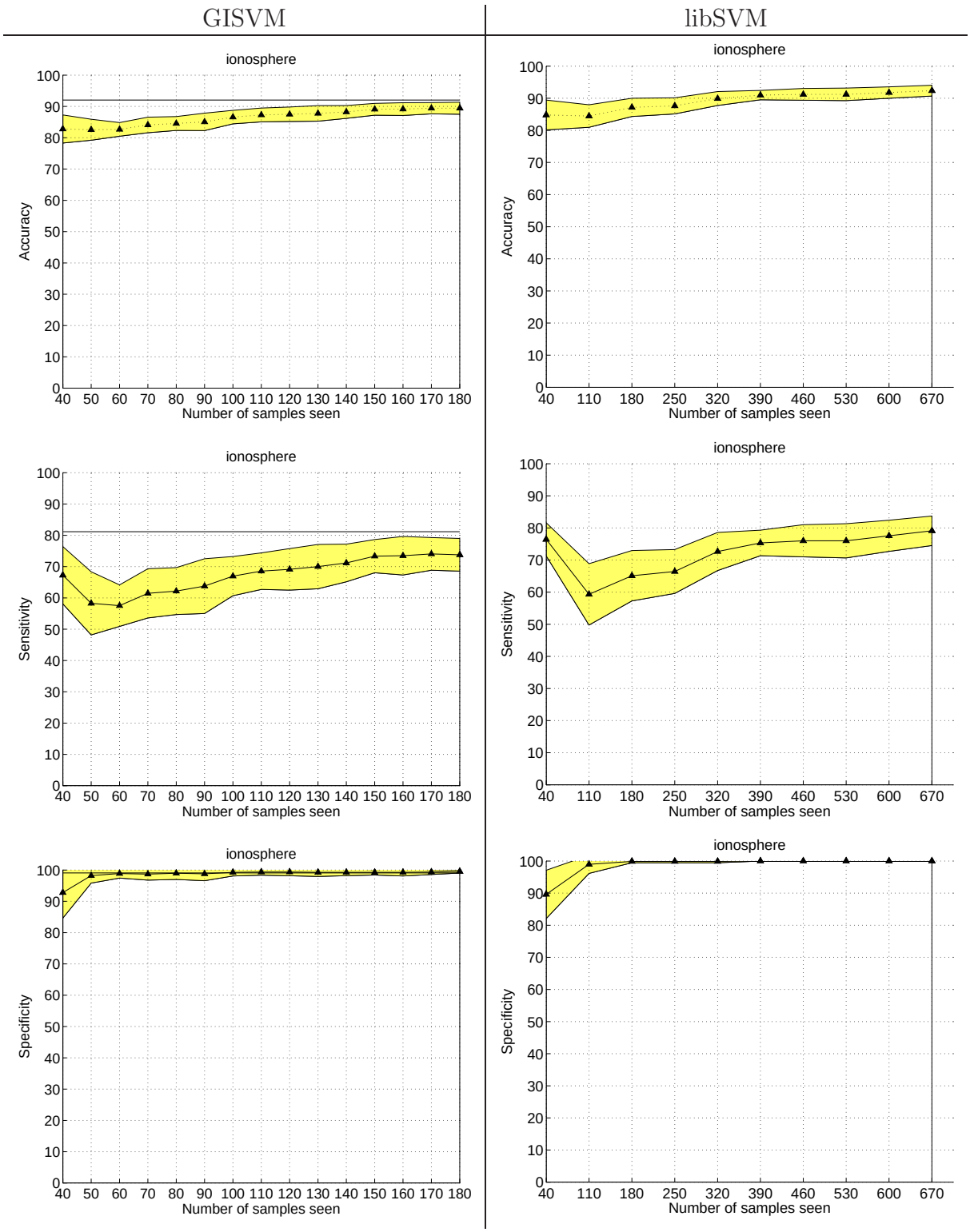


FIGURE 4.12. Results IONOSPHERE dataset. The rows top to bottom accuracy, sensitivity, and specificity.

