12-2012

A submodular optimization framework for never-ending learning : semi-supervised, online, and active learning.

Wael Emara
University of Louisville

A SUBMODULAR OPTIMIZATION FRAMEWORK FOR NEVER-ENDING LEARNING: SEMI-SUPERVISED, ONLINE, AND ACTIVE LEARNING

By

Wael Emara
M.Sc., 2006, in Electrical and Computer Engineering, University of Louisville, USA
B.Sc., 2000, and M.Sc., 2003, in Electronics and Electrical Communications Engineering, Mansoura University, Egypt.

A Dissertation Submitted to the
Faculty of the J.B. Speed School of Engineering
at the University of Louisville
in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

Department of Computer Engineering and Computer Science
University of Louisville
Louisville, Kentucky

December 2012

A SUBMODULAR OPTIMIZATION FRAMEWORK FOR NEVER-ENDING LEARNING: SEMI-SUPERVISED, ONLINE, AND ACTIVE LEARNING

By

Wael Emara
M.Sc., 2006, in Electrical and Computer Engineering, University of Louisville, USA
B.Sc., 2000, and M.Sc., 2003, in Electronics and Electrical Communications Engineering, Mansoura University, Egypt.

A Dissertation Approved On

11/26/2012
Date

by the following Dissertation Committee:

Mehmed Kantardzic, Ph.D.
(Dissertation Director)

Adel Elmaghraby, Ph.D.

Olfa Nasraoui, Ph.D.

Patrick Shafto, Ph.D.

Jacek Zurada, Ph.D.

ACKNOWLEDGEMENTS

I would like to thank Dr. Mehmed Kantardzic, my advisor, for all his help and support and for the great stimulating discussions and comments. I have no doubt in my mind that if it was not for Dr. Kantardzic's patience, guidance, and caring, this work would have never seen the day light. I would like to extend my thanks and gratitude to Dr. Adel Elmaghraby for all his effort to provide a healthy research environment in the computer science department.

I also would like to express my deep gratitude for my parents and sister for all the support they provided me through the years.

Finally, there are no words to describe my debt and appreciation towards my dear wife. Without her unconditional support and encouragement, this work would have never came through. Thank you dear and I wish I could someday pay back my debt towards you.

ABSTRACT

A SUBMODULAR OPTIMIZATION FRAMEWORK FOR NEVER-ENDING LEARNING: SEMI-SUPERVISED, ONLINE, AND ACTIVE LEARNING

Wael Emara

November 26, 2012

The revolution in information technology and the explosion in the use of computing devices in people's everyday activities has forever changed the perspective of the data mining and machine learning fields. The enormous amounts of easily accessible, information rich data is pushing the data analysis community in general towards a shift of paradigm. In the new paradigm, data comes in the form a stream of billions of records received everyday. The dynamic nature of the data and its sheer size makes it impossible to use the traditional notion of offline learning where the whole data is accessible at any time point. Moreover, no amount of human resources is enough to get expert feedback on the data.

In this work we have developed a unified optimization based learning framework that approaches many of the challenges mentioned earlier. Specifically, we developed a *Never-Ending Learning* framework which combines incremental/online, semi-supervised, and active learning under a unified optimization framework. The established framework is based on the class of submodular optimization methods.

At the core of this work we provide a novel formulation of the Semi-Supervised Support Vector Machines (S^3VM) in terms of submodular set functions. The new formulation overcomes the non-convexity issues of the S^3VM and provides a state of the art solution that is orders of magnitude faster than the cutting edge algorithms in the literature.

Next, we provide a stream summarization technique via exemplar selection. This technique makes it possible to keep a fixed size exemplar representation of a data stream

that can be used by any label propagation based semi-supervised learning technique. The compact data steam representation allows a wide range of algorithms to be extended to incremental/online learning scenario. Under the same optimization framework, we provide an active learning algorithm that constitute the feedback between the learning machine and an oracle.

Finally, the developed *Never-Ending Learning* framework is essentially transductive in nature. Therefore, our last contribution is an inductive incremental learning technique for incremental training of SVM using the properties of local kernels. We demonstrated through this work the importance and wide applicability of the proposed methodologies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

CHAPTER I

INTRODUCTION

In today's world, there is an explosive growth of information technology in terms of data generation. The data produced in 2006 alone was estimated to be 161 Exabyte (Billion GByte) [33]. The statistical properties of the generated data are dynamic in nature. Data distributions are evolving, new patterns are emerging/vanishing, and interesting concepts are shifting. Biometric identification systems, multimedia search engines, fraud detection systems, web content classification systems, robotics are some examples of applications that produce large dynamic data volumes. The dynamic nature of the data makes the use of traditional learning paradigms, in which models are learned once using a training data set and then used forever, not useful anymore. In contrast, learning paradigms that mimics learning in humans and other animals, where learning is an ongoing process, is more appropriate for the future of the machine learning field. We refer to such learning paradigms as *Never-Ending Learning* as discussed next.

I.1 Never-Ending Learning

Never-Ending Learning paradigms, also called lifelong learning, have been widely acknowledged in the literature [50, 66, 68, 69, 70]. The basic concept was articulated to handle the proposition that learning tasks in humans are not isolated from each other. In contrast, humans tend to use all their experience from previous tasks to make the learning of a new task easier and faster. As such, most of the literature in this topic study machine learning algorithms capable of transferring knowledge across multiple learning tasks. This is applied in situations where a learning machine faces a never-ending sequence of learning tasks over its entire lifetime. The never-ending learning process happens incrementally where learning occurs at every time step. If such tasks are related, the never-ending learning

paradigm provides the opportunity for synergy. This paradigm has been mostly applied to robotics reinforcement-learning tasks [69] and recently to knowledge extraction from the web [7].

The basic characteristics of never-ending learning have been considered throughout several research directions in machine learning. These directions include transfer learning, multitask learning, incremental learning, sample bias correction, and concept drift analysis. While transfer learning and multitask learning both share the notion of applying knowledge learned from one or more tasks to related tasks, they differ in the direction of knowledge transfer between tasks. Transfer learning [67, 77, 78] performs the knowledge transfer in unidirectional manner from older learned related tasks to a currently pursued one. On the other hand, multitask learning [3, 4, 5, 15] uses bidirectional transfer of knowledge between tasks as it attempts to learn multiple related tasks together. As opposed to transfer and multitask learning, where several tasks are involved at each time step, incremental learning [16, 32, 59] is concerned with only one task (itself) throughout its lifetime. This is why incremental learning can be thought of as a special case of transfer learning where a task transfer knowledge to itself by updating the current model using subsets of training data the become available with time. Other issues that face never-ending learning are those related to the differences in the statistical properties of training data between different tasks or the evolution of single data set with time.The former issue is studied through sample bias correction [11, 12, 29, 39] where the distribution of test and training data are different, while the latter is examined through concept drift analysis [24, 44, 45, 75].

An important aspect for the never-ending learning paradigm is the quality and usefulness of the used training data. While labeled data grant ultimate information content for learning, it is well known that the labeling process of training data is a costly and time consuming process. On the other hand, vast amounts of unlabeled data are being collected all the time. This issue raised the question about the possibility to choose ahead of time some subsets of training data that are expected to be of most usefulness for the learning task at hand. Then, these are the data subsets that will be labeled. This is referred to as active learning [13, 50]. Another question is that if using unlabeled data along with labeled data will improve the learning process. It turned out using unlabeled along with labeled data

reduces dramatically the cost of supervised learning. This issue, called semi-supervised learning, has widely attracted the attention of the machine learning research community recently [50, 81]. Beside being provably beneficial for semi-supervised learning, using unlabeled samples has shown significance in transfer learning [58], multitask learning [4], and sample bias correction [29, 39].

Despite being a broad learning paradigm that encompasses many machine learning research directions, any algorithm for never-ending learning is defined by some basic characteristics that will define its scope. These characteristics include: the number of learning tasks involved (single or multiple), availability of training data labels (labeled, unlabeled, or both), the nature of the data spaces (open or closed data spaces), and the assumption about the training data distributions and concept definitions (fixed or evolving).

I.2 Contributions and Structure of the Dissertation

The goal of this dissertation is to present a unified single-task (classification) and closed-space *Never-Ending Learning* framework for streaming data. We define a *never-ending learning* machine by: (a) its ability to process data sequentially in streams, (b) the streaming data is partially labeled, and (c) the learning process should possess a mechanism of feedback from the machine to the supervising oracle in the form of active learning. A *never-ending learning* framework should be able to run for extended periods of time with little to no supervision from an oracle. Moreover, it must exhibit constant storage and time complexities.

Support vector machines (SVM) [72, 73] are powerful and popular machine learning tools due to their good generalization performance. Therefore, research has been active into extending them to the semi-supervised learning paradigm [17]. The difficulty of the problem and challenge to solve it accurately and efficiently has inspired us to build our *never-ending learning* framework around the Semi-Supervised Support Vector Machine model (S^3VM).

Chapter III represents the core of the dissertation where we developed a novel formulation of the S^3VM in terms of standard quadratic programming optimization. The

new formulation overcomes the non-convexity issues of the S^3VM and makes it feasible to use off the shelf optimization techniques to solve such a hard problem. One important contribution of the new quadratic programing formulation is that it uncovers an intuitive relationship between the two major categories of semi-supervised learning, namely the low-density separation and the graph-based methods. This relationship will help migrate ideas and algorithms between both categories.

The second major contribution in Chapter III is the introduction of submodular set functions optimization to the problem of semi-supervised learning. We transformed the quadratic programming S^3VM into a submodular optimization function. We then showed that in this new form, we could achieve statistically comparable classification accuracies to the state of the art S^3VM algorithms. Meanwhile, the achieved execution times are orders of magnitude better than the state of the art.

In Chapter IV we present a stream summarization algorithm via exemplars selection. This algorithm provides our contributions in Chapter III with a mechanism for achieving constant time and storage complexity. This is achieved by choosing exemplars that preserve the inherent data structures necessary for semi-supervised learning. Once again the exemplar selection algorithm is formulated under the submodular optimization framework which makes it very efficient. One major advantage of the developed stream summarization algorithm is that it is not specific for the S^3VM problem but rather general in the sense that the preserved properties in the exemplars can be used by any label propagation learning algorithm. Under the same submodular optimization framework, in Chapter V we provide an active learning algorithm that constitute the feedback between the learning machine and an oracle.

All the provided algorithms in the previous chapter work under the transductive learning paradigm. In transductive learning only the labels of the unlabeled samples are produced as the output of the learning process. However, no model is available to classify new never seen data. In Chapter VI we present an inductive incremental learning algorithm for supervised SVM. This algorithm uses the properties of local kernels (e.g. RBF) to perform local and efficient updates to an SVM model. The main contribution of this chapter is that we have proved analytically and illustrated experimentally that the well know local-

ity of the RBF-SVM during the testing stage is actually transferrable to the training stage. The provided contributions in Chapter VI complements the proposed *never-ending learning* framework by providing a methodology to keep an inductive model of the data stream. Finally, Chapter VII provides a summary of the conclusions of the dissertation and some directions for future work.

CHAPTER II

PRELIMINARIES

In this chapter we provide an introduction to the basics of Support Vector Machines (SVM) in the supervised and semi-supervised learning paradigms. We also present the algorithms we used to implement the supervised SVM. Moreover, we present an introduction to submodular set function optimization which constitutes the backbone for many of the algorithms developed in this dissertation.

II.1 Background of Support Vector Machines

In order to introduce Support Vector Machines (SVMs), we will first start with discussing linear classifiers and learning in feature spaces.

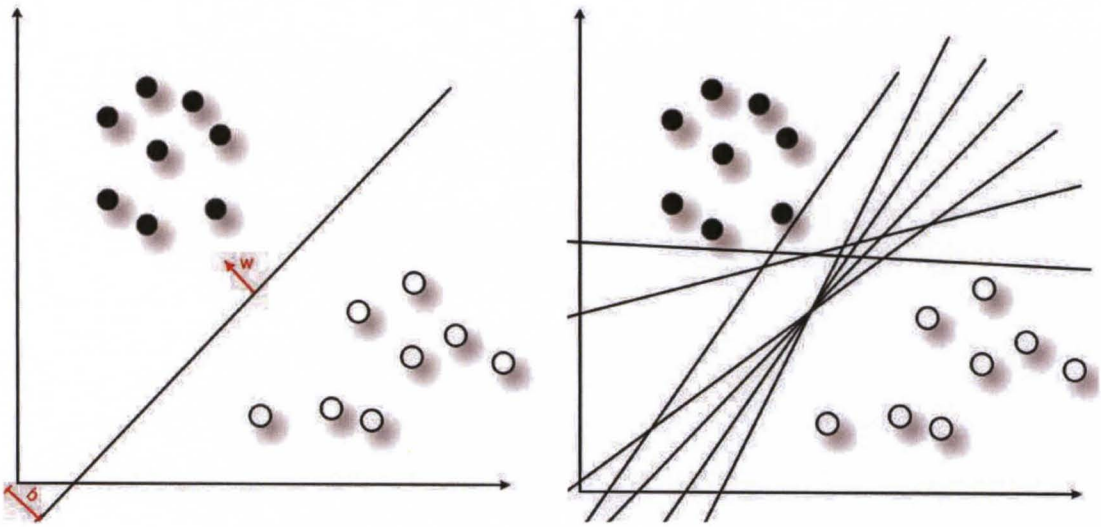
II.1.1 Linear Classifiers

Given a training data set $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, where $\mathbf{x}_i \in \mathbb{R}^n$ and $y_i \in \{+1, -1\}$. A binary classifier is a real valued function $f : \mathbb{R}^n \to \mathbb{R}$ for which $sign(f(\mathbf{x}_i)) = y_i$. A function $f(\mathbf{x})$ is called a linear classifier when it can be written in the following form:

$$\begin{aligned} f(\mathbf{x}) &= \langle \mathbf{w}, \mathbf{x} \rangle + b \\ &= \sum_{i=1}^{n} w_i x_i + b \end{aligned} \tag{II.1}$$

where $\mathbf{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ are the classifier parameters. Linear classifiers can also be interpreted geometrically as a hyperplane in $\mathbb{R}^{n-1}$, where $\mathbf{w}$ is the normal vector to the hyperplane and b is its distance from the origin.

Figure (II.1-a) shows a linear separating hyperplane parameterized by $(\mathbf{w}, b)$. Figure (II.1-b) illustrates the fact that there are many hyperplanes that are capable of separating



(a) (b)

Figure II.1: (a) Illustration of the concept of a linear hyperplane and a depiction of its parameters $\mathbf{w}$ and b. (b) Depiction of a subset of the possible hyperplanes that my separate the two classes.

the shown two classes. Choosing one hyperplane is an issue that will be discussed when we get to support vector machines.

II.1.2 Learning in Feature Space

For many of the real world applications linear separation of the data may not be possible. This leads to the necessity of non-linear separators (decision functions). Non-linear separators my be viewed as linear separators in a different space. Therefore the strategy used in machine learning to accommodate non-linear separators involves mapping the input data $\mathbf{x}_i$ to a new space called the *Feature Space*, where they can be linearly separated.

Figure II.2 shows how non-linear separation in the original data space can be achieved via a non-linear mapping to a feature space via a mapping function $\Phi(\mathbf{x})$. Because a linear

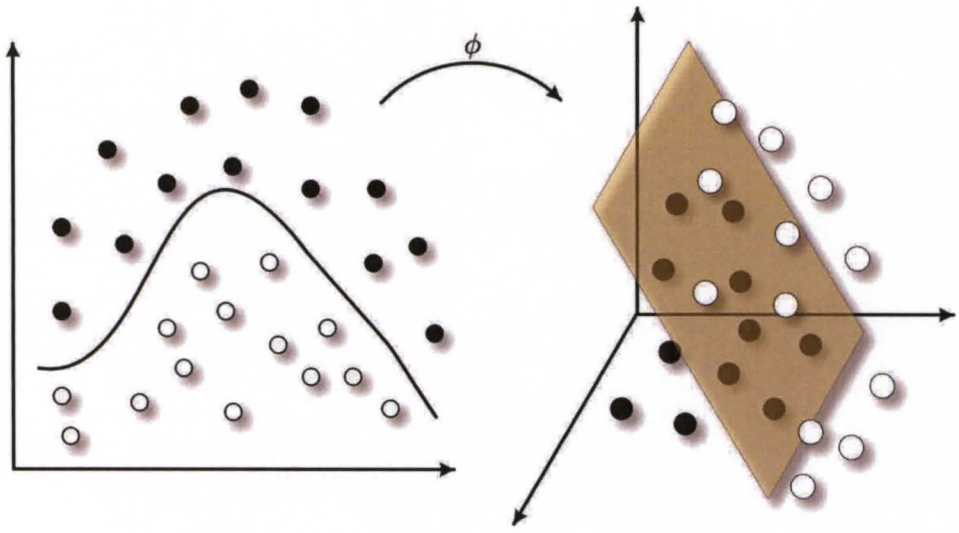


Figure II.2: An illustration of how non-linear separation in the original data space (left) is possible by mapping the data into a higher dimensional feature space (right) where linear separation is possible.

separator is now constructed in the feature space, Eq. (II.2) will have the following form:

$$\begin{aligned} f(\mathbf{x}) &= \langle \mathbf{w}, \mathbf{\Phi}(\mathbf{x}) \rangle + b \\ &= \sum_{i=1}^{n} w_i \Phi(x_i) + b \end{aligned} \tag{II.2}$$

II.1.3 Supervised Support Vector Machines

Support Vector Machines (SVM) were first introduced by Vladimir Vapnik in 1982 [71]. Formally, for a training data set $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, where $\mathbf{x}_i \in \mathbb{R}^n$ and $y_i \in \{+1, -1\}$, SVM in its linear form is a hyperplane that separates the two classes in S with a maximum margin. The samples of S that lie closest to the SVM hyperplane are called *Support Vectors*. The name *Support Vector Machines* is coined to indicate that the obtained hyperplane depends only on the support vectors. To formalize the SVM problem, the margin is defined as follows:

Definition *The functional margin γ_{f_i} of a training sample $(\mathbf{x_i}, y_i)$ with respect to a hyperplane $(\mathbf{w}, b)$ is defined as:*

$$\gamma_{f_i} = y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b)$$

From the definition of a functional margin, it is clear that if a sample $(\mathbf{x_i}, \mathbf{y_i})$ is correctly classified then its functional margin is positive, $\gamma_{f_i} > 0$.

Among all the hyperplanes that separate the training data, SVM finds the one with the maximum margin. However, the functional margin γ_{f_i} can be made arbitrarily large by scaling $\mathbf{w}$ and b. Therefore, a normalized version of γ_{f_i} can be used instead. This normalized margin is called the *Geometric Margin*, γ_{g_i} [23]:

$$\gamma_{g_i} = \frac{\gamma_{f_i}}{\|\mathbf{w}\|} = \frac{y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b)}{\|\mathbf{w}\|} \tag{II.3}$$

The geometric margin measures the Euclidean distances of the data samples from the decision boundary in the input space. The SVM goal of finding the hyperplane that maximizes the geometric margin can be formulated as the solution of the following optimization problem:

$$\begin{aligned} &\max_{\mathbf{w},b} \ \gamma_g \\ s.t. \quad &\frac{y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b)}{\|\mathbf{w}\|} \geq \gamma_g, \qquad i = 1, \ldots, l \end{aligned} \tag{II.4}$$

which can be solved by minimizing $\|\mathbf{w}\|$ instead of maximizing the margin as follows [23]:

$$\begin{aligned} &\min_{\mathbf{w},b} \ \|\mathbf{w}\| \\ s.t. \quad &y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b) \geq 1, \qquad i = 1, \ldots, l \end{aligned} \tag{II.5}$$

This follows from the fact that $\gamma_g = \frac{1}{\|\mathbf{w}\|}$. To show how this fact came about we will again consider the functional margin

$$\gamma_{f_i} = y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b) \tag{II.6}$$

Substituting two support vectors $\mathbf{x}^+$ and $\mathbf{x}^-$ as some positive and negative samples, we obtain the following

$$+1(\langle \mathbf{w}, \mathbf{x}^+ \rangle + b) = \gamma_f \tag{II.7}$$

$$-1(\langle \mathbf{w}, \mathbf{x}^- \rangle + b) = \gamma_f \tag{II.8}$$

Adding (II.7) and (II.8) and using the fact that $\gamma_f = \gamma_g \|\mathbf{w}\|$, we obtain the following:

$$2\gamma_g \|\mathbf{w}\| = (\langle \mathbf{w}, \mathbf{x}^+ \rangle - \langle \mathbf{w}, \mathbf{x}^- \rangle) \tag{II.9}$$

Therefore, for $\gamma_f = 1$

$$(\langle \mathbf{w}, \mathbf{x}^+ \rangle - \langle \mathbf{w}, \mathbf{x}^- \rangle) = 2 \tag{II.10}$$

and hence $\gamma_g = \frac{1}{\|\mathbf{w}\|}$. Figure II.3 provides an illustrative example of a simple linear SVM with depictions of both support vectors and geometric margins.

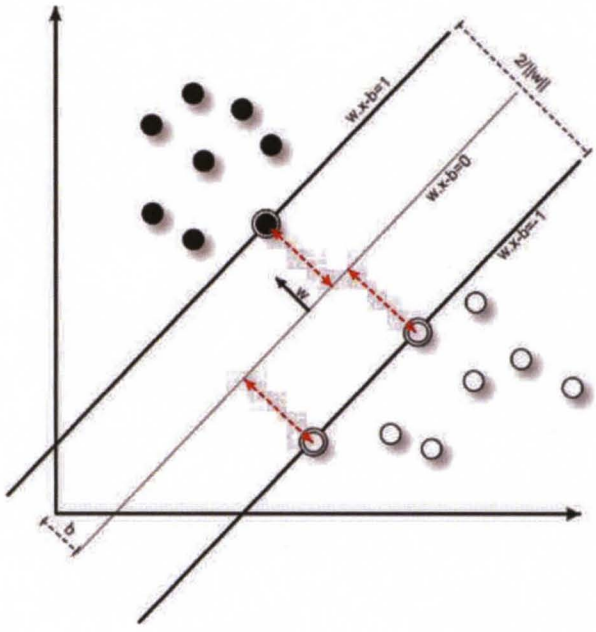


Figure II.3: An illustration of a simple linear SVM parameterized by $(\mathbf{w}, b)$. The circled data samples are the support vectors and the associated dotted arrows depict the margin of each support vector.

II.1.4 Optimization in Support Vector Machines

Many optimization formulations have been proposed for SVMs [56, 57, 71]. A key property of SVMs is that the optimization formulations are often convex and therefore a global optimum can be guaranteed. This section presents basic optimization formulations for SVMs in the cases where the training data examples may and may not be linearly separable.

II.1.4.1 Separable Case

Using the definition of the inner product, the SVM problem for linearly separable training data can be written as:

$$\begin{aligned} &\min_{\mathbf{w},b} \tfrac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle \\ s.t. \quad & y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b) \geq 1, \qquad i = 1, \ldots, l \end{aligned} \tag{II.11}$$

The solution of (II.11) is easier to obtain for the dual representation of the problem. Furthermore, the dual representation is also important for employing kernels with SVMs to obtain non-linear decision functions. The procedure to find the dual representation of (II.11) consists of finding its Lagrangian primal representation and then getting the dual of the Lagrangian.

In the following, we give the definition of the *Lagrangian* and *Kuhn-Tucker* theorem which gives necessary and sufficient conditions for an optimum solution to the optimization problem.

Definition *Given the optimization problem*

$$\begin{aligned} &\min_{\mathbf{X}} f(\mathbf{X}) \\ s.t. \quad & g_i(\mathbf{X}) \leq 0,, \qquad i = 1, \ldots, k \\ & h_j(\mathbf{X}) = 0,, \qquad j = 1, \ldots, m \end{aligned}$$

The Lagrangian primal is defined as:

$$\begin{aligned} \mathbf{L}(\mathbf{X}, \alpha, \beta) &= f(\mathbf{X}) + \sum_{i=1}^{k} \alpha_i g_i(\mathbf{X}) + \sum_{j=1}^{m} \beta_j h_j(\mathbf{X}) \\ &= f(\mathbf{X}) + \alpha \mathbf{g}(\mathbf{X}) + \beta \mathbf{h}(\mathbf{X}) \end{aligned}$$

where α_i and β_j are called the Lagrangian multipliers.

The Lagrangian is usually used to transform a constrained optimization problem into a non-constrained one. The *Kuhn-Tucker* theorem gives necessary and sufficient conditions for the optimal solution in terms of the Lagrangian.

Theorem 1. *(Kuhn-Tucker) Given the optimization problem [23]*

$$\begin{aligned} & \min_{\mathbf{X}} f(\mathbf{X}) && \mathbf{X} \in \Omega \\ s.t. \quad & g_i(\mathbf{X}) \leq 0, , && i = 1, \ldots, k \\ & h_j(\mathbf{X}) = 0, && j = 1, \ldots, m \end{aligned}$$

with the convex domain $\Omega \subseteq \mathbb{R}^n$, *$f$ is convex, and g_i and h_j are affine.* $\mathbf{X}^*$ *is an optimum solution if and only if there exist* α^*, β^* *such that*

$$\begin{aligned} \frac{\partial \mathbf{L}(\mathbf{X}^*, \alpha^*, \beta^*)}{\partial \mathbf{X}} &= 0 \\ \alpha_i^* g_i(\mathbf{X}^*) = 0, \quad & i = 1, \ldots, k \\ g_i(\mathbf{X}^*) \leq 0, \quad & i = 1, \ldots, k \\ \alpha_i^* \geq 0, \quad & i = 1, \ldots, k \end{aligned}$$

Using the definition of a Lagrangian, the Lagrangian primal of (II.11) is

$$\mathbf{L}(\mathbf{w}, b, \alpha) = \frac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle - \sum_{i=1}^{l} \alpha_i [y_i(\langle \mathbf{w_i}, \mathbf{x_i} \rangle + b) - 1] \tag{II.12}$$

Applying the first condition of the Kuhn-Tucker theorem, where the parameters $\mathbf{X}$ in the theorem corresponds to $\mathbf{w}$ and b in (II.11),

$$\frac{\partial \mathbf{L}(\mathbf{w}, b, \alpha)}{\partial \mathbf{w}} = 0 \implies \mathbf{w} = \sum_{i=1}^{l} y_i \alpha_i \mathbf{x_i} \tag{II.13}$$

$$\frac{\partial \mathbf{L}(\mathbf{w}, b, \alpha)}{\partial \mathbf{b}} = 0 \implies \sum_{i=1}^{l} y_i \alpha_i = 0 \tag{II.14}$$

where $\alpha_i \geq 0$ are the Lagrangian multipliers associated with each data sample. Substituting (II.13) and (II.14) in the Lagrangian primal (II.12), we get the following:

$$\begin{aligned} \mathbf{L}(\mathbf{w}, b, \alpha) &= \frac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle - \sum_{i=1}^{l} \alpha_i [y_i(\langle \mathbf{w_i}, \mathbf{x_i} \rangle + b) - 1] \qquad \text{(II.15)} \\ &= \frac{1}{2} \sum_{i=1}^{l} y_i y_j \alpha_i \alpha_j \langle \mathbf{x_i}, \mathbf{x_j} \rangle - \sum_{i=1}^{l} y_i y_j \alpha_i \alpha_j \langle \mathbf{x_i}, \mathbf{x_j} \rangle - b \sum_{i=1}^{l} y_i \alpha_i + \sum_{i=1}^{l} \alpha_i \\ &= \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} y_i y_j \alpha_i \alpha_j \langle \mathbf{x_i}, \mathbf{x_j} \rangle \end{aligned}$$

Therefore, instead of solving (II.11) we solve the following dual quadratic programming problem,

$$\begin{aligned} \max_{\alpha} \; & f(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} y_i y_j \alpha_i \alpha_j \langle \mathbf{x_i}, \mathbf{x_j} \rangle \\ s.t. \quad & \sum_{i=1}^{l} y_i \alpha_i = 0, \qquad i = 1, \ldots, l \\ & \alpha_i \geq 0, \qquad i = 1, \ldots, l \end{aligned} \tag{II.16}$$

Finally, the optimal hyperplane $\mathbf{w}^*$ is obtained by substituting the solution of the dual problem α^* in (II.13), $\mathbf{w}^* = \sum_{i=1}^{l} y_i \alpha_i^* \mathbf{x_i}$.

The Kuhn-Tucker theorem also states that the following set of equations must hold between the optimal solution α^*, $\mathbf{w}^*$, and b:

$$\alpha_\mathbf{i}^* [1 - y_i (\langle \mathbf{w_i^*}, \mathbf{x_i} \rangle + b^*)] = 0 \qquad i = 1, \ldots, l \tag{II.17}$$

These equations imply that $\alpha_\mathbf{i}^*$ are non-zero for only those $\mathbf{x_i}$'s with functional margins equal to one. Those $\mathbf{x_i}$'s with non-zero $\alpha_\mathbf{i}^*$ are called support vectors. Therefore, the SVM depends solely on the support vectors.

II.1.4.2 Non-separable Case

Thus far we have discussed only the SVM optimization case where the training data is assumed to be linearly separable. In this section we show how to handle the linearly non-separable case. Assuming that a data sample $\mathbf{x_i}$ has a label $+1$, however the SVM parameterized by $(\mathbf{w}, b)$ classify it to be labeled -1, that is

$$y_i (\langle \mathbf{w_i^*}, \mathbf{x_i} \rangle + b^*) = -\xi_i \qquad , \xi_i > 0 \tag{II.18}$$

For the constraint $y_i (\langle \mathbf{w_i^*}, \mathbf{x_i} \rangle + b^*) \geq 1$ to be satisfied, we need to add a slack variable ξ_i to each misclassified data sample $\mathbf{x_i}$,

$$\begin{aligned} \min_{\mathbf{w}, b, \xi} \; & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{l} \xi_i \\ s.t. \quad & y_i (\langle \mathbf{w}, \mathbf{x_i} \rangle + b) + \xi_i \geq 1 \\ & \xi_i \geq 0, \qquad i = 1, \ldots, l \end{aligned} \tag{II.19}$$

where C is a fixed parameter to penalize allowing a misclassified data samples. Otherwise, the solution of the optimization problem will take the easy path of assigning slack variables to all the data samples and non of them will be correctly classified. Assigning the values of ξ_i's is done through a function called the *Loss Function*. The loss function should assign $\xi_i = 0$ for correctly classified samples and $\xi_i > 0$ for misclassified ones. A natural choice for the loss function that emerges from the constraints in (II.19) is known as *Hinge* loss function,

$$\xi_i = \ell((\mathbf{w}, \mathbf{b}), (\mathbf{x_i}, y_i)) = max(0, 1 - y_i(\langle \mathbf{w_i}, \mathbf{x_i} \rangle + b)) \tag{II.20}$$

Using the concept of loss functions, the constrained optimization problem in (II.19) can be transformed to an unconstrained form,

$$\min_{\mathbf{w},b,\xi} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l} \ell((\mathbf{w}, \mathbf{b}), (\mathbf{x_i}, y_i)) \tag{II.21}$$

Following the same procedure we employed in the separable case on (II.19), the dual optimization problem is as follows:

$$\begin{aligned} &\max_{\alpha} f(\alpha) = \textstyle\sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i=1}^{l} y_i y_j \alpha_i \alpha_j \langle \mathbf{x_i}, \mathbf{x_j} \rangle && && \text{(II.22)} \\ s.t. \quad & \textstyle\sum_{i=1}^{l} y_i \alpha_i = 0, && i = 1, \ldots, l \\ & C \geq \alpha_i \geq 0, && i = 1, \ldots, l \end{aligned}$$

The solution is reached when all Karush-Kuhn-Tucker (KKT) conditions are being satisfied over all the training samples. KKT conditions are described as follows [57]:

$$\begin{aligned} \alpha_i = 0 &\Longrightarrow y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b) \geqslant 1 \\ 0 < \alpha_i < C &\Longrightarrow y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b) = 1 \\ \alpha_i = C &\Longrightarrow y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b) \leqslant 1 \end{aligned} \tag{II.23}$$

II.1.4.3 Non-linear SVM

As we discussed in Section II.1.2, non-linear decision functions can be learned by using a non-linear function to map the data to some feature space and then construct a linear classifier in that feature space. Generally, the feature space has higher dimensionality

than the original data space. This suggests that the classifier constructed in the feature space will be more computationally expensive to find and utilize. However, as the training data involvement in SVMs is through inner product operations, as shown in (II.22), it is feasible to construct classifiers in high dimensional feature spaces, without compromising the computational complexity, through what is know as the *Kernel Trick* [23].

Definition *Given two data samples* $\mathbf{x}$ *and* $\mathbf{z}$ *from the input space* $\mathbf{X} \subseteq \mathbb{R}^n$*, the function* K *that returns the inner product between their images in the feature space is known as the* ***Kernel Function****:*

$$K(x, z) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{z}) \rangle$$

where Φ *is a mapping from the input data space to the feature space.*

Some of the commonly used kernels are:

- **Identity Kernel**

$$K(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle$$

- **Radial Basis Function Kernel** (with a width parameter γ)

$$K(\mathbf{x}, \mathbf{z}) = exp(-\gamma \|\mathbf{x} - \mathbf{z}\|)$$

- **Polynomial Kernel** (of order p)

$$K(\mathbf{x}, \mathbf{z}) = (\langle \mathbf{x}, \mathbf{z} \rangle + 1)^p$$

Using kernels with SVMs involves replacing any dot product of data sample with the output of the kernel function.

II.1.4.4 Sequential Minimal Optimization

In this section we will present one of the simplest and most efficient techniques to perform training of SVMs, known as *Sequential Minimal Optimization (SMO)*. The rest of this work employs SMO as the core SVM training technique. SMO [57] is a simple algorithm to solve the QP problem arising in training SVMs. It belongs to a family of

algorithms that addresses training SVMs on large data sets by breaking the QP problem into smaller manageable ones [56, 71]. SMO takes the idea of breaking the QP problem to its extreme by choosing to solve the smallest possible QP problem; At each iteration, two Lagrange multipliers are jointly optimized and the SVM is updated accordingly. The advantage of optimizing two Lagrange multipliers lies in the possibility to do it analytically and therefore fast.

Starting with $(\mathbf{x_1}, y_1, \alpha_1^{old})$ and $(\mathbf{x_2}, y_2, \alpha_2^{old})$ and considering the constraints in (II.22), it is clear that the space of possible values for α_1 and α_2 is actually, due to the inequality constraints $C \geq \alpha_i \geq 0$, a square with side length C. Moreover, we can see that the summation constraint, $\sum_{i=1}^{l} y_i \alpha_i = 0$, forces the values of α_1 and α_2 to lie on a diagonal line. These insights narrow down the space of the solution, see Fig. II.4.

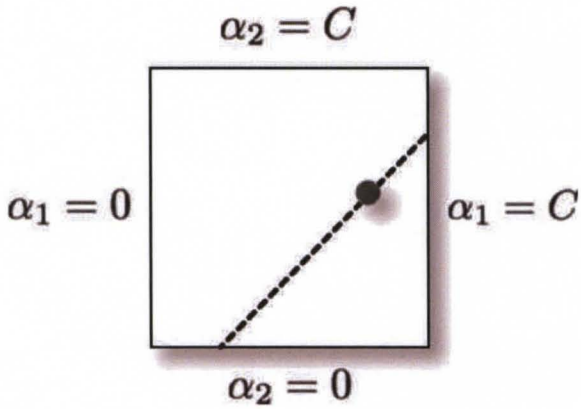


Figure II.4: An depiction of the domain values of α_1 and α_2 imposed by the constraints in (II.22). The square is a result of the inequality constraint $C \geq \alpha_i \geq 0$ and this domain is even reduced to the dotted diagonal line imposed by the constraint that $\sum_{i=1}^{l} y_i \alpha_i = 0$. The shaded circle is sample solution of the optimization problem.

The algorithm starts by finding the bounds on α_2^{new} which depends on the value $y_1 y_2$. If $y_1 y_2 = -1$, then the bounds on α_2^{new} are:

$$\begin{aligned} L &= max(0, \alpha_2^{old} - \alpha_1^{old}) \\ H &= min(C, C + \alpha_2^{old} - \alpha_1^{old}). \end{aligned} \tag{II.24}$$

If $y_1 y_2 = 1$, then the bounds on α_2^{new} are:

$$\begin{aligned} L &= max(0, \alpha_1^{old} + \alpha_2^{old} - C) \\ H &= min(C, \alpha_1^{old} + \alpha_2^{old}). \end{aligned} \tag{II.25}$$

Next, α_2^{new} is obtained by:

$$\alpha_2^{new} = \alpha_2^{old} - \frac{y_2(E_1 - E_2)}{\eta} \tag{II.26}$$

where η is the second derivative of the objective function (II.22) along the diagonal line:

$$\eta = 2\langle \mathbf{x}_1, \mathbf{x}_2 \rangle - \langle \mathbf{x}_1, \mathbf{x}_1 \rangle - \langle \mathbf{x}_2, \mathbf{x}_2 \rangle \tag{II.27}$$

and E_i is the error of the old SVM on the ith training sample. Then, α_2^{new} is clipped according to its bounds in (II.24) or (II.25).

$$\alpha_2^{new,clipped} = \begin{cases} H & \text{if } H \leqslant \alpha_2^{new} \\ \alpha_2^{new} & \text{if } L < \alpha_2^{new} < H \\ L & \text{if } \alpha_2^{new} \leqslant L \end{cases} \tag{II.28}$$

Finally, α_1^{new} is computed by

$$\alpha_1^{new} = \alpha_1^{old} + y_1 y_2 (\alpha_2^{old} - \alpha_2^{new,clipped}) \tag{II.29}$$

SMO provides two heuristics to select the Lagrange multipliers to be optimized. The first choice heuristic selects the first Lagrange multiplier by iterating over the whole training data set until a multiplier violating the KKT conditions is found then the search start for the second Lagrange multiplier by initializing the second choice heuristic. When the first choice heuristic completes one pass through the entire training data set, it starts iterating over the multipliers that are neither 0 or C (non-bound multipliers). Again after it finishes iterating over the non-bound multipliers, it makes another pass through the whole training data set and so on. The second choice heuristic aims at choosing the second Lagrange multiplier that maximizes the step taken during the optimization. SMO proposes using $|E_1 - E_2|$ as an estimate for the step size. Therefore, a cached error E_i is stored for every non-bound multiplier which is then used by the second choice heuristic to choose the second Lagrange multiplier. See Algorithm 1 for a summary of the SMO algorithm.

Algorithm 1: Sequential Minimal Optimization (SMO) [57]

Function $\alpha_i \longleftarrow$ SMO($(\mathbf{x_i}, y_i), i = 1, \ldots, l$)

Input: $(\mathbf{x_i}, y_i)$, Labeled training pairs.

Output: $\alpha_\mathbf{i}$, Lagrangian multipliers.

```
begin
    Num_Changed_α_i = 0
    Examine_All = 1
    while Num_Changed_α_i > 0 OR Examine_All do
        Num_Changed_α_i = 0
        if Examine_All then
            Loop I over all training examples
                Num_Changed_α_i+ = Examine_Example(I)
        end
        else
            Loop I over examples where α_i is not 0 and not C
        end
        if Examine_All == 1 then
            Examine_All = 0
        end
        else if Num_Changed_α_i == 0 then
            Examine_All = 1
        end
    end
end
```

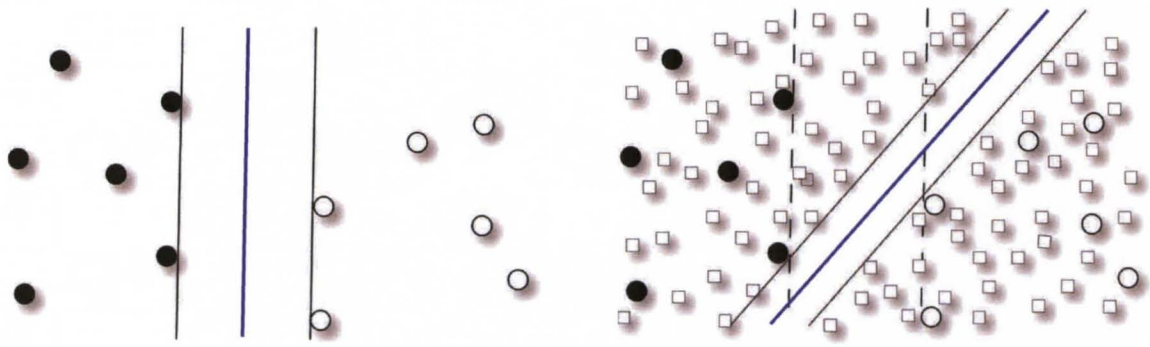
Examine_Example(I): Selects the second Lagrangian multiplier and performs the processes described in Eqn.(II.24-II.29).

II.2 Semi-supervised Learning (SSL)

Traditionally the classification problem in machine learning has been formulated to perform training using only labeled data (feature/label pairs). However, labeled instances are often difficult, expensive, and slow to acquire. Meanwhile, unlabeled instances are abundant and much cheaper to collect. Semi-supervised learning (SSL) addresses the problem of learning from large sets of unlabeled data along with a few labeled samples with the ultimate goal of enhancing the generalization performance of what is learned from the labeled samples using the knowledge from the unlabeled data [81]. The paradigm is also motivated by the fact that SSL is the natural way in which humans learn [82]. Two assumptions form the basis for the usefulness of unlabeled samples in SSL: *The Cluster Assumption and the Smoothness Assumption* [20]. In the cluster assumption, the data of each class is assumed to form a cluster. Therefore, the unlabeled data could be beneficial in finding the boundary of each cluster more accurately [18]. The cluster assumption states that if points are in the same cluster, they are likely to be of the same class. The smoothness assumption is a classic assumption that makes learning from data a possible task. It originally states that if two data samples are close under a certain metric, then so should their corresponding labels. When extended to SSL, the smoothness assumption take into account not only the metric between two data samples but also the density of data between them. Thus the assumption states that if two data samples in a high-density region are close, then so should their corresponding labels. If the data is assumed to lie on a lower dimensional manifold in the original space, the metric used in the smoothness assumption will be defined on that manifold. The smoothness assumption will then be called *The Manifold Assumption* [9].

Much of the literature of SSL algorithms can be categorized according to the assumption they implement; cluster assumption or manifold assumption. Algorithms implementing the cluster assumption are referred to as *"Avoiding Dense Regions"* algorithms. The basic idea behind such algorithms is that no decision function should pass through a cluster [34, 41, 48, 65]. Figure II.5 shows a sample output for an SSL algorithm that implements the cluster assumption. In the figure you see that the decision function avoids

clusters and that unlabeled samples help defining the clusters.



(a) (b)

Figure II.5: An illustration of how SSL algorithms implementing the cluster assumption use unlabeled data. We see that the decision function (bold/blue line) avoids clusters. And that unlabeled samples help defining the clusters. (a) Model constructed using labeled data. (b) Model constructed using labeled/unlabeled data.

Algorithms implementing the manifold (smoothness) assumption are referred to as "graph based" algorithms. In such algorithms, a graph is defined where the nodes are data instances (labeled/unlabeled) and edges connecting the nodes reflect the similarity between the instances. The algorithms can generally be viewed as estimating a function on the constructed graph. This estimate should be close to the labels of the labeled samples and meanwhile be smooth over the whole graph [8, 9, 40, 77, 79]. Figure II.6 depicts an example for a graph based SSL algorithm.

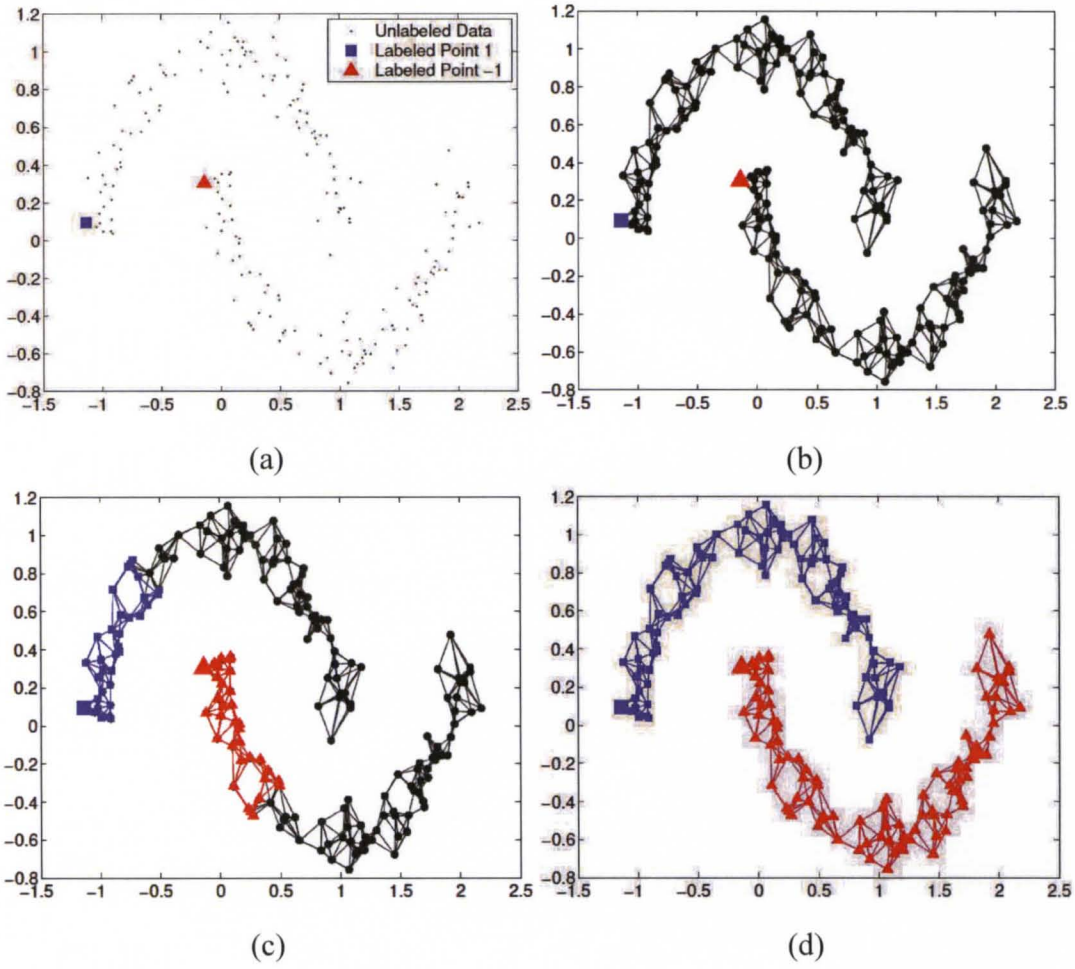


Figure II.6: Example of graph-based SSL algorithms [74]. (a) Labeled/Unlabeled Data. (b) Graph construction. (c) Intermediate step illustrating how the unlabeled samples are assigned to labels. In this case through neighborhood propagation. (d) Final labeling of all unlabeled samples.

Another important taxonomy for SSL algorithms is transductive versus inductive categorization. A learning machine is transductive if it only makes inference about the training data (labeled/unlabeled samples) and can not extend this inference to unseen data. Graph based SSL algorithms are often transductive. On the other hand, inductive learning machines can naturally handle unseen data.

II.3 Semi-Supervised Support Vector Machines (S^{3}VM)

Now that we have introduced the basic concepts of semi-supervised learning, we will present how such ideas are used in the SVM framework. While SVMs work by estimating a hyperplane that maximizes the margin between labeled samples of different classes, S^{3}VM extends the same idea by maximizing the margin jointly for labeled and unlabeled samples. This way S^{3}VM learns a decision boundary that avoids data-dense regions while conforming to the labeled samples. In other words, S^{3}VM is one of the techniques that implement the *cluster assumption* discussed earlier. Figure II.5 shows how S^{3}VM maximizes the margin between classes using both labeled and unlabeled samples.

The major body of work on S^{3}VM is based on the idea of solving a standard SVM while treating unknown labels as additional variables [17]. Formally speaking, if we are given a *partially labeled* data set $S_{lu} = \{(\mathbf{x_i}, y_i)\}_{i=1}^{l} \bigcup \{\mathbf{x_i}\}_{i=l+1}^{l+u}$ where $\mathbf{x_i} \in \mathbb{R}^n$, $y_i \in \{+1, -1\}$, l and u are the number of labeled and unlabeled samples, respectively. The linear S^{3}VM learning problem is to find the solution of the following optimization problem for both the hyperplane parameters $(\mathbf{w}, b)$ and the labels of the unlabeled samples denoted by the vector $\mathbf{y_u} = [y_{l+1} \cdots y_{l+u}]$,

$$\min_{(\mathbf{w},b),\mathbf{y_u}} \mathcal{J}((\mathbf{w}, b), \mathbf{y_u}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{i=l} \ell_l((\mathbf{w}, b); (\mathbf{x_i}, y_i)) + C^* \sum_{i=l+1}^{i=l+u} \ell_u((\mathbf{w}, b); (\mathbf{x_i})) \tag{II.30}$$

where the loss functions for unlabeled samples ℓ_u and labeled samples ℓ_l are defined as follows:

$$\ell_l((\mathbf{w}, b); (\mathbf{x_i}, y_i)) = max\ (0, 1 - y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b)) \tag{II.31}$$

$$\ell_u((\mathbf{w}, b); (\mathbf{x_i})) = \max_{y_i \in \{-1,+1\}}\ (0, 1 - y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b)) \tag{II.32}$$

The first two terms in the objective function $\mathcal{J}$ in (II.30) correspond to a standard SVM and the third term is where the unlabeled data is incorporated. The loss over labeled and unlabeled samples is controlled by two parameters C and C^*, which reflect the confidence in the labels and the cluster assumption respectively.

It is popular in the literature to solve the problem in (II.30) under a class balancing

constraint,

$$\frac{1}{u}\sum_{i=l+1}^{i=l+u} max(y_i, 0) = r \tag{II.33}$$

where r is the ratio of the unlabeled samples that should be labeled $+1$ [42]. This balancing constraint helps overcome the problem of getting totally unbalanced solutions where all the unlabeled samples are assigned to one label. Since the ratio r is not known, it is usually estimated from the labeled samples.

Algorithms that solve (II.30) can be broadly divided into *Combinatorial* and *Continuous* optimization algorithms.

II.3.1 Combinatorial Optimization for S^{3}VM

In combinatorial optimization algorithms, for a given $\mathbf{y_u}$, the optimization for $(\mathbf{w}, b)$ is a standard SVM problem. Therefore, if we define a function $\mathcal{I}(\mathbf{y_u})$ such that

$$\mathcal{I}(\mathbf{y_u}) = \min_{(\mathbf{w},b)} \mathcal{J}((\mathbf{w}, b), \mathbf{y_u}) \tag{II.34}$$

the problem will be transformed to minimizing $\mathcal{I}(\mathbf{y_u})$ over a set of binary variables $\mathbf{y_u}$ where each evaluation of $\mathcal{I}(\mathbf{y_u})$ is a standard SVM optimization problem. The techniques in this category varied between branch-and-bound based algorithms [10, 21], local combinatorial search algorithms [42], and deterministic annealing algorithms [63].

II.3.2 Continuous Optimization for S^{3}VM

In continuous optimization algorithms, for a given fixed $\mathbf{w}, b$, the optimal labels of the unlabeled samples is simply obtained by $\mathbf{y_u} = sign(\langle \mathbf{w}, \mathbf{x_i} \rangle + b)$. This elimination of the variables $\mathbf{y_u}$ converts the problem into a continuous optimization problem over $(\mathbf{w}, b)$,

$$\min_{(\mathbf{w},b)} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{i=l} \ell_l((\mathbf{w}, b); (\mathbf{x_i}, y_i)) + C^*\sum_{i=l+1}^{i=l+u} \ell_u((\mathbf{w}, b); (\mathbf{x_i})) \tag{II.35}$$

where

$$\ell_l((\mathbf{w}, b); (\mathbf{x_i}, y_i)) = max\ (0, 1 - y_i(\langle \mathbf{w}, \mathbf{x_i} \rangle + b)) \tag{II.36}$$

$$\ell_u((\mathbf{w}, b); (\mathbf{x_i})) = max\ (0, 1 - |\langle \mathbf{w}, \mathbf{x_i} \rangle + b|) \tag{II.37}$$

The first two terms in (II.35) correspond to a standard SVM, and the last term drives the boundary away from the unlabeled samples. Also this formulation shows why the S^3VM is a hard problem, this is basically due to the non-convexity of the objective function (II.35). Some of the algorithms in this category attempted to use standard optimization such as the gradient descent [18] of a smoothed version of the objective function in (II.35). Other algorithms employed a continuation approach [19] and convex-concave optimization to overcome the non-convexity of the problem.

II.4 Background of Submodular Optimization

Submodular set functions play a central role in combinatorial optimization [35]. To a great extent they are considered the discrete analogue of convex functions in continuous optimization, in the sense of structural properties that can be benefited from algorithmically. They also emerge as a natural structural form in classic combinatorial problems such as maximum coverage and maximum facility location in location analysis [2], as well as max-cut problems in graphs [36]. More recently submodular set functions have become key concepts in machine learning where problems such as feature selection [53] and active learning [47] are solved by maximizing submodular set functions while other core problems like clustering and learning structures of graph-based models have been formulated as submodular set function minimization [22, 52, 54].

II.4.1 Submodularity Definition and Applications

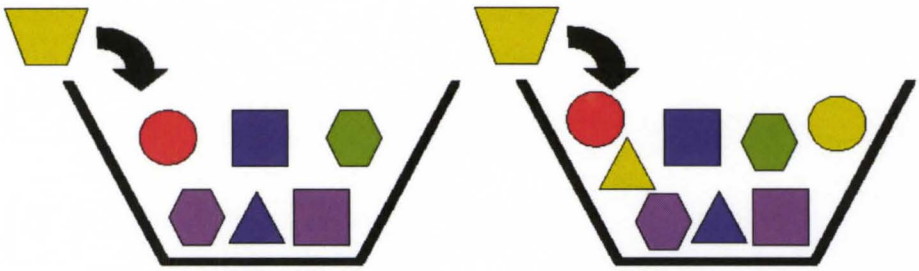
To define submodularity properly, we start by considering a ground set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$. A *set function* defined on $\mathcal{X}$ is a function $f : 2^{\mathcal{X}} \rightarrow \mathbb{R}$. The following definition formalizes the concept of the submodularity of a set function.

Definition 1. *(Submodularity). A set function $f : 2^{\mathcal{X}} \rightarrow \mathbb{R}$ is submodular if and only if, for all $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{X}$ and for each $j \in \mathcal{X} \backslash \mathcal{B}$, it holds that*

$$f(\mathcal{A} \cup \{j\}) - f(\mathcal{A}) \geq f(\mathcal{B} \cup \{j\}) - f(\mathcal{B}).$$

Definition 1 describes the submodularity of a set functions in terms of the *"Law of Diminishing Returns"*, which is a well known principle in economics. In the context of set

functions, it essentially states that adding an element to a smaller set is more influential than adding it to a larger one. To illustrate this idea we provide a simple example of submodular functions in Fig.II.7.



(a) $\mathcal{A} + j$ (b) $\mathcal{B} + j$

Figure II.7: Depiction of the submodular set function $f_{Shapes\&Colors}$, defined as $f_{Shapes\&Colors}(\mathcal{S}) = \#(\text{Distinct Shapes in } \mathcal{S}) + \#(\text{Distinct Colors in } \mathcal{S})$.

The function depicted in Fig.II.7 is a set function $f_{Shapes\&Colors}$ defined over a finite set of shapes and colors. That is for any set $\mathcal{S}$,

$$f_{Shapes\&Colors}(\mathcal{S}) = \#(\text{Distinct Shapes in } \mathcal{S}) + \#(\text{Distinct Colors in } \mathcal{S})$$

First we notice that the set $\mathcal{A}$ in Fig.II.7a is a subset of the set $\mathcal{B}$ in Fig.II.7b. Applying the definition of the $f_{Shapes\&Colors}$ on both sets $\mathcal{A}$ and $\mathcal{B}$, we get the following:

$$f_{Shapes\&Colors}(\mathcal{A}) = 8 \qquad \text{and} \qquad f_{Shapes\&Colors}(\mathcal{B}) = 9$$

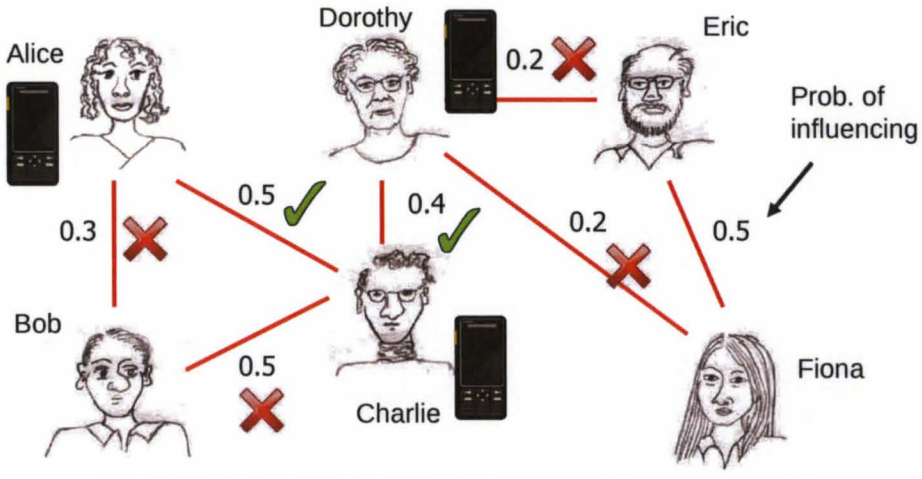
Adding the element j (yellow trapezoid) to both sets will affect their corresponding set values as follows:

$$f_{Shapes\&Colors}(\mathcal{A} \cup j) = 10 \qquad \text{and} \qquad f_{Shapes\&Colors}(\mathcal{B} \cup j) = 10$$

Based on this outcome, we can conclude that $f_{Shapes\&Colors}$ is a submodular set function, as

$$\begin{aligned} f_{Shapes\&Colors}(\mathcal{A} \cup j) &- f_{Shapes\&Colors}(\mathcal{A}) = 2 \\ &\geq \\ f_{Shapes\&Colors}(\mathcal{B} \cup j) &- f_{Shapes\&Colors}(\mathcal{B}) = 1 \end{aligned}$$

The importance of submodularity and the associated *diminishing returns* concept has been attracting increasing attention since many real world combinatorial optimization problems could be modeled as maximizing submodular functions with respect to certain (usually cardinality) constraints [49, 51]. One interesting example is the viral marketing through social networks. The aim of this problem is to select a subset of people in a social network whose influence over of the whole network is maximum [51], see Fig.II.8. Recognizing such influential people is essential in marketing products on social networks where such influential people are given free samples and special offers. This is based on the fact that the expected size of the final influence under many models of influence propagation in networks has been shown to be a submodular function of the set of the initially selected people. Moreover, the budget assigned to any marketing campaign puts a limit on the number of influential people that could be chosen. Therefore, the influence maximization problem can be regarded as maximizing a submodular function subject to cardinality constraints.



Courtesy of Andreas Krause and Carlos Guestrin, 2008 ICML Tutorial: Beyond Convexity – Submodularity in Machine Learning.

Figure II.8: Illustration of viral marketing through social networks [51]. In this example, free cell phones are given away to a small set of people with high social influence in order to encourage others to buy the phones.

Another example is the problem of deciding the optimal positions of sensors for

environmental monitoring [49]. The goal of such a problem is to place sensors in the environment in order to minimize the uncertainty in the recorded observations. The number of used sensors is limited by the assigned budget to the project. This is again a submodular maximization problem as the efficiency of a subset of sensors is a submodular set function. Many other important problems fall in this category of optimization problems, e.g. the capital budgeting problem where it is required to decide the optimal assignment of limited investments among different projects, and the feature selection problem in pattern recognition problems [46].

II.4.2 Optimization of Submodular Set Functions

In this work we make use of a well acknowledged result by Nemhauser *et al.* [55, 31], which provides a lower bound on the performance of using a simple greedy approach to maximize a *monotonic submodular set function* subject to a cardinality constraint. We first provide the definition of monotonicity as follows:

Definition 2. *(Monotonicity). A set function $f : 2^{\mathcal{X}} \rightarrow \mathbb{R}$ is monotonic if and only if, for all $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{X}$, it holds that*

$$f(\mathcal{A}) \leq f(\mathcal{B}).$$

Theorem 2 provides the formal statement for the performance of the greedy algorithm,

Theorem 2. *Given a finite set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ and a monotonic submodular function $f(\mathcal{A})$, where $\mathcal{A} \subseteq \mathcal{X}$ and $f(\emptyset) = 0$. For the following maximization problem,*

$$\mathcal{A}^* = \underset{|\mathcal{A}| \leq k}{argmax}\, f(\mathcal{A}).$$

The greedy maximization algorithm returns $\mathcal{A}_{Greedy}$ such that $f(\mathcal{A}_{Greedy}) \geq (1 - \frac{1}{e}) f(\mathcal{A}^*)$, where e is Euler's number.

Proof. Let $\mathcal{A}_i$ denote the first i elements selected by the greedy algorithm and let $\mathcal{A}^*$ denote the actual optimum, $f(\mathcal{A}^*) = \text{OPT}$. Greedy will select exactly k elements, i.e. $\mathcal{A}_k$ is the set returned by the algorithm. We claim via induction that for $0 \leq i \leq k$,

$$f(\mathcal{A}^*) - f(\mathcal{A}_i) \leq (1 - 1/k)^i f(\mathcal{A}^*) \tag{II.38}$$

The base case of $i = 0$ is trivially true. Suppose that $i > 0$ and in the $i-$th step, Greedy selects element a_i, maximizing $f_{\mathcal{A}_{i-1}}(a_i)$ among the remaining elements. Observe that the remaining elements include $\mathcal{A}^* \backslash \mathcal{A}_{i-1}$, a set of size at most k. By submodularity, we have

$$f(\mathcal{A}^*) - f(\mathcal{A}_{i-1}) \leq \sum_{a \in \mathcal{A}^* \backslash \mathcal{A}_{i-1}} f_{\mathcal{A}_{i-1}}(a) \tag{II.39}$$

and this implies that the element a_i has marginal value

$$f_{\mathcal{A}_{i-1}}(a_i) \geq \frac{1}{|\mathcal{A}^* \backslash \mathcal{A}_{i-1}|} \sum_{a \in \mathcal{A}^* \backslash \mathcal{A}_{i-1}} f_{\mathcal{A}_{i-1}}(a) \geq \frac{1}{k}(f(\mathcal{A}^*) - f(\mathcal{A}_{i-1})). \tag{II.40}$$

Assuming that Eq.(II.38) holds true for $\mathcal{A}_{i-1}$, we have

$$\begin{aligned} f(\mathcal{A}^*) - f(\mathcal{A}_i) &= f(\mathcal{A}^*) - f(\mathcal{A}_{i-1}) - f_{\mathcal{A}_{i-1}}(a_i) \\ &\leq f(\mathcal{A}^*) - f(\mathcal{A}_{i-1}) - \frac{1}{k}(f(\mathcal{A}^*) - f(\mathcal{A}_{i-1})) \\ &= (1 - 1/k)(f(\mathcal{A}^*) - f(\mathcal{A}_{i-1})) \\ &\leq (1 - 1/k)^i f(\mathcal{A}^*) \end{aligned}$$

which proves Eq.(II.38). Using the claim for $i = k$, we get

$$f(\mathcal{A}^*) - f(\mathcal{A}_k) \leq (1 - 1/k)^k f(\mathcal{A}^*) \leq e^{-1} f(\mathcal{A}^*). \tag{II.41}$$

□

The simple greedy algorithms, see Algorithm 2, basically works by adding the element that maximally increases the objective value. According to Theorem 2, this simple procedure is guaranteed to achieve at least a constant fraction $(1 - 1/e)$ of the optimal solution, where e is the natural exponential. One point to emphasize is that the provided constant fraction $(1 - 1/e)$ is just a lower bound on the performance. However, in practice the greedy algorithm achieves significantly better than this lower bound.

What makes the greedy algorithm even more interesting is that it has been proved by Feige [30] that the approximation factor $(1 - 1/e)$ is the optimal approximation for this problem. Specifically, given any fixed $\epsilon > 0$, the problem of achieving a $(1 - 1/e + \epsilon)$-approximation for the Max k-cover problem is NP-hard. Keep in mind that the Max k-cover problem is a special case of $\max\{f(\mathcal{A}) : |\mathcal{A}| \leq k\}$ for f monotone submodular

Algorithm 2: Greedy Algorithm for Submodular Function Maximization with Cardinality Constraint [55, 64]

Input : Submodular Function $f(\mathcal{A})$, where $\mathcal{A} \subseteq \mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$.
Cardinality Parameter k.

Output: $\mathcal{A}^* \approx \underset{|\mathcal{A}| \leq k}{\arg\max}\, f(\mathcal{A})$

begin
  Set $\mathcal{A}_0 := \phi$
  for $i := 1$ **to** k **do**
    $\mathbf{x}^* := \underset{\mathbf{x} \in \mathcal{X} \setminus \mathcal{A}_{i-1}}{\arg\max}\, f(\mathcal{A}_{i-1} \cup \{\mathbf{x}\}) - f(\mathcal{A}_{i-1})$
    (Find greedy maximizer)
    $\mathcal{A}_i := \mathcal{A}_{i-1} \cup \{\mathbf{x}^*\}$ (Update current set $\mathcal{A}_i$)
  end
  $\mathcal{A}^* := \mathcal{A}_k$
end

set function. Thus, the greedy algorithm constitutes the best approximation obtainable for these problems. The same problem has been studied for more complicated domains as well. Specifically, for maximizing a submodular function over a matroid, which is a general class that includes the cardinality constraint as a special case. Recent results by Vondrak [14] shows that the $(1 - 1/e)$-approximation persists for the matroid constraints as well.

CHAPTER III

EFFICIENT SEMI-SUPERVISED SUPPORT VECTOR MACHINE THROUGH SUBMODULAR OPTIMIZATION (SUBMOD-S^{3}VM)

In this chapter we present a quadratic programming approximation of the Semi-Supervised Support Vector Machine (S^{3}VM) problem, namely approximate QP-S^{3}VM, that can be efficiently solved using off the shelf optimization packages. We show that this new formulation establishes a relationship between the low density separation and the graph-based models of semi-supervised learning (SSL), which is important to develop a unifying framework for semi-supervised learning methods. Furthermore, we propose the novel idea of representing SSL problems as submodular set functions and use efficient submodular optimization algorithms to solve them. Using this new idea we develop a representation of the approximate QP-S^{3}VM as a maximization of a submodular set function (SUBMOD-S^{3}VM) which makes it possible to optimize using efficient greedy algorithms. We demonstrate that the proposed methods are highly competitive and provide significant improvement in time complexity (up to $300X$) over the state of the art in the literature.

The proposed approximate QP-S^{3}VM is detailed in Section III.1. In Section III.2 we present the submodular formulation SUBMOD-S^{3}VM. Experimental results are provided in Section III.4, followed by the discussion in Section III.5.

III.1 Quadratic Programming Approximation of S^{3}VM (QP-S^{3}VM)

As we have mentioned in Chapter I, the objective function of the S^{3}VM problem has the form:

$$\underset{\mathbf{w},y_j}{\arg\min}\ \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i\in\mathcal{L}}\ell_l(\mathbf{w},(\mathbf{x}_i,y_i)) + C^*\sum_{j\in\mathcal{U}}\ell_u(\mathbf{w},\mathbf{x}_j) \tag{III.1}$$

where $\mathcal{L}$ and $\mathcal{U}$ are the labeled and unlabeled sample sets, respectively. The loss functions for unlabeled samples ℓ_u and labeled samples ℓ_l are defined as:

$$\ell_u(\mathbf{w}, \mathbf{x}_j) = \min_{y_j \in \{-1,+1\}} \max\{0, 1 - y_j(\langle \mathbf{w}, \mathbf{x}_j \rangle + b)\} \tag{III.2}$$

$$\ell_l(\mathbf{w}, (\mathbf{x}_i, y_i)) = \max\{0, 1 - y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b)\} \tag{III.3}$$

The combinatorial path to solve this problem works as follows: a) The solution space for the problem is the space of all vectors $\{\pm 1\}^{|\mathcal{U}|}$, b) Each evaluation of a solution in the vector space, is performed through a standard supervised SVM optimization process.

III.1.1 Continuous Formulation of S^{3}VM Problem

To simplify this hard mixed-integer programming, the loss of setting $y_j = 1$, denoted by ℓ_j^+, is assigned a new variable p_j, where $0 \leq p_j \leq 1$. This variable represents the probability that the assignment $y_j = 1$ is correct. Similarly, the loss of setting $y_j = -1$, denoted by ℓ_j^-, is represented by the probability $1 - p_j$. The balancing constraint, which prevents obtaining trivial solutions as discussed earlier, will have the form $\sum_{j \in \mathcal{U}} p_j = r|\mathcal{U}|$, where r is the ratio of all $y_j = 1$ assignments in the final solution. This way the problem has been modified from mixed-integer to a completely continuous problem. The modified formulation is formalized in Problem 1 [63, 76]:

Problem 1. *Continuous Optimization Formulation of the Combinatorial S^{3}VM Problem.*

$$\underset{\mathbf{p}'=[p_1,\ldots,p_{|\mathcal{U}|}]}{\arg\min} \quad \min_{\mathbf{w}} \mathcal{J}(\mathbf{w}, \mathbf{p}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i \in \mathcal{L}} \zeta_i + C^* \sum_{j \in \mathcal{U}} p_j \ell_j^+ + C^* \sum_{j \in \mathcal{U}} (1 - p_j)\ell_j^-$$

subject to

$$\begin{aligned} y_i[\langle \mathbf{w}, \mathbf{x}_i \rangle + b] &\geq 1 - \zeta_i \\ \langle \mathbf{w}, \mathbf{x}_j \rangle + b &\geq 1 - \ell_j^+ \\ -\langle \mathbf{w}, \mathbf{x}_j \rangle - b &\geq 1 - \ell_j^- \\ \zeta_i \geq 0, \ell_j^+ \geq 0, \ell_j^- &\geq 0 \\ 0 \leq p_j \leq 1, \sum_{j \in \mathcal{U}} p_j &= r|\mathcal{U}| \end{aligned}$$

To get more insight about Problem 1, Figure III.1 provides a visualization of a simple version of the objective function $\mathcal{J}(\mathbf{w}, \mathbf{p})$ when $\mathbf{w}$ and $\mathbf{p} = [p_1, \ldots, p_{|\mathcal{U}|}]$ are exclusively fixed. Fixing $\mathbf{p}$ means that all unlabeled samples are assigned labels, where all samples with $p_j \approx 1$ have the labels $y_j = 1$ and all samples with $p_j \approx 0$ are assigned $y_j = -1$. Therefore, our objective function is now a function only in $\mathbf{w}$ and has the following form,

$$\begin{aligned} &\arg\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i \in \{\mathcal{L} \cup \mathcal{U}\}} \zeta_i \\ &\text{subject to} \qquad y_i[\langle \mathbf{w}, \mathbf{x}_i \rangle + b] \geq 1 - \zeta_i \quad , \quad \zeta_i \geq 0 \end{aligned} \tag{III.4}$$

This is a standard supervised SVM problem which is a convex quadratic optimization problem where the global minimum can be efficiently found. This idea is depicted in Fig. III.1 by the convex functions on the planes perpendicular to the $\mathbf{p}$ axis.

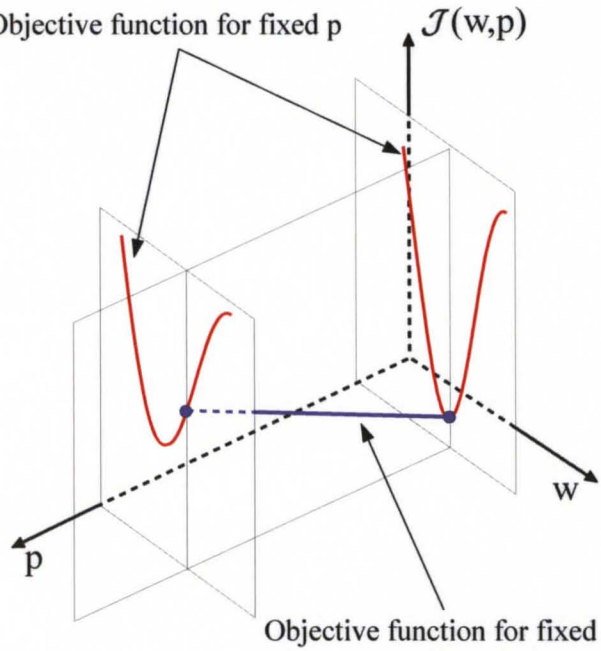


Figure III.1: Illustration of the nature of the continuous S^{3}VM objective function $\mathcal{J}(\mathbf{w}, \mathbf{p})$. The illustration visualizes a cross-section in $\mathcal{J}(\mathbf{w}, \mathbf{p})$ with respect to $\mathbf{w}$ and $\mathbf{p}$. The cross-section with respect to $\mathbf{p}$ shows the convexity of $\mathcal{J}(\mathbf{w}, \mathbf{p})$ for fixed $\mathbf{p}$ as it is reduced to an SVM problem. The cross-section with respect to $\mathbf{w}$ is linear in $\mathbf{p}$.

On the other hand, when the hyperplane $\mathbf{w}$ is fixed, Problem 1 will be reduced to the

following problem:

$$\begin{aligned} \underset{\mathbf{p}'=[p_1,\ldots,p_{|\mathcal{U}|}]}{\arg\min} \quad & C^* \sum_{j\in\mathcal{U}} p_j \ell_j^+ \; + \; C^* \sum_{j\in\mathcal{U}} (1-p_j)\ell_j^- \\ \text{subject to} \quad & \sum_{j\in\mathcal{U}} p_j = r|\mathcal{U}| \quad , \quad 0 \leq p_j \leq 1 \end{aligned} \tag{III.5}$$

This is a linear programming problem in $\mathbf{p}$, which makes it simple and efficient to optimize. This is illustrated in Fig.III.1 by the linear function on the plane perpendicular to the $\mathbf{w}$ axis.

The provided insights about the objective function $\mathcal{J}(\mathbf{w}, \mathbf{p})$ in terms of being convex when $\mathbf{p}$ is fixed and linear when $\mathbf{w}$ is fixed, might give the impression that $\mathcal{J}(\mathbf{w}, \mathbf{p})$ is simple to optimize. This actually would have been true if it was not for the dependence between the variables $\mathbf{w}$ and $\mathbf{p}$. The optimal $\mathbf{w}$ is the hyperplane that maximizes the margin using all labeled and unlabeled data samples, and therefore it implicitly assigns values for members of $\mathbf{p}$, and vice versa for the optimal $\mathbf{p}$ which should result in finding $\mathbf{w}$ that maximizes the margin according to the label assignments in $\mathbf{p}$. Therefore, $\mathcal{J}(\mathbf{w}, \mathbf{p})$ is a non-convex function that is hard to optimize.

III.1.2 Proposed Technique for Continuous S^{3}VM Optimization

Our strategy to solve the non-convex optimization in Problem 1 is based on proposing a surrogate objective function that is simpler to optimize and meanwhile preserves the optimal solution. To that end, we propose to use an approximation of $min_{\mathbf{w}} \mathcal{J}(\mathbf{w}, \mathbf{p})$ that is a function in $\mathbf{p}$ only. This way Problem 1 will be a function in $\mathbf{p}$ and as we will see later, we end up with a quadratic optimization problem.

To find an approximation for $min_{\mathbf{w}} \mathcal{J}(\mathbf{w}, \mathbf{p})$ we proceed to find its dual form. Deriving the *Lagrangian* and applying the *Karush-Kuhn-Tucker* conditions to it, the obtained dual form is presented in Problem 2.

Problem 2. *Dual form of* $\min_{\mathbf{w}} \mathcal{J}(\mathbf{w}, \mathbf{p})$ *in Problem 1.*

$$\underset{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma}{max} \; \mathcal{I}_{\mathbf{Dual}} \tag{III.6}$$

where

$$\begin{aligned}\mathcal{I}_{\mathbf{Dual}} &= \boldsymbol{\alpha}'\mathbf{1}_{|\mathcal{L}|} + (\boldsymbol{\gamma}+\boldsymbol{\beta})'\mathbf{1}_{|\mathcal{U}|} - \frac{1}{2}(\boldsymbol{\alpha}\circ\mathbf{y})'\mathbf{K}_{\mathbf{ll}}(\boldsymbol{\alpha}\circ\mathbf{y}) \\ &\quad - \frac{1}{2}(\boldsymbol{\gamma}-\boldsymbol{\beta})'\mathbf{K}_{\mathbf{uu}}(\boldsymbol{\gamma}-\boldsymbol{\beta}) - (\boldsymbol{\alpha}\circ\mathbf{y})'\mathbf{K}_{\mathbf{lu}}(\boldsymbol{\gamma}-\boldsymbol{\beta})\end{aligned} \tag{III.7}$$

subject to

$$\begin{aligned}&\mathbf{0} \leq \boldsymbol{\alpha} \leq C\mathbf{1}_{|\mathcal{L}|} \\ &\mathbf{0} \leq \boldsymbol{\gamma} \leq C^*\mathbf{p} \\ &\mathbf{0} \leq \boldsymbol{\beta} \leq C^*(\mathbf{1}_{|\mathcal{U}|} - \mathbf{p})\end{aligned}$$

where

$\mathbf{1}_{|\mathcal{L}|}$: *A ones vector of length* $|\mathcal{L}|$. *Similarly is* $\mathbf{1}_{|\mathcal{U}|}$.

α_i:*Lagrangian Multiplier of labeled loss constraint* ζ_i.

γ_j:*Lagrangian Multiplier of unlabeled loss constraint* ℓ_j^+.

β_j:*Lagrangian Multiplier of unlabeled loss constraint* ℓ_j^-.

$\boldsymbol{\alpha}' = [\alpha_1, \ldots, \alpha_{|\mathcal{L}|}], \boldsymbol{\beta}' = [\beta_1, \ldots, \beta_{|\mathcal{U}|}], \boldsymbol{\gamma}' = [\gamma_1, \ldots, \gamma_{|\mathcal{U}|}]$

$\mathbf{p}' = [p_1, \ldots, p_{|\mathcal{U}|}]$, $\mathbf{y}' = [y_1, \ldots, y_{|\mathcal{L}|}]$,

$\mathbf{K_{ll}} = \mathbf{K}_{i,i'}\ \forall i, i' \in \mathcal{L}$, $\mathbf{K_{uu}} = \mathbf{K}_{j,j'}\ \forall j, j' \in \mathcal{U}$,

$\mathbf{K_{lu}} = \mathbf{K}_{i,j}\ \forall i \in \mathcal{L}, j \in \mathcal{U}$.

One issue to clarify at this point is that $min_{\mathbf{w}}\mathcal{J}(\mathbf{w}, \mathbf{p}) = max_{\alpha,\beta,\gamma}\mathcal{I}_{Dual}$ for any fixed $\mathbf{p}$. This is true because for any fixed $\mathbf{p}$, $min_{\mathbf{w}}\mathcal{J}(\mathbf{w}, \mathbf{p})$ is basically a standard SVM problem which is known to be convex and therefore there is no duality gap. Using the derived dual form in Problem 2, we approximate $min_{\mathbf{w}}\mathcal{J}(\mathbf{w}, \mathbf{p})$ by deriving an upper bound function for $max_{\alpha,\beta,\gamma}\mathcal{I}_{Dual}$. Theorem 3 provides the details of deriving the upper bound function.