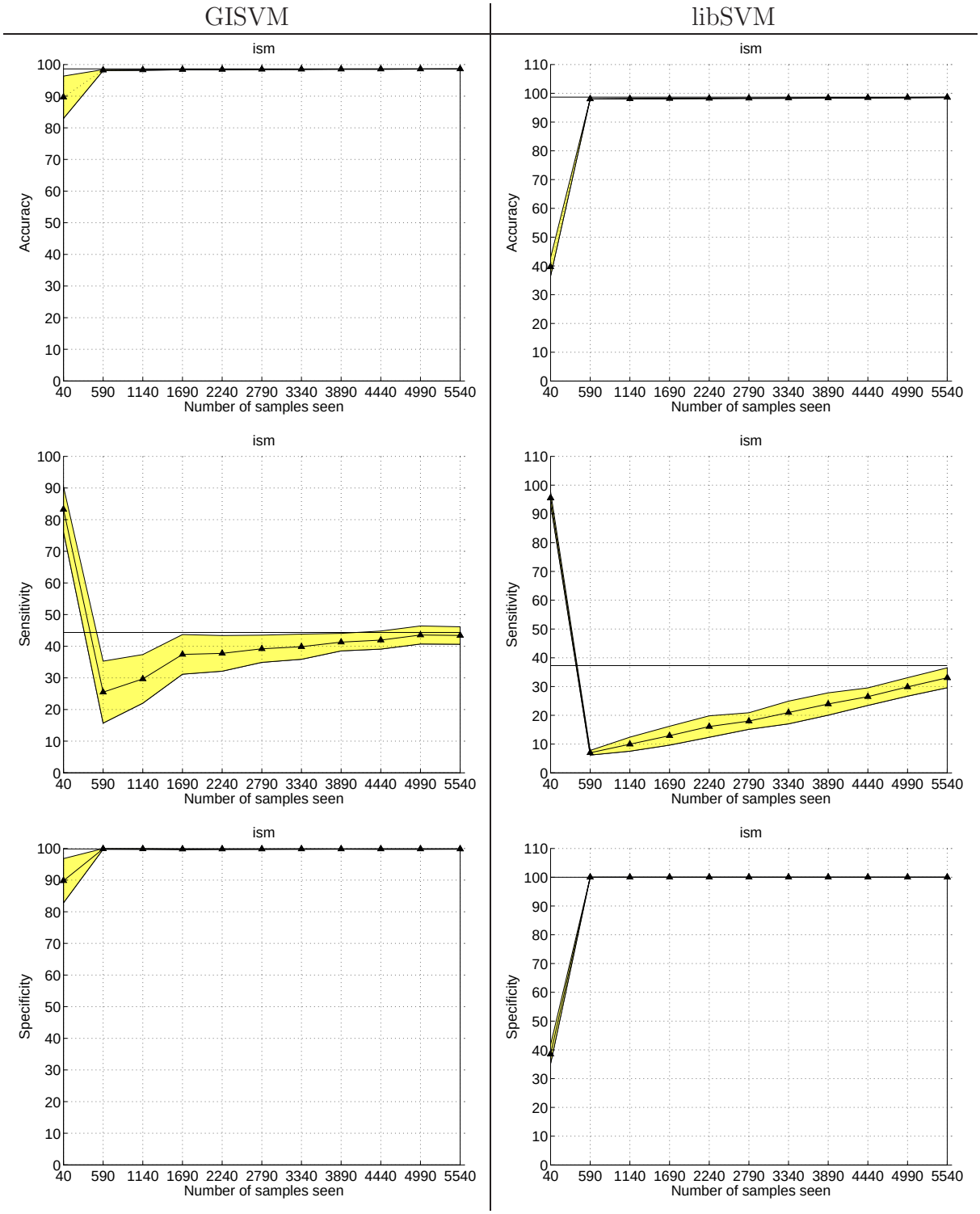


FIGURE 4.13. Results ISM dataset. The rows top to bottom accuracy, sensitivity, and specificity.

CHAPTER 5

APPLICATION - COMPUTER AIDED DIAGNOSIS FOR CAPSULE ENDOSCOPY

5.1. Capsule Endoscopy

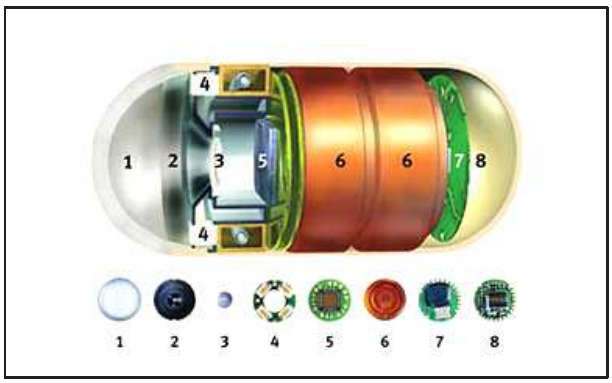


FIGURE 5.1. Contents of CE Device 1) Optical Dome 2) Lens holder 3) Lens 4) Illuminating LEDs (Light Emitting Diode) 5) CMOS 6) Imager 7) Battery 8) ASIC transmitter Antenna

Wireless capsule Endoscopy (WCE) is a recently established technology that requires no wired device intrusion and can be used to examine the entire small intestine non-invasively. The imaging component of this system is a vitamin-sized capsule that is composed of a color CMOS camera, a battery, a light source and a wireless transmitter. It provides a 140-degree field of view and generates 256×256 images. Once the device is activated, it is ready to take pictures. The camera acquires two pictures every second for approximately eight hours, transmitting images to a recording device worn by the patient. By using a lens of short focal length, images are obtained as the capsule is propelled through the tract. Unlike conventional fiber-optic Endoscopy, WCE requires little patient preparation and can potentially image any section in the digestive system. The ability of WCE to detect undersized lesions in the small intestine is ideally suited for this particular role. It enables physicians to examine the entire small intestine, a region that was previously difficult to view at all, and provide a new non-invasive gastrointestinal (GI) visualization technology. The diagnostic yield using WCE is much higher compared to other endoscopic imaging methods. Capsule Endoscopy has the potential for use in a wide variety of illnesses.

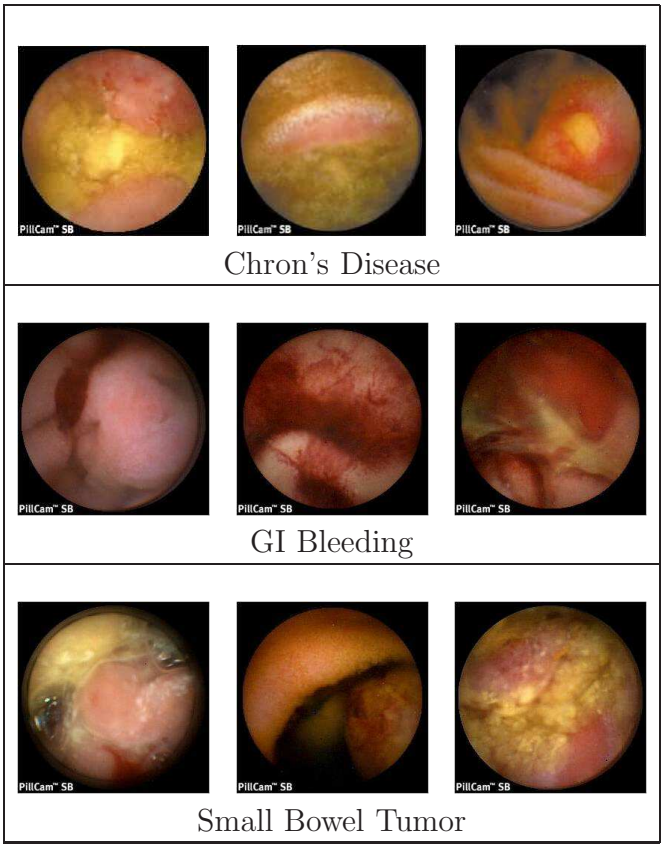
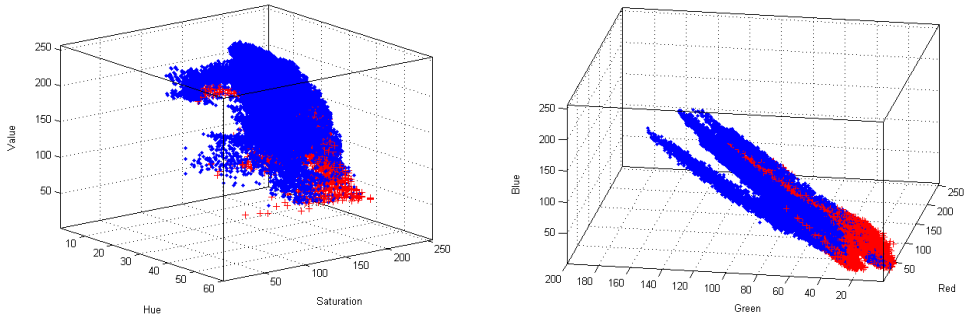