Theorem 3. *Proposed upper bound for* $max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}}\mathcal{I}_{\mathbf{Dual}}$:

$$\max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}} \mathcal{I}_{\mathbf{Dual}} \leq \mathcal{I}(\mathbf{w}^*) + C^*|\mathcal{U}| + \mathcal{M}_1 + \mathcal{M}_2 \tag{III.8}$$

where

$$\mathcal{I}(\mathbf{w}^*) = \min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i\in\mathcal{L}}\zeta_i$$

$$\mathcal{M}_1 = \frac{1}{2}C^{*2}(\mathbf{1}_{|\mathcal{U}|} - \mathbf{p})'\mathbf{K_{uu}}\mathbf{p} \ , \quad \mathcal{M}_2 = CC^*\mathbf{y}'\mathbf{K_{lu}}(\mathbf{1}_{|\mathcal{U}|} - \mathbf{p}) \tag{III.9}$$

Proof. To get an upper bound for $\mathcal{I}_{\mathbf{Dual}}$ we divide it into several components as follows:

$$\mathcal{I}_{\mathbf{Dual}} = \mathcal{N}_1 + \mathcal{N}_2 + \mathcal{N}_3 \tag{III.10}$$

where

$$\begin{aligned} \mathcal{N}_1 &= \boldsymbol{\alpha}'\mathbf{1}_{|\mathcal{L}|} - \frac{1}{2}(\boldsymbol{\alpha} \circ \mathbf{y})'\mathbf{K_{ll}}(\boldsymbol{\alpha} \circ \mathbf{y}) \\ \mathcal{N}_2 &= (\boldsymbol{\gamma} + \boldsymbol{\beta})'\mathbf{1}_{|\mathcal{U}|} - \frac{1}{2}(\boldsymbol{\gamma} - \boldsymbol{\beta})'\mathbf{K_{uu}}(\boldsymbol{\gamma} - \boldsymbol{\beta}) \\ \mathcal{N}_3 &= -(\boldsymbol{\alpha} \circ \mathbf{y})'\mathbf{K_{lu}}(\boldsymbol{\gamma} - \boldsymbol{\beta}). \end{aligned} \tag{III.11}$$

Then

$$\max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}} \mathcal{I}_{\mathbf{Dual}} \leq \max_{\boldsymbol{\alpha}} \mathcal{N}_1 + \max_{\boldsymbol{\beta},\boldsymbol{\gamma}} \mathcal{N}_2 + \max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}} \mathcal{N}_3 \tag{III.12}$$

$\max_{\boldsymbol{\alpha}} \mathcal{N}_1$ is the dual form of a standard supervised SVM problem using the label data, i.e.

$$\max_{\boldsymbol{\alpha}} \mathcal{N}_1 = \min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i\in\mathcal{L}} \zeta_i \tag{III.13}$$

Furthermore, using the value limits of $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$, i.e. $\mathbf{0} \leq \boldsymbol{\alpha} \leq C\mathbf{1}_{|\mathcal{L}|}$, $\mathbf{0} \leq \boldsymbol{\beta} \leq C^*(\mathbf{1}_{|\mathcal{U}|} - \mathbf{p})$ and $\mathbf{0} \leq \boldsymbol{\gamma} \leq C^*\mathbf{p}$, we can derive the following upper bounds of $\mathcal{N}_2$ and $\mathcal{N}_3$,

$$\max_{\boldsymbol{\beta},\boldsymbol{\gamma}} \mathcal{N}_2 \leq C^*|\mathcal{U}| + \frac{1}{2}C^{*2}(\mathbf{1}_{|\mathcal{U}|} - \mathbf{p})'\mathbf{K_{uu}}\mathbf{p} \tag{III.14}$$

and

$$\max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}} \mathcal{N}_3 \leq CC^*\mathbf{y}'\mathbf{K_{lu}}(\mathbf{1}_{|\mathcal{U}|} - \mathbf{p}). \tag{III.15}$$

Combining the three upper bounds we get the provided bound in the theorem.

□

Examining the upper bound in Theorem 3 we observe that $\mathcal{I}(\mathbf{w}^*)$ is the objective function value of optimizing a standard supervised SVM on the labeled samples $\mathcal{L}$. Therefore, it is a constant value as well as the term $C^*|\mathcal{U}|$. The rest of the upper bound, namely $\mathcal{M}_1 + \mathcal{M}_2$, is a function of $\mathbf{p}$. Therefore, the proposed upper bound is a function in $\mathbf{p}$ only. The optimal $\mathbf{p}$ is now obtainable through solving Problem 3, which is a standard quadratic programming problem that can be solved by off the shelf optimization techniques.

Problem 3. *Quadratic Programming Approximation of Semi-supervised Support Vector Machines (QP-S^{3}VM):*

$$\underset{\mathbf{p}'=[p_1,\ldots,p_{|\mathcal{U}|}]}{\arg\min} \frac{1}{2}C^{*2}(\mathbf{1}_{|\mathcal{U}|}-\mathbf{p})'\mathbf{K_{uu}}\mathbf{p}+CC^{*}\mathbf{y}'\mathbf{K}_{lu}(\mathbf{1}_{|\mathcal{U}|}-\mathbf{p}) \tag{III.16}$$

subject to

$$\mathbf{p}'\mathbf{1}_{|\mathcal{U}|}=r|\mathcal{U}|, \quad \mathbf{0}\leq\mathbf{p}\leq\mathbf{1}_{|\mathcal{U}|}. \tag{III.17}$$

Note: Equation (III.16) *can be rewritten in the standard quadratic programming form as follows:*

$$\underset{\mathbf{p}'=[p_1,\ldots,p_{|\mathcal{U}|}]}{\arg\min} -\frac{1}{2}C^{*2}\mathbf{p}'\mathbf{K_{uu}}\mathbf{p}+(\frac{1}{2}C^{*2}\mathbf{1}'_{|\mathcal{U}|}\mathbf{K_{uu}}-CC^{*}\mathbf{y}'\mathbf{K}_{lu})\mathbf{p} \tag{III.18}$$

In order to avoid trivial solutions to the problem where all the variables p_j are $zero$, we add the constraint $\mathbf{p}'\mathbf{1}=r|\mathcal{U}|$ which makes sure that a certain ratio of the unlabeled samples, r, be assigned to class $+1$.

An intuitive illustration of the proposed surrogate objective function is depicted in Fig.III.2, where we can see the upper bound function, which is a function in $\mathbf{p}$ only, tracking the optimal value of $max_{\alpha,\beta,\gamma}\mathcal{I}_{Dual}$, which is a function in both $\mathbf{w}$ and $\mathbf{p}$, for any $\mathbf{p}$. Therefore, the optimal $\mathbf{p}$ solution of the upper bound function will occur at the same vector $\mathbf{p}$ obtained from solving Problem 1. The concavity of the upper bound function in Fig.III.2 is actually inferred from the standard quadratic programming form in Eq.(III.18), where the Hessian matrix for the objective function in Eq.(III.18) is $-0.5\ C^{*2}\mathbf{K_{uu}}$. Remember that $\mathbf{K_{uu}}$ is essentially the kernel matrix of all unlabeled samples in $\mathcal{U}$, therefore $\mathbf{K_{uu}}$ is a positive semidefinite matrix. Hence, the Hessian matrix is negative semidefinite, which makes the objective function in Problem 3 a concave function.

III.1.3 QP-S^{3}VM Model Verification

Now that we have derived a quadratic programming surrogate objective function for S^{3}VM in Problem 3, one essential step is to validate the correctness of this objective function. The definition of being correct as a surrogate objective function implies that both

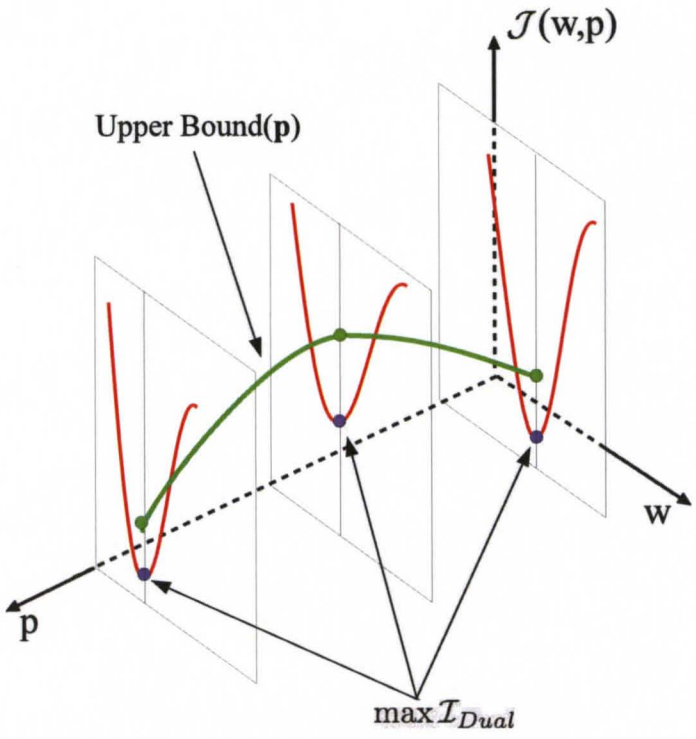


Figure III.2: Illustration of the proposed upper bound function and how it tracks the $max_{\alpha,\beta,\gamma}\mathcal{I}_{Dual}$ at all possible $\mathbf{p}$.

the original and the surrogate function always produce proportional values and thus the surrogate function can be used to simplify the process of optimizing the original objective function. In our case, to show that the proposed surrogate in Problem 3 is correct with respect to the original Problem 1, we need to show that,

$$\min_{\mathbf{w}} \mathcal{J}(\mathbf{w},\mathbf{p}) \quad \propto \quad \text{UpperBound}(\mathbf{p}) \qquad \forall \mathbf{p} \tag{III.19}$$

where

$$\begin{aligned}
\mathcal{J}(\mathbf{w},\mathbf{p}) &= \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i\in\mathcal{L}}\zeta_i + C^*\sum_{j\in\mathcal{U}}p_j\ell_j^+ + C^*\sum_{j\in\mathcal{U}}(1-p_j)\ell_j^-, \\
\text{UpperBound}(\mathbf{p}) &= \frac{1}{2}C^{*2}(\mathbf{1}_{|\mathcal{U}|}-\mathbf{p})'\mathbf{K}_{\mathbf{uu}}\mathbf{p} + CC^*\mathbf{y}'\mathbf{K}_{\mathbf{lu}}(\mathbf{1}_{|\mathcal{U}|}-\mathbf{p}).
\end{aligned}$$

To that end, we use an experimental setup that evaluates both sides of Eq.(III.19) with respect to a wide spectrum of $\mathbf{p}$ assignments and then examine the output using scatter plots.

III.1.3.1 Sampling the p-space

The members of the vector $\mathbf{p}' = [p_1, \ldots, p_j, \ldots, p_{|\mathcal{U}|}]$ represent the probabilities of unlabeled samples to belong to the positive class. Despite the the continuous nature of the p-space, the members of any vector $\mathbf{p}$ will be thresholded to decide the membership to the classes. Therefore, when sampling from p-space we can discretize the continuous space into finitely countable combinatorial space which significantly reduces the difficulty of the sampling process. To sample from such large combinatorial space we start by setting the center of the space to a certain vector, denoted by $\mathbf{p}_{Center}$. Samples from the combinatorial space are then generated by flipping the values of the members of $\mathbf{p}_{Center}$. Since we validate the relationship in Eq.(III.19) using data sets, we set $\mathbf{p}_{Center}$ to the original labels of an examined data set, that is

$$\mathbf{p}'_{Center} = [p_{Center_1}, \ldots, p_{Center_j}, \ldots, p_{Center_{|\mathcal{U}|}}], \quad p_{Center_j} = \begin{cases} 1 & \forall \quad y_j = \quad 1 \\ 0 & \forall \quad y_j = \quad -1 \end{cases}$$

III.1.3.2 Covering the p-space

To cover the whole range of p-space, we need to use a measure to indicate how far a sample is from the center of the space, $\mathbf{p}_{Center}$. We use the Hamming distance which is basically the number of mismatches between the corresponding components of two combinatorial vectors. The Hamming distance (HD) between any vector in the p-space and $\mathbf{p}_{Center}$ has the range, HD $\in \{0, |\mathbf{p}|\}$. Therefore, to provide a good coverage of the p-space, we sample vectors with all possible Hamming distances from $\mathbf{p}_{Center}$.

III.1.3.3 QP-S^{3}VM Model Verification Experimental Setup

We validate the correctness of Eq.(III.19) by experimentally evaluating both of its sides with respect to sampled $\mathbf{p}$ vectors. For each data set, the samples were split into 99.5% unlabeled samples and the rest are labeled. This means that the length of the vectors we use for evaluation is $|\mathbf{p}| = 0.995 * \#$ Samples. As mentioned earlier, $\mathbf{p}_{Center}$ is set to the original labels and sampling takes place by randomly flipping individual components in $\mathbf{p}_{Center}$. The number of samples to be flipped depends on the Hamming distance we are examining.

In our experiments, we use all possible Hamming distances, HD $\in \{0, |\mathbf{p}|\}$. Once a vector $\mathbf{p}$ is sampled, the value for the Upper Bound($\mathbf{p}$) is easily obtained by direct substitution. However, the value for $min_{\mathbf{w}}\mathcal{J}(\mathbf{w}, \mathbf{p})$ is obtained through an SVM training procedure. The output of the experiment is illustrated through scatter plots of $min_{\mathbf{w}}\mathcal{J}(\mathbf{w}, \mathbf{p})$ versus the Upper Bound($\mathbf{p}$) for all sampled $\mathbf{p}$ vectors.

III.1.3.4 QP-S^{3}VM Model Verification Experiments Discussion

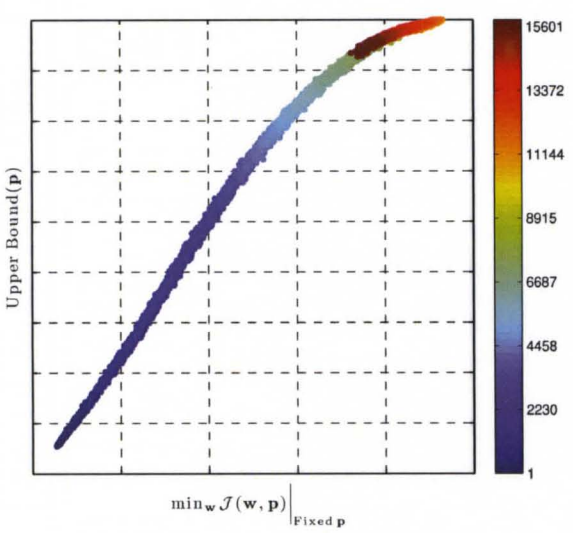
Figure III.3 shows the outcome of the verification experiments for a wide range of data sets. The shade of each point on the scatter plots indicate the Hamming distance of the $\mathbf{p}$ vector from $\mathbf{p}_{Center}$. The key for shade-Hamming distance correspondence is visualized in the color bar associated with each scatter plot. All the scatter plots shown in Fig.III.3 illustrate that the relationship between both sides of Eq.(III.19) is *direct proportionality* over the whole range of $\mathbf{p}$, which validates using the Upper Bound($\mathbf{p}$) as a surrogate for $min_{\mathbf{w}}\mathcal{J}(\mathbf{w}, \mathbf{p})$.

Upon closer examination of the obtained scatter plots, we notice that the proportional relationship between Upper Bound($\mathbf{p}$) and $min_{\mathbf{w}}\mathcal{J}(\mathbf{w}, \mathbf{p})$ is almost strictly linear for very high dimensional data sets, see Fig.(III.3h, III.3i, III.3j, and III.3k). This means that for high dimensional data sets, the Upper Bound($\mathbf{p}$) works as a perfect surrogate objective function. On the other hand, it is clear that the degree of proportionality varies with the Hamming distance for data sets with few dimensions. For such data sets, while the proportional relationship is linear over almost all the $\mathbf{p}$-space, it tends to saturate for $\mathbf{p}$ vectors with very large Hamming distances as can be seen in Fig.(III.3a-III.3g, III.3i, and III.3l). However, that does not take away from the quality of the Upper Bound($\mathbf{p}$) as a surrogate function. It just indicates that the surrogate function becomes less sensitive to the variations in $min_{\mathbf{w}}\mathcal{J}(\mathbf{w}, \mathbf{p})$ at the edges of $\mathbf{p}$-space centered at $\mathbf{p}_{Center}$. In other words, the Upper Bound(($\mathbf{p}$)) is very tight for high dimensional data sets and most of the $\mathbf{p}$-space for low dimensional data sets. The tightness weakens a bit for low dimensional data set around the edges of the $\mathbf{p}$-space. These observations about the relationship between the dimensionality of the data sets and the Upper Bound($\mathbf{p}$) will render very important in the results section where the performance of the approximate formulation in Problem 3 is examined.

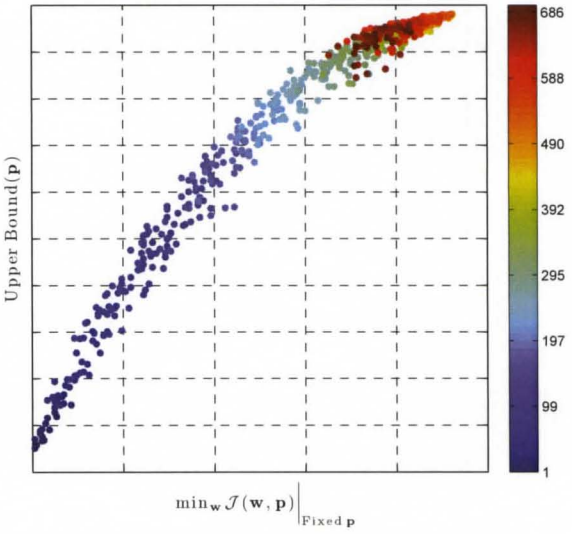
It is also noticeable in the scatter plots that some data sets show more dispersion with respect to the linear relationship between $min_{\mathbf{w}}\mathcal{J}(\mathbf{w},\mathbf{p})$ and Upper Bound($(\mathbf{p})$) than others, see Fig.(III.3b,III.3d,III.3e,III.3g, and III.3i). As the performance of different data sets varies with the type of kernel used, the observed variation in dispersion is basically due to the use of the linear kernel for all the data sets in the verification experiment. This is evident from the small dispersion observed for data sets that are known to work well with the linear kernel, see Fig.(III.3h, III.3j, and III.3k). Other data sets that achieve better performance with non-linear kernels show larger dispersion. For instance, the RBF kernel achieves better performance for the Diabetes data set which explains the dispersion in Fig.(III.3e).

While the scatter plots in Fig.III.3 show the good performance of the Upper Bound as a surrogate function for the $min_{\mathbf{w}}\mathcal{J}(\mathbf{w},\mathbf{p})$ they might wrongfully give the impression that it is an easy problem. Actually through the experiments we noticed that some of Hamming distances are not visible on the scatter plots. For instance, in Fig.III.3a the $\mathbf{p}$ vectors with Hamming distances HD $\in \{7000, 11000\}$ are not visible on the scatter plot. It turned out that $min_{\mathbf{w}}\mathcal{J}(\mathbf{w},\mathbf{p})$ is not linear in the Hamming distance but rather concave, see Fig.III.4a. That is after a certain Hamming distance, see point $\mathbf{M}$ in Fig.III.4a, $min_{\mathbf{w}}\mathcal{J}(\mathbf{w},\mathbf{p})$ decreases, backtracking its way on the points with smaller Hamming distance. Figure III.4 shows the behavior of $min_{\mathbf{w}}\mathcal{J}(\mathbf{w},\mathbf{p})$ with increasing Hamming distance. Figure III.4c illustrates the trajectory of $\mathbf{p}$ vectors with increasing Hamming distance from $\mathbf{p}_{Center}$ on the scatter plots, where though the point $\mathbf{E}$ has the farthest distance from $\mathbf{p}_{Center}$, as shown in Fig.III.4a, it exits inside the scatter plot in Fig.III.4c not at its edge.

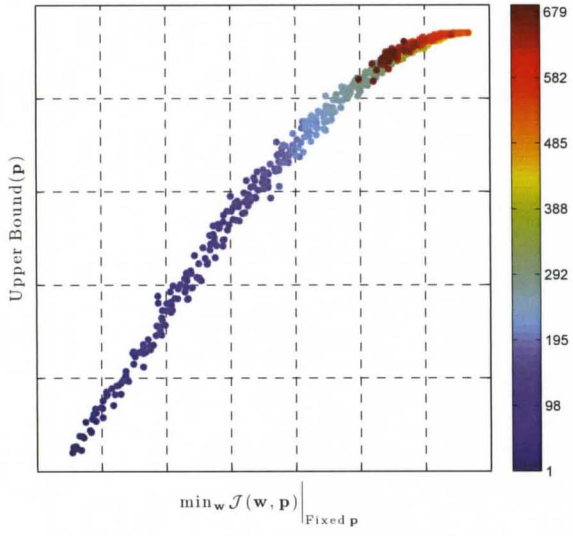
The concave behavior of the $min_{\mathbf{w}}\mathcal{J}(\mathbf{w},\mathbf{p})$ with changing $\mathbf{p}$ can be explained using the nature of the SVM training process. In our verification experiment we start with the original labels of the unlabeled samples as the origin of the $\mathbf{p}$-space and then we sample vectors that have increasing Hamming distances from $\mathbf{p}_{Center}$. Assuming that $\mathbf{p}_{Center}$ are good labels, i.e. the data set is well separated using SVM, then any $\mathbf{p}$ vector farther away from $\mathbf{p}_{Center}$ will force SVM to just add the differences between $\mathbf{p}$ and $\mathbf{p}_{Center}$ as tolerable classification mistakes, while sticking mostly to the $\mathbf{p}_{Center}$ labels. Such mistakes are handled by slack variables that increase the value of the SVM objective function. This is



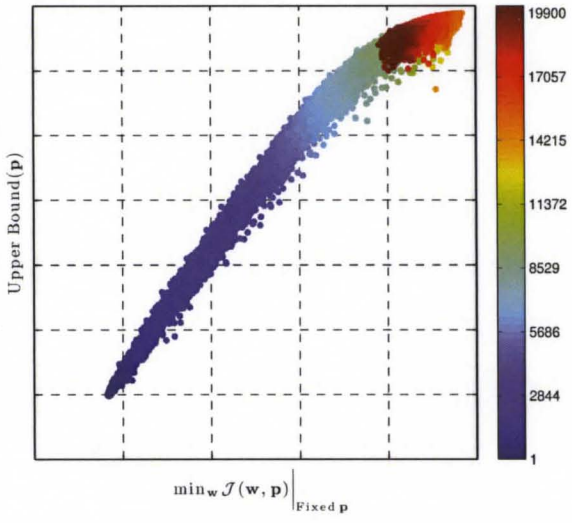
(a) A9a: # Samples = 15,680, # Features = 123.



(b) Australian: # Samples = 690, # Features = 14.

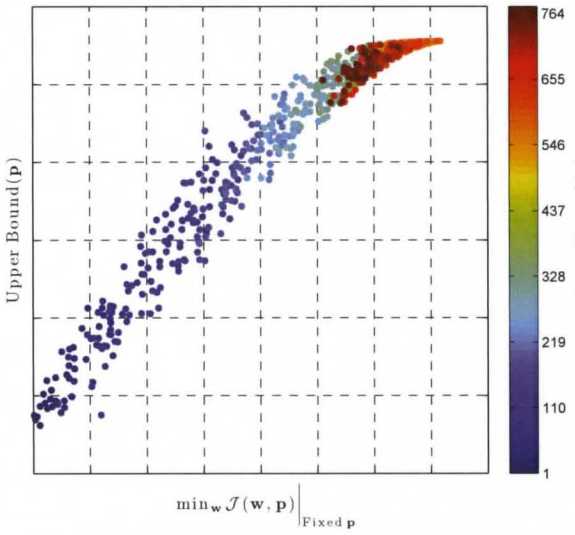


(c) Breast Cancer: # Samples = 683, # Features = 10.

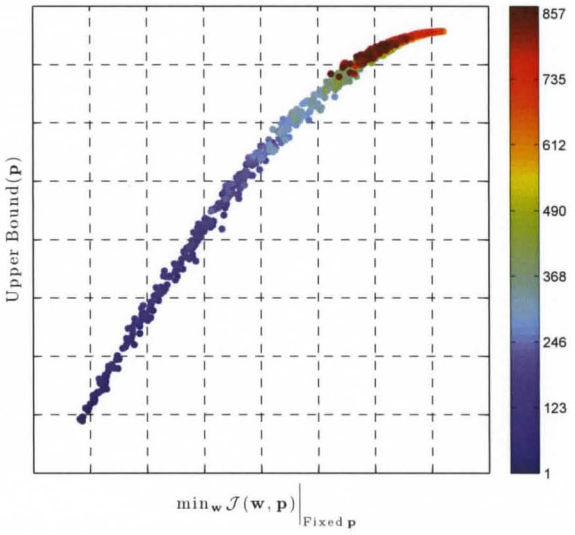


(d) Covtype.Binary: # Samples = 20,000, # Features = 54.

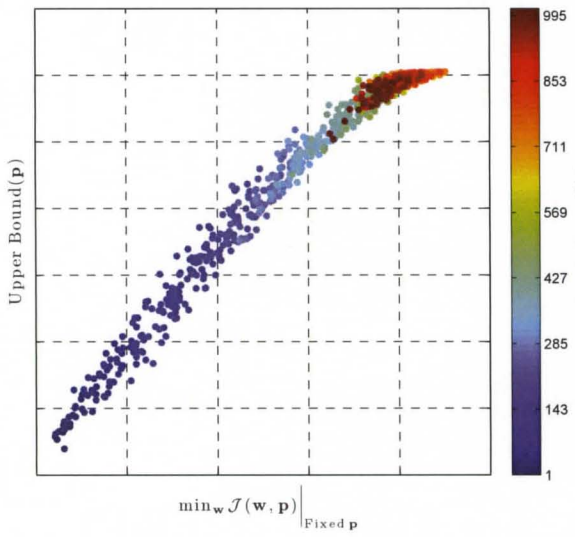
Figure III.3: Scatter plots of $min_{\mathbf{w}}\mathcal{J}(\mathbf{w}, \mathbf{p})$ versus the Upper Bound($\mathbf{p}$) for several data sets. The shade of each point represent the Hamming distance of the examined combinatorial vector $\mathbf{p}$ from $\mathbf{p}_{Center}$. The key for the Hamming distances shades is depicted in the color bar attached to each plot.



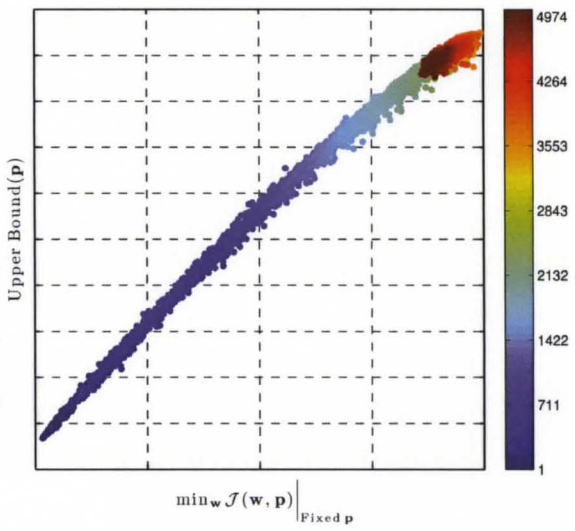
(e) Diabetes: # Samples = 768, # Features = 8.



(f) Four Class: # Samples = 862, # Features = 2.

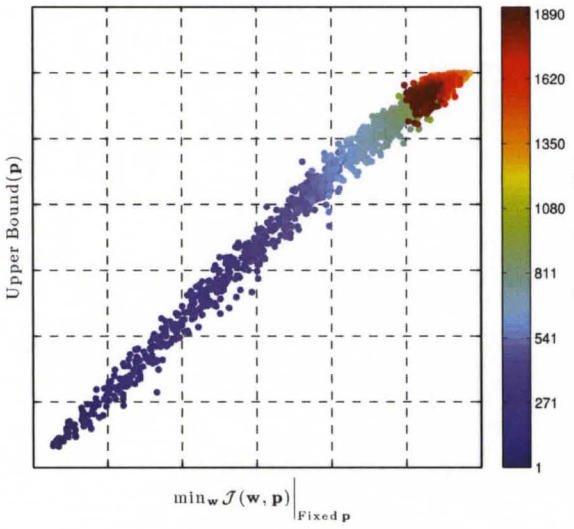


(g) German-Numer: # Samples = 1,000, # Features = 24.

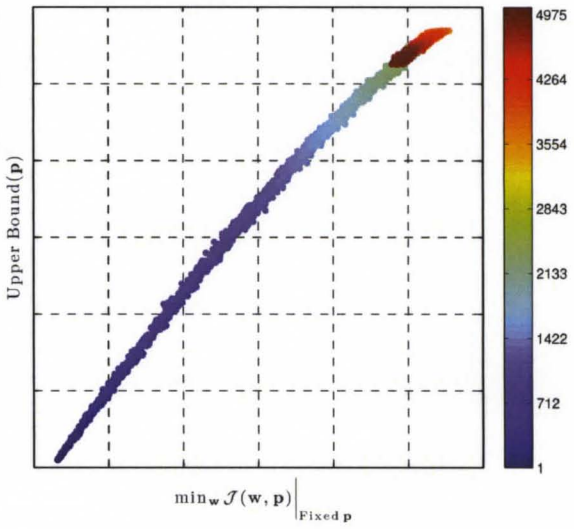


(h) News20.Binary: # Samples = 5,000, # Features = 1,355,191.

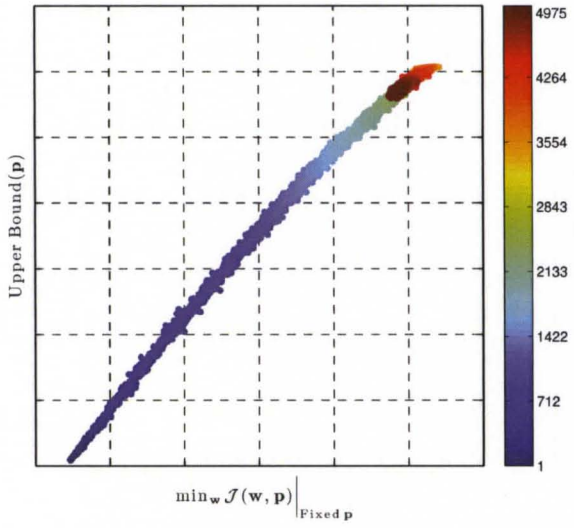
Figure III.3: Continued: Scatter plots of $min_{\mathbf{w}} \mathcal{J}(\mathbf{w}, \mathbf{p})$ versus the Upper Bound($\mathbf{p}$) for several data sets. The shade of each point represent the Hamming distance of the examined combinatorial vector $\mathbf{p}$ from $\mathbf{p}_{Center}$. The key for the Hamming distances shades is depicted in the color bar attached to each plot.



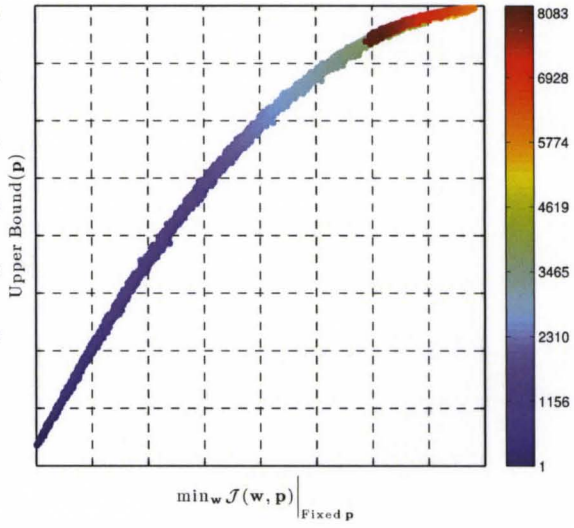
(i) W6a: # Samples = 1,900, # Features = 300.



(j) RCV1.Binary: # Samples = 5,000, # Features = 47,236.



(k) Real-Sim: # Samples = 5,000, # Features = 20,958.



(l) Mushrooms: # Samples = 8,124, # Features = 112.

Figure III.3: Continued: Scatter plots of $min_{\mathbf{w}}\mathcal{J}(\mathbf{w}, \mathbf{p})$ versus the Upper Bound($\mathbf{p}$) for several data sets. The shade of each point represent the Hamming distance of the examined combinatorial vector $\mathbf{p}$ from $\mathbf{p}_{Center}$. The key for the Hamming distances shades is depicted in the color bar attached to each plot.

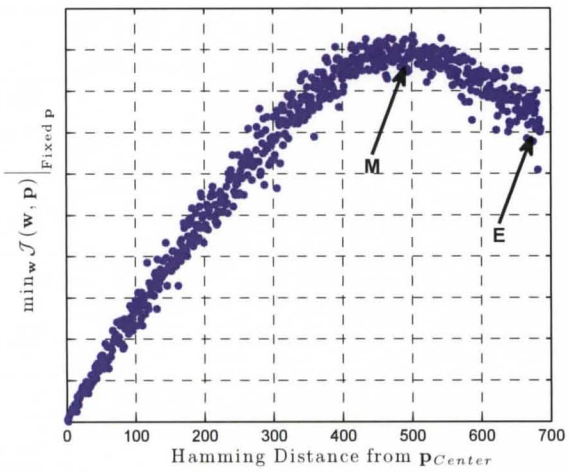
why the the value $min_{\mathbf{w}}\mathcal{J}(\mathbf{w},\mathbf{p})$ increases with increasing the Hamming distance from $\mathbf{p}_{Center}$. As the Hamming distance between $\mathbf{p}$ and $\mathbf{p}_{Center}$ gets very large, see point **M** in Fig.III.4a, the majority of the label assignments in the vector $\mathbf{p}$ are opposite to those in $\mathbf{p}_{Center}$. Therefore, it is cheaper for an SVM trained using $\mathbf{p}$ to accommodate the labels of $\mathbf{p}$ and will handle the differences from $\mathbf{p}_{Center}$ as tolerable mistakes which result in smaller value for the SVM objective function. This explains why the value of $min_{\mathbf{w}}\mathcal{J}(\mathbf{w},\mathbf{p})$ tends to decrease after the point **M** in Fig.III.4a. Figure III.4b highlights the flexibility of the Upper Bound to track $min_{\mathbf{w}}\mathcal{J}(\mathbf{w},\mathbf{p})$ in various regions of the $\mathbf{p}$-space.

It is important to clarify at this point that through out the verification experiments we wanted to show that the UpperBound($\mathbf{p}$) is a proper surrogate objective function for $min_{\mathbf{w}}\mathcal{J}(\mathbf{w},\mathbf{p})$, this is why we experimented with all possible areas of the $\mathbf{p}$-space. However, it is important to notice that the solution space for Problem 3 will be smaller than the general $\mathbf{p}$-space we considered so far. This is because the solution of Problem 3 has to adhere to the labels of the labeled samples. In other words, the area of the $\mathbf{p}$-space where the output labels are mostly opposite to $\mathbf{p}_{Center}$, e.g. $\mathbf{p}$-space from **M** to **E** in Fig.III.4a, is outside of the solution space for Problem 3.

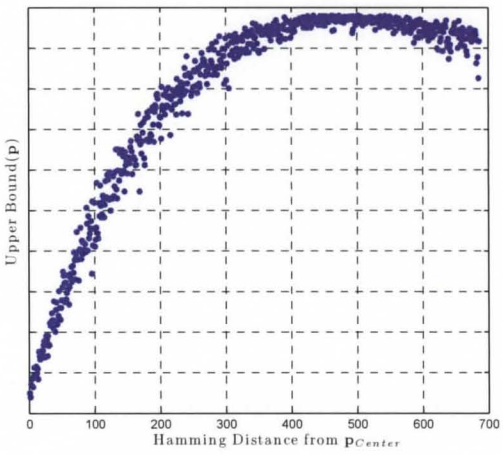
In the presented verification experiments thus far, due to the computational burden of performing SVM training thousands of times, we sampled one $\mathbf{p}$ vector for each hamming distance. This can be considered limiting considering the fact that the number of $\mathbf{p}$ vectors at a Hamming distance H from $\mathbf{p}_{Center}$ is $\binom{|\mathbf{p}|}{H}$. Therefore, In Fig.III.4c we show the output of repeatedly sampling $\mathbf{p}$ vectors at the same Hamming distance. The figure provides a magnification of the scatter plot where 300 $\mathbf{p}$ vectors are sampled at the same Hamming distance. It is clear that the direct proportionality behavior is maintained and is consistent with our previous results in Fig.III.3.

III.1.3.5 QP-S^{3}VM Model Verification Experiments Conclusion

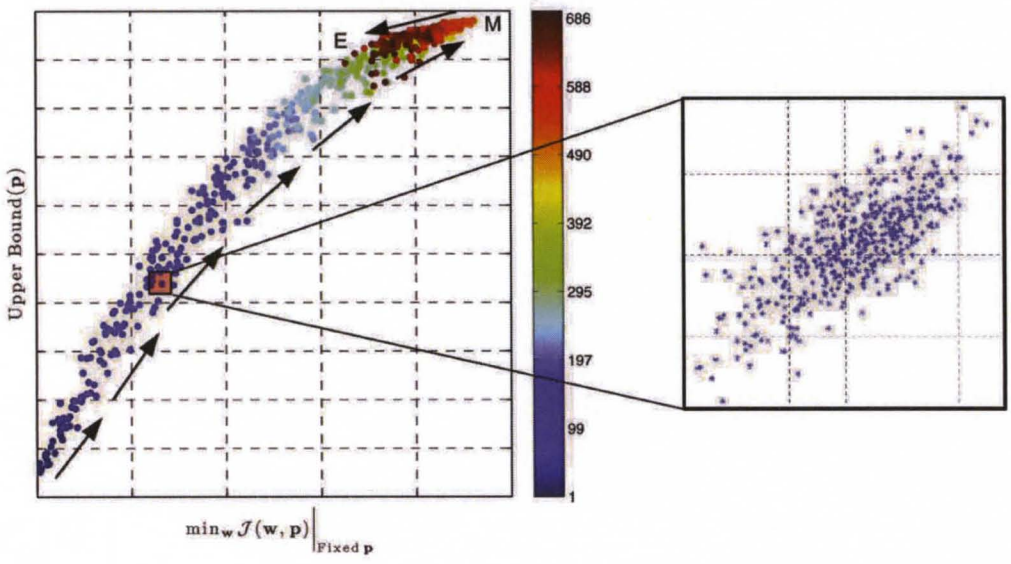
In this section we provided an experimental verification of using the upper bound proposed in Theorem 3 as a surrogate objective function for $min_{\mathbf{w}}\mathcal{J}(\mathbf{w},\mathbf{p})$ which transformed the non-convex S^{3}VM objective function in Eq.III.2 into the quadratic programming in Problem 3.



(a)



(b)



(c)

Figure III.4: (a) and (b) provide an illustration of the behavior of $min_{\mathbf{w}}\mathcal{J}(\mathbf{w}, \mathbf{p})$ and Upper Bound($\mathbf{p}$), respectively, versus the Hamming distance of the examined combinatorial vector $\mathbf{p}$ from $\mathbf{p}_{Center}$. (c) Path of increasing Hamming distance on scatter plots and an illustration of sampling 300 $\mathbf{p}$ vectors at the same Hamming distance.

III.1.4 QP-S^{3}VM Model Interpretation

In this section we provide analytical interpretation of the quadratic programming S^{3}VM (QP-S^{3}VM) obtained in Problem 3 which has the following form,

$$\underset{\mathbf{p}'=[p_1,\ldots,p_{|\mathcal{U}|}]}{\arg\min} \frac{1}{2}C^{*2}(\mathbf{1}_{|\mathcal{U}|}-\mathbf{p})'\mathbf{K_{uu}}\mathbf{p}+CC^{*}\mathbf{y}'\mathbf{K}_{lu}(\mathbf{1}_{|\mathcal{U}|}-\mathbf{p}) \tag{III.16}$$

subject to

$$\mathbf{p}'\mathbf{1}_{|\mathcal{U}|}=r|\mathcal{U}|, \quad \mathbf{0}\leq\mathbf{p}\leq\mathbf{1}_{|\mathcal{U}|}.$$

This step is necessary to ensure that the proposed approximate QP-S^{3}VM model does not conceptually deviate from the original S^{3}VM problem.

III.1.4.1 Interpreting the First Term in Eq.(III.16)

The first term in Eqn.(III.16) can be expanded as follows:

$$\begin{aligned}\frac{1}{2}C^{*2}(\mathbf{1}_{|\mathcal{U}|}-\mathbf{p})'\mathbf{K_{uu}}\mathbf{p} = & \underbrace{\frac{1}{2}C^{*2}\sum_{\substack{j,j'=\{1,\ldots,|\mathcal{U}|\}\\ j=j'}}[\mathbf{K_{uu}}]_{j,j'}p_{j'}(1-p_j)}_{\mathcal{Q}_1} \\ & \underbrace{+\frac{1}{2}C^{*2}\sum_{\substack{j=\{1,\ldots,|\mathcal{U}|-1\}\\ j'=\{j+1,\ldots,|\mathcal{U}|\}}}[\mathbf{K_{uu}}]_{j,j'}(p_j+p_{j'}-2p_jp_{j'})}_{\mathcal{Q}_2}\end{aligned} \tag{III.20}$$

As $\mathcal{Q}_1$ is negative quadratic in p_j, minimizing $\mathcal{Q}_1$ enforces the variables p_j's to take values at the extremes of their possible range, i.e. either 0 or 1. In other words, minimizing $\mathcal{Q}_1$ helps making clear assignments of the labels to the unlabeled samples. The notion of clear assignments can actually be traced back to the idea behind the S^{3}VM where decision functions are sought in low density regions, thus the name *Low Density Separation* algorithms. Therefore by avoiding confusing midrange label assignments like $p_j \simeq 0.5$, $\mathcal{Q}_1$ essentially implements the low density separation criterion of S^{3}VM where it is preferred for no unlabeled samples to exist inside the margin or near the decision boundary.