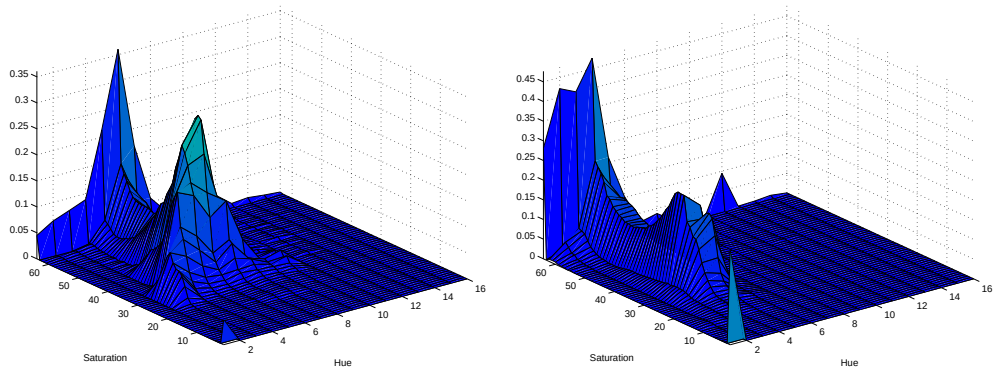
FIGURE 5.2. Video frames from CE videos showing symptoms of disease.

5.2. Feature Extraction



(a) (b)

FIGURE 5.3. (a) HSV & (b) RGB Color spaces of pixels with obscure bleeding(red), and non-bleeding(blue) regions.



(a) (b)

FIGURE 5.4. Average Hue-Saturation Histogram of (a) Non bleeding frames (b) Bleeding frames of the training set.

We employ three image features in our method: color histograms, dominant color, and color co-occurrence.

Color Histogram (CH)

Color histogram is widely used due to its concise representation of color information. Among many color spaces, HSV separates the luminance from chromaticness (Figs 5.4,5.3). It is usually represented with a hexacone, the central vertical axis of which denotes the luminance. Hue is defined as an angle relative to the red and ranges in $[0, 2\pi]$. Saturation is measured as a radial distance from the central axis of the hexacone. Its chromatic components describe color in a way that is most suitable to bleeding detection [55]. Hence, video frames are converted to HSV color space and each color component is normalized to $[0, 1]$ and sampled with 256 bins. In previous experiments with CE videos, it was noted that HSV color space gives better classification performance on average [33]. In addition, using histogram significantly reduces the dimensionality (Each frame is a 256$\times$ 256 color image. If pixel color is used, the dimensionality of each instance is up to 196,608.) Hence, we adopted the color histogram in the HSV space as features. The color histogram is very large and sparse matrix as shown in Figure 5.5. With n bins used in each color component, there are n^3 features using HSV histogram for every video frame, most of which are zeros or close to zeros. To suppress sparseness and the number of values in features, only the hue and saturation (HS) components were used. As observed in our previous experiments [33, 55], an advantage of using HS components is improved robustness in handling lighting variations in the GI tract. As shown

in Figure 5.5, the 2D space spanned by HS components is dominated with small values. Hence, a minimum bounding rectangle region of the HS space with non zero values was identified from the training images. Only the values within the rectangular region of HS histogram were used as features for our classification.

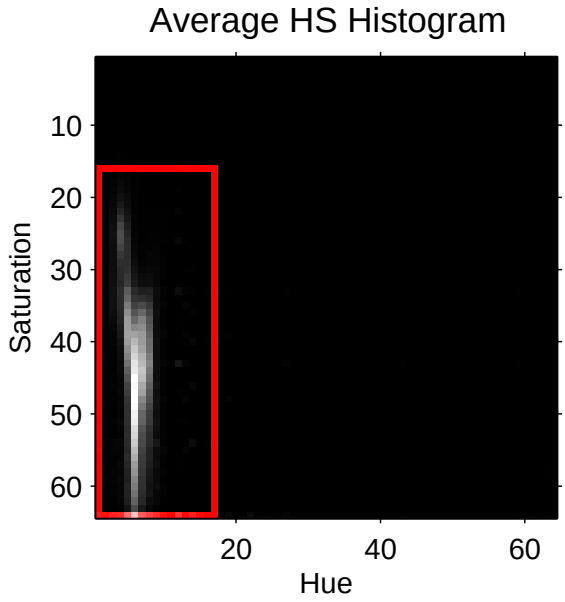


FIGURE 5.5. The average HS histogram of the CE video used in training. The rectangle denotes the color space used in our learning and classification processes.

Dominant Color (DC)

The dominant color consists of eight representative colors, variances for each color, and their percentages in the image [60]. The descriptor is presented as a vector in the following format and the total percentages of the colors in the image sum to 1.

$$DC = \{c_i, v_i, p_i\}, \text{ and } i = \{1, \ldots, 8\} \tag{23}$$

where c_i is the i^{th} dominant color, p_i is its percentage, v_i is the color variance.

For each video frame, colors are clustered and the mean color is used to represent each cluster. This results in a much smaller number of colors. The variance of dominant colors is computed for bleeding and non-bleeding frames. Despite possible information overlap with CH, DC delivers a more concise color description and suppresses the color variance as well as the number of colors.

Color Co-occurrence (CC)

The color co-occurrence matrix follows the classical computation of co-occurrence matrix and contains the frequency of color pair within a pre-defined distance, i.e., $(\triangle x, \triangle y)$. In an 8-bit color image, there are possible 2^{24} colors. To reduce the matrix size, we quantize the color into a set of representative ones. In addition, to eliminate rotation variance in the image plane, we omit the direction of the spatial location of two pixels and only keep track of the pixel distance, i.e., $d = \sqrt{\triangle x^2 + \triangle y^2}$. Because the matrix is symmetric with respect to the major diagonal line, our feature vector only uses the components in the upper triangle matrix.

5.3. Obscure Bleeding Detection in CE Videos

Among many efforts in computer aided diagnosis with CE videos, bleeding detection has been investigated the most due to its clinical importance. The "Suspected Blood Indicator" function by the Given Imaging, a CE manufacturer, provides the capability of detecting blood in video frames. A study by Liangpunsakul et al. [53] showed that the overall sensitivity and accuracy of SBI were 25% and 34.8%, respectively. It exhibits better performance for active bleeding lesions in the small bowel with reported sensitivity and accuracy of 81.2% and 83.3%. This deficiency motivated studies in automatic bleeding detection. Color feature is adopted in many detection studies [47, 39, 37] and texture features are used in applications particularly for detecting heterogeneous objects, e.g., ulcer and polyps [9, 82, 12]. The combination of color and texture has also been heavily experimented [51, 21]. On the other hand, neural networks [82] Support Vector Machines (SVMs) [57, 33], and thresholding [82] are used to make decisions. Despite the encouraging improvement, many previous studies were evaluated with a small number of samples and, to the best of our knowledge, no performance was reported with respect to entire videos. An important question awaits investigation: "Given relatively small number of positive examples from CE videos, how to train learning algorithms to achieve minimal false negative detections?"

Automatic detection of obscure bleeding in CE videos has been studied and Table 5.1 summarizes 12 related work. Despite different features and classification methods used, the experimental data and performances vary greatly. Among these studies, results in eight studies were generated from experiments using 1000 examples or less [52, 9, 82, 41, 51, 21, 40, 46, 56]. Two studies [48, 39]

used moderately larger number of examples. Comparing to the number of frames available in a CE video (approximately 50,000), however, the training data set size is much smaller. Ideally, if the training set is well-selected and unbiased, the classifier can achieve satisfactory generalization performance. It is unclear how the samples are selected and if the cohort formed represents the true data distribution.

TABLE 5.1. Experimental data and detection outcomes. '-' indicates not reported in the paper.

Reported Studies	Data Set Size			Performance		
	Total	**Abnormal**	**Normal**	**Sensitivity**	**Specificity**	**Accuracy**
Kodogiannis and Boulougoura [46]	140	35	35	-	-	95.7%
Kodogiannis and Lygouras [56]	140	35	35	-	-	97.1%
Vilarino et al. [82]	400	100	300	-	-	95.5%
Coimbra and Cunha [21]	1000	-	-	-	-	87%
Lau and Correia [48]	1705	577	1128	88.3%	-	-
Li and Meng [52]	60	30	30	65.2%	82.5%	-
Li and Meng [51]	400	200	200	91%	93%	-
Jung et al. [39]	2000	1000	1000	92.8%	89.5%	-
Barbosa et al. [9]	204	100	104	98.7%	96.6%	-
Karargyris and Bourbakis [41]	-	20	30	75%	73.3%	-
Karargyris and Bourbakis [40]	50	10	40	100%	67.5%	-

5.3.1. Experimental Setup

CE videos were manually annotated by gastroenterologists. The positive training set of 1000 frames was constructed by extracting the frames with signs of obscure bleeding from two videos of human subjects. And the negative training set showing the normal GI tract was obtained be extracting all the frames from a CE video of a normal subject. Seven full length videos of seven subjects with signs of obscure bleeding and three videos with no signs of bleeding was used for evaluation. Each frame in a CE video is 256×256 pixels in size with a color depth of 8-bits. As part of preprocessing, the black border, which is outside the field-of-view, was removed to avoid confusion and reduce the computational expense. The MPEG descriptors Scalable color and the dominant color were used as features of each frame.

First, a suitable μ_u is found such that the RCHs becomes linearly separable. Using the same μ_u for both classes, to minimize the total number of mis-classified samples from both classes would result in the decision boundary being biased to towards the majority class. This would result in large number of false negatives. To avoid this two separate μ_{u+}, μ_{u-} are determined, by weighting using the ratio of number of samples from each class seen. i.e. when making the convex hulls linearly separable for the CE frames one frame with bleeding is weighted to be equal to 150 frames containing no signs of bleeding. Next, to avoid over-fitting, a $\mu_{li}(< \mu_{ui})$ is selected and the skins $S(\Phi_i, \mu_{li}, \mu_{ui})$ $i = \{+, -\}$, denoted by filled triangles in Fig-3.4-b are used to find the decision boundary for the current incremental step. The decision boundary is determined by finding the nearest points between the skins $S(\Phi_i, \mu_{li}, \mu_{ui})$ $i = \{+, -\}$. For the incremental steps skins $S(\Phi_i, \mu_{li}, 1)$ $i = \{+, -\}$ (Fig-3.4-d) are retained.

Further color histogram is invariant to image scaling, translation, rotation as well as partial occlusion. Radial basis function with $\sigma = 0.01$ was used as kernel to train the support vector machine, the samples were weighted by using an imbalanced ratio of 1:150. The Table 5.2 list the performance on two test videos, by adding a video at each incremental step.

5.4. Segmentation of CE Videos by organs in the Gastrointenstinal Tract

Recognizing where a WCE frame is taken in the digestive tract is vital to diagnosis and treatment deployment. An important question that often arises at diagnosis is where the lesion

TABLE 5.2. Results of incrementally training by adding one CE video at each step.

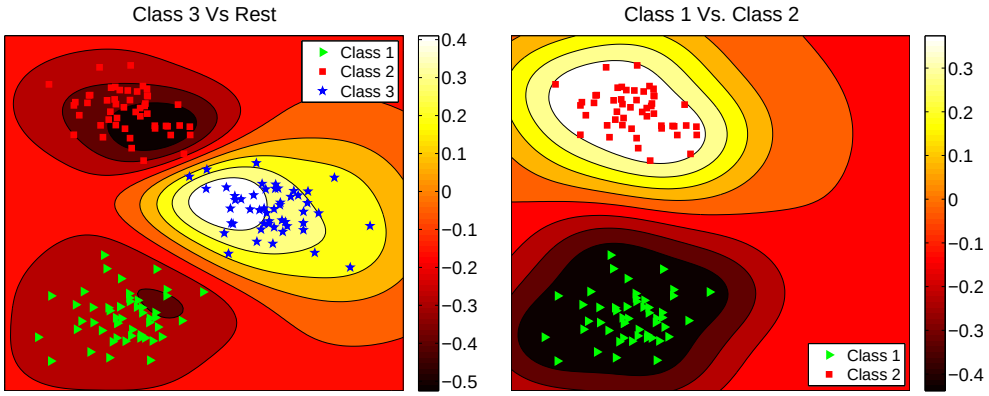
	Sensitivity	Specificity
Step #1	77.2%	98.7%
Step #2	79.5%	98.5%
Step # 3	78.3%	98.6%
Step # 4	80.9%	98.3%
Step # 5	81.7%	98.5%

is found. Reviewing WCE videos and estimating the anatomical locations of WCE frames are, however, very difficult, even for experienced readers. The primary reasons are inconsistent speed of WCE device and lack of physical landmarks. The current technology relies on wireless signal strength, which is used to calculate the distance of the device to the data receiver. This method provides very coarse anatomical trace. A common practice by physicians is reading WCE frames to identify some specific gastrointestinal (GI) images, known as GI landmarks, that indicate entrance to a GI section [23, 57].