To understand the implications of minimizing $\mathcal{Q}_2$ on the solution of Problem 3, we start by plotting $z_{j,j'}=(p_j+p_{j'}-2p_jp_{j'})$, for all $p_j, p_{j'}\in[0,1]$, as shown in Fig.III.5.

In Fig.III.5 we see that small values of $z_{j,j'}$, i.e. $z_{j,j'}\simeq 0$, means that $p_j\simeq p_{j'}$ while large values of $z_{j,j'}$, i.e. $z_{j,j'}\simeq 1$, means that $p_jp_{j'}\simeq 0$. In $\mathcal{Q}_2$, since $z_{j,j'}$ is

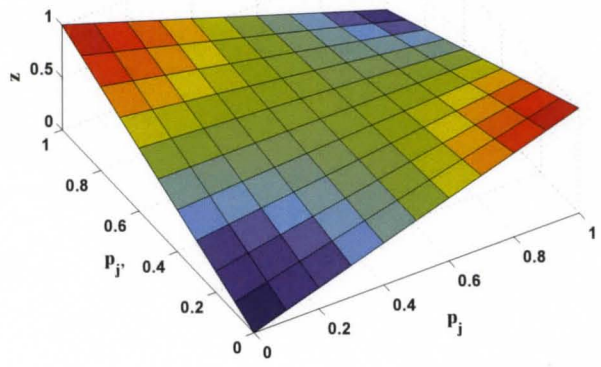


Figure III.5: Plot of $z_{j,j'} = (p_j + p_{j'} - 2p_j p_{j'})$ for all $p_j, p_{j'} \in [0, 1]$.

multiplied by $[\mathbf{K_{uu}}]_{j,j'}$, then the absolute minimum of $\mathcal{Q}_2$ can be achieved by assigning all $z_{j,j'}$ values to $zero$, which basically assigns $p_j = p_{j'}$ for all j and j', i.e. we get a degenerate solution where all the unlabeled samples are assigned to one class. This situation is highly undesirable, therefore we must assign a certain portion of $z_{j,j'}$ to small values, indicating pairs with the same label, and the rest of $z_{j,j'}$ should be assigned high values to indicate pairs with opposite labels. This is why the balancing constraint, $\mathbf{p}'\mathbf{1}_{|\mathcal{U}|} = r|\mathcal{U}|$, is important in the approximate formulation in Problem 3.

With the previous argument in mind, to minimize $\mathcal{Q}_2$ we assign small $z_{j,j'}$ to large valued $[\mathbf{K_{uu}}]_{j,j'}$. This means that when two *unlabeled samples* $\mathbf{x}_j$ and $\mathbf{x}_{j'}$ are similar, $[\mathbf{K_{uu}}]_{j,j'}$ is large, the assigned small valued $z_{j,j'}$ will force them to assume the same label, i.e. $p_j \simeq p_{j'}$. On the other hand, if $[\mathbf{K_{uu}}]_{j,j'}$ is small, indicating dissimilarlity, assigning large $z_{j,j'}$, i.e. $p_j p_{j'} \simeq 0$, will ensure that the involved unlabeled samples assume opposite labels. It is clear to see now how minimizing $\mathcal{Q}_2$ basically implements the *clustering assumption* [61] of semi-supervised learning algorithms where unlabeled samples form clusters and all samples in the same cluster have the same label.

III.1.4.2 Interpreting the Second Term in Eq.(III.16)

Next we study the second term in Eq.(III.16). We start by rewriting it as follows:

$$
\begin{aligned}
CC^*\mathbf{y}'\mathbf{K_{lu}}(\mathbf{1}_{|\mathcal{U}|}-\mathbf{p}) &= CC^* \sum_{i\in\mathcal{L},j\in\mathcal{U}} y_i[\mathbf{K_{lu}}]_{i,j}(1-p_j) \\
&= \underbrace{CC^* \sum_{\substack{i\in\mathcal{L},j\in\mathcal{U}\\ y_i=+1}} [\mathbf{K_{lu}}]_{i,j}(1-p_j)}_{\mathcal{Q}_3} + \underbrace{CC^* \sum_{\substack{i\in\mathcal{L},j\in\mathcal{U}\\ y_i=-1}} [\mathbf{K_{lu}}]_{i,j}(p_j-1)}_{\mathcal{Q}_4}
\end{aligned}
\tag{III.21}
$$

We split Eq.(III.21) into terms associated with labeled samples with $y_i = +1$, $\mathcal{Q}_3$, and those with $y_i = -1$, $\mathcal{Q}_4$. This is necessary because of the dependence of the interpretation on the labels y_i. Since $p_j \in [0, 1]$, minimizing $\mathcal{Q}_3$ involves assigning small $(1-p_j)$, i.e. $p_j \simeq 1$, to $[\mathbf{K_{lu}}]_{i,j}$ with large values and vice versa, small valued $[\mathbf{K_{lu}}]_{i,j}$ are assigned large $(1-p_j)$, i.e. $p_j \simeq 0$. In other words, if an *unlabeled sample* $\mathbf{x}_j$ that is close to, i.e. large $[\mathbf{K_{lu}}]_{i,j}$, a *labeled sample* $(\mathbf{x}_i, y_i = +1)$, then this unlabeled sample should have the same label as the labeled sample, that is $p_j \simeq 1$ and $y_j = +1$. On the other hand, if the *unlabeled sample* $\mathbf{x}_j$ is far from, i.e. small $[\mathbf{K_{lu}}]_{i,j}$, the *labeled sample* $(\mathbf{x}_i, y_i = +1)$, then this unlabeled sample should have an opposite label to that of the labeled sample, that is $p_j \simeq 0$ and $y_j = -1$. The same argument holds for minimizing $\mathcal{Q}_4$ where unlabeled samples with large/small similarity to a labeled sample $(\mathbf{x}_i, y_i = -1)$ will be assigned small/large $(p_j - 1)$, i.e. $p_j \simeq 0$ and $p_j \simeq 1$, respectively.

It is worth mentioning here that the balancing constraint, $\mathbf{p}'\mathbf{1}_{|\mathcal{U}|} = r|\mathcal{U}|$, is not of significant effect in minimizing $\mathcal{Q}_3 + \mathcal{Q}_4$. This is because of the competition between $\mathcal{Q}_3$ and $\mathcal{Q}_4$ in the assignment of p_j. Essentially, if minimizing $\mathcal{Q}_3$ took the easy road of assigning all unlabeled samples to the same label, $\mathcal{Q}_4$ will exert a high cost on that solution, and vice versa for $\mathcal{Q}_4$. Therefore, optimizing $\mathcal{Q}_3 + \mathcal{Q}_4$ has its own inherent balancing mechanism.

To summarize, the provided interpretation of Eq.(III.16) shows that the proposed QP-S^{3}VM model adheres to the core intuition of S^{3}VM by implementing the *clustering* assumption as well as the *low density separation* principle. Therefore, the approximation approach we used to derive the QP-S^{3}VM model will not affect the output of the semi-

supervised learning process.

III.1.4.3 QP-S^{3}VM Model Relationship to Graph-based Semi-supervised Learning Methods

The proposed QP-S^{3}VM model inherently belongs to the *low density separation* semi-supervised leaning methods. In this section we discuss another aspect of the QP-S^{3}VM model in terms of its relationship to graph-based methods. In graph-based semi-supervised learning methods [79, 80], a weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is constructed, where the vertex set $\mathcal{V} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{|\mathcal{L}|+|\mathcal{U}|}\}$ consists of all labeled and unlabeled data samples, and $\mathcal{E} = \{w_{ij}\}$ is the set of edges where w_{ij} represents the similarity between the vertices $\mathbf{x}_i$ and $\mathbf{x}_j$. Once the graph is obtained, learning basically assigns labels y_i to the vertices set $\mathcal{V}$ by using the edges connecting the labeled vertices to the unlabeled ones. Given that the edges w_{ij} represent the similarity between vertices, the idea behind semi-supervised learning on the graph is that for any two vertices $\mathbf{x}_i$ and $\mathbf{x}_j$ with large edge weight w_{ij}, the assigned labels y_i and y_j are expected to be the same. Based on this simple notion, semi-supervised learning on a graph can be intuitively described as the process where labels are propagated from labeled vertices to unlabeled ones via graph edges, where the extent of propagation is controlled by the edge wights. This is why this class of algorithms are also called *Label Propagation* algorithms.

One thing to clarify is that it is not necessary for every unlabeled vertex to have a direct edge with a labeled vertex. However, it is enough for an unlabeled vertex to be connected to a labeled vertex through many intermediate strongly connected unlabeled vertices where such intermediate unlabeled vertices act as stepping stones for the label to propagate from the labeled vertex.

Graph-based semi-supervised learning algorithms formalize the idea of label propagation by estimating a labeling function f on the graph $\mathcal{G}$ with two important constraints: First, the function f must respect the labeled vertices. That is, for all labeled vertices $\mathbf{x}_i$ with given labels y_i, the output of $f(\mathbf{x}_i)$ should be very close to y_i. Second, the function f should be smooth with respect to the graph. Estimating f is usually formulated in a regularization framework, where a loss function is used to enforce consistency with the la-

beled vertices and the smoothness is imposed via regularization using various forms of the graph Laplacian. One popular algorithm for graph-based semi-supervised learning is the *harmonic function* algorithm [80] where estimating the function f is formulated as follows,

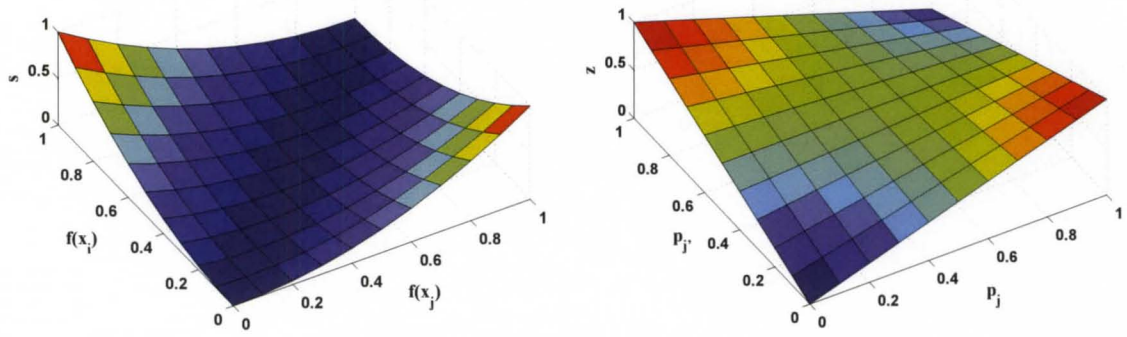
$$\min_{f:f(\mathbf{x})\in\mathbb{R}} \underbrace{\infty \sum_{i\in\mathcal{L}} (y_i - f(\mathbf{x}_i))^2}_{\text{Loss}} + \underbrace{\sum_{i,j\in\mathcal{L}\cup\mathcal{U}} w_{ij}(f(\mathbf{x}_i) - f(\mathbf{x}_j))^2}_{\text{Regularization via Graph Laplacian}}. \tag{III.22}$$

Looking at Eq.(III.22), it is clear that minimizing regularizer term enforces $f(\mathbf{x}_i) \approx f(\mathbf{x}_j)$ for large w_{ij}, thus achieving label smoothness over the graph. This regularizer has another form, $2\,\mathbf{f}'\mathbf{L}\mathbf{f}$, where $\mathbf{f}' = \left[f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_{|\mathcal{L}|+|\mathcal{U}|})\right]$ and $\mathbf{L}$ is the *combinatorial graph Laplacian*, where $\mathbf{L} = D - W$, W is the similarity matrix with all weights w_{ij}, and D is the diagonal degree matrix with diagonal elements $d_{ii} = \sum_j w_{ij}$.

With the graph regularization formulation of Eq.(III.22) in mind, the QP-S^{3}VM model can be interpreted under the graph regularization framework where $\mathcal{Q}_2$ is the regularization term with respect to the similarity kernel matrix $\mathbf{K}$. To illustrate this idea we compare the regularization behavior of both the graph Laplacian in Eq.(III.22) and that of $\mathcal{Q}_2$ in Fig.III.6. For the sake of clarity we assume that the function $f(\mathbf{x})$ estimated in Eq.(III.22) has the same range as the variables q_j in $\mathcal{Q}_2$, i.e. $f(\mathbf{x}) \in [0, 1]$. Figure III.6a depicts the regularization effect of graph Laplacian by plotting $s_{i,j} = (f(\mathbf{x}_i) - f(\mathbf{x}_j))^2$, for all $f(\mathbf{x}) \in [0, 1]$. For easy reference Fig.III.6b depicts the smoothness behavior of $\mathcal{Q}_2$ provided earlier in Fig.III.5.

Figure III.6 shows that the minimum of both the graph Laplacian regularization and $\mathcal{Q}_2$ encourages giving similar labels to similar samples, i.e. smoothness, as shown in the blue colored areas of both depictions. One important difference though is that the graph Laplacian regularization, Fig.III.6a, enforces the label smoothness without regard to value of $f(\mathbf{x})$; $f(\mathbf{x}_i) \approx f(\mathbf{x}_j)$ for all $f(\mathbf{x}) \in [0, 1]$. On the other hand, $\mathcal{Q}_2$ not only enforces smoothness but also encourages the values of labels to be localized around 0 or 1, see Fig.III.6b, which is consistent with the assumptions of the low density separation methods explained earlier. Therefore, $\mathcal{Q}_2$ can be thought of as a graph regularization with the additional property of clear label assignments.

Finally, the process of jointly minimizing $\mathcal{Q}_2$, which is basically a regularization



(a) (b)

Figure III.6: (a) Plot of the graph Laplacian regularizer, $s_{i,j} = (f(\mathbf{x}_i) - f(\mathbf{x}_j))^2$, for all $f(\mathbf{x}) \in [0, 1]$. (b) Plot of $z_{j,j'} = (p_j + p_{j'} - 2p_j p_{j'})$ for all $p_j, p_{j'} \in [0, 1]$.

term, and $\mathcal{Q}_3 + \mathcal{Q}_4$, where unlabeled samples are assigned labels by their similarity to labeled samples, results in a formulation that follows the same intuition behind *label propagation* algorithms [80] for semi-supervised learning. This shows that both categories of semi-supervised learning algorithms, namely *graph-based methods (label propagation methods)* and *low density separation methods*, are connected in a well principled manner despite their different origins. The impact of this connection goes beyond producing a universal semi-supervised learning framework where algorithms from both *graph-based SSL methods* and *low density separation SSL methods* provide their best properties. This connection actually extends to establishing relationships with spectral and kernel unsupervised methods [25].

III.1.4.4 QP-S^{3}VM Model Interpretation Conclusion

In this section, we provided a detailed discussion of how the formulation in Problem 3 (QP-S^{3}VM), though approximate, does not deviate from the general paradigm of the semi-supervised learning. Moreover, we showed that the provided formulation presents an insight into the connection between the *Low Density Separation* semi-supervised algorithms, which include S^{3}VM, and the *Graph-based* algorithms.

III.2 Submodular Optimization of QP-S^{3}VM

The approximate QP-S^{3}VM formulation proposed in Problem 3 is simple and intuitive. However, due to the fact that it is a quadratic minimization of a concave function, the computational complexity of finding a solution will become a hindering issue specially for semi-supervised learning problems which are inherently large scale. In this section we use the concepts of submodular set functions to provide a simple and efficient algorithm for the proposed approximate QP-S^{3}VM problem.

III.2.1 Semi-supervised Learning as a Set Function Optimization

As discussed in Section III.1, the solution of the QP-S^{3}VM problem provides a value for the variable p_j associated with each unlabeled sample $\mathbf{x}_j, j \in \mathcal{U}$, such that $p_j = 1$ for $y_j = +1$ and $p_j = 0$ for $y_j = -1$. In this section, we use a different perspective of the problem. In this new perspective the problem of binary semi-supervised classification in general is concerned with choosing a subset $\mathcal{A}$ from the pool of all unlabeled samples $\mathcal{U}$. All the unlabeled samples $\mathbf{x}_j, j \in \mathcal{A}$, should be assigned the label $y_j = +1$, and the rest of them, $\mathbf{x}_j, j \in \mathcal{U}\backslash\mathcal{A}$, will be assigned the label $y_j = -1$. Each possible subset $\mathcal{A}$ is assigned a value by a *set function* $f(\mathcal{A})$ that has the same optimal solution, in terms of $\mathcal{A}$ and $\mathcal{U}\backslash\mathcal{A}$, as the original semi-supervised classification problem. Reformulating the semi-supervised learning problem into optimizing set functions, more precisely submodular set functions, puts at our disposal a vast arsenal of efficient optimization techniques that were not used in this capacity before [35].

III.2.2 Solving QP-S^{3}VM Using Submodular Optimization (SUBMOD-S^{3}VM)

In this section we use the concepts of submodular functions maximization to provide an efficient and simple algorithm for solving the approximate QP-S^{3}VM problem. Towards this goal we propose a submodular maximization problem, in Problem 4, that is equivalent to the approximate QP-S^{3}VM in Problem 3. For the sake of quick reference, we restate Problem 3 here.

Problem 3. *Quadratic Programming Approximation of Semi-supervised Support Vector*

Machines (QP-S^{3}VM):

$$\underset{\mathbf{p}'=[p_1,\ldots,p_{|\mathcal{U}|}]}{\arg\min} \frac{1}{2}C^{*2}(\mathbf{1}_{|\mathcal{U}|}-\mathbf{p})'\mathbf{K_{uu}}\mathbf{p}+CC^{*}\mathbf{y}'\mathbf{K}_{lu}(\mathbf{1}_{|\mathcal{U}|}-\mathbf{p}) \tag{III.16}$$

subject to

$$\mathbf{p}'\mathbf{1}_{|\mathcal{U}|}=r|\mathcal{U}|, \quad \mathbf{0}\leq\mathbf{p}\leq\mathbf{1}_{|\mathcal{U}|}.$$

Note: Equation (III.16) *can be rewritten in the standard quadratic programming form as follows:*

$$\underset{\mathbf{p}'=[p_1,\ldots,p_{|\mathcal{U}|}]}{\arg\min} (\frac{1}{2}C^{*2}\mathbf{1}'_{|\mathcal{U}|}\mathbf{K_{uu}}-CC^{*}\mathbf{y}'\mathbf{K_{lu}})\mathbf{p}-\frac{1}{2}C^{*2}\mathbf{p}'\mathbf{K_{uu}}\mathbf{p} \tag{III.18}$$

III.2.2.1 Discrete Representation

Before stating the proposed submodular maximization problem, we start by providing the discrete version of the quadratic objective function in Eq.(III.18):

• *Discretizing* $\frac{1}{2}C^{*2}\mathbf{1}'_{|\mathcal{U}|}\mathbf{K_{uu}}\mathbf{p}$:

Let $\mathbf{T}=[t_1,\ldots,t_{|\mathcal{U}|}]$, such that $t_{j'}=\sum_{j\in\mathcal{U}}[\mathbf{K_{uu}}]_{j,j'}$, then

$$\begin{aligned}\frac{1}{2}C^{*2}\mathbf{1}'_{|\mathcal{U}|}\mathbf{K_{uu}}\mathbf{p} &= \frac{1}{2}C^{*2}\mathbf{T}\mathbf{p}\\ &= \frac{1}{2}C^{*2}\sum_{j\in\mathcal{U}}t_jp_j \ .\end{aligned}$$

As described earlier, in Sec.III.2.1, the set function formulation of semi-supervised learning makes $p_j\in\{0,1\}$. Therefore, only nonzero p_j's, i.e. $p_j,\forall j\in\mathcal{A}$, will contribute to the term in hand. Thus,

$$\begin{aligned}\frac{1}{2}C^{*2}\mathbf{1}'_{|\mathcal{U}|}\mathbf{K_{uu}}\mathbf{p} &= \frac{1}{2}C^{*2}\sum_{j\in\mathcal{A}}t_j\\ &= \frac{1}{2}C^{*2}\sum_{j\in\mathcal{A}}\sum_{j'\in\mathcal{U}}[\mathbf{K_{uu}}]_{j,j'} \ .\end{aligned} \tag{III.23}$$

• *Discretizing* $-CC^{*}\mathbf{y}'\mathbf{K_{lu}}\mathbf{p}$:

Let $\mathbf{T}=[t_1,\ldots,t_{|\mathcal{U}|}]$, such that $t_j=\sum_{i\in\mathcal{L}}y_i[\mathbf{K_{lu}}]_{i,j}$, then

$$\begin{aligned}-CC^{*}\mathbf{y}'\mathbf{K_{lu}}\mathbf{p} &= -CC^{*}\mathbf{T}\mathbf{p}\\ &= -CC^{*}\sum_{j\in\mathcal{U}}t_jp_j=-CC^{*}\sum_{j\in\mathcal{A}}t_j\\ &= -CC^{*}\sum_{j\in\mathcal{A}}\sum_{i\in\mathcal{L}}y_i[\mathbf{K_{lu}}]_{i,j}\end{aligned} \tag{III.24}$$

• *Discretizing* $-\frac{1}{2}C^{*2}\mathbf{p}'\mathbf{K_{uu}}\mathbf{p}$:

Let $\mathbf{T} = [t_1, \ldots, t_{|\mathcal{U}|}]$, such that $t_{j'} = \sum_{j\in\mathcal{U}} p_j[\mathbf{K_{uu}}]_{j,j'} = \sum_{j\in\mathcal{A}}[\mathbf{K_{uu}}]_{j,j'}$, then

$$\begin{aligned} -\frac{1}{2}C^{*2}\mathbf{p}'\mathbf{K_{uu}}\mathbf{p} &= -\frac{1}{2}C^{*2}\mathbf{T}\mathbf{p} \\ &= -\frac{1}{2}C^{*2}\sum_{j\in\mathcal{U}} t_j p_j = -\frac{1}{2}C^{*2}\sum_{j\in\mathcal{A}} t_j \\ &= -\frac{1}{2}C^{*2}\sum_{j\in\mathcal{A}}\sum_{j'\in\mathcal{A}}[\mathbf{K_{uu}}]_{j,j'} \end{aligned} \tag{III.25}$$

Using the discretized version of Eq.III.18, we propose the following submodular maximization formulation:

Problem 4. *Submodular maximization formulation that is equivalent to Problem 3:*

$$\max_{|\mathcal{A}|\leq r|\mathcal{U}|} \mathcal{S}(\mathcal{A}) \tag{III.26}$$

where

$$\begin{aligned} \mathcal{S}(\mathcal{A}) = &-\frac{1}{2}C^{*2}\sum_{j\in\mathcal{A},j'\in\mathcal{U}}[\mathbf{K_{uu}}]_{j,j'} + CC^*\sum_{j\in\mathcal{A},i\in\mathcal{L}} y_i\,[\mathbf{K_{lu}}]_{i,j} + \frac{1}{2}C^{*2}\sum_{j,j'\in\mathcal{A}}[\mathbf{K_{uu}}]_{j,j'} \\ &+ \underbrace{\sum_{j,j'\in\mathcal{A}} d\left[\delta_{j,j'}\left(\frac{3}{2}C^{*2}|\mathcal{U}| + CC^*|\mathcal{L}|\right) - \frac{1}{2}C^{*2}\right]}_{\mathcal{Q}_5}, \end{aligned} \tag{III.27}$$

and $\mathcal{S}$ is a submodular set function defined on all subsets $\mathcal{A} \subset \mathcal{U}$ of unlabeled samples assigned to the class $y_j = +1$, $0 \leq \mathbf{K}_{.,.} \leq d$, and $\delta_{j,j'} = 1$ for $j = j'$ and 0 otherwise.

Problem 4 basically maximizes the negative of a discrete version of the objective function in Eq.(III.18). The first three terms in $\mathcal{S}(\mathcal{A})$ are those derived in Eq.(III.23), (III.24) and (III.25). However, the term $\mathcal{Q}_5$ is of our design, and it is added to ensure the monotonicity and submodularity of $\mathcal{S}(\mathcal{A})$, as will be shown in Theorem 4.

III.2.2.2 $\mathcal{Q}_5$ Design

The initial submodular maximization formulation of (III.18) has the the following form,

$$\mathcal{S}_{Initial}(\mathcal{A}) = -\frac{1}{2}C^{*2}\sum_{j\in\mathcal{A},j'\in\mathcal{U}}[\mathbf{K_{uu}}]_{j,j'} + CC^*\sum_{j\in\mathcal{A},i\in\mathcal{L}} y_i\,[\mathbf{K_{lu}}]_{i,j} + \frac{1}{2}C^{*2}\sum_{j,j'\in\mathcal{A}}[\mathbf{K_{uu}}]_{j,j'}$$

The design of $\mathcal{Q}_5$ took place through several iterations of examining the monotonicity and submodularity properties of $\mathcal{S}_{Initial}(\mathcal{A})$, then compensating for the unsatisfied conditions with terms that should not affect the optimal solution of the optimization problem.

In the first design iteration, we examined the monotonicity of $\mathcal{S}_{Initial}(\mathcal{A})$, where $\forall m \notin \mathcal{A}$,

$$\mathcal{S}_{Initial}(\mathcal{A} \cup m) - \mathcal{S}_{Initial}(\mathcal{A}) = \underbrace{-\frac{1}{2}C^{*2}\sum_{j' \in \mathcal{U}}[\mathbf{K_{uu}}]_{j',m}}_{\mathbf{D_1}} + \underbrace{CC^*\sum_{i \in \mathcal{L}} y_i[\mathbf{K_{lu}}]_{i,m}}_{\mathbf{D_2}} + \underbrace{C^{*2}\sum_{j \in \mathcal{A}}[\mathbf{K_{uu}}]_{m,j}}_{\mathbf{D_3}} + \underbrace{\frac{1}{2}C^{*2}[\mathbf{K_{uu}}]_{m,m}}_{\mathbf{D_4}}.$$

It is clear that $\mathbf{D}_3$ and $\mathbf{D}_4$ are non-negative, as $0 \leq \mathbf{K}_{.,.} \leq d$. On the other hand, $\mathbf{D}_1$ is non-positive and the sign of $\mathbf{D}_2$ depends on how many of the labels $y_i = -1$. Therefore, the monotonicity of $\mathcal{S}_{Initial}$ is conditional on the available data labels and the associated kernel matrix. To overcome this dependence and achieve absolute monotonicity for $\mathcal{S}_{Initial}$, we add the constant term $\left(\frac{1}{2}C^{*2}|\mathcal{U}| + CC^*|\mathcal{L}|\right)|\mathcal{A}|d$, which ensures that possible negative values produced by $\mathbf{D}_1$ and $\mathbf{D}_2$ are cancelled out. The new updated function, $\mathcal{S}_{Updated}$, have the form,

$$\begin{aligned}\mathcal{S}_{Updated-1}(\mathcal{A}) &= \mathcal{S}_{Initial}(\mathcal{A}) + \left(\frac{1}{2}C^{*2}|\mathcal{U}| + CC^*|\mathcal{L}|\right)|\mathcal{A}|d \\ &= -\frac{1}{2}C^{*2}\sum_{j \in \mathcal{A}, j' \in \mathcal{U}}[\mathbf{K_{uu}}]_{j,j'} + CC^*\sum_{j \in \mathcal{A}, i \in \mathcal{L}} y_i\,[\mathbf{K_{lu}}]_{i,j} + \frac{1}{2}C^{*2}\sum_{j,j' \in \mathcal{A}}[\mathbf{K_{uu}}]_{j,j'} \\ &\quad + \left(\frac{1}{2}C^{*2}|\mathcal{U}| + CC^*|\mathcal{L}|\right)|\mathcal{A}|d.\end{aligned}$$

In the next design iteration, we examined the submodularity of $\mathcal{S}_{Updated-1}(\mathcal{A})$. We applied the submodularity condition in Def.1 on the smallest possible case of diminishing return law, where a set $\mathcal{A} \subset \mathcal{B}$ and $\mathcal{B} = \{\mathcal{A} \cup q\}$. That is $\forall m \notin \mathcal{B}$,

$$\begin{aligned}\mathcal{S}_{Updated-1}(\mathcal{A} \cup m) - \mathcal{S}_{Updated-1}(\mathcal{B} \cup m) &= \mathcal{S}_{Updated-1}(\mathcal{A} \cup m) - \mathcal{S}_{Updated-1}(\mathcal{A} \cup q \cup m) \\ &= -C^{*2}[\mathbf{K}_{uu}]_{q,m} \quad ,\end{aligned}$$

which means that the current set function, $\mathcal{S}_{Updated-1}$, is not submodular. We then trace this result back to the original formulation of $\mathcal{S}_{Updated-1}$ and we find that adding the constant $-\frac{1}{2}C^{*2}|\mathcal{A}|^2 d$ overcomes the non-submodularity issue. The updated set function now has the form,

$$\begin{aligned}\mathcal{S}_{Updated-2}(\mathcal{A}) &= \mathcal{S}_{Updated-1}(\mathcal{A}) - \frac{1}{2}C^{*2}|\mathcal{A}|^2 d \\ &= -\frac{1}{2}C^{*2}\sum_{j\in\mathcal{A},j'\in\mathcal{U}}[\mathbf{K_{uu}}]_{j,j'} + CC^*\sum_{j\in\mathcal{A},i\in\mathcal{L}} y_i\,[\mathbf{K_{lu}}]_{i,j} + \frac{1}{2}C^{*2}\sum_{j,j'\in\mathcal{A}}[\mathbf{K_{uu}}]_{j,j'} \\ &\quad + \left(\frac{1}{2}C^{*2}|\mathcal{U}| + CC^*|\mathcal{L}|\right)|\mathcal{A}|d - \frac{1}{2}C^{*2}|\mathcal{A}|^2 d.\end{aligned}$$

However, the last update performed on the set function violated the monotonicity condition again. Therefore, we repeated the design process one more time and we found that adding a constant $C^{*2}|\mathcal{U}||\mathcal{A}|d$ will suffice to reach the final monotone submodular function, $\mathcal{S}_{Final}(\mathcal{A})$ described in Problem 4,

$$\begin{aligned}\mathcal{S}_{Final}(\mathcal{A}) &= -\frac{1}{2}C^{*2}\sum_{j\in\mathcal{A},j'\in\mathcal{U}}[\mathbf{K_{uu}}]_{j,j'} + CC^*\sum_{j\in\mathcal{A},i\in\mathcal{L}} y_i\,[\mathbf{K_{lu}}]_{i,j} + \frac{1}{2}C^{*2}\sum_{j,j'\in\mathcal{A}}[\mathbf{K_{uu}}]_{j,j'} \\ &\quad + \underbrace{\sum_{j,j'\in\mathcal{A}} d\left[\delta_{j,j'}\left(\frac{3}{2}C^{*2}|\mathcal{U}| + CC^*|\mathcal{L}|\right) - \frac{1}{2}C^{*2}\right]}_{\mathcal{Q}_5},\end{aligned}$$

It is worth mentioning that $\mathcal{S}_{Final}(\emptyset) = 0$, which is a necessary condition for using the greedy maximization approach.

III.2.2.3 Implications of Using $\mathcal{Q}_5$ to Satisfy Monotonicity and Submodularity

Since for a fixed $|\mathcal{A}|$ the value of $\mathcal{Q}_5$ is constant, then the solution obtained by optimizing $\mathcal{S}(\mathcal{A})$ is not affected by adding $\mathcal{Q}_5$. In other words, examining $\mathcal{Q}_5$

$$\begin{aligned}\mathcal{Q}_5 &= \sum_{j,j'\in\mathcal{A}} d\left[\delta_{j,j'}\left(\frac{3}{2}C^{*2}|\mathcal{U}| + CC^*|\mathcal{L}|\right) - \frac{C^{*2}}{2}\right] \\ &= \left(\frac{3}{2}C^{*2}|\mathcal{U}| + CC^*|\mathcal{L}|\right)|\mathcal{A}|d - \frac{C^{*2}}{2}|\mathcal{A}|^2 d\end{aligned}$$

shows that it depends on the cardinality of $\mathcal{A}$ not its contents.

Although adding $\mathcal{Q}_5$ in Problem 4 (SUBMOD-S^{3}VM) does not change the fact that both Problem 3 (QP-S^{3}VM) and the proposed submodular SUBMOD-S^{3}VM have the same optimal solution, adding $\mathcal{Q}_5$ will affect the approximation guarantee of the greedy approach when applied on SUBMOD-S^{3}VM. To better understand this issue, let us denote the discretized version of Problem 3 by $\mathcal{D}(\mathcal{A})$ such that

$$\mathcal{D}(\mathcal{A}) = -\frac{1}{2}C^{*2}\sum_{j\in\mathcal{A},j'\in\mathcal{U}}[\mathbf{K_{uu}}]_{j,j'} + CC^*\sum_{j\in\mathcal{A},i\in\mathcal{L}} y_i\,[\mathbf{K_{lu}}]_{i,j} + \frac{1}{2}C^{*2}\sum_{j,j'\in\mathcal{A}}[\mathbf{K_{uu}}]_{j,j'}\,.$$

Then, for the optimal solution $\mathcal{A}^*$, the proposed $\mathcal{S}(\mathcal{A})$ in SUBMOD-S^{3}VM can be written as,

$$\mathcal{S}(\mathcal{A}^*) = \mathcal{D}(\mathcal{A}^*) + \mathcal{Q}_5,$$

with $\mathcal{Q}_5$ being a constant as mentioned earlier. Using the *submodular greedy* algorithm to maximize $\mathcal{S}(\mathcal{A})$ will result in a solution $\mathcal{A}_{Greedy}$ such that,

$$\mathcal{S}(\mathcal{A}_{Greedy}) \geq (1-1/e)\mathcal{S}(\mathcal{A}^*),$$

which can be written in terms of $\mathcal{D}(\mathcal{A})$ as follows,

$$\mathcal{D}(\mathcal{A}_{Greedy}) + \mathcal{Q}_5 \;\geq\; (1-1/e)[\mathcal{D}(\mathcal{A}^*) + \mathcal{Q}_5] \tag{III.28}$$

$$\mathcal{D}(\mathcal{A}_{Greedy}) \;\geq\; (1-1/e)\mathcal{D}(\mathcal{A}^*) - (1/e)\mathcal{Q}_5. \tag{III.29}$$

Therefore, we see that the actual obtained lower bound on the approximation of $\mathcal{D}$ is less than the $(1-1/e)$, i.e. 63.21%, promised by the *submodular greedy maximization* algorithm. It is important here to remember that this is a lower bound on the performance and that the actual performance is usually much higher. Section III.4.3.1 provides imperical estimates for the approximation percentage $\mathcal{D}(\mathcal{A}_{Greedy})/\mathcal{D}(\mathcal{A}^*)$ achieved by the greedy algorithm for several of data sets. For most of the data sets, the obtained approximation is more than 98% of the optimal value and the lowest reached approximation is 87.5%.

III.2.2.4 Proposed Submodular Optimization of S^{3}VM (SUBMOD-S^{3}VM)

In Theorem 4, we provide the detailed proof that the proposed set function $\mathcal{S}(\mathcal{A})$ is monotone submodular, and thus can be maximized using the simple submodular greedy maximization algorithm.

Theorem 4. *The set function $\mathcal{S}(\mathcal{A})$ in Problem 4 is monotone (non-decreasing), submodular, and $\mathcal{S}(\emptyset) = 0$.*

Proof. First, $\mathcal{S}(\emptyset) = 0$ follows directly from the definition in Eq.(III.27) where all the summations are on elements in the set $\mathcal{A}$. Therefore if $\mathcal{A} = \emptyset$ then $\mathcal{S}(\emptyset) = 0$. Next we prove the *monotonicity property*. Using the definition of $\mathcal{S}(\mathcal{A})$, we can show that for any $m \in \mathcal{U}$ and $m \notin \mathcal{A}$, the increase in the objective value of $\mathcal{S}$ due to adding m is,

$$\begin{aligned}\mathcal{S}(\mathcal{A} \cup m) - \mathcal{S}(\mathcal{A}) \;=\; & -\frac{1}{2}C^{*2}\sum_{j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{m,j'} + CC^* \sum_{i \in \mathcal{L}} y_i \, [\mathbf{K_{lu}}]_{i,m} \\ & + C^{*2}\sum_{j' \in \mathcal{A}} [\mathbf{K_{uu}}]_{m,j'} - C^{*2}|\mathcal{A}|d \\ & + \frac{1}{2}C^{*2}\Big([\mathbf{K_{uu}}]_{m,m} - d\Big) + \frac{3}{2}C^{*2}|\mathcal{U}|d + CC^*|\mathcal{L}|d\end{aligned} \tag{III.30}$$

Since we are dealing with semi-supervised learning problems, then $|\mathcal{U}| \gg |\mathcal{L}|$. For any kernel matrix $\mathbf{K}$, where $0 \leq \mathbf{K}_{i,j} \leq d$, since

$$\begin{aligned} & C^{*2}\sum_{j' \in \mathcal{A}} [\mathbf{K_{uu}}]_{m,j'} \geq 0 \\ & CC^*|\mathcal{L}|d + CC^* \sum_{i \in \mathcal{L}} y_i \, [\mathbf{K_{lu}}]_{i,m} \geq 0 \\ & \frac{1}{2}C^{*2}|\mathcal{U}|d \geq \frac{1}{2}C^{*2}\sum_{j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{m,j'} \\ & C^{*2}|\mathcal{U}|d + \frac{1}{2}C^{*2}\Big([\mathbf{K_{uu}}]_{m,m} - d\Big) \geq C^{*2}|\mathcal{A}|d \end{aligned} \tag{III.31}$$

then,

$$\mathcal{S}(\mathcal{A} \cup m) - \mathcal{S}(\mathcal{A}) \geq 0.$$

Thus the monotonicity property of $\mathcal{S}(\mathcal{A})$ holds true.

Now we prove the *submodularity* of $\mathcal{S}(\mathcal{A})$ by assuming the set $\mathcal{B} = \{\mathcal{A} \cup q\}$, where $q \in \mathcal{U}$. Using the same set element m we used earlier, i.e. $m \in \mathcal{U}$ and $m \notin \mathcal{A}$, we need to show that adding m to the set $\mathcal{A}$ has more effect than adding it to the set $\mathcal{B}$ as stated in Def. 2. Since

$$\begin{aligned}\mathcal{S}(\mathcal{B}) \;=\; & -\frac{1}{2}C^{*2}\sum_{j \in \{\mathcal{A} \cup q\}, j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{j,j'} + CC^* \sum_{j \in \{\mathcal{A} \cup q\}, i \in \mathcal{L}} y_i \, [\mathbf{K_{lu}}]_{i,j} + \frac{1}{2}C^{*2}\sum_{j,j' \in \{\mathcal{A} \cup q\}} [\mathbf{K_{uu}}]_{j,j'} \\ & + \sum_{j,j' \in \{\mathcal{A} \cup q\}} d\left[\delta_{j,j'}\left(\frac{3}{2}C^{*2}|\mathcal{U}| + CC^*|\mathcal{L}|\right) - \frac{1}{2}C^{*2}\right]\end{aligned} \tag{III.32}$$

then

$$\begin{aligned}\mathcal{S}(\mathcal{B}\cup m)-\mathcal{S}(\mathcal{B}) = &-\frac{1}{2}C^{*2}\sum_{j'\in\mathcal{U}}\left[\mathbf{K_{uu}}\right]_{m,j'}+CC^{*}\sum_{i\in\mathcal{L}}y_i\left[\mathbf{K_{lu}}\right]_{i,m}\\ &+C^{*2}\sum_{j'\in\{\mathcal{A}\cup q\}}\left[\mathbf{K_{uu}}\right]_{m,j'}-C^{*2}\left(|\mathcal{A}|+1\right)d \qquad \text{(III.33)}\\ &+\frac{1}{2}C^{*2}\Big(\left[\mathbf{K_{uu}}\right]_{m,m}-d\Big)+\frac{3}{2}C^{*2}|\mathcal{U}|d+CC^{*}|\mathcal{L}|d\end{aligned}$$

Therefore

$$\left[\mathcal{S}(\mathcal{A}\cup m)-\mathcal{S}(\mathcal{A})\right]-\left[\mathcal{S}(\mathcal{B}\cup m)-\mathcal{S}(\mathcal{B})\right] \;=\; C^{*2}\Big(d-\left[\mathbf{K_{uu}}\right]_{q,m}\Big)\geq 0 \qquad \text{(III.34)}$$

Hence the set function $\mathcal{S}(\mathcal{A})$ is submodular.

□

Now that we have shown that $\mathcal{S}(\mathcal{A})$ is monotonic, submodular, and $\mathcal{S}(\emptyset)=0$ this means that the greedy maximization algorithm can be used to optimize Problem 4 and the performance guarantee in Theorem 2 holds true.