The software that comes with Pillcam camera displays the route and relative position of the capsule on a graphical model of the torso, so that diagnosis could be performed. However these are displayed only after the medical specialist manually annotates Pylorus, where the capsule leaves the stomach and enters the intestine, and the Ileocecal valve, between the intestine and colon. The experiments on CE videos were performed to automate the classification of the frames of CE videos into four digestive organs namely esophagus, stomach, intestine and colon.

5.4.1. Multi-class Classification

Figure 5.6 illustrates the contour plots of decision boundary trained with synthetic data sets of three classes. The color depicts the distance to the center of each class. Two classifiers were trained to partition the space into three parts that correspond to the class distribution. The decision boundary for this example is denoted with the zero level contour. Figure 5.6(a) illustrates the decision boundary of class 3 verses classes 1 and 2, whereas Figure 5.6(b) illustrates the decision boundary of classes 1 and 2. The 2D contour plots of the classifier demonstrate that our method predicts class labels successfully in the case of multi-class classification.



(a) (b)

FIGURE 5.6. Results of hierarchical multi class classification. Contours of decision boundaries of (a) Class 3 Vs Class(1&2) (b) Class(1)

TABLE 5.3. Order and parameters of hierarchical classification of organs of CE videos

		Inc GeoSVM	LibSVM
		Kernel	
1	Esophagus Vs Rest	RBF($\sigma = 0.15$)	
2	Intestine Vs Stomach & Colon	RBF($\sigma = 0.1$)	
3	Stomach Vs Colon	RBF($\sigma = 0.5$)	

5.4.2. Experimental Setup

The order of classification of multi-class SVM was determined based on the preliminary classification evaluation. In our experiments, identification of esophagus gives the best accuracy followed by the identification of small intestine. Hence, the order is determined and listed in Table 5.4. The kernels used to train SVM are also included in this table.

TABLE 5.4. Order and parameters of hierarchical classification of organs of CE videos

Order	Dividing classes	Kernel
1	Esophagus vs. Rest	RBF($\sigma = 0.15$)
2	Small intestine vs. Stomach and Colon	RBF($\sigma = 0.1$)
3	Stomach vs. Colon	RBF($\sigma = 0.5$)

In the learning process, 50 frames were randomly selected from each class of the training video to train a SVM. In each incremental step, randomly selected 20 frames from the remaining training video frames were used to update the classifier. The iteration repeats until the training examples exhaust. Table 5.5 lists the results of the final classifiers tested using the test videos. The

performance of our method is highly satisfactory. With the majority of frames acquired in stomach and small intestine, the average accuracies are 86.9% and 94.4%, respectively. Images acquired in colon are disturbed with noise from feces and fluid, which results in large number of dark images and causes performance drop in the classification accuracy.

TABLE 5.5. Classification performance of digestive organs in CE videos

Video	Esophagus	Stomach	Small Intestine	Colon
1	100.0%	87.6%	94.2%	85.3%
2	94.4%	85.8%	95.3%	82.2%
3	95.0%	87.2%	94.7%	84.3%
4	100.0%	86.4%	94.1%	83.7%
5	90.0%	87.7%	93.9%	94.3%

At the end of incremental training only 12% of the frames were part of the skins among the four classes for the hierarchical SVMs. Apparently, the smaller number of examples demands much less memory space for learning process and, hence, provides a plausible mechanism for handling large amount of data set. When new samples are added, the classifier is updated efficiently in contrast to the conventional batch learning methods.

CHAPTER 6

CONCLUSION

With advance in data acquisition technology, data available for research have become large and dynamic. Efficient and scalable approaches are needed that can modify knowledge structures in an incremental fashion without having to revisit all previously processed data.

This dissertation presents GISVM to learn from large data set with emerging trend and dynamic patterns. To overcome high computational demands from large data set, this dissertation describes a method to identify and employ a subset of samples in the training process. GISVM, extends the reduced convex hull concept and defines the approximate skin segments of convex hulls. The data points in the skin are found by identifying the extreme points. A recursive method is developed to find such points in reduced convex hulls. The intuition of our incremental learning is that only the samples within the skin are retained in training. When additional samples are provided, they will be used together with the skin of the convex hull constructed from previous dataset. This results in a small number of instances used in incremental steps of the training process. The set of extreme points are found by recursively searching along the direction defined by a pair of extreme points. Therefore, a fewer number of instances is used in the training process.

Besides the advantages in computational efficiency, our method handles linearly non-separable cases naturally. This is achieved with the representation of disjoint datasets of the distinct classes.

Experiments were conducted with synthetic, benchmark and CE dataset. The results demonstrated highly competitive performance that requires much fewer resource. Based on the experiments, the following conclusions can be drawn.

With the synthetic datasets, the proposed method achieves satisfactory classifiers that closely model the underlying data distribution with appropriate kernels. The choice of RBF kernel for the synthetic datasets provides a good description of the datasets and retains much fewer number of samples in the training iterations.

From the experiments on benchmark datasets it was demonstrated that the GISVM learning handles data efficiently and update the classifier in approximately 13.4% of the time for batch learn-

ing SVM. Performance over the incremental steps demonstrates the proposed method's stability, improvement, and recoverability. The accuracy over the incremental steps increases steadily. Since the implementation is optimized for accuracy, the sensitivity or the specificity drops at certain steps. As the skin containing useful samples is retained, the classifier is able to recover to previous levels in future iterations. Furthermore, the improvement in the performance measures over the incremental steps by deleting samples other than the ones on skin indicates that retaining the samples on the skin preserves adequate information about the decision boundary of SVMs.

From the experiments on CE videos it was noted that the average performance of classifying CE video is above 86.9%, which is very competitive. The amount of memory space required in the training process could be one eighth of what is required by the conventional SVM, which cast new light on processing large data set within constrained resource. The accuracy for our CE video segmentation could be further improved if temporal information is utilized. Further experiments on CE videos demonstrated that GISVM can handled data that could not be handled by libSVM.

6.1. Future Work

This dissertation presents GISVM to learn from large data set with emerging trend and dynamic patterns to overcome high computational demands. Primary forcus of this work has been to identyfy more informative samples and use them in incremental steps. Thereby reducing the number of samples used in incremental steps and enable learning in limited memory situations. On direction of future work would be to efficiently store the computed values, so that they could be reused.

Parameters selection for the GISVM is done by deciding on parameter ranges, and to then do an exhaustive grid search over the parameter space to find the best setting manually. This at times results in a large number of evaluations and long training run times. Automating parameter selection of parameters a using a more coarse to fine refinement is another direction for improvement.

GISVM proposed in this dissertation has been validated using synthetic and benchmark data set. Theoretical validation or estimates on bound on error between batch and incremental learning would add value to the proposed method.

BIBLIOGRAPHY

[1] Shilton A, Palaniswami M, Ralph D, and Ah Chung Tsoi, *Incremental training of support vector machines*, IEEE Transactions on Neural Networks 16 (Jan. 2005), 114– 131.

[2] D. G. Adler and C. J. Gostout, *Wireless capsule endoscopy*, Hospital Physician 39 (2003), no. 5, 14–22.

[3] Sumeet Agarwal, V. Vijaya Saradhi, and Harish Karnick, *Kernel-based online machine learning and support vector reduction*, Neurocomput. 71 (2008), 1230–1237.

[4] Habiboulaye Amadou Boubacar, Stéphane Lecoeuche, and Salah Maouche, *Sakm: Self-adaptive kernel machine a kernel-based algorithm for online clustering*, Neural Netw. 21 (2008), 1287–1301.

[5] M. Appleyard, Z. Fireman, A. Glukhovsky, H. Jacob, R. Shreiver, S. Kadirkamanathan, A. Lavy, S. Lewkowicz, E. Scapa, R. Shofti, P. Swain, and A. Zaretsky, *A randomized trial comparing wireless capsule endoscopy with push enteroscopy for the detection of small-bowel lesions*, Gastroenterology 119 (2000), no. 6, 1431–1438.

[6] F. Arguelles-Arias, A. Caunedo, J. Romero, A. Sanchez, M. Rodriguez-Tellez, F. J. Pellicer, F. Arguelles-Martin, and J. M. Herrerias, *The value of capsule endoscopy in pediatric patients with a suspicion of crohn's disease*, Endoscopy 36 (2004), no. 10, 869–873.

[7] A. Asuncion and D.J. Newman, *UCI machine learning repository*, 2007.

[8] C. Bradford Barber, David P. Dobkin, and Hannu Huhdanpaa, *The quickhull algorithm for convex hulls*, ACM Trans. Math. Softw. 22 (1996), 469–483.

[9] Daniel J. C. Barbosa, Jaime Ramos, and Carlos S. Lima, *Detection of small bowel tumors in capsule endoscopy frames using texture analysis based on the discrete wavelet transform*, Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE, aug. 2008, pp. 3012–3015.

[10] Kristin P. Bennett and Colin Campbell, *Support vector machines: hype or hallelujah?*, SIGKDD Explor. Newsl. 2 (2000), no. 2, 1–13.

[11] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik, *A training algorithm for optimal margin classifiers*, COLT '92: Proceedings of the fifth annual workshop on Computational learning theory (New York, NY, USA), ACM, 1992, pp. 144–152.

[12] Nikolaos Bourbakis, *Detecting abnormal patterns in wce images*, Bioinformatics and Bioengineering, 2005. BIBE 2005. Fifth IEEE Symposium on (2005), 232–238.

[13] Christopher J. C. Burges, *A tutorial on support vector machines for pattern recognition*, Data Mining Knowledge Discovery 2 (1998), 121–167.

[14] Doina Caragea, Adrian Silvescu, and Vasant G. Honavar, *Analysis and synthesis of agents that learn from distributed dynamic data sources*, (2001), 547–559.

[15] Gert Cauwenberghs and Tomaso Poggio, *Incremental and decremental support vector machine learning*, NIPS, 2000, pp. 409–415.

[16] Chih-Chung Chang and Chih-Jen Lin, *LIBSVM: a library for support vector machines*, 2001.

[17] Nitesh V. Chawla, K. W. Bowyer, L. O. Hall, and P. Kegelmeyer, *SMOTE: Synthetic minority over-sampling technique*, Journal of Artificial Intelligence and Research 16 (2002), 321–357.

[18] Pai-Hsuen Chen, Chih-Jen Lin, and Bernhard Schölkopf, *A tutorial on ν-support vector machines:*, Appl. Stoch. Model. Bus. Ind. 21 (2005), no. 2, 111–136.

[19] Shouxian Cheng and Frank Y. Shih, *An improved incremental training algorithm for support vector machines using active query*, Pattern Recogn. 40 (2007), 964–971.

[20] A. K. H. Chong, A. Taylor, A. Miller, O. Hennessy, W. Connell, and P. Desmond, *Capsule endoscopy vs. push enteroscopy and enteroclysis in suspected small-bowel crohn's disease*, Gastrointestinal Endoscopy 61 (2005), no. 2, 255–261.

[21] Miguel Tavares Coimbra and Joo Paulo Silva Cunha, *Mpeg-7 visual descriptors contributions for automated feature extraction in capsule endoscopy*, IEEE Transactions on Circuits and Systems for Video Technology (2006).

[22] David J. Crisp and Christopher J. C. Burges, *A geometric interpretation of ν-SVM classifiers*, 1999.

[23] J. P. S. Cunha, M. Coimbra, P. Campos, and J. M. Soares, *Automated topographic segmenta-*

tion and transit time estimation in endoscopic capsule exams, IEEE Transactions on Medical Imaging 27 (2008), no. 1, 19–27.

[24] C.P. Diehl and G. Cauwenberghs, *SVM incremental learning, adaptation and optimization*, Proceedings of the International Joint Conference on Neural Networks 4 (2003), 2685–2690.

[25] C. Domeniconi and D. Gunopulos, *Incremental support vector machine construction*, Proceedings IEEE International Conference on Data Mining (2001), 589–592.

[26] Hua Duan, Xiaojian Shao, Weizhen Hou, Guoping He, and Qingtian Zeng, *An incremental learning algorithm for lagrangian support vector machines*, Pattern Recogn. Lett. 30 (2009), no. 15, 1384–1391.

[27] R. Eliakim, A. Suissa, K. Yassin, D. Katz, and D. Fischer, *Wireless capsule video endoscopy compared to barium follow-through and computerised tomography in patients with suspected crohnś disease-final report*, Digest and Liver Disease 36 (2004), 519–522.

[28] S. Ertekin, L. Bottou, and C.L. Giles, *Nonconvex online support vector machines*, Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (2011), no. 2, 368–381.

[29] G. Fung and O. Mangasarian, *Incremental support vector machine classification*, Technical Report 01-08, Data Mining Institute, Computer Sciences Department, University of Wisconsin, Madison (2001).