To sum up, the proposed equivalent submodular maximization in Problem 4 is defined on the all subsets $\mathcal{A}$ of samples belonging to the class labeled $y_j=+1$. The efficient greedy algorithm in Algorithm 2 is used to the solve the problem efficiently. Once the optimum solution A^* is determined, the rest of the unlabeled samples, i.e. $\mathcal{U}\backslash\mathcal{A}^*$, will belong to class with labels $y_j=-1$. We use the proposed algorithm in the transductive setting of semi-supervised learning. However, if the inductive setting is needed, a standard supervised SVM training can be performed to give the final hyperplane $\mathbf{w}$.

III.2.2.5 SUBMOD-S^{3}VM Algorithm and Implementation Details

SUBMOD-S^{3}VM is an iterative greedy algorithm. In each iteration all available unlabeled samples $\mathbf{x}_j, j\in\mathcal{U}$ are evaluated against the *marginal benefit function*, defined in Eq.(III.30), and the sample $\mathbf{x}_m$ with the most *benefit* is added to the set of selections $\mathcal{A}$. The process is repeated $r|\mathcal{U}|$ times, i.e. the expected number of samples with $y_j=+1$ and $j\in|\mathcal{U}|$, where only previously unselected samples, $\mathbf{x}_j, j\in\mathcal{U}\backslash\mathcal{A}$, are considered as illustrated

in Algorithm 3.

$$\begin{aligned}\mathcal{S}(\mathcal{A}\cup m)-\mathcal{S}(\mathcal{A}) \;=\; & \underbrace{-\frac{1}{2}C^{*2}\sum_{j'\in\mathcal{U}}[\mathbf{K_{uu}}]_{m,j'}}_{\text{Term}_1}+\underbrace{CC^{*}\sum_{i\in\mathcal{L}}y_i\,[\mathbf{K_{lu}}]_{i,m}}_{\text{Term}_2}\\ & +\underbrace{C^{*2}\sum_{j'\in\mathcal{A}}[\mathbf{K_{uu}}]_{m,j'}}_{\text{Term}_3}+\underbrace{(-C^{*2}|\mathcal{A}|d)}_{\text{Term}_4}\\ & +\underbrace{\frac{1}{2}C^{*2}\left([\mathbf{K_{uu}}]_{m,m}-d\right)}_{\text{Term}_5}+\underbrace{\frac{3}{2}C^{*2}|\mathcal{U}|d+CC^{*}|\mathcal{L}|d}_{\text{Term}_{\text{Const}}}\end{aligned} \tag{III.30}$$

Algorithm 3: Greedy Algorithm to Optimize SUBMOD-S^{3}VM [55, 64]

Input : Set of Labeled Samples $\{(\mathbf{x}_i, y_i)\}$, $i \in \mathcal{L}$.
Set of Unlabeled Samples $\{\mathbf{x}_j\}$, $j \in \mathcal{U}$.
Ratio of Positive Labels in the set of Unlabeled Samples; $r = \frac{|\{y_j=+1, j\in\mathcal{U}\}|}{|\mathcal{U}|}$.

Output: Positively Labeled Samples in $\mathcal{U}$; $\mathcal{A}^* = \{j \in \mathcal{U} : y_j = +1\}$.
Negatively Labeled Samples in $\mathcal{U}$; $\mathcal{U}\backslash\mathcal{A}^* = \{j \in \mathcal{U} : y_j = -1\}$.

```
begin
    Set A_0 := φ
    for i := 1 to r|U| do
        foreach m ∈ U\A_{i-1} do
            MarginalBenefit [m] := S(A_{i-1} ∪ m) − S(A_{i-1})
        end
        m* := arg max_m MarginalBenefit [m]
        A_i := A_{i-1} ∪ m*
    end
    A* := A_{r|U|}
end
```

As the focus of this work is handling very large data, it is important to emphasize that, despite being iterative, the SUBMOD-S^{3}VM algorithm is computationally efficient. To illustrate this idea we examine the *marginal benefit function* in Eq.(III.30) which is considered the computational bottleneck of the algorithm as it is for all available unlabeled samples reevaluated in each iteration. Looking at Eq.(III.30), it is notable that Term_1, Term_2, and Term_5 are independent of the selected set $\mathcal{A}$ and hence they are evaluated once, for each unlabeled sample, during the first iteration of the algorithm and

then used throughout all iterations. Therefore, only Term_3 and Term_4 need to be reevaluated in-between iterations. However, both of them can be written in a recursive form, see Eq.(III.35), where they use values from previous iterations and the values' updates are efficient to calculate.

$$\text{For iteration } i+1, \quad \begin{aligned} \text{Term}_3\big|_{i+1} &= C^{*2} \sum_{j' \in \mathcal{A}_i} [\mathbf{K_{uu}}]_{m,j'} \\ &= \text{Term}_3\big|_i + C^{*2} [\mathbf{K_{uu}}]_{m,j'=\{\mathcal{A}_i \setminus \mathcal{A}_{i-1}\}} \\ \text{Term}_4\big|_{i+1} &= -C^{*2} id \\ &= \text{Term}_4\big|_i - C^{*2} d \end{aligned} \tag{III.35}$$

So far we have discussed the efficiency of evaluating the marginal benefit function involved in the SUBMOD-S^{3}VM algorithm. However, as we perform this evaluation for all possible unlabeled samples in each iteration, we end up with number of evaluations of order $O(r|\mathcal{U}|^2)$. The submodular property can be exploited to dramatically reduce the number of required evaluations by using *lazy evaluations*. Let $\delta_m(\mathcal{A})$ be the marginal benefit of the unlabeled samples $\mathbf{x}_m$ with respect to the selected set $\mathcal{A}$ as defined in Eq.(III.30). The key idea is that for a fixed m, the function $\delta_m(A)$ is monotonically non-increasing in $\mathcal{A}$: For any $\mathcal{A} \subseteq \mathcal{A}'$, $\delta_m(\mathcal{A}) \leq \delta_m(\mathcal{A}')$ holds true. Moreover, the greedy algorithm produces a monotonically increasing sequence of sets $\mathcal{A}_i \subseteq \mathcal{A}_{i+1}$ during the iterations. Therefore, for a fixed m, δ_m can never increase as the greedy iterations proceed. This idea is employed by evaluating δ_m for all $m \in \mathcal{U}$ in the first greedy iteration. For all next iterations we go through $m \in \mathcal{U} \setminus \mathcal{A}_i$ in a decreasing order of their δ_m value by using a sorted list data structure. In each iteration, the highest δ_m, located at the top of the list, is reevaluated and then reinserted in the proper spot with respect to the order of all δ_m. In many cases, the reevaluation does not change the value much and therefore it is quite often that the reevaluated δ_m keeps its position at the top of the list. When this occurs, it is not necessary to reevaluate any other smaller δ_m as we know they will not increase to be above the obtained top δ_m. We perform evaluations only for the highest δ_m, thus the name *lazy evaluations*.

Algorithm 4 presents the SUBMOD-S^{3}VM lazy evaluations greedy algorithm for general kernels. The first loop (Steps 3-9) evaluates the marginal benefit, Eq.(III.30), for

all unlabeled samples. All the obtained evaluations are then sorted and kept in a list (MrgB-nfList). The top of the list is then picked as the first unlabeled sample with positive label and the index for the head of the list is moved to the second item. Lazy evaluations occur in the while loop (Steps 13-24) where only the current head of the list is re-evaluated. If after re-evaluation the head of the list remains at the top, the while loop is broken and the current head of the list is chosen to be positively labeled. If after re-evaluation the head of the list is less than its successor value, then the current head is pushed down the list such that it is larger than the next item. The while loop then continues to re-evaluate only the top item until a new head is found. The process is repeated in the for loop (Steps 12-26) until all unlabeled samples that should positively labeled are picked.

Finally for the sake of easy reproducibility of this work we provide in Algorithm 5 the detailed lazy evaluations greedy algorithm for optimizing SUBMOD-S^{3}VM with linear kernel.

III.3 Automatic Estimation of Unlabeled Positive Samples Ratio r

One of the most important open problems in the semi-supervised learning (SSL) field is that many algorithms produce imbalanced solutions, sometimes called *degenerate solutions*, where almost all unlabeled samples are assigned to just one class. Looking at the continuous variable formulation of S^{3}VM that we introduced earlier,

$$\underset{\mathbf{p}'=[p_1,\ldots,p_{|\mathcal{U}|}]}{\arg\min} \quad \min_{\mathbf{w}} \mathcal{J}(\mathbf{w},\mathbf{p}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i\in\mathcal{L}}\zeta_i + C^*\sum_{j\in\mathcal{U}} p_j\ell_j^+ + C^*\sum_{j\in\mathcal{U}}(1-p_j)\ell_j^-$$

we see that a very small value of the objective function is achievable by assigning all $p_j = 1$ (or all $p_j = 0$). In this case, third and forth terms of the objective function will be zero, either because of the values of p_j or the zero loss terms associated. The only loss endured in this case, is due to the misclassification of the labeled samples in the second term.

To overcome this problem, a wide range of semi-supervised algorithms put a restriction on the number of samples assigned to each class. In particular, it is a common practice to impose a constraint on the output of SSL problems such that a certain ratio of the unlabeled samples are to be assigned to the positive class, namely the *unlabeled positive ratio* r. What makes this issue a pressing problem is that all algorithms assign a value to r before

Algorithm 4: Lazy Evaluations Greedy Algorithm to Optimize SUBMOD-S^{3}VM with general kernels [28].

Input : Set of Labeled Samples $\{(\mathbf{x}_i, y_i)\}, i \in \mathcal{L}$.
Set of Unlabeled Samples $\{\mathbf{x}_j\}, j \in \mathcal{U}$.
Ratio of Positive Labels in the set of Unlabeled Samples; $r = \frac{|\{y_j=+1, j\in\mathcal{U}\}|}{|\mathcal{U}|}$.

Output: Positive Labeled Samples in $\mathcal{U}$; $\mathcal{A}^* = \{j \in \mathcal{U} : y_j = +1\}$.
Negative Labeled Samples in $\mathcal{U}$; $\mathcal{U}\backslash\mathcal{A}^* = \{j \in \mathcal{U} : y_j = -1\}$.

begin
  Set $\mathcal{A}_0 := \phi$
  for $m := 1$ **to** $|\mathcal{U}|$ **do**
    New MrgBnfObj
    MrgBnfObj.Index $:=$ m
    MrgBnfObj. δ $:= \mathcal{S}(\mathcal{A}_0 \cup m) - \mathcal{S}(\mathcal{A}_0)$
    MrgBnfObj.UpdateIteration $:=$ 1
    MrgBnfList.PushBack $\leftarrow$ MrgBnfObj
  end
  MrgBnfList $\leftarrow$ MrgBnfList.Sort

  $\mathcal{A}_1$ $:=$ $\mathcal{A}_0 \cup$ MrgBnfList [1].Index
  for $l := 2$ **to** $r|\mathcal{U}|$ **do**
    while *True* **do**
      if *(*MrgBnfList *[l].UpdateIteration=l)* **then**
        Break while loop.
      end
      MrgBnfList [l]. δ $:= \mathcal{S}(\mathcal{A}_{l-1} \cup m) - \mathcal{S}(\mathcal{A}_{l-1})$
      MrgBnfList [l].UpdateIteration $:=$ l
      if *(*MrgBnfList *[l].*$\delta \geq$ MrgBnfList *[*$l+1$*].*δ*)* **then**
        Break while loop.
      else
        Push down MrgBnfList [l] to proper position to keep list sorted.
      end
    end
    $\mathcal{A}_l$ $:=$ $\mathcal{A}_{l-1} \cup$ MrgBnfList [l].Index
  end
  $\mathcal{A}^*$ $:=$ $\mathcal{A}_{r|\mathcal{U}|}$
end

Algorithm 5: Greedy Algorithm with Lazy Evaluations to Optimize SUBMOD-S^{3}VM with linear kernel [28].

Input : Set of Labeled Samples $\{(\mathbf{x}_i, y_i)\}, i \in \mathcal{L}$.
Set of Unlabeled Samples $\{\mathbf{x}_j\}, j \in \mathcal{U}$.
Ratio of Positive Labels in the set of Unlabeled Samples; $r = \frac{|\{y_j=+1, j\in\mathcal{U}\}|}{|\mathcal{U}|}$.

Output: Positive Labeled Samples in $\mathcal{U}$; $\mathcal{A}^* = \{j \in \mathcal{U} : y_j = +1\}$.
Negative Labeled Samples in $\mathcal{U}$; $\mathcal{U} \backslash \mathcal{A}^* = \{j \in \mathcal{U} : y_j = -1\}$.

begin
  Set $\mathcal{A}_0 := \phi$
  US $:= \sum_{j\in\mathcal{U}} \mathbf{x}_j$ \\ Sum of Unlabeled Samples.
  SLS $:= \sum_{i\in\mathcal{L}} y_i \mathbf{x}_i$ \\ Sum of Signed Labeled Samples.
  for $j := 1$ **to** $|\mathcal{U}|$ **do** \\ Evaluate Marginal Benefit of All Unlabeled Samples w.r.t $\mathcal{A}_0$.
    New MrgBnfObj \\ Instantiate a Marginal Benefit Object.
    MrgBnfObj.Index $:= j$
    MrgBnfObj.SSim $:= \mathbf{x'}_j \cdot$ US \\ Sum of Similarities to All Unlabeled Samples.
    MrgBnfObj.UpdateIteration $:= 1$
    MrgBnfObj.Term$_1 := -0.5\, C^{*2}(\mathbf{x'}_j \cdot$ US$)$
    MrgBnfObj.Term$_2 := CC^*(\mathbf{x'}_j \cdot$ SLS$)$
    MrgBnfObj.Term$_3 := 0$ \\ Since Term$_3 = C^{*2} \sum_{j'\in\mathcal{A}} \mathbf{x}_j \cdot \mathbf{x}_{j'}$ and $\mathcal{A}_0 = \phi$.
    MrgBnfObj.Term$_4 := 0$ \\ Since Term$_4 = -C^{*2}|\mathcal{A}|d$, and $|\mathcal{A}_0| = 0$.
    MrgBnfObj.Term$_5 := 0.5\, C^{*2}(\mathbf{x'}_j \cdot \mathbf{x}_j - d)$
    MrgBnfObj. $\delta := \sum_{t=1}^{5}$ Term$_t$ \\ Marginal Benefit Value of Unlabeled Sample $\mathbf{x}_j$.
    MrgBnfList.PushBack $\leftarrow$ MrgBnfObj \\ Pushing Objects into a List.
  end
  MrgBnfList $\leftarrow$ MrgBnfList.Sort \\ Descending Sorting of List Objects Using Values.
  $\mathcal{A}_1 := \mathcal{A}_0 \cup$ MrgBnfList [1].Index
  SA $:= \mathbf{x}_j,\ j \in \mathcal{A}_1$ \\ Sum of Unlabeled Samples with Indeces in $\mathcal{A}$.
  for $l := 2$ **to** $r|\mathcal{U}|$ **do**
    while *True* **do**
      if *(*MrgBnfList *[l].UpdateIteration=l)* **then**
        Break while loop.
      end
      MrgBnfList [l].Term$_3 := C^{*2}$(SA$' \cdot \mathbf{x}_j$) , $j =$ MrgBnfList [l].Index
      MrgBnfList [l].Term$_4 := -C^{*2}|\mathcal{A}_{l-1}|d$
      MrgBnfList [l]. $\delta := \sum_{t=1}^{5}$ Term$_t$
      MrgBnfList [l].UpdateIteration $:= l$
      if *(*MrgBnfList *[l].*$\delta \geq$ MrgBnfList *[*$l+1$*].*δ*)* **then**
        Break while loop.
      else
        Push down MrgBnfList [l] to proper position to keep list sorted.
      end
    end
    $\mathcal{A}_l := \mathcal{A}_{l-1} \cup$ MrgBnfList [l].Index
    SA $:=$ SA $+\mathbf{x}_j$, $j =$ MrgBnfList [l].Index
  end
  $\mathcal{A}^* := \mathcal{A}_{r|\mathcal{U}|}$
end

starting to optimize for the solution. This ratio assignment is usually estimated using the labeled samples, which can be a misleading estimate especially as in SSL problems the labeled data set is usually too small to be informative.

The proposed SUBMOD-S^{3}VM algorithm provides an advantage over the other S^{3}VM techniques in the sense that the involved greedy procedure produces the classification of individual unlabeled samples sequentially. Therefore, after n iterations of the greedy procedure on a set $\mathcal{U}$ of unlabeled samples, we have a set $\mathcal{A}$ with $|\mathcal{A}| = m$ of *positively labeled samples* and the rest of the samples $\mathcal{U} \backslash \mathcal{A}$ are negatively labeled. This means that the greedy procedure imposes a structure over the space of all possible label assignments to $\mathcal{U}$, which is exponentially large with size $2^{|\mathcal{U}|}$, and reduce it to be of size $|\mathcal{U}|$. In other words, the SUBMOD-S^{3}VM algorithm produces $|\mathcal{U}|$ possible label assignments, and they are produced sequentially. This allows the opportunity to examine the various assignments and decide when to terminate the learning process. Figure III.7 shows a sample sequence of label assignments obtained during the iterations of the SUBMOD-S^{3}VM algorithm along with their corresponding SVM decision boundaries. The reason the SVM models are depicted is to show how some of the label assignments are better than the others in terms of data separability, and therefore generalization performance, which is the core idea behind SVM/S^{3}VM.

Figure III.8 presents the proposed approach for estimation of unlabeled positive samples ratio r. As discussed earlier the iterations of the SUBMOD-S^{3}VM algorithm produces a sequence of possible label assignment such that each two consecutive label assignments differ in only one sample label. We use the inverse width of the SVM margin, Eq.(III.36), as an estimate of the generalization performance of the SVM model associated with every label assignment as a measure of the best label assignment.

$$\begin{aligned} &\arg\min_{\mathbf{w},\xi_i} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \\ &\text{subject to} \qquad y_i[\langle \mathbf{w}, \mathbf{x}_i\rangle + b] \geq 1 - \xi_i \quad , \quad \xi_i \geq 0 \end{aligned} \tag{III.36}$$

The proposed approach, as seen in Fig.III.8, starts by training an SVM f_1 using the first set of label assignments produced by the SUBMOD-S^{3}VM algorithm. The inverse norm $1/\|\mathbf{w}\|^2$ is then used to estimate the generalization estimate for f_1. The process is

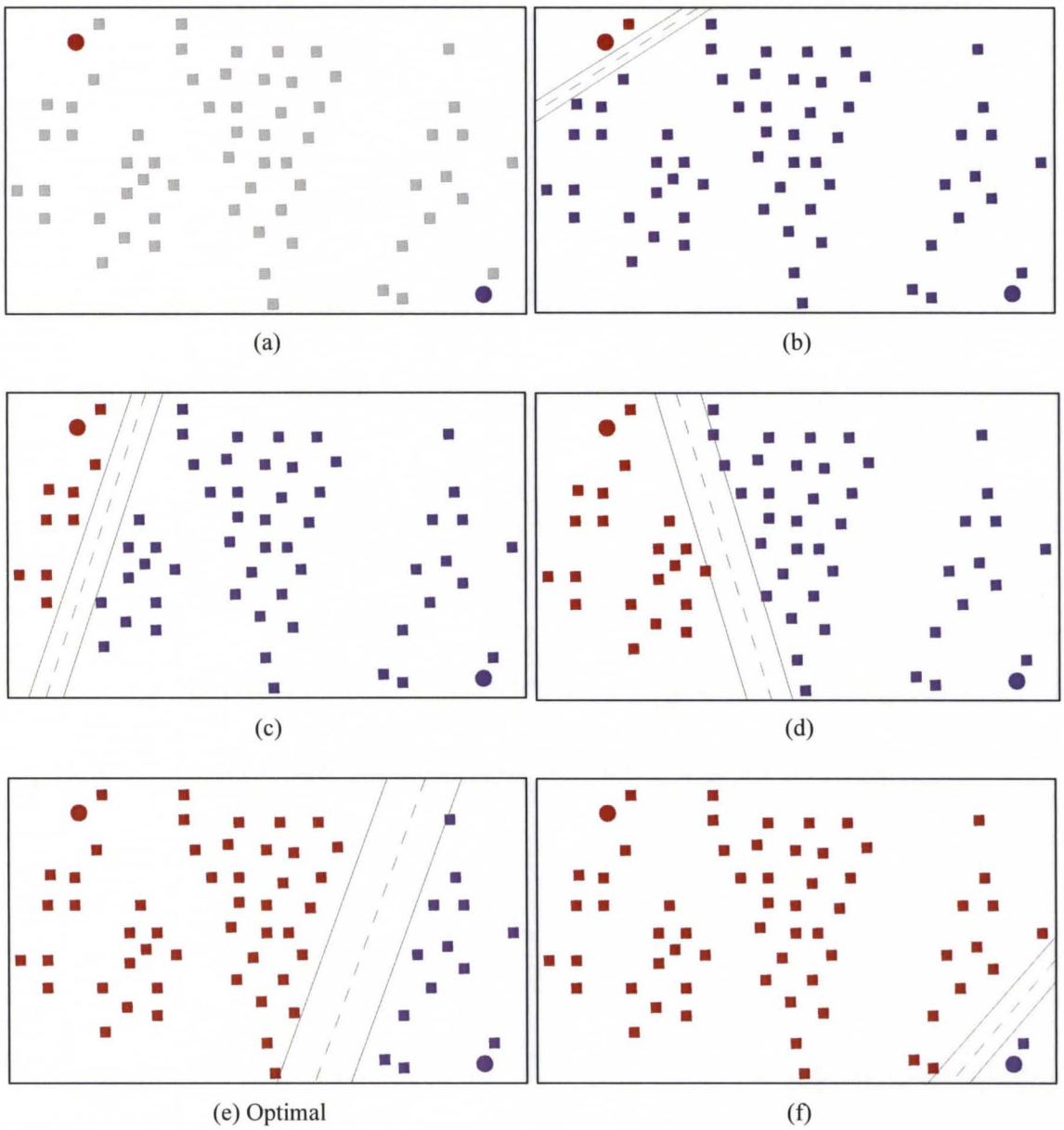


Figure III.7: Sample sequential label assignments obtained during the iterations of the SUBMOD-S^3VM algorithm. The lines depict the SVM model corresponding to the obtained label assignment. (a) Original SSL data set. (b) First iteration label assignments. (c,d) Intermediate iterations. (e) The optimal label assignment and the corresponding SVM model. (f) Last iteration, only one sample is assigned a negative label.

repeated for all iterations of the SUBMOD-S^{3}VM algorithm to get $f_2, \ldots, f_{|U|}$ and the corresponding margin width estimators. After the first iteration, SVM's do not have to be trained from scratch but rather incremental SVM is used to update previous ones. This is especially helpful given that each two consecutive SVM's have the same exact training data except for one sample with flipped label. We use *incremental-decremental SVM* introduced in [16] to perform incremental SVM training where exact closed form, and thus very efficient, formulation is utilized to account for adding or removing one sample from the training set of an SVM. Once all iterations are finished, the label assignment with the best margin is chosen as the output of the SUBMOD-S^{3}VM algorithm. However, if there exists prior knowledge about a proper threshold for the generalization performance, the iterations can be stopped when this threshold is reached. Figure III.9 shows a sample output of the proposed approach on the imbalanced (Unlabeled Positive Sample Ration $r = 65\%$) Breast-Cancer data set. We see that the margin width has a clear minimal value which corresponds to the proper inherent value for r. To validate the obtained ratio, we also depict the accuracy as well as the F-Score of the label assignments to show that they achieve highest values at the chosen positive ratio r.

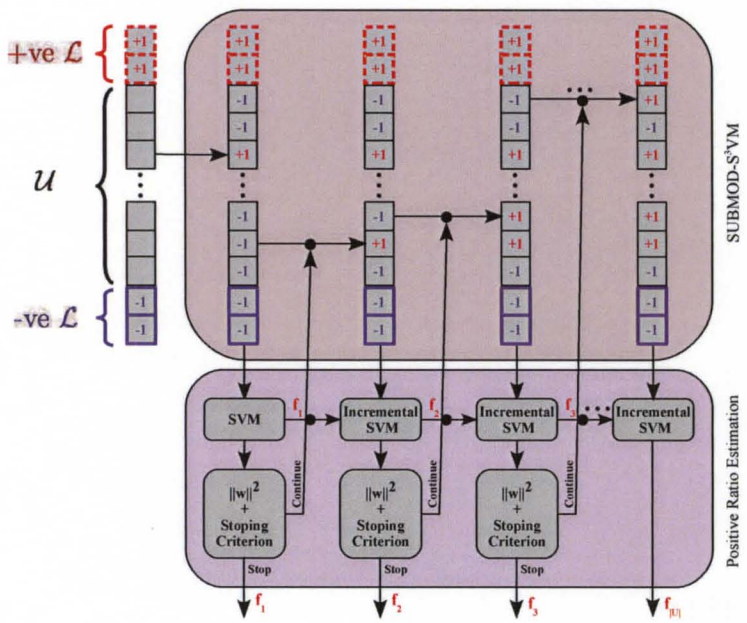


Figure III.8: Proposed approach for automatic estimation of unlabeled positive samples ratio r.

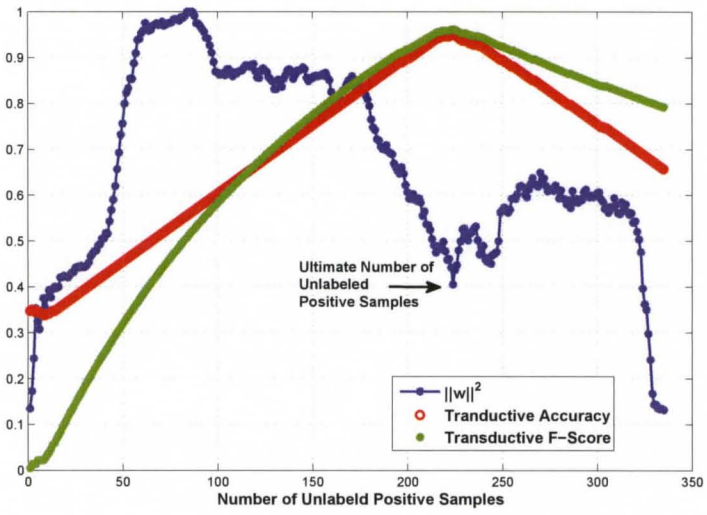


Figure III.9: Sample result of the proposed approach for automatic estimation of unlabeled positive samples ratio r on the Breast-Cancer data set. The global internal minimum of the margin norm ($1/\|\mathbf{w}\|^2$ is the margin width) corresponds to the correct value for the number of positive samples.

III.4 Experimental Results

In this section we illustrate the superior performance, in terms of accuracy and time efficiency, of the proposed QP-S^{3}VM and its submodular formulation SUBMOD-S^{3}VM. We compare the performance of QP-S^{3}VM and SUBMOD-S^{3}VM with state of the art S^{3}VM algorithms, namely *Transductive Support Vector Machine* (TSVM) [41], *Deterministic Annealing for Semi-supervised Kernel Machines* (DA) [62], and *Gradient Transductive Support Vector Machine* (∇TSVM) [18]. All experiments are performed on a 2.7 GHZ Intel Core2 Duo machine with 8 GB RAM. Moreover, we provide experiments to examine and illustrate the validity and effectiveness of the proposed contributions in this chapter.

III.4.1 Experiments Description

Figure III.10 provides an outline of the work presented in this chapter and the corresponding experiments. As shown in the figure, our first contribution is the *quadratic*

programming surrogate objective function QP-S^{3}VM. The verification experiments for this model are presented in Sec.III.1.3. For scalability purposes we transformed the QP-S^{3}VM into a submodular optimization problem SUBMOD-S^{3}VM and used an efficient greedy approach for optimization. Therefore, in Sec.III.4.3.2 both QP-S^{3}VM and SUBMOD-S^{3}VM are compared against the state of the art S^{3}VM techniques stated above in terms of their *transductive accuracy* (accuracy of labels assigned to unlabeled samples). Since some of the used benchmark data sets are imbalanced, we also use the sensitivity, specificity, and Matthews Correlation Coefficient (MCC) for comparison:

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

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP}+\text{FN}}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN}+\text{FP}}$$

$$\text{Matthews Correlation Coefficient (MCC)} = \frac{\text{TP}\cdot\text{TN}-\text{FP}\cdot\text{FN}}{\sqrt{(\text{TP}+\text{FP})(\text{TP}+\text{FN})(\text{TN}+\text{FP})(\text{TN}+\text{FN})}}$$

where TP and FN refer to the *true positive* and *false negative* counts of the positive-labeled unlabeled samples, respectively. Similarly, TN and FP are defined with respect to negative-labeled unlabeled samples. We use the *Matthews Correlation Coefficient (MCC)* rather than the F-Score due to the dependence of the F-Score on the positive class where the TN is not considered. Therefore, the F-Score is not symmetric with respect to the labels of the classes while the MCC is. Therefore, MCC is preferred in our experiments as the cost of erroneous assignments to both classes is equal.

The proposed SUBMOD-S^{3}VM is a discrete form of QP-S^{3}VM with an added constant term to achieve submodularity and monotonicity. This means that the optimal solution of both problems is the same. However, we have shown in Sec.III.2.2.3 that the added constant affects the approximation lower bound achievable by the greedy approach to optimize SUBMOD-S^{3}VM. In Sec.III.4.3.1 we perform experiments to examine the empirical optimization approximation achieved by the greedy algorithm when applied on SUBMOD-S^{3}VM. The last set of experiments, Sec.III.4.5, examines the time efficiency and scalability of the SUBMOD-S^{3}VM given that the submodular property allows the greedy approach to perform lazy evaluations which significantly reduces its computational complexity.

The experiments are performed on several real world and simulated data sets with

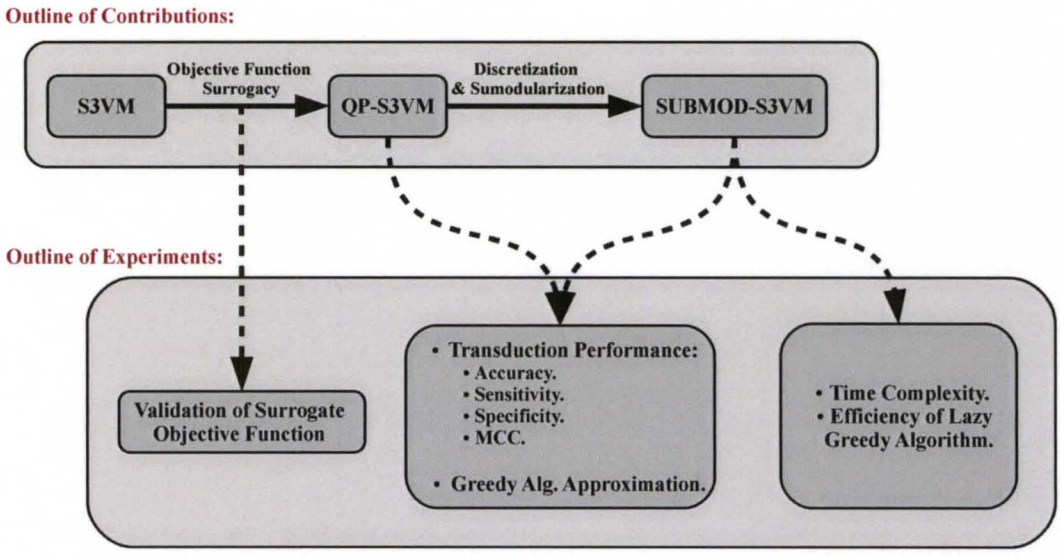


Figure III.10: Outline of the proposed contributions in this chapter and the corresponding experimental outline.

a wide spectrum of dimensions and size. Not all S^3VM techniques in the literature are scalable, therefore we divide the experiments into small and medium/Large scale experiments. In the small scale experiments, Sec.III.4.3, all the techniques are used for comparison. Whereas in medium/large scale experiments, Sec.III.4.4, we compare the proposed SUBMOD-S^3VM only to the DA as it is the most scalable techniques in the literature.

III.4.2 Experimental Setup

All experiments are performed on pessimistic 10 labeled/unlabeled splits out of 100 random splits of the data sets and the average is reported. SVM's are known for their very good generalization performance using only few data samples for learning. Therefore, it is not uncommon to obtain very few labeled samples and they produce highly good labeling of the unlabeled samples in SSL. In other words, the unlabeled samples in this case do not carry extra knowledge. Figure III.11 illustrates this idea, where the decision boundary obtained in Fig.III.11b is almost the same as that in Fig.III.11a. This shows that the few chosen labeled samples are quite powerful and that the unlabeled in this example are not of much importance. To overcome the issue of sampling useless unlabeled samples, we

choose our labeled/unlabeled samples to be challenging for the standard SVM. We acquire 100 random labeled/unlabeled splits of any data set and use the 10 splits that achieve the worst classification performance with a standard supervised SVM. This way we guarantee that the SSL decision boundary is very different from that of the supervised SVM, see Fig.III.11c. This will be clear when examining Table III.2 and Table III.7, where the supervised SVM have hard time achieving a good accuracy. One of the state of the art S^3VM techniques is the Deterministic Annealing (DA) S^3VM. DA uses linear kernels. Therefore, we chose to perform all the experiments using the linear kernel. Moreover, using the linear kernel is one of the key aspects of applying SVM based techniques to very large data sets. Finally, the ratio of positive samples in the output r is set to the correct ratio in the unlabeled samples.

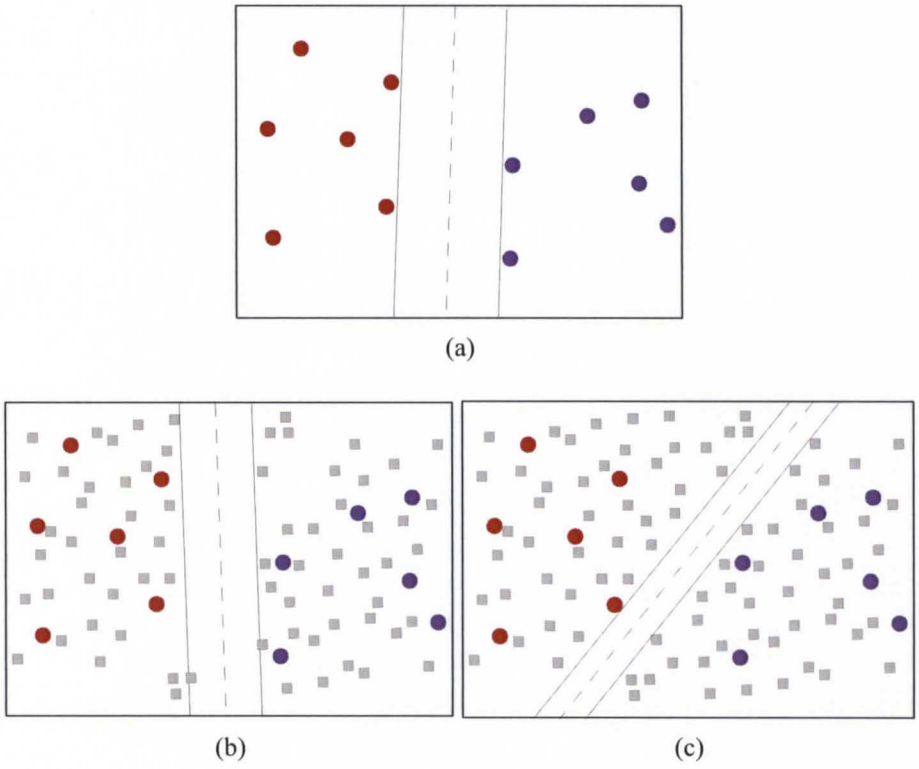


Figure III.11: (a) Decision function obtained by supervised SVM on few labeled samples. (b) Labeled (circles) and Unlabeled (squares) samples where a supervised SVM on the labeled samples is good enough to effectively label all the unlabeled samples. Decision boundary similar to (a). (c) Unlabeled samples distribution is challenging for decision boundary of the supervised SVM. Decision boundary is very different from (a).

III.4.3 Small Scale Data Sets Experiments

Table III.1 presents the small scale data sets used for comprehensive comparison of the proposed algorithms to the literature S^3VM techniques. The data sets have been chosen to cover a wide spectrum of data characteristics and applications. Most of the data sets come from real applications, except for G50c, Four-Class, and Digit1 which are synthetic. G50c consists of samples from two Gaussian classes in a 50-dimensional spaces. The Four-Class data set is a challenging 2-dimensional data set used to test non-linear separation. Digit1 is an image data set with different variations of the number one. It is designed to include samples close to a low-dimensional manifold that is embedded into a high-dimensional space. However, the samples do not show a pronounced cluster structure.

The Breast-Cancer and Diabetes data sets are real world data for predicting breast cancer malignancy and diabetes in females of Pima Indian heritage, respectively. From the financial applications, we used the Australian and German-Numer data sets. Both are used for credit approval. We also experimented with data sets from physics based applications. Svmguide1 is a data set for classifying astroparticles, Sonar is a data set for classifying rocks versus mine like objects, and the Ionosphere data set is used for detection of structures in the ionosphere. The USPS data set [20] is derived from the famous USPS data set of handwritten digits. The positive class consists of the digits "2" and "5", and the rest of the digits constitute the negatives class.

Finally, from text classification applications we used the News20-Binary and the Text data sets for Newsgroup classification, and the W6a data sets for web page classification. News20-Binary is a size-balanced two-class variant of the UCI "20 Newsgroups" generated in [43]. The positive class consisting of 10 groups with names of the form sci.*, comp.*, or misc.forsale. The negative class includes the other 10 groups. The Text data set [20] consists of the 5 comp.* groups from the Newsgroups data set and the goal is to classify the *ibm* category versus the rest.

TABLE III.1

Small scale data sets used in the experiments [6, 38, 20].

Data set	Samples	Features	Labeled	C	C^*/C	r
Sonar	208	60	2	1	1	0.47
Ionosphere	350	34	2	1	1	0.35
G50C	550	50	5	10^{-1}	10^{-2}	0.5
Breast-Cancer	675	10	4	1	10^{-2}	0.65
Australian	690	14	4	1	10^{-1}	0.44
Diabetes	768	8	8	1	10^{-2}	0.65
Four-Class	862	2	11	1	10^{-2}	0.36
News20-Binary-Small	997	1,355,191	15	1	10^{-4}	0.5
German-Numer	1000	24	3	1	1	0.3
W6A	1488	300	15	1	10^{-2}	0.44
Text	1494	11,960	30	1	10^{-3}	0.5
USPS	1500	241	4	1	10^{-3}	0.2
Digit1	1500	241	8	1	10^{-1}	0.49
Svmguide1	3025	4	16	10^{-2}	10^{-6}	0.66

III.4.3.1 Greedy Approximation Experiments

Section III.2.2.2 presented the design process of the term $\mathcal{Q}_5$ that achieves the monotonicity and submodularity of the SUBMOD-S^{3}VM objective function. Furthermore, in Sec.III.2.2.3 we discussed the implications of using the term $\mathcal{Q}_5$, Eq.(III.29), on the approximation achieved by the greedy algorithm. That is, while using the greedy algorithm has an approximation low bound of 63%, adding $\mathcal{Q}_5$ will further reduce the approximation lower bound.

In this section the approximation percentage $\mathcal{D}(\mathcal{A}_{Greedy})/\mathcal{D}(\mathcal{A}^*)$ achieved by the greedy algorithm is calculated for all the data sets in Table III.1. Figure III.12 shows the obtained approximation percentages. It is clear that despite adding the constant $\mathcal{Q}_5$, the greedy algorithm still reaches a very good approximation of the optimal maximum value. Most of the shown data sets achieves more than 98% of the optimal value and the lowest reached approximation is 87.5% for the W6a data set.

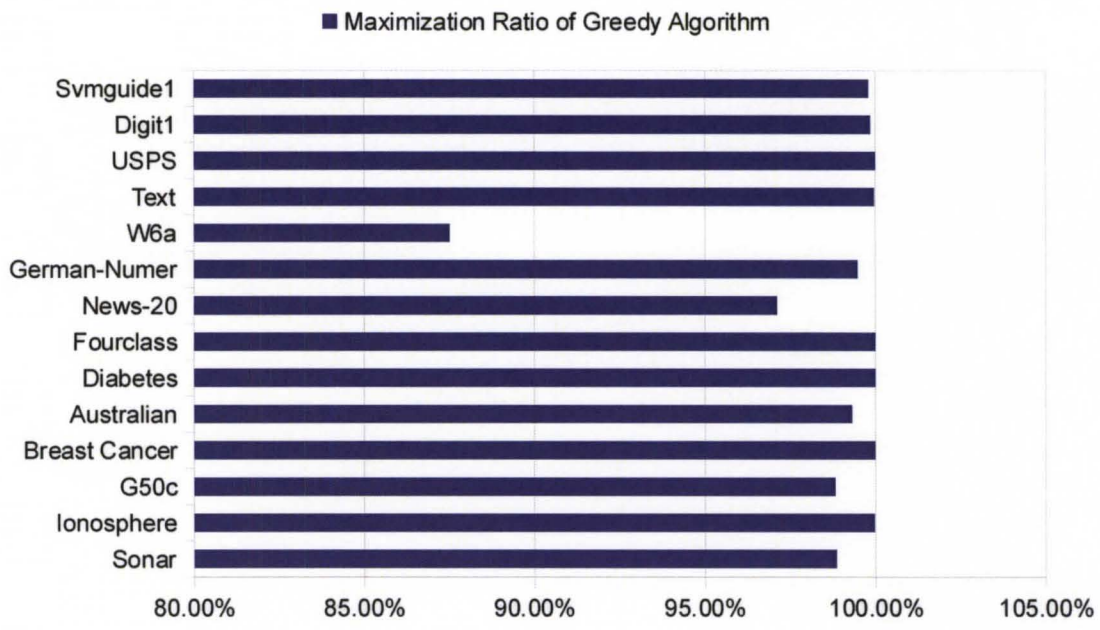


Figure III.12: Approximation achieved by the greedy approach.

III.4.3.2 Small Data Set Accuracy and Efficiency Experiments

Table III.2 provides the outcome of the *transductive accuracy* experiments. The bold and the underlined numbers indicate the best and the second best transductive accuracy, respectively, amongst all the tested techniques. It is clear that in almost all the tested data sets, the proposed QP-S^3VM and SUBMOD-S^3VM techniques achieve the best or the second best transductive accuracy. This shows the stability and the consistent performance of the proposed methods. For statistically robust conclusions, we have performed a series of paired-sample hypothesis tests, at the 5% significance level, to examine the transductive accuracy of SUBMOD-S^3VM versus the state of the art, using both Table III.2 and Table III.7. The output p-values of the hypothesis tests are shown in Table III.3. The p-values are small enough so we can reject the null hypothesis that the SUBMOD-S^3VM have the same performance as the state of art S^3VM algorithms.

Table III.4 provides further comparison measures, namely *sensitivity*, *specificity*, and *MCC*, for the small imbalanced data sets, where we see the proposed techniques are very comparable to the literature.

TABLE III.2

Transductive accuracy for small scale data sets. All algorithms are tested on the unlabeled samples. SVM is trained only using the labeled samples while the semi-supervised techniques are trained using both labeled and unlabeled samples.

Data set	SVM	TSVM	DA	∇TSVM	QP-S^{3}VM	SUBMOD-S^{3}VM
Sonar	45.15	54.45	50.24	45.83	55.53	**57.09**
Ionosphere	34.22	51.15	**76.44**	45.92	65.23	65.17
G50C	49.54	**94.70**	94.26	49.54	94.64	93.83
Breast-Cancer	58.78	96.66	93.28	76.54	**96.69**	96.63
Australian	40.26	64.46	63.13	63.43	**82.83**	79.30
Diabetes	46.16	66.14	**67.74**	67.16	61.13	61.11
Fourclass	55.24	60.71	63.34	**64.49**	63.41	63.43
News-20-Small	49.12	67.67	65.60	N/A	57.23	**67.78**
German-Numer	33.01	60.12	61.24	61.81	**63.75**	63.43
W6A	45.09	53.90	60.65	44.82	**64.13**	62.30
Text	56.67	73.71	**75.96**	N/A	74.25	74.04
USPS	50.67	70.95	**74.88**	62.67	71.40	71.42
Digit1	50.56	83.90	84.95	49.56	66.55	**85.92**
Svmguide1	65.11	**94.88**	81.77	84.05	92.38	92.55

TABLE III.3

P-values of paired hypothesis tests examining transductive accuracy of SUBMOD-S^{3}VM.

	TSVM	DA	∇TSVM
p-value	0.0427	0.0307	0.0025

TABLE III.4

Transductive accuracy evaluation for imbalanced small scale data sets.

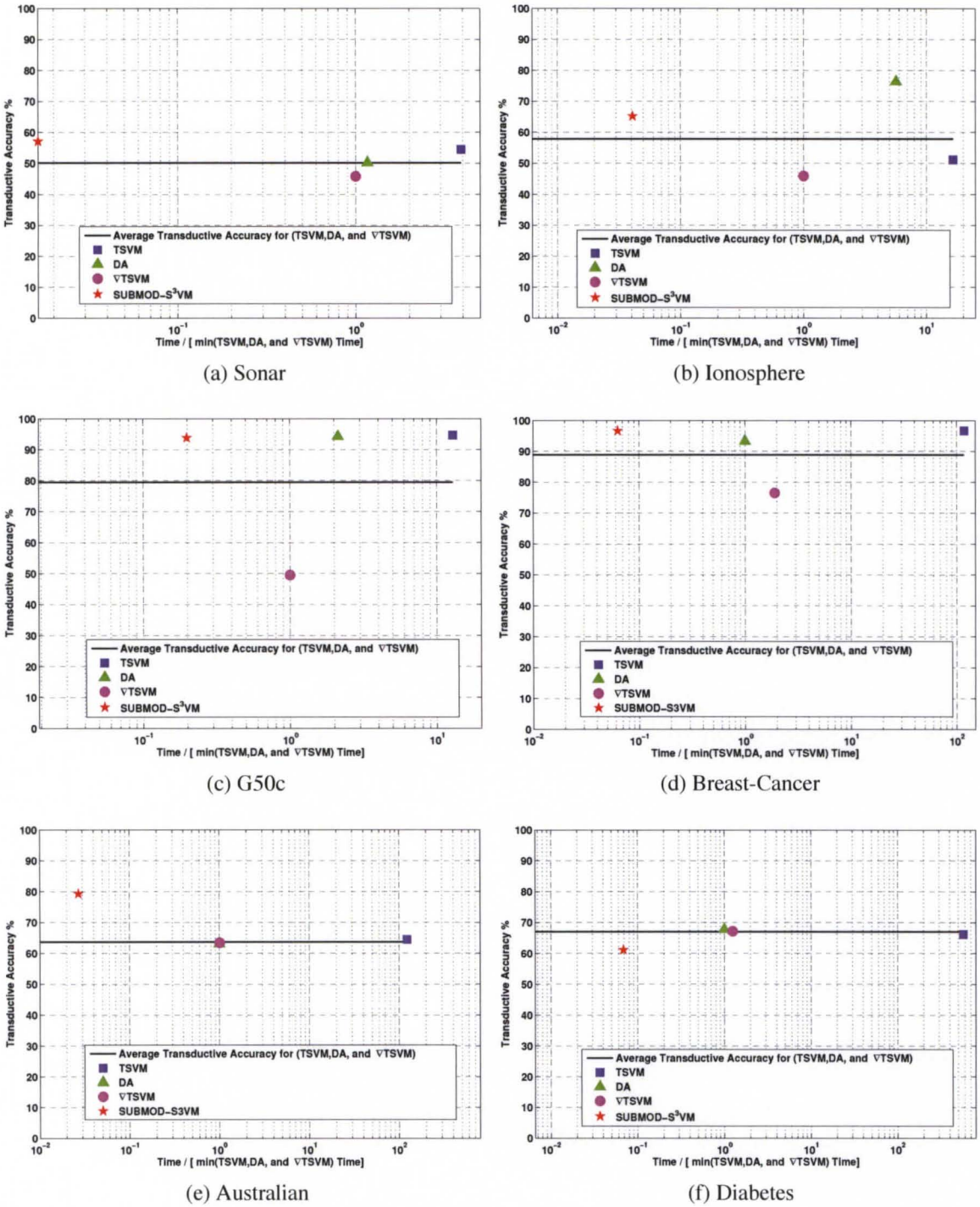
Data set		TSVM	DA	∇TSVM	QP-S^{3}VM	SUBMOD-S^{3}VM
Ionosphere	Sens.	31.73	66.29	84.52	50.97	50.89
	Spec.	61.81	82.01	24.82	73.06	73.02
	MCC	-0.06	**0.48**	N/A	0.24	0.24
Breast-Cancer	Sens.	97.41	98.60	69.01	97.45	97.41
	Spec.	95.28	83.45	90.26	95.28	95.19
	MCC	**0.93**	0.85	0.61	**0.93**	**0.93**
Diabetes	Sens.	72.88	81.19	79.64	70.28	70.26
	Spec.	53.42	42.35	43.53	43.83	43.80
	MCC	**0.26**	0.25	0.26	0.14	0.14
Four-Class	Sens.	-	49.96	52.36	48.58	48.61
	Spec.	-	70.72	71.24	71.60	71.62
	MCC	-	0.21	**0.24**	0.20	0.20
German-Numer	Sens.	33.75	33.83	21.46	39.30	38.76
	Spec.	71.34	72.92	78.92	74.16	73.93
	MCC	0.05	0.07	0.09	**0.13**	**0.13**
USPS	Sens.	27.08	22.58	76.21	28.22	28.26
	Spec.	81.86	87.89	59.31	82.15	82.15
	MCC	0.09	0.14	**0.28**	0.10	0.10
Svmguide1	Sens.	95.78	86.73	94.49	94.24	94.37
	Spec.	93.12	72.08	63.72	88.75	89.00
	MCC	**0.89**	0.59	0.66	0.83	0.83

The outcomes of the time efficiency comparison are listed in Table III.5. We can see that the proposed SUBMOD-S^{3}VM achieves a speed-up of $1.5X - 60X$ over the most efficient literature algorithm for each data set while maintaining a statistically significant better performance as evidenced in Table III.3. In Fig.III.13 we provide a visualization that summarizes the transductive performance and time efficiency of the proposed techniques. For each data set, the average transductive accuracy of the literature S^{3}VM techniques (TSVM, DA, and ∇TSVM) is plotted as a solid black line. Moreover, for each data set, the time access is scaled with respect to the most time efficient algorithm in the literature, i.e. min[time(TSVM), time(DA), time(∇TSVM)]. This scaling makes the time reading corresponding to each algorithm represents the time efficiency compared to the best algorithm. Examining Fig.III.13 we see that in all data sets the proposed SUBMOD-S^{3}VM performs better or close to the average transductive accuracy of the state of the art S^{3}VM algorithms while achieving a much better time efficiency.

TABLE III.5

CPU time (Seconds) experiments for the small scale data sets.

Data set	TSVM	DA	∇TSVM	QP-S^{3}VM	SUBMOD-S^{3}VM
Sonar	1.412	0.421	0.361	1.705	0.006
Ionosphere	2.521	0.863	0.153	5.853	0.006
G50C	1.254	0.207	0.097	27.38	0.019
Breast Cancer	7.957	0.068	0.129	40.73	0.004
Australian	36.03	0.294	0.297	105.5	0.008
Diabetes	55.18	0.096	0.120	76.31	0.007
Fourclass	54.48	0.127	0.108	115.4	0.005
News-20-Small	369.0	21.87	N/A	195.0	0.583
German-Numer	4.402	0.849	0.384	211.2	0.012
W6A	28.31	0.960	0.536	767.2	0.163
Text	46.98	0.628	N/A	756.2	0.130
USPS	24.70	2.708	1.143	734.8	0.016
Digit1	23.33	2.472	0.137	782.4	0.094
Svmguide1	34.43	0.974	0.214	2823	0.030

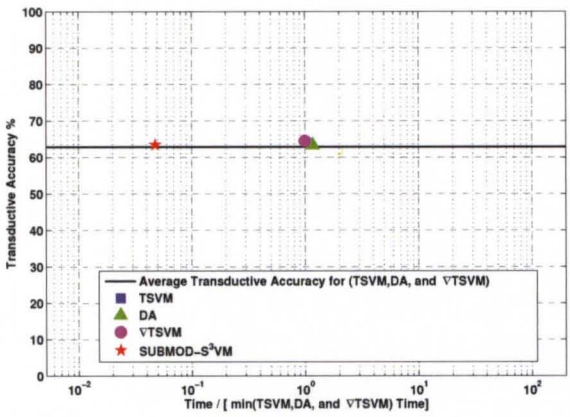


(a) Sonar (b) Ionosphere

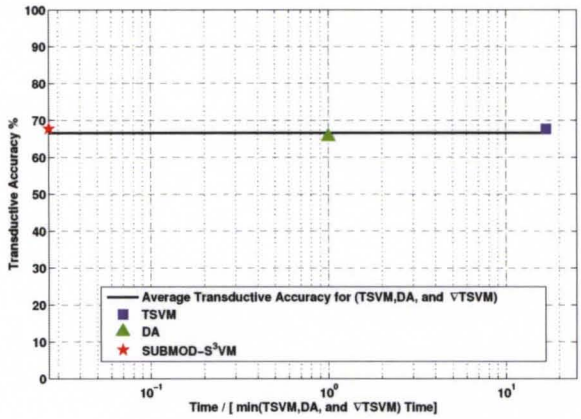
(c) G50c (d) Breast-Cancer

(e) Australian (f) Diabetes

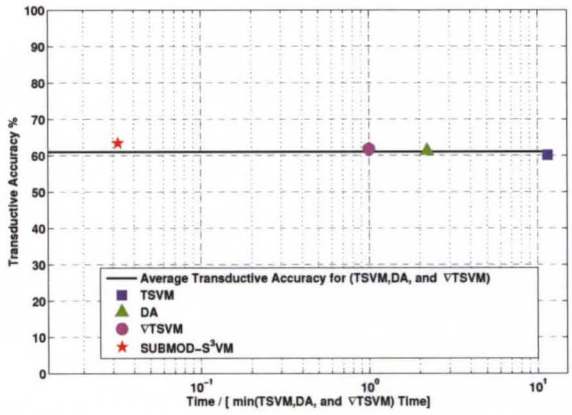
Figure III.13: Summary visualization of the transductive accuracy and time efficiency of the proposed SUBMOD-S^3VM versus the state of the art S^3VM techniques.



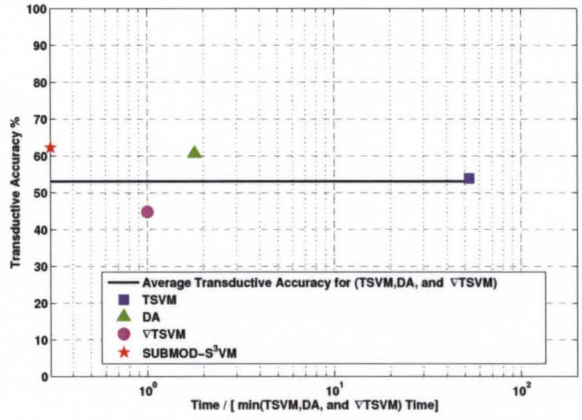
(g) Four-Class



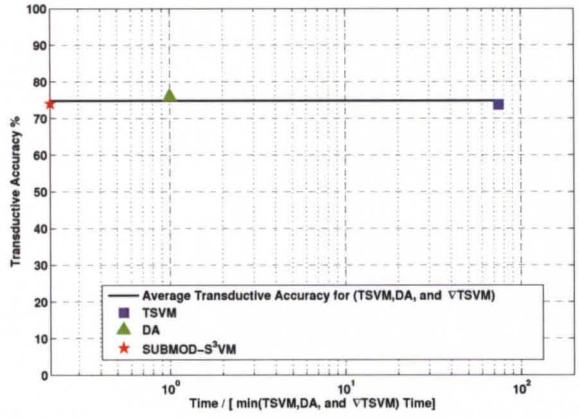
(h) News-20-Small



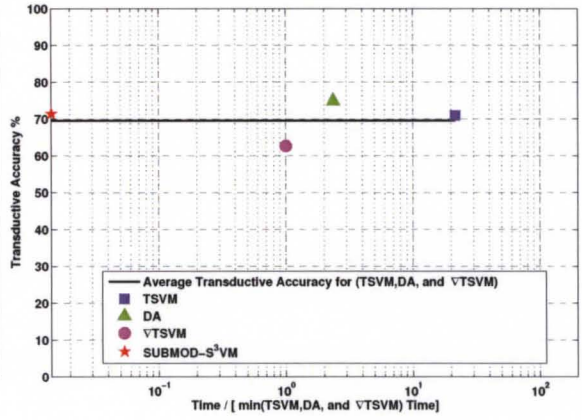
(i) German-Numer



(j) W6A

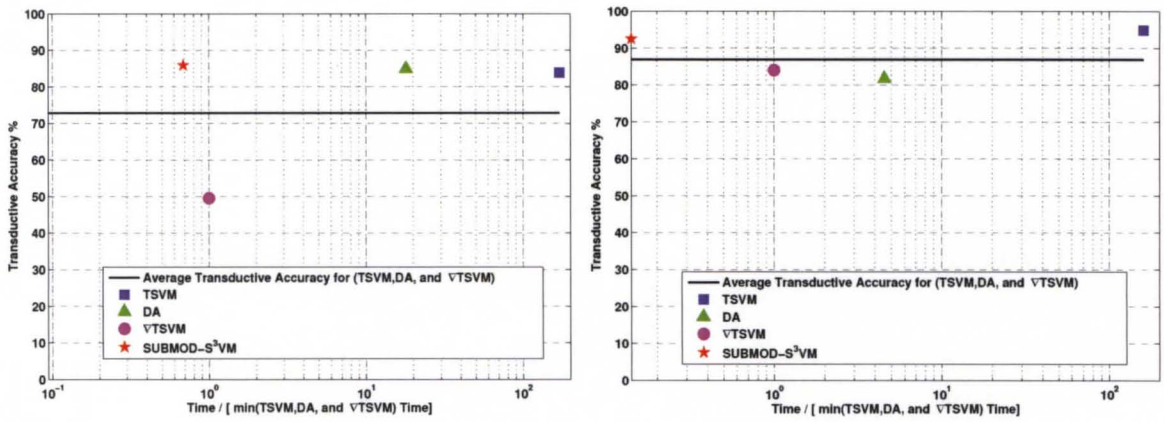


(k) Text



(l) USPS

Figure III.13: Continued: Summary visualization of the transductive accuracy and time efficiency of the proposed SUBMOD-S^3VM versus the state of the art S^3VM techniques.



(m) Digit1 (n) Svmguide1

Figure III.13: Continued: Summary visualization of the transductive accuracy and time efficiency of the proposed SUBMOD-S^3VM versus the state of the art S^3VM techniques.

III.4.4 Medium/Large Scale Data Sets Experiments

In this section we provide the transductive accuracy and time efficiency experiments for medium and large scale data sets with sizes ranging from few tens of thousands to more than a million samples as listed in Table III.6. For the current experiments we focused more on using with high dimensionality. The W8a and the New20.Binary are just larger versions of the data sets used in Sec.III.4.3. The RCV1.Binary is extracted from the Reuters Corpus Volume 1 data set, where CCAT(Corporate/Industrial) and ECAT(Economics) constitute the positive class and GCAT(Government/Social) and MCAT(Markets) constitute the negative class. We also considered classifying the CCAT against all other topics. The A9A is a Census data set that is used to redact if an adult earns more than \$50K a year. Aut-Avn and Real-Sim data sets are articles discussion groups for simulated auto racing, simulated aviation, real autos, real aviation. The Aut-Avn data set is used to classify between the aviation and auto articles, while the Real-Sim is used for classifying real versus simulation articles. Cod-Rna is used for the detection of non-coding Rna in bioinformatics. Finally, the KDD-99 data set comes from computer network analysis where network intrusion needs to be detected. The used KDD-99 data set is a close to balanced version of the full (training/testing) original data set.

III.4.4.1 Medium/Large Data Set Accuracy and Efficiency Experiments

We consider only the DA algorithm for comparison as it is the only algorithm that can scale to large data sets and it has shown consistent good performance in the small scale data sets. Table III.7 and Table III.8 provide the output of the transductive accuracy experiments. It is clear that SUBMOD-S^{3}VM again shows statistically significant better better performance while achieving a $8X - 300X$ speed up as clear from Table III.9.

TABLE III.6

Medium and large scale data sets used in the experiments [6, 38].

Data set	Samples	Features	Labeled	C	C^*/C	r
News20.Binary	19,954	1,355,191	50	1	10^{-3}	0.50
CCAT	23,149	47,236	30	1	10^{-6}	0.47
GCAT	23,149	47,236	30	1	10^{-6}	0.40
A9A	35,276	122	9	1	10^{-3}	0.23
W8A	59,245	300	15	1	10^{-6}	0.02
Aut-Avn	70,166	20,702	9	1	10^{-6}	0.65
Real-Sim	72,201	20,958	8	1	10^{-6}	0.31
Cod-Rna	488,516	8	98	1	10^{-6}	0.33
Cov-Type	581,012	54	24	0.1	10^{-6}	0.49
RCV1.Binary	697,641	47,236	35	1	10^{-8}	0.52
KDD-99	1,500,000	122	9	1	10^{-5}	0.47

TABLE III.7

Transductive accuracy for medium and large scale data sets.

Data set	SVM	DA	SUBMOD-S^{3}VM
News20-Binary	50.74	69.51	**71.10**
CCAT	51.13	62.44	**63.32**
GCAT	57.63	65.81	**79.29**
A9A	56.04	**66.55**	65.11
W8A	80.74	**97.08**	95.00
Aut-Avn	62.08	64.78	**70.12**
Real-Sim	54.09	61.64	**70.83**
Cod-Rna	33.33	56.67	**70.08**
Cov-Type	48.52	50.13	**55.09**
RCV1.Binary	69.91	75.60	**76.16**
KDD-99	71.95	**98.62**	95.38

TABLE III.8

Transductive accuracy evaluation for imbalanced medium and large scale data sets.

Data set		DA	SUBMOD-S^3VM
News20-Binary	Sens.	65.89	71.06
	Spec.	73.12	71.15
	MCC	0.391	0.422
CCAT	Sens.	53.60	61.12
	Spec.	70.33	65.29
	MCC	0.24	0.264
GCAT	Sens.	10.64	66.06
	Spec.	90.04	85.09
	MCC	N/A	0.530
A9A	Sens.	23.44	24.76
	Spec.	79.57	77.29
	MCC	0.028	0.021
W8A	Sens.	0.243	14.33
	Spec.	99.99	97.42
	MCC	N/A	0.118
Aut-Avn	Sens.	100.0	76.94
	Spec.	0.000	57.58
	MCC	N/A	0.345
Real-Sim	Sens.	20.03	52.58
	Spec.	80.13	78.93
	MCC	N/A	0.315
Cod-Rna	Sens.	27.32	55.12
	Spec.	71.35	77.56
	MCC	-0.014	0.327
Cov-Type	Sens.	86.85	53.95
	Spec.	15.19	56.18
	MCC	N/A	0.101
RCV1.Binary	Sens.	76.75	77.27
	Spec.	74.34	74.93
	MCC	0.51	0.522
KDD-99	Sens.	99.08	95.13
	Spec.	98.20	95.60
	MCC	0.97	0.907

TABLE III.9

CPU time (Seconds) experiments.

Data set	DA	SUBMOD-S^3VM
News20-Binary	175.2	0.497
CCAT	12.59	1.150
GCAT	5.891	1.09
A9A	11.61	0.166
W8A	6.228	1.573
Aut-Avn	24.34	0.347
Real-Sim	12.35	1.457
Cod-Rna	119.7	9.302
Cov-Type	59.38	14.44
RCV1.Binary	2,190	23.22
KDD-99	425.45	15.72

III.4.5 Time Complexity Experiments

In Sec.III.2.2.5 we presented the implementation details of the SUBMOD-S^3VM algorithm and we discussed the idea of implementing the greedy algorithm using lazy evaluations which is made possible by the submodularity property of the objective function. In this section we provide experiments comparing the number of function evaluations saved through using lazy greedy evaluations. Table III.10 and Fig.III.14 provide the number of function evaluations used by the lazy greedy approach compared to the total number of evaluations required for the standard greedy approach and they show that reductions in the range 71-99% are achieved. Table III.11 presents the output of the same experiment on the medium and large scale data sets where the reductions are above 99% for all large scale data sets. It is noticeable by comparing TableIII.10 and Table III.10 that the reduction in the number of the function evaluations achieved gets better with larger data sets.

TABLE III.10

Number of Evaluations for Lazy Greedy and Standard Greedy Approaches for Small Scale Data Set.

Data set	Lazy Evaluations	Complete Greedy Evaluations	Reduction %
Sonar	2.74E+03	1.53E+04	82.04%
Ionosphere	3.88E+03	3.56E+04	89.11%
G50C	2.11E+04	1.12E+05	81.16%
Breast Cancer	2.03E+03	1.98E+05	98.97%
Australian	6.40E+03	1.63E+05	96.07%
Diabetes	5.30E+03	2.55E+05	97.92%
Fourclass	4.34E+03	2.13E+05	97.96%
News-20-Small	1.11E+05	3.59E+05	69.01%
German-Numer	7.92E+03	2.53E+05	96.87%
W6A	2.16E+05	7.50E+05	71.23%
Text	1.44E+05	8.00E+05	82.00%
USPS	4.59E+03	4.03E+05	98.86%
Digit1	4.04E+04	8.23E+05	95.10%
Svmguide1	1.26E+04	4.01E+06	99.69%

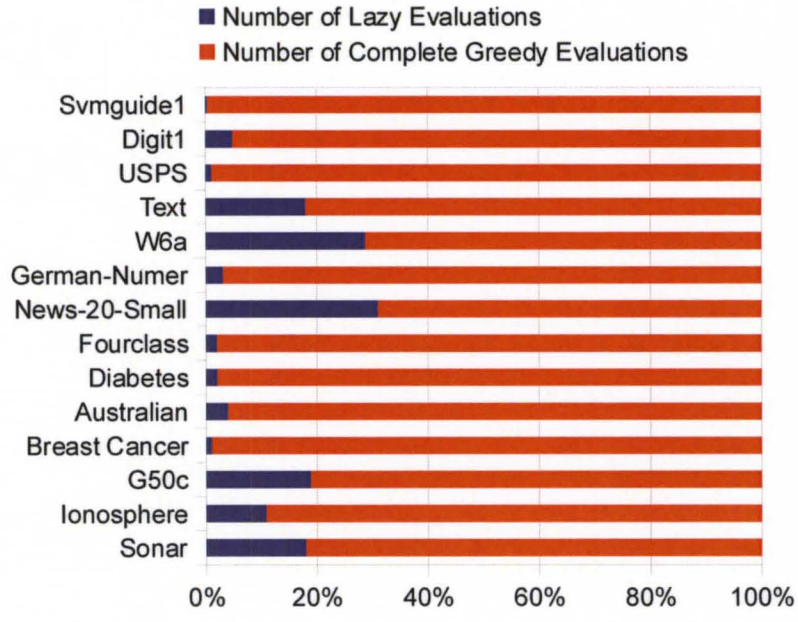


Figure III.14: Depiction of the number of evaluations used by the lazy greedy algorithm compared to the total number evaluations of a standard greedy approach for the small scale data sets.

TABLE III.11

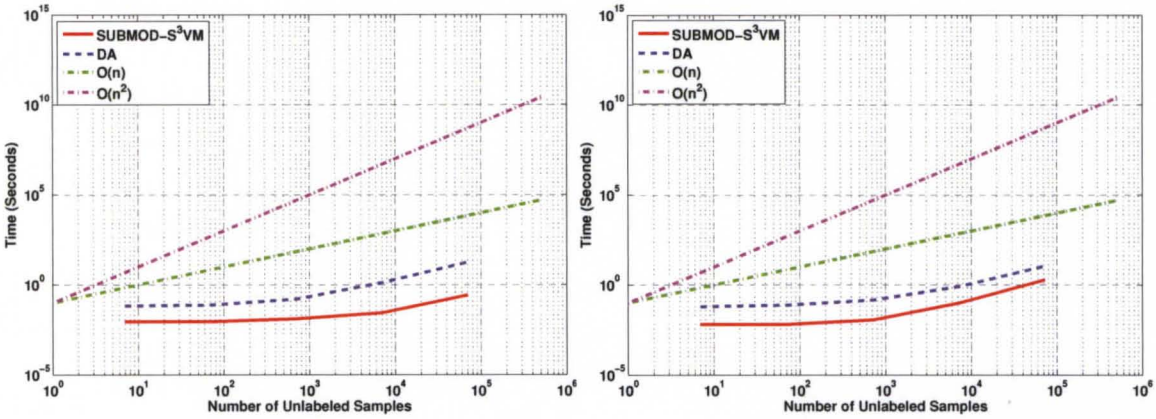
Number of Evaluations for Lazy Greedy and Standard Greedy Approaches for Large Scale Data Set.

Data set	Lazy Evaluations	Complete Greedy Evaluations	Reduction %
News20-Binary	3.08E+004	1.48E+008	99.9793%
CCAT	1.45E+005	1.81E+008	99.9196%
GCAT	1.30E+005	1.30E+008	99.9100%
A9A	6.45E+004	2.55E+008	99.9747%
W8A	2.06E+005	1.01E+008	99.7959%
Aut-Avn	7.52E+005	2.16E+009	99.9651%
Real-Sim	2.30E+005	1.36E+009	99.9830%
Cod-Rna	9.06E+005	6.63E+010	99.9986%
Cov-Type	1.27E+006	1.24E+011	99.9990%
RCV1.Binary	1.42E+006	1.88E+011	99.9992%
KDD-99	2.41E+006	8.15E+011	99.9997%

Next, we examine the efficiency of SUBMOD-S^{3}VM with increasing data set sizes. For this experiment we repeated the original experiment with varying number of total samples. Specifically, for the large scale data sets we run the SUBMOD-S^{3}VM on the full set of unlabeled samples and then reduce the number of samples by one order of magnitude and re-run. The process is repeated until the number of unlabeled samples is less than 10. Fig.III.15 shows the time complexity of the set of large scale data sets compared to the DA algorithm. We also provide the $O(n)$ and $O(n^2)$ complexities as references for comparison. The results shown in Fig.III.15 illustrate that the SUBMOD-S^{3}VM time complexity increases almost linearly with the size of the unlabeled data set.

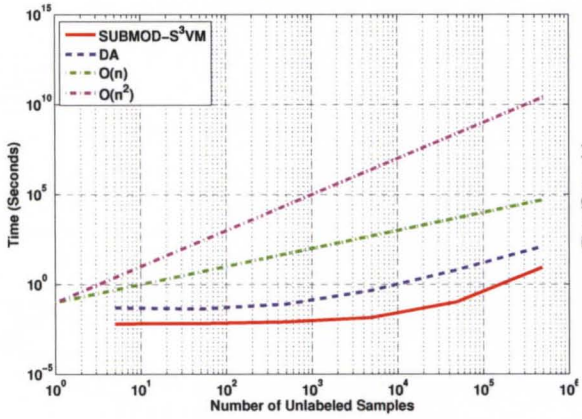
III.5 Discussion

In this chapter we proposed a quadratic programming approximation of the semi-supervised SVM problem (QP-S^{3}VM) that proved to be efficient to solve using standard optimization techniques. One major contribution of the proposed QP-S^{3}VM is that it establishes a link between the two major paradigms of semi-supervised learning, namely low density separation methods and graph-based methods. Such link is considered a significant step towards a unifying framework for semi-supervised learning methods. Furthermore, we propose a novel formulation of the semi-supervised learning problems in terms of submod-

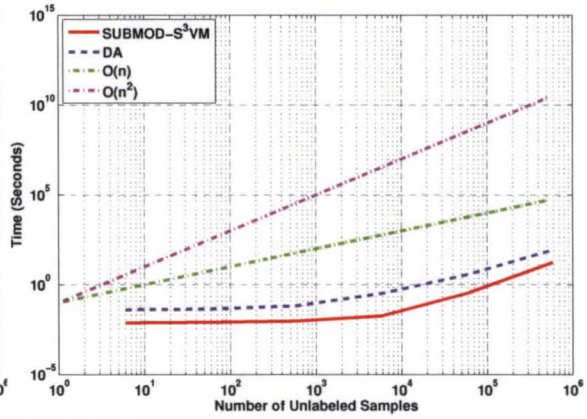


(a) Aut-Avn

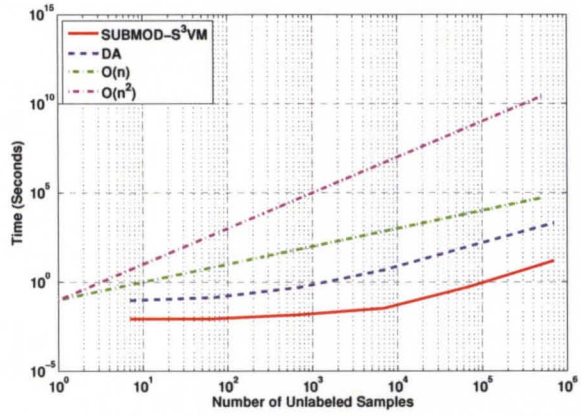
(b) Real-Sim



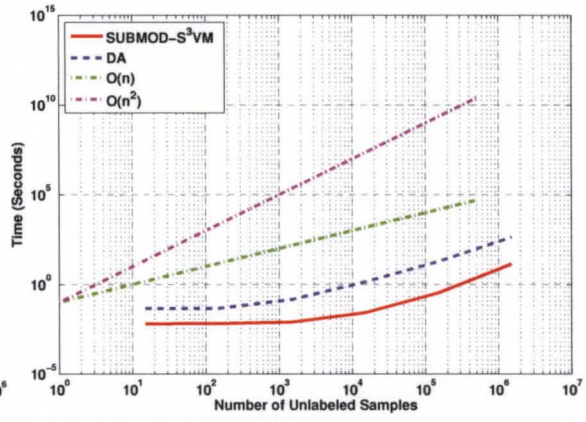
(c) Cod-Rna



(d) Cov-Type



(e) RCV1.Binary



(f) KDD-99

Figure III.15: Time complexity of the proposed SUBMOD-S^{3}VM as a function of the data sets size compared to the DA algorithm.

ular set functions which is, up to our knowledge, is the first time such idea is presented. Using this new formulation we present a methodology to use submodular optimization techniques to efficiently solve the proposed QP-S^{3}VM problem, namely SUBMOD-S^{3}VM. We showed through the chapter that the proposed SUBMOD-S^{3}VM algorithm is very competitive with the sate of the art S^{3}VM algorithms and moreover it achieves a $1.5X - 300X$ speed up which constitutes a significant improvement to the field as well as our general purpose of using such algorithms for incremental/online learning.

CHAPTER IV

SEMI-SUPERVISED SVM LEARNING FOR STREAMING DATA

IV.1 Semi-supervised SVM (S^3VM) for Streaming Data

In Chapter III we have developed the QP-S^3VM and SUBMOD-S^3VM algorithms to efficiently solve the S^3VM problem. However, both algorithms assume the existence of the all data prior to the learning process. The goal of the current chapter is to extend these algorithms to the more flexible case where parts of the data need to be processed sequentially. This constitutes the core of the proposed *never-ending learning* framework. In general, sequential data processing is important for two scenarios: a) If the data is too large to fit in the main memory. Therefore, the learning algorithm has to work on smaller manageable portions of the data sequentially. b) If the data is not available at the beginning of the algorithm operation but rather is being generated over time. The first scenario is known as *Incremental Learning* while the second one is referred to as *Online Learning*. Incremental/online learning

The target of the proposed algorithms in this chapter is to achieve a classification performance that is not affected by the sequential processing of the data as well as reaching a constant time and storage complexity for processing streaming data.

For the rest of this chapter data is presented to a learning algorithm in a stream of batches $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_t$. We consider a batch to be partially labeled, that is $\mathcal{B}_t = \{\mathcal{U}_t, \mathcal{L}_t\}$, where $\mathcal{U}_t$ and $\mathcal{L}_t$ are the unlabeled and labeled sample subsets, respectively. The availability of the labels $\mathcal{L}_t$ depends on the learning scenario. In *incremental learning*, the labels $\mathcal{L}_t$ are revealed to the algorithm instantly. However, in *online learning* they are presented as feedback from the environment after the learning algorithm has classified the samples. Labeled samples are usually expensive to acquire and therefore the majority of the batches are unlabeled ($\mathcal{L}_t = \phi$).

The proposed algorithm proceeds in ordered iterations of *testing*, *supervision* (i.e. environment feedback), and *updating* steps. For *incremental learning* the testing and supervision steps coincide as the labels of $\mathcal{L}_t$ are revealed to the algorithm at the testing step. The following summarize these steps for both the incremental and learning algorithms.

1. At every time step t, the environment chooses a batch $\mathcal{B}_t = \{\mathcal{U}_t, \mathcal{L}_t\}$ and present it to the incremental/online transductive learning algorithm.

2. *Testing Step:* Using its current model (predictor) $\mathcal{E}_{t-1}$:

   (a) The incremental transductive algorithm produces labels for only the *unlabeled* samples in $\mathcal{U}_t$.

   (b) The online transductive algorithm produces labels for all *labeled/unlabeled* samples in $\mathcal{B}_t$.

3. *Supervision Step:* Occasionally, $\mathcal{L}_t \neq \phi$:

   (a) For the incremental transductive algorithm, the environment provides $\mathcal{L}_t$ instantly to the learning machine to use it along with the model $\mathcal{E}_{t-1}$ to predict the labels of $\mathcal{U}_t$.

   (b) The online transductive algorithm labels all samples in $\mathcal{B}_t$ and then the correct labels of the samples in $\mathcal{L}_t$ are provided by the environment as a feedback.

4. *Updating Step:* The incremental/online transductive algorithm updates its current model to $\mathcal{E}_t$ using the new available data.

Note that the QP-S^{3}VM and SUBMOD-S^{3}VM algorithms inherently performs label propagation to predict labels of unlabeled samples. Therefore, the model $\mathcal{E}_{t-1}$ used by the incremental/online extensions consist basically of all samples seen thus far by the algorithm, $\mathcal{E}_{t-1} = \{\mathcal{B}_1, \ldots, \mathcal{B}_{t-1}\}$. This is a hindering problem in practice as no memory is ever enough to store all samples in a data stream. In the proposed work, we use compact exemplar based representations as will be presented in the following sections.

The direct dependence of the model $\mathcal{E}_t$ on the data samples makes all components of a data batch $\mathcal{B}_t = \{\mathcal{U}_t, \mathcal{L}_t\}$ necessary for the the model update process in step 4. Clearly

the labels for the samples in $\mathcal{L}_t$ are very valuable as they present a direct feedback from the environment (oracle) about the performance of the algorithm. However, even the samples in $\mathcal{U}_t$ are of utmost importance as they delineate dense regions in the data space that are necessary for the operation of the QP-S^{3}VM and SUBMOD-S^{3}VM algorithms.

IV.2 From Batch to Online Learning of QP-S^{3}VM

To introduce the proposed *Incremental/Online Transductive Learning Approaches* we start by describing the incremental/online version of the QP-S^{3}VM proposed in Section III.1 and then we proceed to its efficient submodular version. Recall that the QP-S^{3}VM problem has the following form,

$$\begin{aligned} \underset{\mathbf{p}'=[p_1,\ldots,p_{|\mathcal{U}|}]}{\arg\min} \quad & -\tfrac{1}{2}C^{*2}\mathbf{p}'\mathbf{K}_{\mathcal{U}\mathcal{U}}\mathbf{p} + (\tfrac{1}{2}C^{*2}\mathbf{1}'_{|\mathcal{U}|}\mathbf{K}_{\mathcal{U}\mathcal{U}} - CC^*\mathbf{y}'\mathbf{K}_{\mathcal{L}\mathcal{U}})\mathbf{p} \\ \text{subject to} \quad & \mathbf{p}'\mathbf{1}_{|\mathcal{U}|} = r|\mathcal{U}| \quad , \quad \mathbf{0} \leq \mathbf{p} \leq \mathbf{1}_{|\mathcal{U}|} \end{aligned} \tag{IV.1}$$

Assuming that the input data is divided into batches $\{\mathcal{B}_1, \ldots, \mathcal{B}_t, \ldots \mathcal{B}_T\}$, the objective function in Eq.(IV.1) can be rewritten as follows:

$$\sum_{t=1}^{T}\left(-\frac{1}{2}C^{*2}\sum_{s=1}^{T}\mathbf{p}'_{\mathcal{U}_s}\mathbf{K}_{\mathcal{U}_s\mathcal{U}_t} + \frac{1}{2}C^{*2}\sum_{s=1}^{T}\mathbf{1}'_{|\mathcal{U}_s|}\mathbf{K}_{\mathcal{U}_s\mathcal{U}_t} - CC^*\sum_{s=1}^{T}\mathbf{y}'_{\mathcal{L}_s}\mathbf{K}_{\mathcal{L}_s\mathcal{U}_t}\right)\mathbf{p}_{\mathcal{U}_t} \tag{IV.2}$$

where $\mathbf{K}_{\mathcal{U}_s\mathcal{U}_t}$ and $\mathbf{K}_{\mathcal{L}_s\mathcal{U}_t}$ are kernel matrices with $\mathcal{U}_s, \mathcal{L}_s \in \mathcal{B}_s$ and $\mathcal{U}_t \in \mathcal{B}_t$. Eq.(IV.2) shows that the QP-S^{3}VM objective is additive over the data batches.