[30] G. Fung and O. L. Mangasarian, *Proximal support vector machine classifiers*, Knowledge Discovery and Data Mining (2001), 77–86.

[31] Giovanni Gallo, Eliana Granata, and Giuseppe Scarpulla, *Wireless capsule endoscopy video segmentation*, IEEE Transactions on International Workshop on Medical Measurements and Applications (2009), 236–240.

[32] E.G. Gilbert, *An iterative procedure for computing the minimum of a quadratic form on a convex set.*, SIAM Journal of Control 4 (1966), no. 1.

[33] Balathasan Giritharan, Xiaohui Yuan, Jianguo Liu, Bill Buckle, JungHwan Oh, and Shou Jiang Tang, *Bleeding detection from capsule endoscopy videos*, 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (2008), 4780–4783.

[34] N. I. Goldfarb, L. T. Pizzi, J. P. Fuhr, C. Salvador, V. Sikirica, A. Kornbluth, and B. Lewis,

Diagnosing crohn's disease: An economic analysis comparing wireless capsule endoscopy with traditional procedures, Disease Management 4 (2004), no. 7, 292–304.

[35] A. K. Hara, J. A. Leighton, V. K. Sharma, R. I. Heigh, and D. E. Fleischer, *Imaging of small bowel disease: Comparison of capsule endoscopy, standard endoscopy, barium examination, and ct*, Radiographics 25 (2005), 697–711.

[36] D. Hartmann, H. Schmidt, Bolz, D. Schilling, F. Kinzel, A. Eickhoff, W. Huschner, K. Moller, R. Jakobs, P. Reitzig, U. Weickert, K. Gellert, H. Schultz, K. Guenther, H. Hollerbuhl, K. Schoenleben, H. J. Schulz, and J. Riemann, *A prospective two-center study comparing wireless capsule endoscopy with intraoperative enteroscopy in patients with obscure GI bleeding*, Gastrointest Endoscopy 61 (2005), no. 7, 826–832.

[37] Sae Hwang, JungHwan Oh, Jay Cox, Shou Jiang Tang, and Harry F. Tibbals, *Blood detection in wireless capsule endoscopy using expectation maximization clustering*, SPIE Medical Imaging 2006 (2006).

[38] G. Iddan, G. Meron, A. Glukhovsky, and P. Swain, *Wireless capsule endoscopy*, Nature 405 (2000), 417.

[39] Yun Sub Jung, Young Ho Kim, Dong Ha Lee, and Jong Hyo Kim, *Active blood detection in a high resolution capsule endoscopy using color spectrum transformation*, International Conference on BioMedical Engineering and Informatics 1 (2008), 859–862.

[40] A. Karargyris and N. Bourbakis, *Identification of polyps in wireless capsule endoscopy videos using log gabor filters*, Life Science Systems and Applications Workshop (2009), 143–147.

[41] Alexandros Karargyris and Nikolaos Bourbakis, *Identification of ulcers in wireless capsule endoscopy videos*, ISBI'09: Proceedings of the Sixth IEEE international conference on Symposium on Biomedical Imaging (Piscataway, NJ, USA), IEEE Press, 2009, pp. 554–557.

[42] D. A. Kastin, A. L. Buchman, T. Barrett, A. Halverson, and A. Wallin, *Strictures from crohn's disease diagnosed by video capsule endoscopy*, Journal of Clinical Gastroenterology 38 (2004), no. 4, 346–349.

[43] Shinya Katagiri and Shigeo Abe, *Incremental training of support vector machines using hyperspheres*, Pattern Recognition Letters 27 (2006), no. 13, 1495–1507.

[44] S.S. Keerthi, S.K. Shevade, C. Bhattacharyya, and K.R.K. Murthy, *A fast iterative nearest point algorithm for support vector machine classifier design*, Neural Networks, IEEE Transactions on 11 (2000), no. 1, 124–136.

[45] Ralf Klinkenberg and Thorsten Joachims, *Detecting concept drift with support vector machines*, ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning (2000), 487–494.

[46] V. S. Kodogiannis and M. Boulougoura, *An adaptive neurofuzzy approach for the diagnosis in wireless capsule endoscopy imaging*, International Journal of Information Technology 13 (2007), no. 1, 46–54.

[47] Vassilis Kodogiannis, *Computer-aided diagnosis in clinical endoscopy using neuro-fuzzy systems*, IEEE International Conference on Fuzzy Systems (2004).

[48] Phooi Yee Lau and P.L. Correia, *Detection of bleeding patterns in wce video using multiple features*, Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE, aug. 2007, pp. 5601–5604.

[49] J.A. Leighton, K. Srivathsan, E. J. Carey, V. K. Sharma, R. I. Heigh, J. K. Post, P. J. Erickson, S. R. Robinson, J. L. Bazzell, and D. E. Fleischer, *Safety of wireless capsule endoscopy in patients with implantable cardiac defibrillators*, American Journal of Gastroenterology 100 (2005), 1–4.

[50] B. S. Lewis and P. Swain, *Capsule endoscopy in the evaluation of patients with suspected small intestinal bleeding: results of a pilot study*, Gastrointestinal Endoscopy 56 (2002), no. 3, 349–353.

[51] Baopu Li and M.Q.-H. Meng, *Computer-aided detection of bleeding regions for capsule endoscopy images*, Biomedical Engineering, IEEE Transactions on 56 (2009), no. 4, 1032 –1039.

[52] Bapou Li and M.Q.-H. Meng, *Analysis of wireless capsule endoscopy images using chromaticity moments*, IEEE International Conference on Robotics and Biomimetics (2007), 87–92.

[53] Suthat Liangpunsakul, Vidyasree Chadalawada, Douglas K Rex, Dean Maglinte, and John Lappas, *Wireless capsule endoscopy detects small bowel ulcers in patients with normal results*

from state of the art enteroclysis, The American College of Gastroenterology 98 (2003), no. 1, 1295–1298.

[54] Hsuan-Tien Lin and Chih-Jen Lin, *A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods*, Tech. report, Department of Computer Science, National Taiwan University, 2003.

[55] J. Liu and X Yuan, *Obscure bleeding detection in endoscopy images using support vector machines*, Optimization and Engineering (2009), 289–299.

[56] Yang Liu, Aijun An, and Xiangji Huang, *Boosting prediction accuracy on imbalanced datasets with SVM ensembles*, Advances in Knowledge Discovery and Data Mining (2006), 107–118.

[57] M. Mackiewicz, J. Berens, and M. Fisher, *Wireless capsule endoscopy color video segmentation*, IEEE Transactions on Medical Imaging 27 (2008), no. 12, 1769–1781.

[58] Michal W. Mackiewicz, Mark Fisher, and Crawford Jamieson, *Bleeding detection in wireless capsule endoscopy using adaptive colour histogram model and support vector classification*, Medical Imaging 2008: Image Processing 6914 (2008), no. 1, 69140R.

[59] O. L. Mangasarian and David R. Musicant, *Active support vector machine classification*, Advances in Neurawl Information Processing Systems (2000), 577–583.

[60] B.S. Manjunath, J.-R. Ohm, V.V. Vasudevan, and A. Yamada, *Color and texture descriptors*, Circuits and Systems for Video Technology, IEEE Transactions on 11 (2001), no. 6, 703–715.

[61] M.E. Mavroforakis, M. Sdralis, and S. Theodoridis, *A geometric nearest point algorithm for the efficient solution of the SVM classification task*, IEEE Transactions on Neural Networks 18 (2007), no. 5, 1545–1549.

[62] M.E. Mavroforakis and S. Theodoridis, *A geometric approach to support vector machine (SVM) classification*, IEEE Transactions on Neural Networks 17 (2006), no. 3, 671–682.

[63] P. Mitra, C.A. Murthy, and S.K. Pal, *Data condensation in large databases by incremental learning with support vector machines*, Proceedings. 15th International Conference on Pattern Recognition, vol. 2, 2000, pp. 708–711.

[64] Pabitra Mitra, Student Member, C. A. Murthy, and Sankar K. Pal, *A probabilistic active sup-*

port vector learning algorithm, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (2004), 2004.

[65] M. Muhlbaier, A. Topalis, and R. Polikar, *Incremental learning from unbalanced data*, 2004 IEEE International Joint Conference on Neural Networks, vol. 2, July 2004, pp. 1057–1062.

[66] Robert Nisbet, John Elder, and Gary Miner, *Handbook of statistical analysis and data mining applications*, Academic Press, 2009.

[67] Francesco Orabona, Claudio Castellini, Barbara Caputo, Luo Jie, and Giulio Sandini, *On-line independent support vector machines*, Pattern Recogn. 43 (2010), 1402–1412.

[68] Edgar Osuna and Osberth De Castro, *Convex hull in feature space for support vector machines*, IBERAMIA 2002: Proceedings of the 8th Ibero-American Conference on AI (London, UK), Springer-Verlag, 2002, pp. 411–419.

[69] E. Parrado-Hernndez, I. Mora-Jimnez, J. Arenas-Garca, A. R. Figueiras-Vidal, and A. Navia-Vzquez, *Growing support vector classifiers with controlled complexity*, Pattern Recognition 36 (2003), no. 7, 1479 – 1488.

[70] Xin-jun Peng and Yi-fei Wang, *CCH-based geometric algorithms for SVM and applications*, Applied Mathematics and Mechanics 30 (2009), 89–100.

[71] R.A. and Jarvis, *On the identification of the convex hull of a finite set of points in the plane*, Information Processing Letters 2 (1973), no. 1, 18–21.

[72] Carlos Renjifo, David Barsic, Craig Carmen, Kevin Norman, and G. Scott Peacock, *Improving radial basis function kernel classification through incremental learning and automatic parameter selection*, Neurocomput. 72 (2008), 3–14.

[73] A. Ruano-Ravinia and T. Rey-Liste, *Effectiveness of endoscopic capsule for the detection of small bowel bleeding of unknown origin and for the diagnosis of crohnś disease*, Medicina Clnica 123 (2004), no. 2, 70–76.

[74] Bernhard Schölkopf, John C. Platt, John C. Shawe-Taylor, Alex J. Smola, and Robert C. Williamson, *Estimating the support of a high-dimensional distribution*, Neural Comput. 13 (2001), no. 7.

[75] Nicola Segata and Enrico Blanzieri, *Fast local support vector machines for large datasets*, Pro-

ceedings of the 6th International Conference on Machine Learning and Data Mining in Pattern Recognition, 2009, pp. 295–310.

[76] Nadeem A Syed, Huan Liu, and Kah Kay, *Incremental learning with support vector machines*, In Proceedings of the Workshop on Support Vector Machines at the International Joint Conference on Articial Intelligence (IJCAI-99) (1999).

[77] S. J. Tang and G. B. Haber, *Capsule endoscopy in obscure gastrointestinal bleeding*, Gastrointestinal Endoscopy Clinics of North America 14 (2004), 87–100.

[78] S. J. Tang, S. Zanati, E. Dubcenco, D. Christodoulou, M. Cirocco, G. Kandel, P. Kortan, G. B. Haber, and N. E. Marcon, *Capsule endoscopy regional transit abnormality: a sign of underlying small bowel pathology*, Gastrointestinal Endoscopy 58 (2003), no. 4, 598–602.

[79] S. J. Tang, S. Zanati, G. Kandel, N. E. Marcon, and P. Kortan, *Gastric intestinal vascular ectasia syndrome: Findings on capsule endoscopy*, Endoscopy 37 (2005), no. 12, 1244–1247.

[80] Amund Tveit, Magnus Hetland, and Håavard Engum, *Incremental and decremental proximal support vector classification using decay coefficients*, Data Warehousing and Knowledge Discovery (2003), 422–423.

[81] Vladimir N. Vapnik, *The nature of statistical learning theory*, Springer–Verlag (1995).

[82] Fernando Vilarino, Ludmila Kuncheva, and Petia Radeva, *ROC curves and video analysis optimization in intestinal capsule endoscopy*, Pattern Recognition Letters 27 (2006), 875–881.

[83] Xue wen Chen, B. Gerlach, and D. Casasent, *Pruning support vectors for imbalanced data classification*, Proceedings, IEEE International Joint Conference on Neural Networks 3 (2005), 1883–1888.

[84] Chongming Wu, Xiaodan Wang, Dongying Bai, and Hongda Zhang, *Fast incremental learning algorithm of svm on kkt conditions*, Proceedings of the 6th international conference on Fuzzy systems and knowledge discovery - Volume 1, 2009, pp. 551–554.

[85] Rong Xiao, Jicheng Wang, and Fayan Zhang, *An approach to incremental SVM learning algorithm*, 12th IEEE International Conference on Tools with Artificial Intelligence (2000), 268–273.

[86] Dengyong Zhou, Baihua Xiao, Huibin Zhou, and Ruwei Dai, *Global geometry of SVM classifiers*, 2001.