In incremental/online learning, the batches $\mathcal{B}_t$ arrive sequentially and thus the algorithm has access only to the input batches up to the present time. Keeping in mind that the objective function in Eq.(IV.2) is decomposable over the data batches, a natural incremental/online extension of the QP-S^{3}VM algorithm presents itself where for each new arriving data batch $\mathcal{B}_t$ an instantaneous objective functions of the following form is optimized:

$$\left(-\frac{1}{2}C^{*2}\sum_{s=1}^{t}\mathbf{p}'_{\mathcal{U}_s}\mathbf{K}_{\mathcal{U}_s\mathcal{U}_t} + \frac{1}{2}C^{*2}\sum_{s=1}^{t}\mathbf{1}'_{|\mathcal{U}_s|}\mathbf{K}_{\mathcal{U}_s\mathcal{U}_t} - CC^*\sum_{s=1}^{t}\mathbf{y}'_{\mathcal{L}_s}\mathbf{K}_{\mathcal{L}_s\mathcal{U}_t}\right)\mathbf{p}_{\mathcal{U}_t} \tag{IV.3}$$

Naturally, the instantaneous objective function in Eq.(IV.3) is optimized with respect to $\mathbf{p}_{\mathcal{U}_t}$ where labels are assigned to the most recent batch of unlabeled samples $\mathcal{U}_t$. This form of instantaneous objective optimization depends on the decisions made in previous

iterations for $\mathbf{p}_{\mathcal{U}_s}$ where $s < t$. Since there is no assumption about the sequence in which the data batches are presented to the learning algorithm, the values of $\mathbf{p}_{\mathcal{U}_s}$ obtained with partial observation of the data are not guaranteed to be consistent with their values if the whole data set is observed. To illustrate this idea, we examine two possibilities for batch sequence arrival of the sliced cube data set in Fig.IV.1.

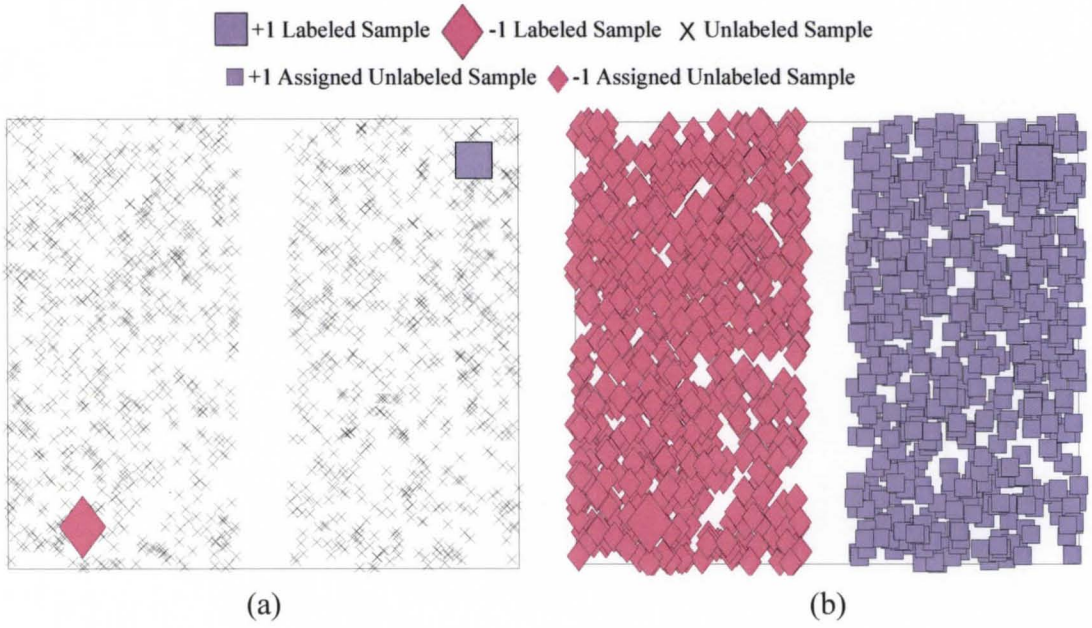


Figure IV.1: (a) Partially labeled sample of the Sliced Cube data set. (b) The ideal outcome after using semi-supervised learning to label the unlabeled samples.

Figure IV.2 and Fig.IV.3 depict the ideal and the realistic scenarios for batch sequence arrival of the sliced cube data set, respectively. In both figures, the first column (a-d) shows the partially labeled data available at each time step. Each data batch is titled with a number indicating the time step when it arrived. Figure IV.2 depicts a rather ideal scenario for data batches arrival, where the new unlabeled batches of the same class arrive in sequence so as to be closer to each other than to the opposite class, see Fig.IV.2(a-d). Using the instantaneous objective function in Eq.(IV.3) for this scenario will result in a perfect solution as seen in Fig.IV.2(e-h). This is because the data batch arrival is coherent with the inherent data structure and thus the instantaneous objective function does not make intermediate mistakes during the algorithm operation.

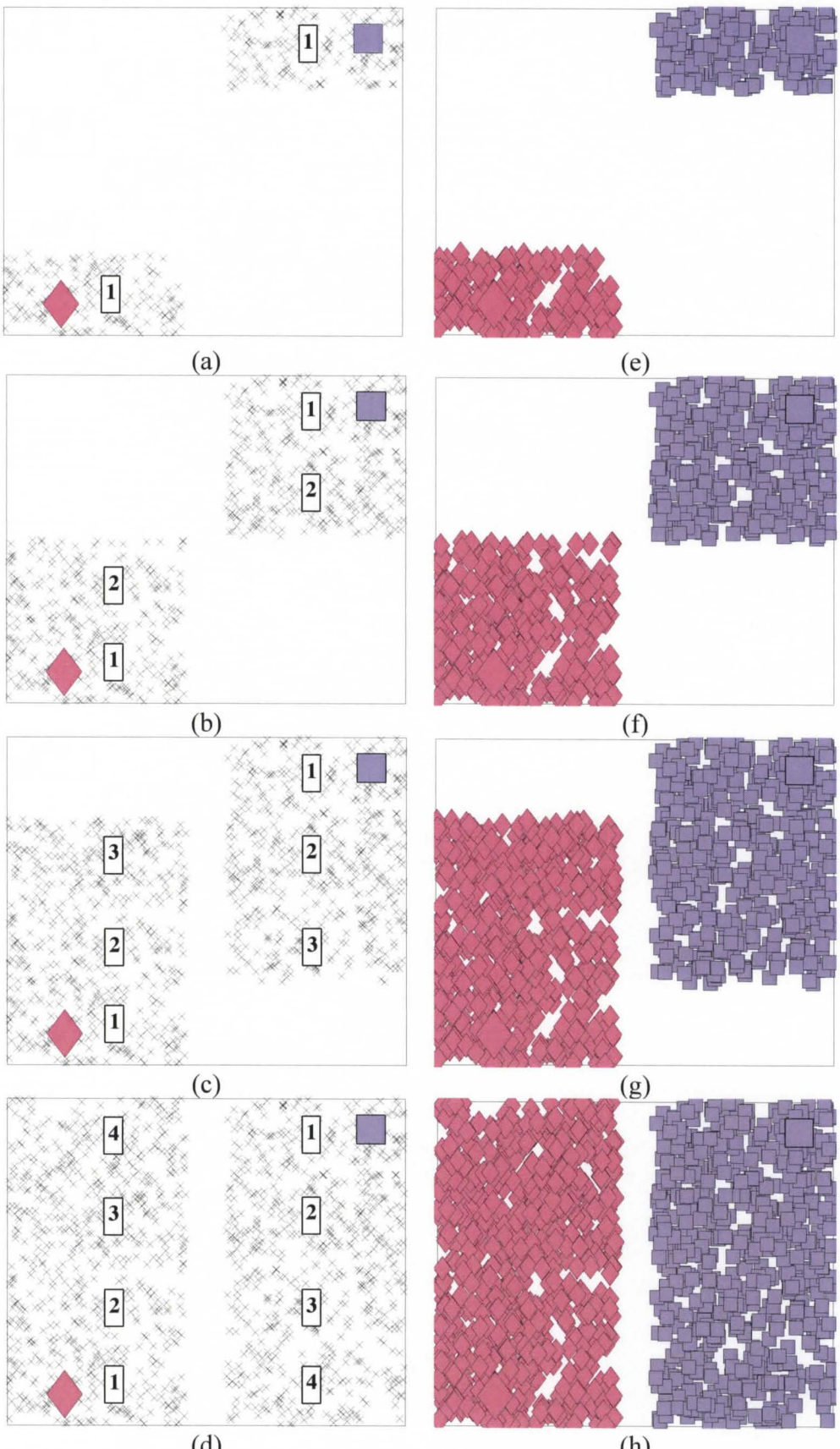


Figure IV.2: (a-d) Ideal scenario for data batches arrival. (e-h) Corresponding labeling using instantaneous objective function in Eq.(IV.3).

A more realistic scenario for batch data arrival is depicted in Fig.IV.3(a-d), for instance in the second time step of the algorithm, Fig.IV.3(b), two batches arrive that are closer to their opposite classes than to their own. Therefore, optimizing the instantaneous objective in Eq.IV.3 will results in the labeling shown in Fig.IV.3(f) which is clearly correct with respect to the currently available data but rather wrong when more data becomes available and the correct structure of the data is uncovered. As mentioned earlier, the instantaneous objective function finds the labeling for the most recent data batch using the labelings obtained in previous iterations. Therefore, we see that that the wrongful labeling in Fig.IV.3(f) carries over to Fig.IV.3(g-h) and thus the intermediate mistake made in Fig.IV.3(f) ruins the results for all further iterations.

One suggestion to overcome the problem of error propagation through iterations is to optimize the instantaneous objective function in Eq.(IV.2) for both $\mathbf{p}_{\mathcal{U}_s}, \forall s < t$, and $\mathbf{p}_{\mathcal{U}_t}$ for each new data batch $\mathcal{B}_t$,

$$\underset{\mathbf{p}_{\mathcal{U}_s} \ \forall s \leq t}{\arg\min} \Bigg(\underbrace{-\frac{1}{2} C^{*2} \sum_{s=1}^{t} \mathbf{p}'_{\mathcal{U}_s} \mathbf{K}_{\mathcal{U}_s \mathcal{U}_t}}_{\mathcal{T}_1} + \underbrace{\frac{1}{2} C^{*2} \sum_{s=1}^{t} \mathbf{1}'_{|\mathcal{U}_s|} \mathbf{K}_{\mathcal{U}_s \mathcal{U}_t}}_{\mathcal{T}_2} - \underbrace{C C^* \sum_{s=1}^{t} \mathbf{y}'_{\mathcal{L}_s} \mathbf{K}_{\mathcal{L}_s \mathcal{U}_t}}_{\mathcal{T}_3} \Bigg) \mathbf{p}_{\mathcal{U}_t} \quad \text{(IV.4)}$$

This suggestion might look promising specially that it maintains label propagation and similarities between the current batch unlabeled/labeled samples and all previous unlabeled/labeled samples through the terms $\mathcal{T}_2$ and $\mathcal{T}_3$, respectively. It also maintains label assignment smoothness between the current batch and all previous ones through the term $\mathcal{T}_1$. However, one crucial part is missing from Eq.(IV.4) which maintains the labeling smoothness among the previous batches themselves. The absence of such smoothness results in data labeling that is not consistent with underlying geometry of the data.

The previous argument tells us that using the QP-S^{3}VM algorithm for streaming data is only possible by repetitively applying it on all the data available at each iteration as follows:

Problem 5. Incremental/Online QP-S^{3}VM:

For a sequence of data batches $\mathcal{B}_1, \ldots, \mathcal{B}_\tau, \ldots, \mathcal{B}_T$, as each batch become available

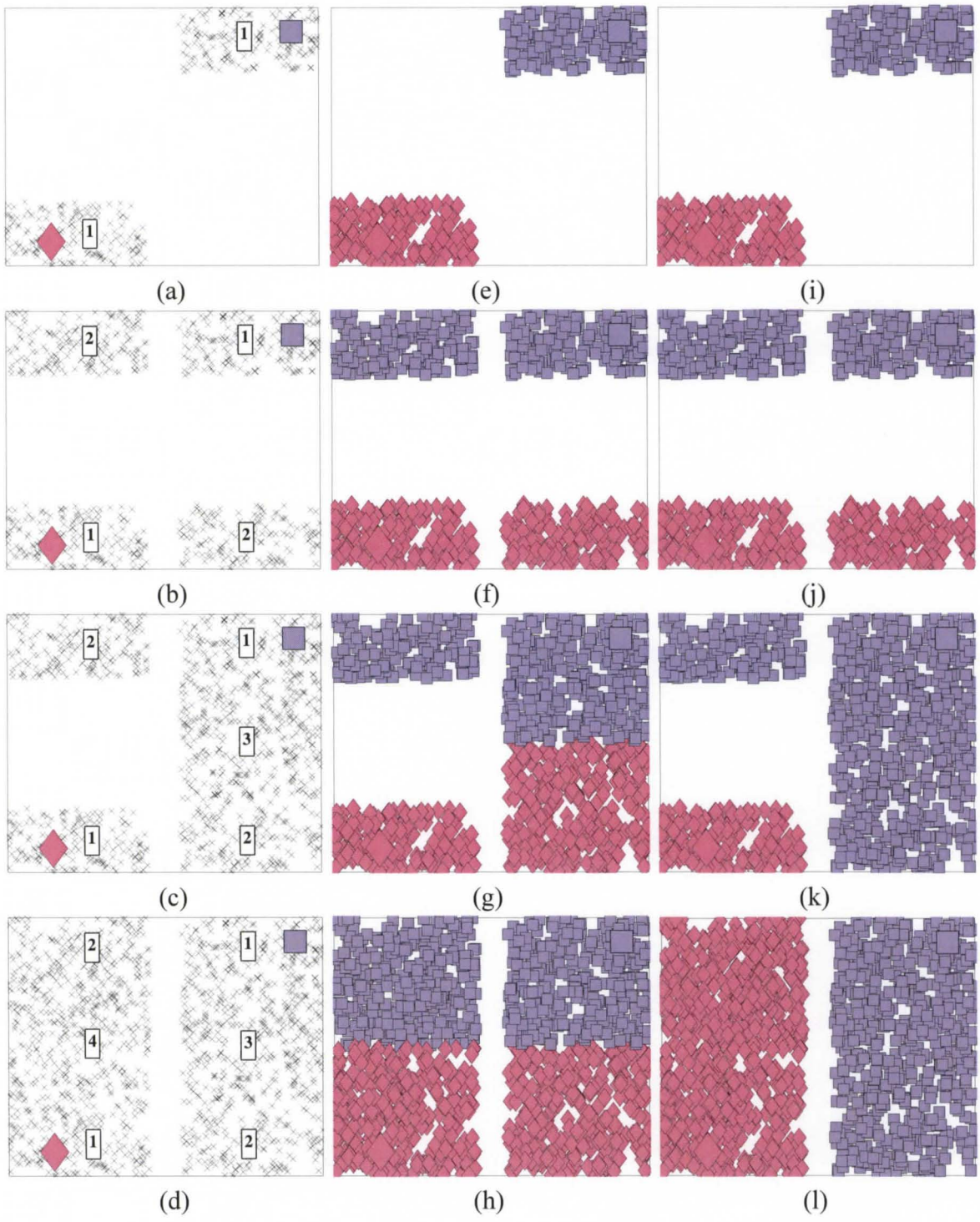


Figure IV.3: (a-d) Realistic scenario for data batches arrival. (e-h) Corresponding labeling using instantaneous objective function in Eq.(IV.3) which exhibits errors to carry over between iterations. (i-l) Corresponding correct labeling without error carry over.

find the solution of,

$$\underset{\mathbf{p}_{\mathcal{U}_t}}{\arg\min} \underset{\forall t \leq T}{\sum_{t=1}^{T}} \left(-\frac{1}{2} C^{*2} \sum_{s=1}^{T} \mathbf{p}'_{\mathcal{U}_s} \mathbf{K}_{\mathcal{U}_s \mathcal{U}_t} + \frac{1}{2} C^{*2} \sum_{s=1}^{T} \mathbf{1}'_{|\mathcal{U}_s|} \mathbf{K}_{\mathcal{U}_s \mathcal{U}_t} - C C^* \sum_{s=1}^{T} \mathbf{y}'_{\mathcal{L}_s} \mathbf{K}_{\mathcal{L}_s \mathcal{U}_t} \right) \mathbf{p}_{\mathcal{U}_t}$$

This way we make sure that all label assignments to all unlabeled samples available thus far are based on the current view of data and that any previous mistakes due to partial observation of the data are fixed as more data become available and finally we are certain that all smooth label propagation paths (dense regions) are preserved.

IV.2.1 Proposed QP-Exemplar Selection Model for Online QP-S^{3}VM Learning (QP-EXMP)

We have shown that the QP-S^{3}VM algorithm can only be used for streaming data via repetitive application on all data available at each time iteration. However, this entails storing all samples of a data stream. This deems the applicability of the approach impossible. In this section we propose a stream summarization, i.e. exemplars selection, technique for the specific use with QP-S^{3}VM algorithm. The exemplars are selected to keep a compact representation of the data stream that retains the key properties important to semi-supervised learning with respect to dense regions and similarities to labeled samples.

Figure IV.4 provides an illustration of the concept of exemplar selection for incremental/online learning. Figure IV.4(a) shows a batch $\mathcal{B}1$ of partially labeled data arriving at the disposal of the learning algorithm. $\mathcal{B}1$ will be processed by S^{3}VM and the labels of the unlabeled samples ($+$ and $\times$) are predicted. To proceed with processing more batches, $\mathcal{B}1$ has to be summarized using few exemplar samples. The summarizations aims at choosing few representative exemplars that fit into the main memory. The shaded samples in Fig.IV.4(b) represent the chosen exemplars. The exemplars at this time iteration form the model of the data stream ($\mathcal{E}_t$) up until this moment. In the next time iteration, a new batch $\mathcal{B}2$ arrives, Fig.IV.4(c). The stream model $\mathcal{E}_t$ is then appended to $\mathcal{B}2$ for further processing by S^{3}VM.

The proposed exemplar selection technique is formulated as follows:

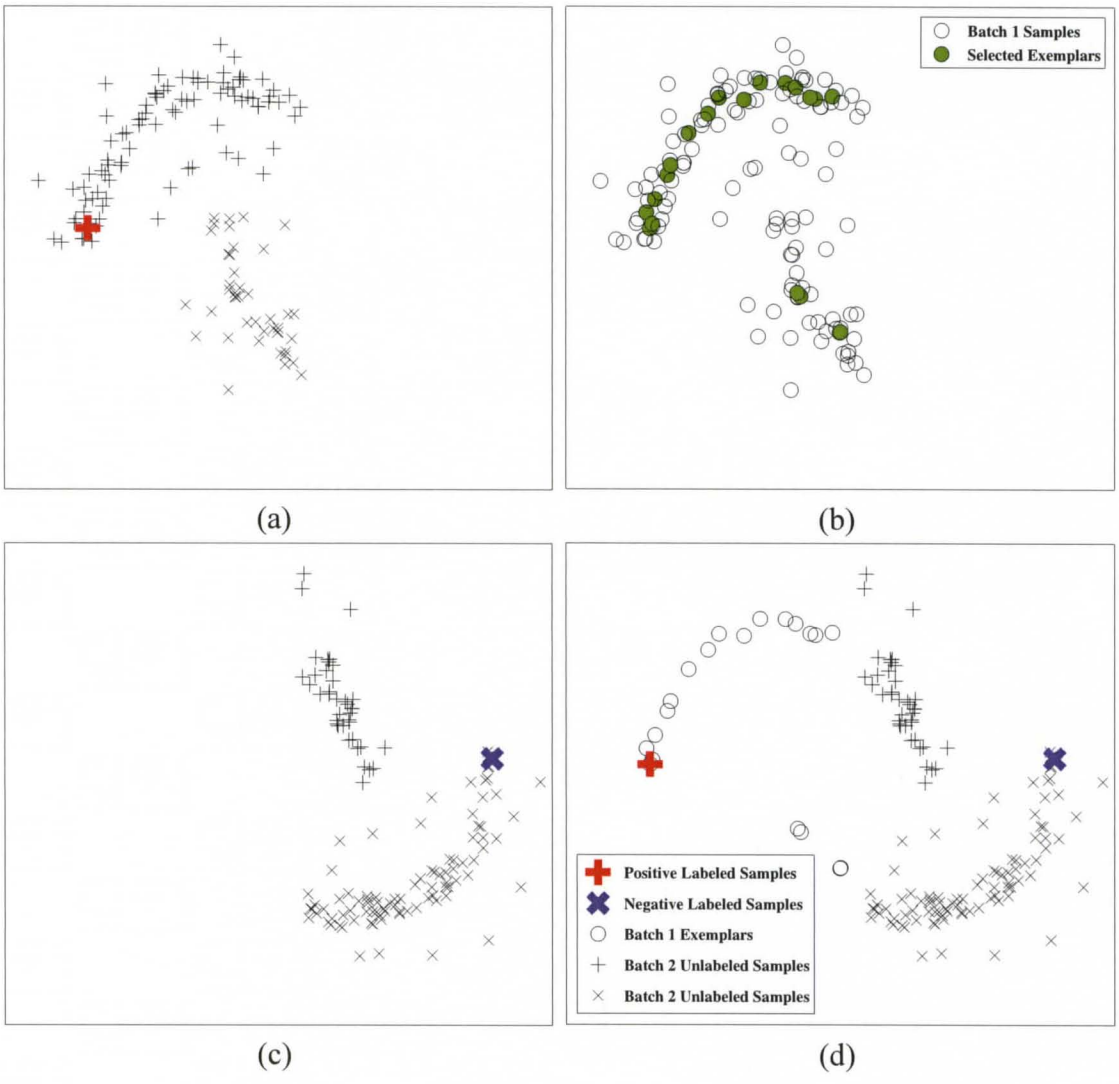


Figure IV.4: Illustration of exemplar selection for stream summarization. (a) $\mathcal{B}1$ of data with a single labeled sample and many unlabeled ones. (b) Exemplars selected to summarize $\mathcal{B}1$. (c) Batch $\mathcal{B}2$. (d) Exemplars from (b) are appended to $\mathcal{B}2$ for further learning iterations.

Problem 6. *Quadratic Programming Exemplar Selection Algorithm for Incremental/Online QP-S^{3}VM (QP-EXMP):*

Given a data batch $\mathcal{B} = \{\mathcal{U}, \mathcal{L}\}$, with $\mathcal{U}$ and $\mathcal{L}$ being subsets of unlabeled and labeled samples, respectively. Assuming that the subset $\mathcal{L}$ is very small and to be kept in its entirety. The proposed QP-EXMP selects exemplars from the subset $\mathcal{U}$ using the following optimization problem:

$$\underset{\mathbf{e}'=[e_1,\ldots,e_{|\mathcal{U}|}]}{\arg\min} \quad \lambda_1 \mathbf{e}'\mathbf{K_{uu}}\mathbf{e} + \lambda_2 \mathbf{1}'_{|\mathcal{L}|}\mathbf{K}_{lu}(\mathbf{1}_{|\mathcal{U}|} - \mathbf{e}) + \lambda_3 \mathbf{1}'_{|\mathcal{U}|}\mathbf{K}_{uu}(\mathbf{1}_{|\mathcal{U}|} - \mathbf{e}) \tag{IV.5}$$

subject to

$$\mathbf{e}'\mathbf{1}_{|\mathcal{U}|} = \mathcal{M}_{\mathcal{E}}, \quad \mathbf{0} \leq \mathbf{e} \leq \mathbf{1}_{|\mathcal{U}|}. \tag{IV.6}$$

where

$\mathbf{1}_{|\mathcal{L}|}$*: A ones vector of length* $|\mathcal{L}|$*. Similarly is* $\mathbf{1}_{|\mathcal{U}|}$*.*

$\mathbf{e}' = [e_1, \ldots, e_{|\mathcal{U}|}]$, $e_j = 1$ *for selected exemplars and* $e_j = 0$ *otherwise.*

$\mathcal{M}_{\mathcal{E}}$*: The number of exemplars to be chosen.*

$\mathbf{K_{ll}} = \mathbf{K}_{i,i'}\ \forall i, i' \in \mathcal{L}, \quad \mathbf{K_{uu}} = \mathbf{K}_{j,j'}\ \forall j, j' \in \mathcal{U}, \quad \mathbf{K_{lu}} = \mathbf{K}_{i,j}\ \forall i \in \mathcal{L}, j \in \mathcal{U}.$

Note: Equation (IV.5) *can be rewritten in the standard quadratic programming form as follows:*

$$\underset{\mathbf{e}'=[e_1,\ldots,e_{|\mathcal{U}|}]}{\arg\min} \quad \lambda_1 \mathbf{e}'\mathbf{K_{uu}}\mathbf{e} - (\lambda_2 \mathbf{1}'_{|\mathcal{L}|}\mathbf{K}_{lu} + \lambda_3 \mathbf{1}'_{|\mathcal{U}|}\mathbf{K}_{uu})\mathbf{e} \tag{IV.7}$$

Figure IV.5 provides a depiction of the proposed incremental/online QP-S^{3}VM learning procedure using the QP-EXMP exemplar selection algorithm. Each column in Fig.IV.5 demonstrates the operation of the algorithm at a single point in time. The first row contains the new data batch arriving at that time iteration, $\mathcal{B}_1, \ldots, \mathcal{B}_t$. The last row shows the corresponding labeled batches after applying QP-S^{3}VM, $\tilde{\mathcal{B}}_1, \ldots, \tilde{\mathcal{B}}_t$. The second row contains the stream model used for the current time iteration. Notice that in the first column, the stream model is empty as $\mathcal{B}_1$ is the first batch to process. The third row in the figure contains the aggregation of the input batch in the first row and the stream model in the second row. Basically the contents of the third row represent the input to the QP-S^{3}VM at each time iteration.

In the proposed algorithm, we assume that all batches $\mathcal{B}_t$ have fixed size. We further assume that the memory budget for the stream model is set at twice the size of the processed batches, i.e. $2|\mathcal{B}_t|$. These assumptions are quite general as data streams can be processed at arbitrary batch sizes. In streaming data processing, it is usually assumed that the correlation between data samples is inversely proportional to the time elapsed between their arrival. Therefore, in the proposed incremental/online learning procedure, we give more weight to the last arriving batch against being summarized to construct the stream model. Basically, at each time t the used stream model consists of the last data batch $\mathcal{B}_{t-1}$ in its entirety and the QP-EXMP exemplar summarization of all previous batches of the stream $\mathcal{E}_{t-2}$ starting with $\mathcal{B}_{t-2}$ until the first batch. This is illustrated in Fig.IV.5 where at $t = 3$ the stream model in the second row consists of the last batch $\mathcal{B}_2$ combined with the stream summarization $\mathcal{E}_1$; $\mathcal{E}_1$ is essentially the QP-EXMP summary of the batch $\mathcal{B}_1$. On the other hand, at $t = 2$ the stream model is $\mathcal{B}_1$ as simply $\mathcal{E}_1 = \phi$.

Each curved arrow in Fig.IV.5 represents applying the QP-EXMP algorithm to obtain an exemplar representation of the stream $\mathcal{E}_t$. This illustrated in the third row at $t = 3$ where the QP-EXMP algorithm summarizes $\mathcal{E}_1\mathcal{B}_2$ into $\mathcal{E}_2$.

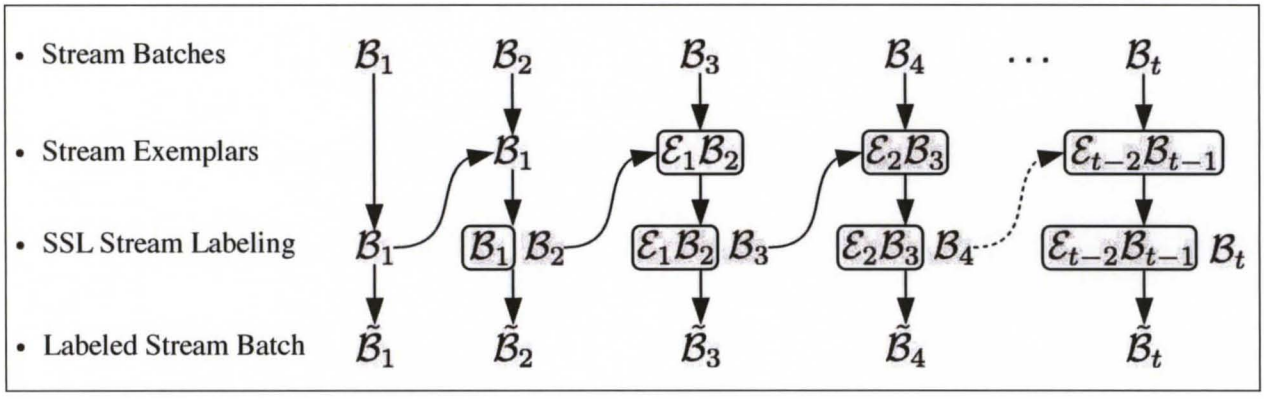


Figure IV.5: Illustration of the incremental/online QP-S^{3}VM learning procedure using the proposed QP-EXMP exemplar selection algorithm.

IV.2.2 QP-EXMP Model Interpretation

In this section we provide an interpretation of the proposed QP-EXMP model which will give an insight into understanding why this model works and how it is appropriate for stream summarization for the QP-S^{3}VM algorithm. The QP-EXMP model has the following formulation:

$$\underset{\mathbf{e}'=[e_1,\ldots,e_{|\mathcal{U}|}]}{\arg\min} \; \lambda_1 \mathbf{e}'\mathbf{K_{uu}}\mathbf{e} + \lambda_2 \mathbf{1}'_{|\mathcal{L}|}\mathbf{K}_{lu}(\mathbf{1}_{|\mathcal{U}|} - \mathbf{e}) + \lambda_3 \mathbf{1}'_{|\mathcal{U}|}\mathbf{K}_{uu}(\mathbf{1}_{|\mathcal{U}|} - \mathbf{e}) \tag{IV.8}$$

subject to

$$\mathbf{e}'\mathbf{1}_{|\mathcal{U}|} = \mathcal{M}_{\mathcal{E}}, \quad \mathbf{0} \leq \mathbf{e} \leq \mathbf{1}_{|\mathcal{U}|}.$$

The first term in Eq.(IV.8) can be expanded in the following manner:

$$\lambda_1 \mathbf{e}'\mathbf{K_{uu}}\mathbf{e} = \underbrace{\lambda_1 \sum_{\substack{j,j'=\{1,\ldots,|\mathcal{U}|\} \\ j=j'}} [\mathbf{K_{uu}}]_{j,j'} e_{j'} e_j}_{\mathcal{Q}_1} + \underbrace{\lambda_1 \sum_{\substack{j=\{1,\ldots,|\mathcal{U}|-1\} \\ j'=\{j+1,\ldots,|\mathcal{U}|\}}} [\mathbf{K_{uu}}]_{j,j'} (2 e_j e_{j'})}_{\mathcal{Q}_2} \tag{IV.9}$$

The term $\mathcal{Q}_1$ is quadratic in e_j, thus $\mathcal{Q}_1$ is minimized by setting all e_j value to zero. Therefore, $\mathcal{Q}_1$ induces sparsity of the solution by discouraging samples from being chosen as exemplars, which is a desirable notion in our problem where only few important exemplars are to be chosen. Minimizing the term $\mathcal{Q}_2$ is explained through Fig.IV.6 where we plot the function $z_{j,j'} = 2e_j e_{j'}$, for all $e_j, e_{j'} \in [0, 1]$. Minimizing $\mathcal{Q}_2$ with respect to the cardinality constraint $\mathbf{e}'\mathbf{1}_{|\mathcal{U}|} = |\mathcal{E}|$ means that terms with large $[\mathbf{K_{uu}}]_{j,j'}$ values should be assigned a small $2e_j e_{j'}$ value and vice versa for small $[\mathbf{K_{uu}}]_{j,j'}$. This basically means that when two samples are similar, large $[\mathbf{K_{uu}}]_{j,j'}$, then only one or neither of them should be selected as an exemplar $e_j = 1$. However, if the two samples are dissimilar, small $[\mathbf{K_{uu}}]_{j,j'}$, then both samples are selected to be exemplars. Therefore, minimizing $\mathcal{Q}_2$ encourages selecting diverse samples which is once more an important criterion in exemplar selection as it provides the ability to model rich environments using few samples.

The second term in Eq.(IV.8) can be expanded as follows:

$$\lambda_2 \mathbf{1}'_{|\mathcal{L}|}\mathbf{K}_{lu}(\mathbf{1}_{|\mathcal{U}|} - \mathbf{e}) = \lambda_2 \sum_{i \in \mathcal{L}, j \in \mathcal{U}} [\mathbf{K_{lu}}]_{i,j}(1 - e_j) \tag{IV.10}$$

Since $e_j \in [0, 1]$, minimizing Eq.(IV.10) would involve setting small $(1 - e_j)$, i.e. $e_j \simeq 1$, to $[\mathbf{K_{lu}}]_{i,j}$ with large values and vice versa. This means that samples with high similarity

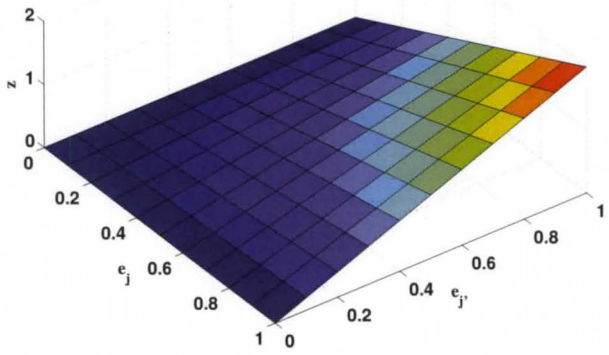


Figure IV.6: Plot of $z_{j,j'} = (2e_j e_{j'})$ for all $e_j, e_{j'} \in [0, 1]$.

to labeled samples are encouraged to be picked as exemplars. This idea makes sense in the context of exemplar selection for the QP-S^{3}VM algorithm where label propagation occurs from labeled samples through dense regions to unlabeled samples. Therefore, to ensure proper label propagation exemplars close to labeled samples are important to be included.

The same argument we used for the second term in Eq.(IV.10) can be used to interpret the third term,

$$\lambda_3 \mathbf{1}'_{|\mathcal{U}|} \mathbf{K}_{uu} (\mathbf{1}_{|\mathcal{U}|} - \mathbf{e}) = \lambda_3 \sum_{j,j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{j,j'} (1 - e_j) \tag{IV.11}$$

We see that the minimizing the third term encourages selecting exemplars that are similar or close to other unlabeled samples. In other words, this term chooses exemplars that exist in dense regions. This is once more an important criterion for exemplar selection for semi-supervised learning paradigm where unlabeled samples delineate the data space and moreover provide dense paths for label propagation. Therefore, choosing exemplars in dense regions helps preserving the underlying dense regions structure of the data space for use semi-supervised learning algorithms.

In the rest of this section we provide a set of illustrative figures to show in action the effect and importance of the different components of the proposed QP-EXMP model in Eq.(IV.8). The basic set up for these illustrations starts with the Two Moons data set in Figure IV.7. In Fig. IV.7(a), the bold samples are labeled samples while the thin ones are unlabeled. The unlabeled samples have the marker of their ideal labeling. Therefore, for

an unlabeled sample to be labeled correctly, the color assigned to it should match the color of the labeled sample of the same marker shape. This makes it easier to visually judge the quality of the achieved labeling.

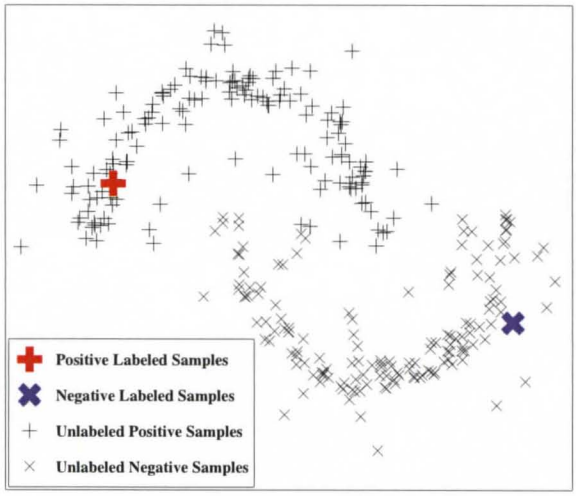
To simulate the streaming scenario, the data set in Fig.IV.7(a) is split into two batches $\mathcal{B}1$ and $\mathcal{B}2$ that arrive in order, see Fig.IV.7 (b) and (c), respectively. The illustration setup continues by applying the QP-EXMP algorithm on $\mathcal{B}1$ to select few summary exemplars $\mathcal{E}_1$. The exemplars $\mathcal{E}_1$ along with the batch $\mathcal{B}2$ are used as input to the QP-S^3VM algorithm to assign labels to $\mathcal{B}2$. In each of the following illustrative experiments we will demonstrate the importance of the various components of the QP-EXMP model by showing the output of the exemplar selection process if each component was ignored from the model. We further show how the differences in the chosen exemplars reflect on the labeling of the batch $\mathcal{B}_2$.

First we examine the effect of the first term in Eq.(IV.8), which induces diversity among the chosen exemplars. Figure IV.8(a) depicts the exemplars selected by the original QP-EXMP model, and Fig. IV.8(b) provides the exemplars when the diversity inducing term is removed. The exemplars chosen without the diversity term (Right Column) are clumped and therefore does not well represent the dense regions in $\mathcal{B}1$. As can be seen in the bottom row, this resulted in a severe distortion in the labeling of the batch $\mathcal{B}_2$.

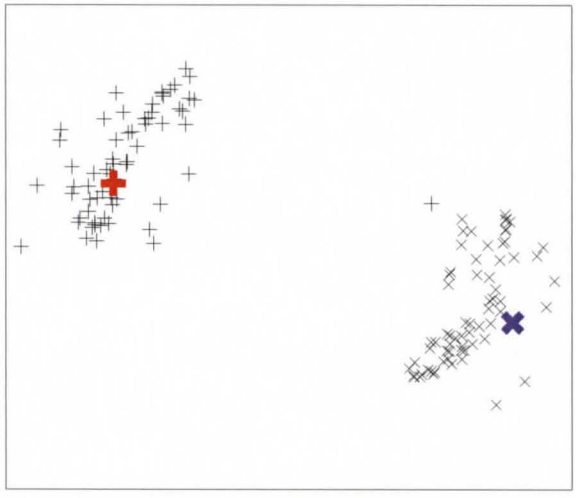
Figure IV.9 illustrates the importance of exemplars selection in dense regions. As we can see, ignoring the third term in Eq.(IV.8) results in selecting exemplars that are more likely to belong to noise (Right Column) rather than meaningful exemplars representing the underlying dense regions of the data space. This also has a degenerative effect on the labeling obtained for the batch $\mathcal{B}2$ as shown in the bottom row of the figure.

To illustrate the importance of the exemplar selections with high similarity to labeled samples, i.e. the second term in Eq.(IV.8), we use the Long Tail Two Moons data set in Fig.IV.10. Figure IV.11 shows the effect of exemplar selections without this term, where ignoring the similarity to labeled samples resulted in cutting the path for labels to propagate from the labeled samples to unlabeled samples and thus distorting the final labeling of the batch $\mathcal{B}2$ as shown in the bottom row of the figure.

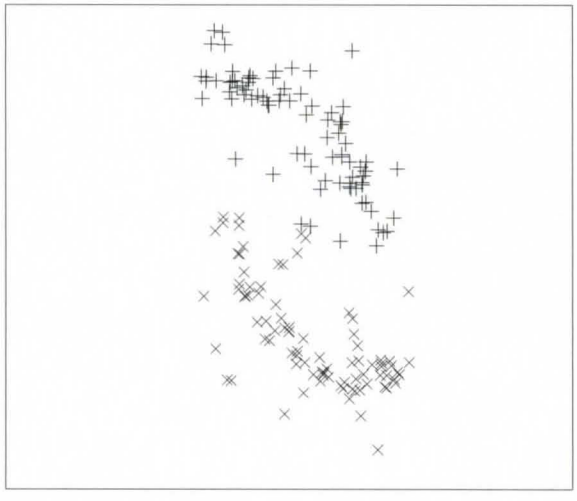
One important property of the proposed QP-EXMP model is its ability to select



(a)



(b)



(c)

Figure IV.7: (a) Two moons partially labeled data set. (b) Batch $\mathcal{B}1$. (c) Batch $\mathcal{B}2$.

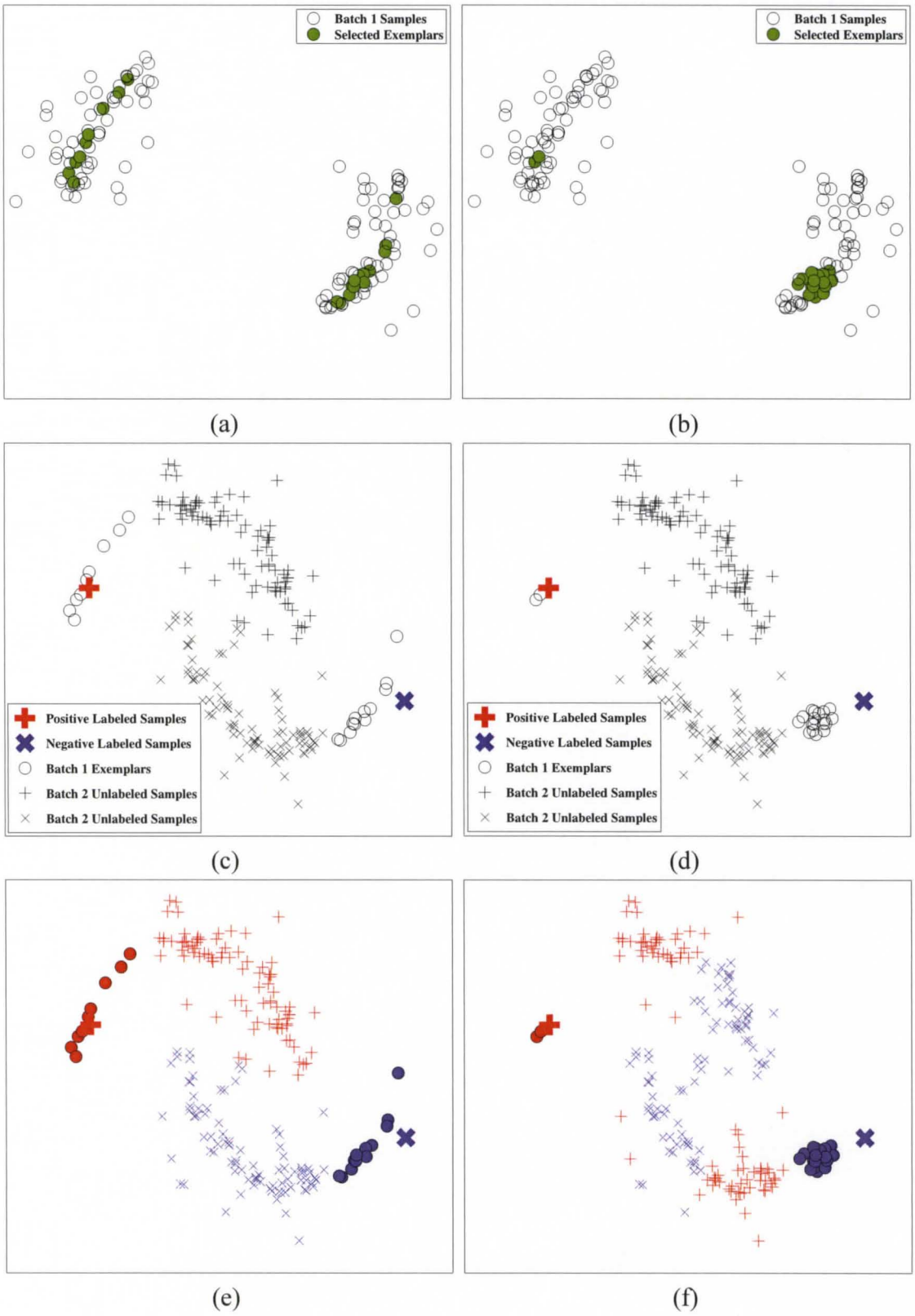


Figure IV.8: Importance of diversity in exemplars. Left Column: Output of QP-EXMP model. Right Column: Output if diversity enforcement is ignored. (a-b) Selecting exemplars from batch $\mathcal{B}1$. (c-d) Using exemplars and batch $\mathcal{B}2$ as input for SSL. (e-f) Output of SSL.

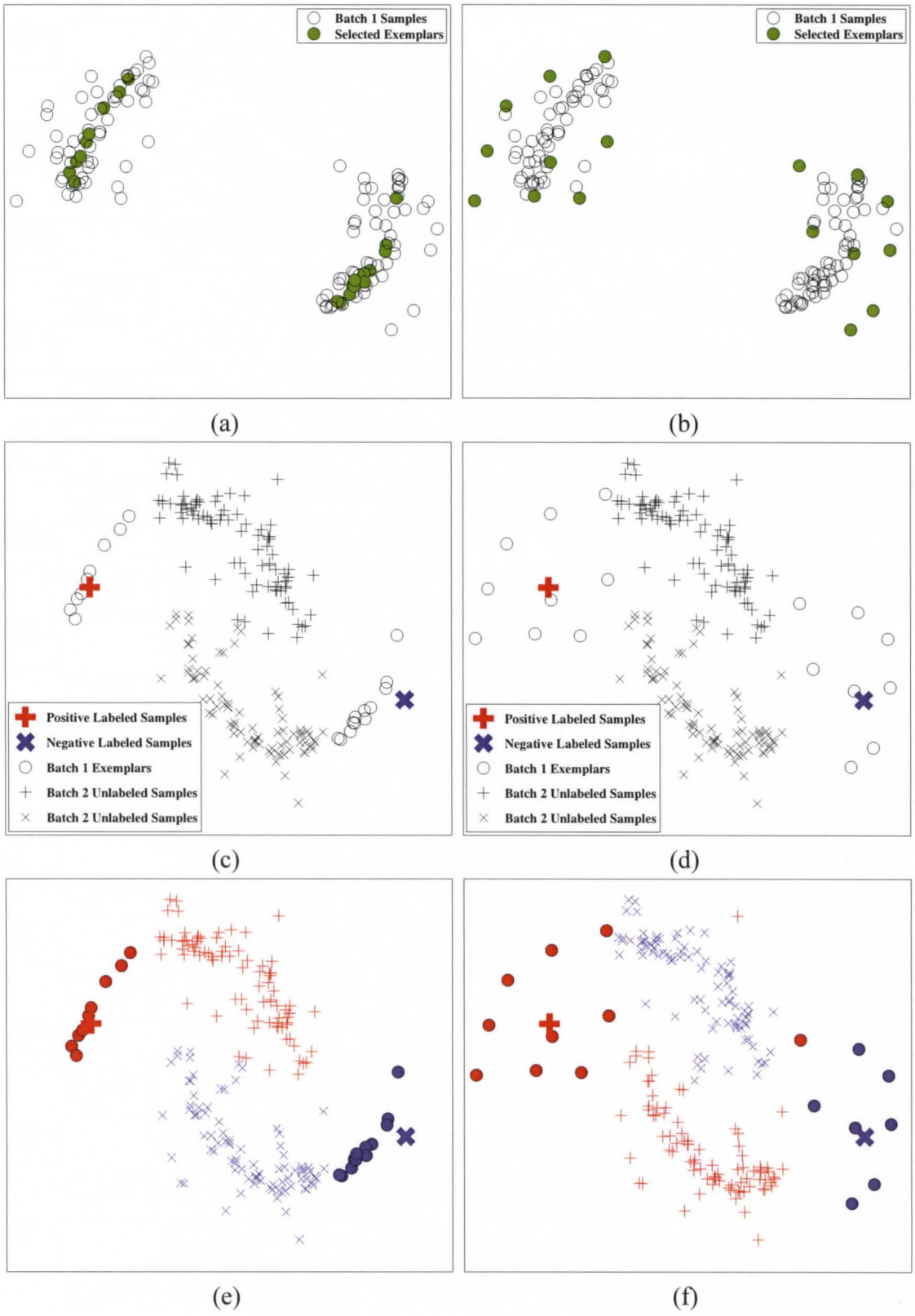
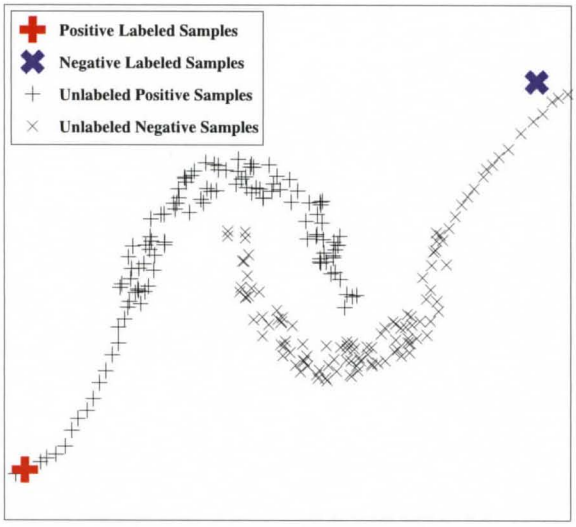
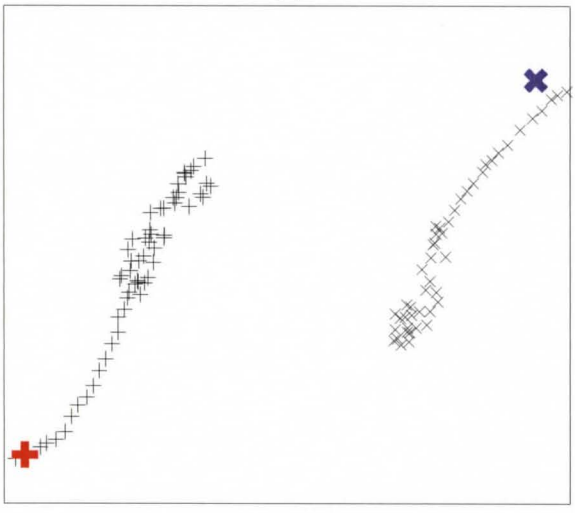


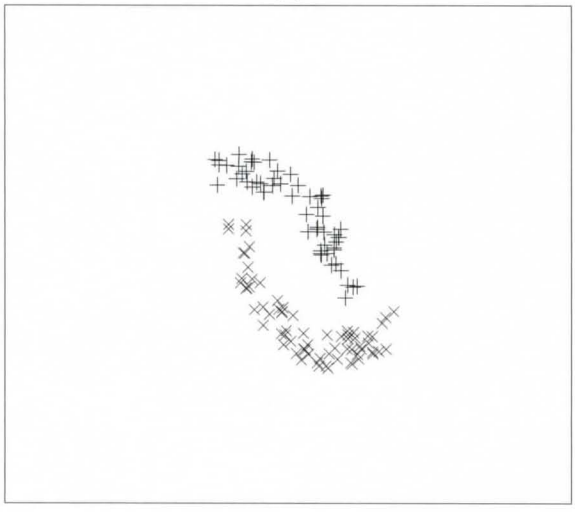
Figure IV.9: Importance of choosing exemplars in dense regions. Left Column: Output of QP-EXMP model. Right Column: Output if dense region enforcement is ignored. (a-b) Selecting exemplars from batch $\mathcal{B}1$. (c-d) Using exemplars and batch $\mathcal{B}2$ as input for SSL. (e-f) Output of SSL.



(a)



(b)



(c)

Figure IV.10: (a) Long tail two moons partially labeled data set. (b) Batch $\mathcal{B}1$. (c) Batch $\mathcal{B}2$.

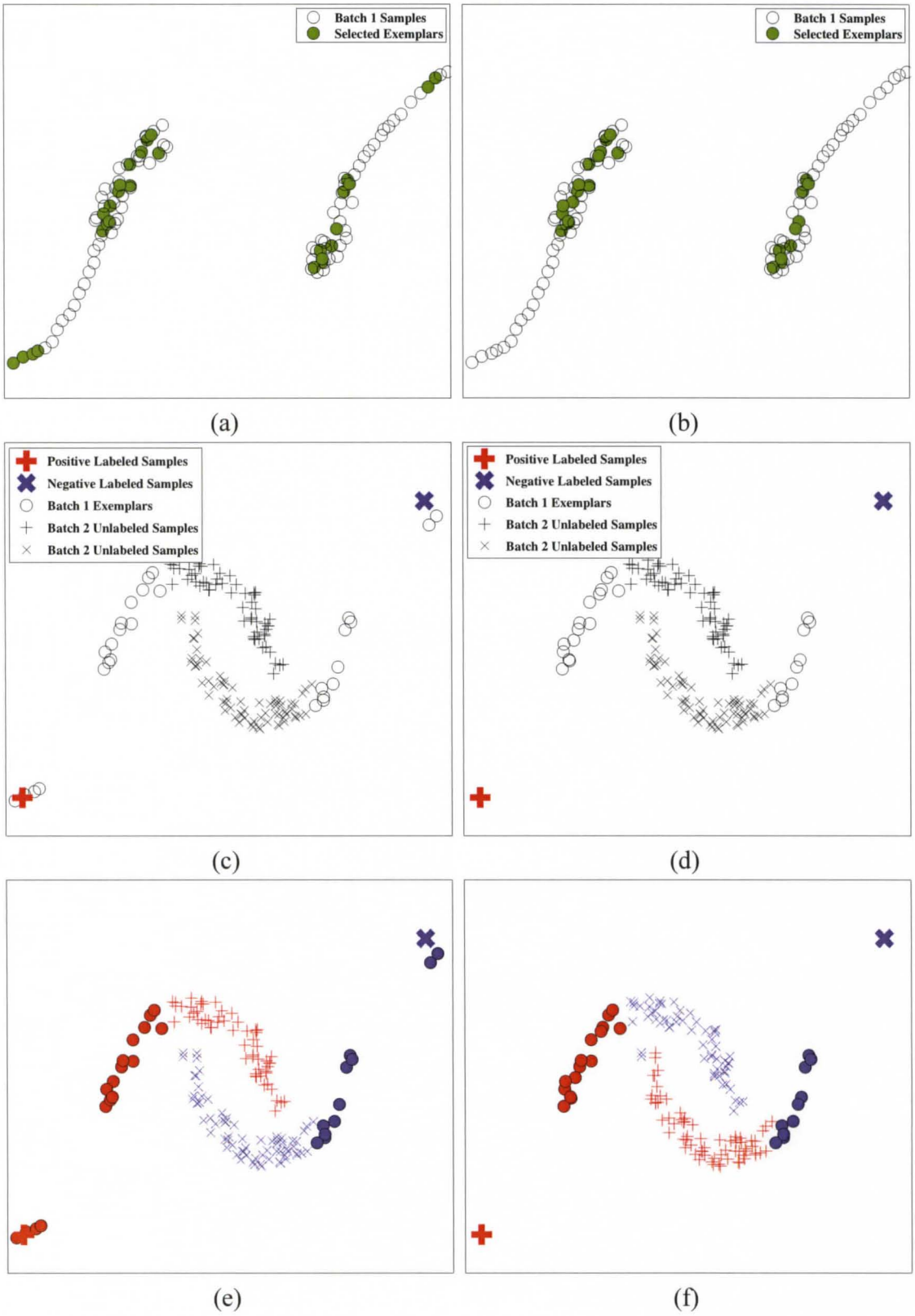


Figure IV.11: Importance of choosing exemplars with high similarity to labeled samples. Left Column: Output of QP-EXMP model. Right Column: Output if enforcement of similarity to labeled samples is ignored. (a-b) Selecting exemplars from batch $\mathcal{B}1$. (c-d) Using exemplars and batch $\mathcal{B}2$ as input for SSL. (e-f) Output of SSL.

exemplars from dense regions that might not belong to any existing know class. This scenario is illustrated in Fig.IV.12, where the batch $\mathcal{B}1$ has clusters of samples that belong to two classes. However, it includes only one labeled sample. Despite the lack of a clear path between the second unknown cluster and any labeled samples, the QP-EXMP selected some exemplars from the unknown cluster in anticipation that it might receive labeled samples in the future that clarifies its identity.

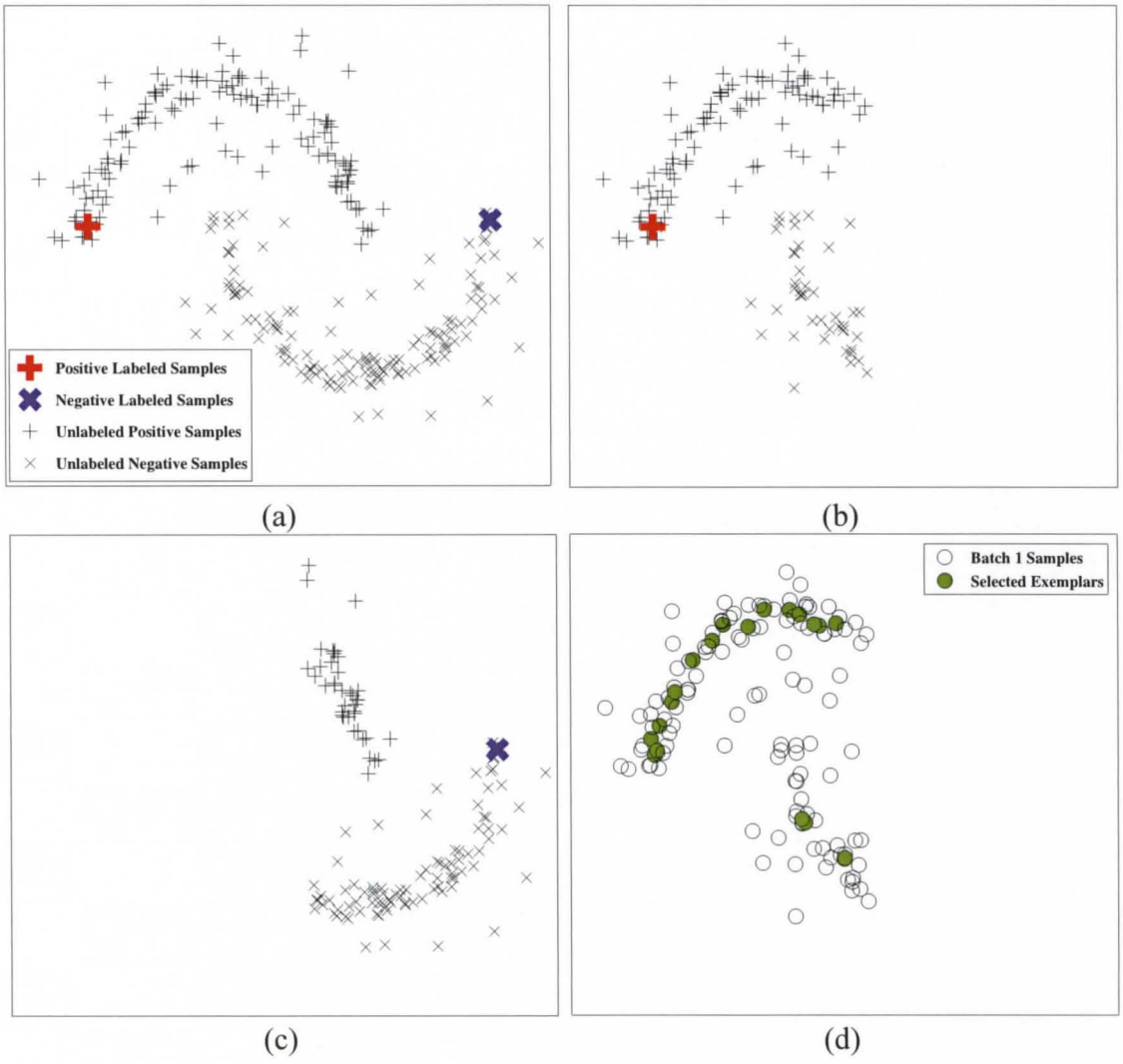
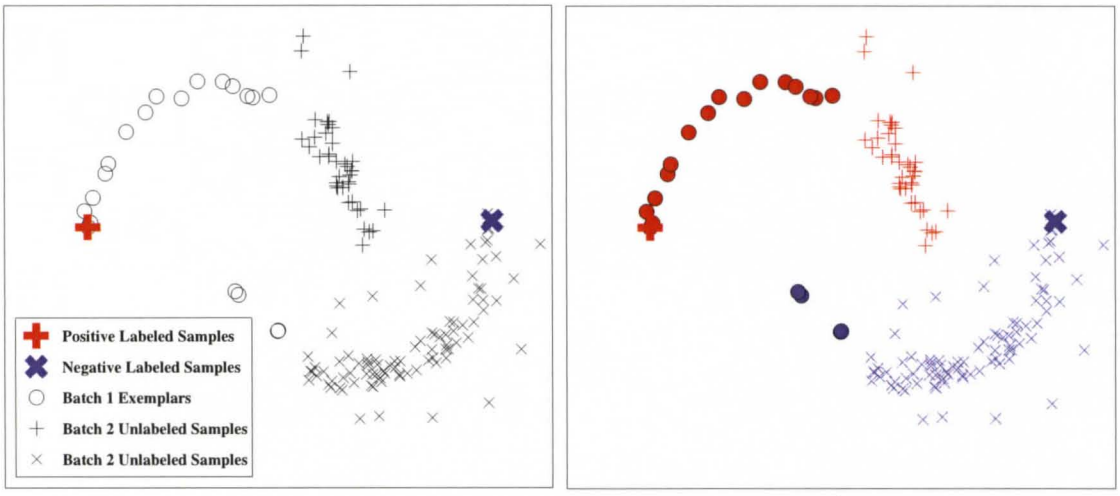


Figure IV.12: Exemplar selection from dense regions with no labeled samples. (a) Two moons data set. (b) Batch $\mathcal{B}1$ with two cluster from opposite classes and only one labeled sample. (c) Batch $\mathcal{B}2$. (d) Exemplars selected by QP-EXMP. (e) Exemplars and Batch 2 input to SSL. (f) Output of SSL.



(e) (f)

Figure IV.12: Continued: Exemplar selection from dense regions with no labeled samples. (a) Two moons data set. (b) Batch $\mathcal{B}1$ with two cluster from opposite classes and only one labeled sample. (c) Batch $\mathcal{B}2$. (d) Exemplars selected by QP-EXMP. (e) Exemplars and Batch 2 input to SSL. (f) Output of SSL.

IV.2.3 QP-EXMP Model Verification

Now that we have formulated and interpreted the QP-EXMP model, we proceed in this section by verifying if the proposed model actually works for its intended purpose which is selecting exemplars that preserve the inherent data properties for use with the QP-S^{3}VM during incremental/online learning.

The used experimental setup starts by selecting two batches, $\mathcal{B}1$ and $\mathcal{B}2$, from any given data set. Batch $\mathcal{B}1$ represents all previous data from a data stream, while $\mathcal{B}2$ represents a new arriving batch that needs to be labeled using the QP-S^{3}VM algorithm. The purpose of the experiment is to show that batch $\mathcal{B}1$ can be significantly summarized into a set small set of exemplars $\mathcal{E}1$ without affecting the final labeling of $\mathcal{B}2$. In other words, we need to show that:

$$\text{QP-S}^3\text{VM Labeling of } \mathcal{B}2 \text{ Given } \mathcal{B}1 \approx \text{QP-S}^3\text{VM Labeling of } \mathcal{B}2 \text{ Given } \mathcal{E}1$$

One issue that we considered while sampling batches for the experiment is the mutual relationship between $\mathcal{B}1$ and $\mathcal{B}2$ in the sense that using $\mathcal{B}1$ during the semi-supervised labeling of $\mathcal{B}2$ provides significant improvement over the case if only $\mathcal{B}2$ is used. This way we make sure that sure that any significant reduction in the number of exemplars $\mathcal{E}1$

while maintaining the labeling accuracy of $\mathcal{B}2$ is indeed due to the good performance of the QP-EXMP algorithm and not because $\mathcal{B}1$ is invaluable to the semi-supervised learning process. We use what we called the *Dissimilarity Indicator of* $\mathcal{B}1$ *versus* $\mathcal{B}2$ as a measure for the value of $\mathcal{B}1$ with respect to semi-supervised labeling of $\mathcal{B}2$,

$$\text{Dissimilarity Indicator of } \mathcal{B}1 \text{ versus } \mathcal{B}2 = \frac{\left|\text{Difference(QP-S}^3\text{VM Labeling of } \mathcal{B}2, \text{QP-S}^3\text{VM Labeling of } \mathcal{B}2 \text{ Given } \mathcal{B}1)\right|}{|\mathcal{B}2|} \times 100\%$$

For each selected $\mathcal{B}1$ and $\mathcal{B}2$, the experiment proceed by applying the QP-EXMP model to select an exemplar set $\mathcal{E}1$ and the comparative transductive accuracy of using $\mathcal{E}1$ versus using $\mathcal{B}1$ in the semi-supervised labeling of $\mathcal{B}2$ is calculated as follows,

$$\text{Transductive Accuracy of } \mathcal{B}1 \text{ Exemplars } \mathcal{E}1 = \frac{\text{Transductive Accuracy of QP-S}^3\text{VM Labeling of } \mathcal{B}2 \text{ Given } \mathcal{E}1}{\text{Transductive Accuracy of QP-S}^3\text{VM Labeling of } \mathcal{B}2 \text{ Given } \mathcal{B}1} \times 100\%$$

The process is repeated for several sizes of $\mathcal{E}1$ to examine the effectiveness of the QP-EXMP model in selecting smallest exemplar sets that maintains the transductive accuracy.

We performed the experiment on seven data sets spanning a wide spectrum of dimensionality. We used two types of kernels during the experiments, *Linear* and *RBF* as shown in the following figures. For each data set, batches $\mathcal{B}1$ and $\mathcal{B}2$ of size 500 samples each are randomly chosen and the QP-S^{3}VM algorithm is applied on both of them to get the labels for $\mathcal{B}2$. Next the QP-EXMP is applied on $\mathcal{B}1$ to extract the exemplars set $\mathcal{E}1$. $\mathcal{E}1$ is then fed back to the QP-S^{3}VM algorithm to get new labels for $\mathcal{B}2$ as described earlier. The process starts with initial size of the exemplar set as small as 0.5% of $|\mathcal{B}1|$ and keep increasing to see when satisfactory performance is achieved. We repeated this procedure on 200 pairs of batches. Samples results for three of the data sets are presented in Fig.(IV.13-IV.15). The rest of the experiments are provided in Appendix.

Examining the figures we see that the QP-EXMP model performs very well in terms of selecting very small exemplar sets, as low as 1% of $\mathcal{B}1$ in Fig.IV.15, while maintaining more than 95% of the comparative transductive accuracy with respect to $\mathcal{B}1$. The results also prove to be robust in the sense of being consistent even for batches with very large dissimilarity indicator, up to 95% in Fig.IV.13. The figures also show that the performance

is consistent with both *Linear* and *RBF* kernels which suggests expected good performance with other types of kernels.

Finally, through these experiments we have verified that the proposed QP-EXMP model is a proper model for the problem of exemplar selection and summarization of data streams for the purposes of incremental/online semi-supervised learning.

IV.3 Submodular Optimization of QP-EXMP (SUBMOD-EXMP)

The proposed QP-EXMP has time efficiency issues due to the quadratic programming problem involved. Similar to what we have presented in Chapter III, in this section we propose a submodular maximization problem that is equivalent to the QP-EXMP problem and that is highly scalable. By using submodular optimization to solve the QP-EXMP, the problem is transformed into a set function optimization problem where the goal is to select the best set of exemplars among all possible samples. We start by restating the QP-EXMP problem for the sake of quick reference.

Problem 7. *Quadratic Programming Exemplar Selection Algorithm for Incremental/Online QP-S^{3}VM (QP-EXMP):*

$$\underset{\mathbf{e}'=[e_1,\ldots,e_{|\mathcal{U}|}]}{\arg\min} \; \lambda_1 \mathbf{e}'\mathbf{K_{uu}}\mathbf{e} + \lambda_2 \mathbf{1}'_{|\mathcal{L}|}\mathbf{K}_{lu}(\mathbf{1}_{|\mathcal{U}|} - \mathbf{e}) + \lambda_3 \mathbf{1}'_{|\mathcal{U}|}\mathbf{K}_{uu}(\mathbf{1}_{|\mathcal{U}|} - \mathbf{e}) \tag{IV.12}$$

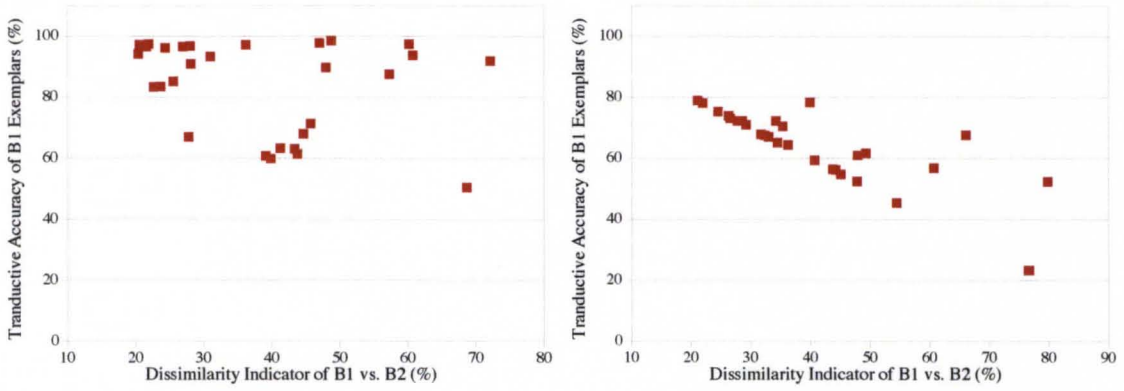
subject to

$$\mathbf{e}'\mathbf{1}_{|\mathcal{U}|} = \mathcal{M}_{\mathcal{E}}, \quad \mathbf{0} \leq \mathbf{e} \leq \mathbf{1}_{|\mathcal{U}|}. \tag{IV.13}$$

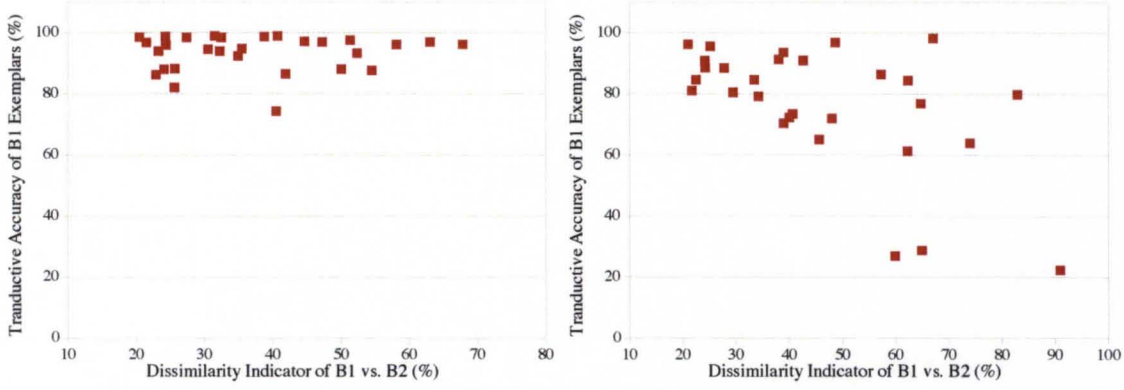
Note: Equation (IV.12) *can be rewritten in the standard quadratic programming form as follows:*

$$\underset{\mathbf{e}'=[e_1,\ldots,e_{|\mathcal{U}|}]}{\arg\min} \; \lambda_1 \mathbf{e}'\mathbf{K_{uu}}\mathbf{e} - (\lambda_2 \mathbf{1}'_{|\mathcal{L}|}\mathbf{K}_{lu} + \lambda_3 \mathbf{1}'_{|\mathcal{U}|}\mathbf{K}_{uu})\mathbf{e} \tag{IV.14}$$

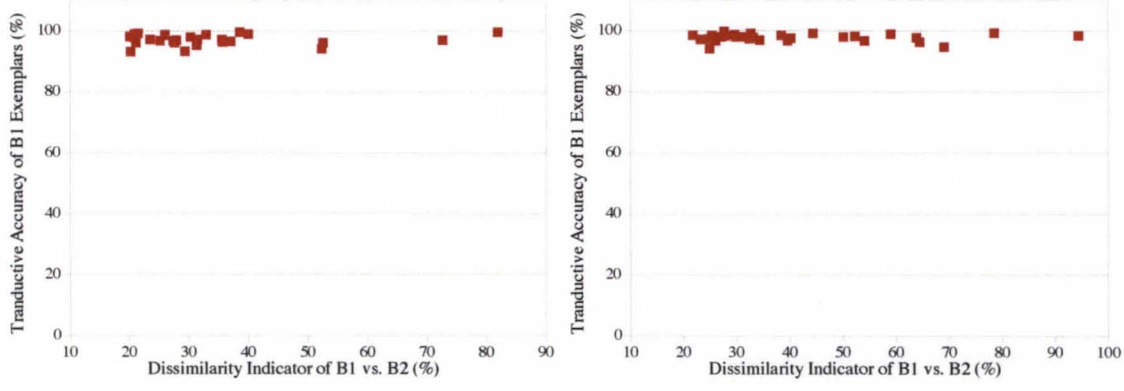
Following the same procedure used in Section III.2.2, we use the discrete representation of Eq.IV.14 and propose the following submodular maximization formulation of the QP-EXMP:



(a) RBF: $|\mathcal{E}1| = 0.01|\mathcal{B}1|$



(b) RBF: $|\mathcal{E}1| = 0.1|\mathcal{B}1|$



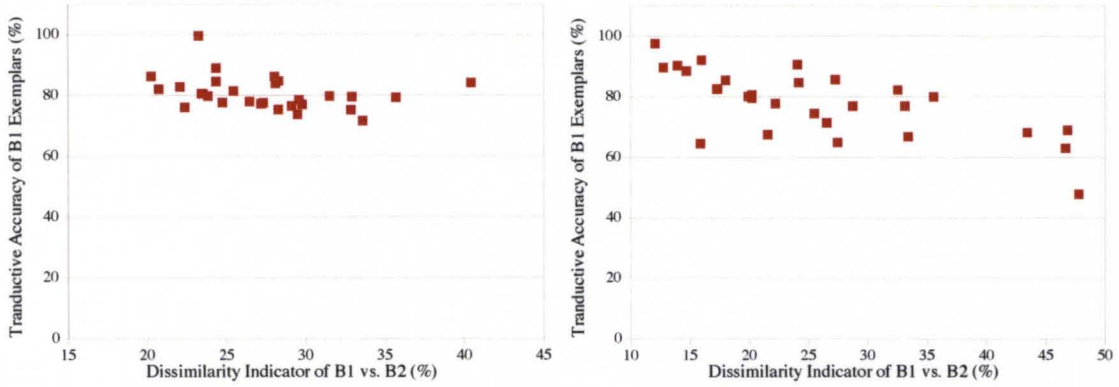
(c) RBF: $|\mathcal{E}1| = 0.2|\mathcal{B}1|$

(d) Linear: $|\mathcal{E}1| = 0.01|\mathcal{B}1|$

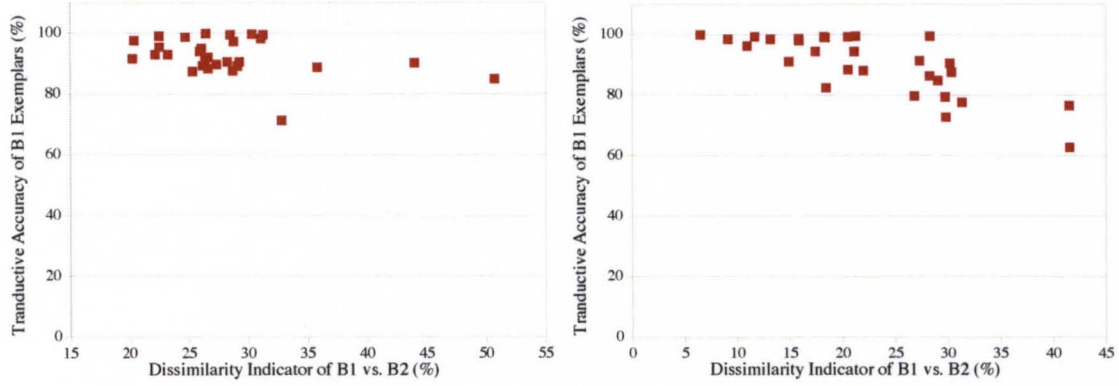
(e) Linear: $|\mathcal{E}1| = 0.2|\mathcal{B}1|$

(f) Linear: $|\mathcal{E}1| = 0.8|\mathcal{B}1|$

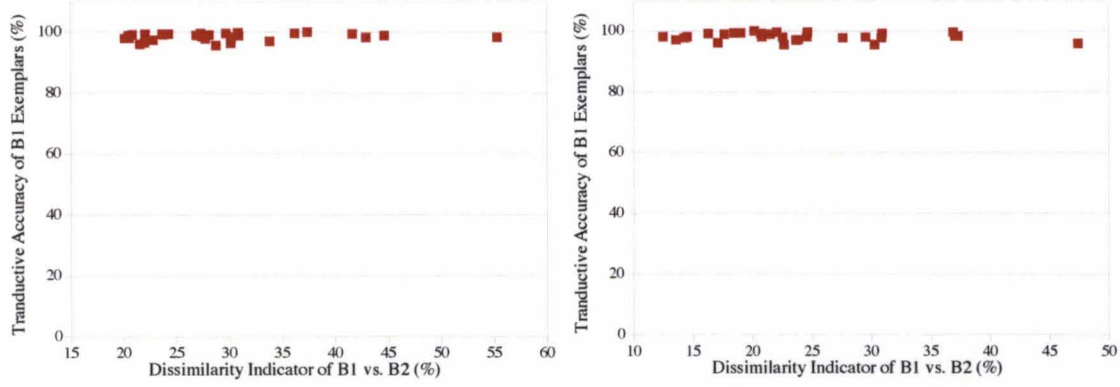
Figure IV.13: QP-EXMP model verification for the **Cov-Type** data set using linear and RBF kernels.



(a) Linear: $|\mathcal{E}1| = 0.005|\mathcal{B}1|$



(b) Linear: $|\mathcal{E}1| = 0.01|\mathcal{B}1|$



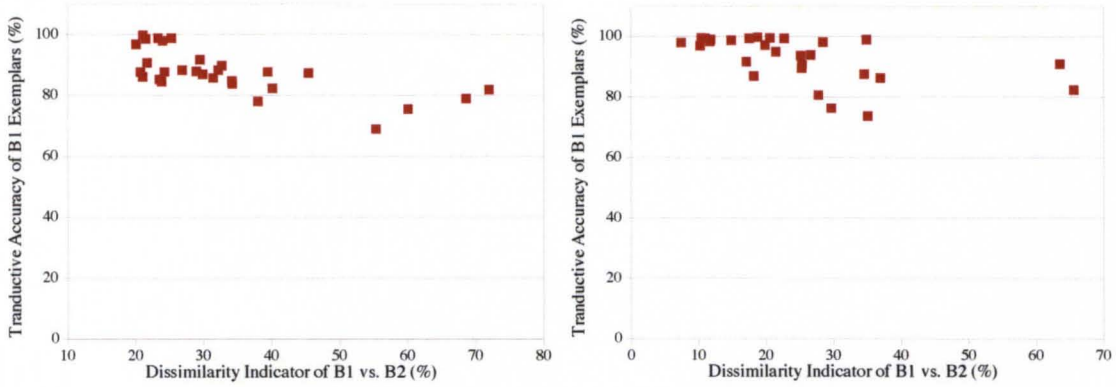
(c) Linear: $|\mathcal{E}1| = 0.03|\mathcal{B}1|$

(d) RBF: $|\mathcal{E}1| = 0.005|\mathcal{B}1|$

(e) RBF: $|\mathcal{E}1| = 0.01|\mathcal{B}1|$

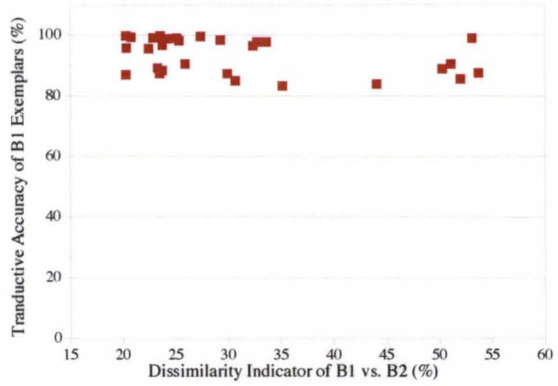
(f) RBF: $|\mathcal{E}1| = 0.03|\mathcal{B}1|$

Figure IV.14: QP-EXMP model verification for the **News20.Binary** data set using linear and RBF kernels.

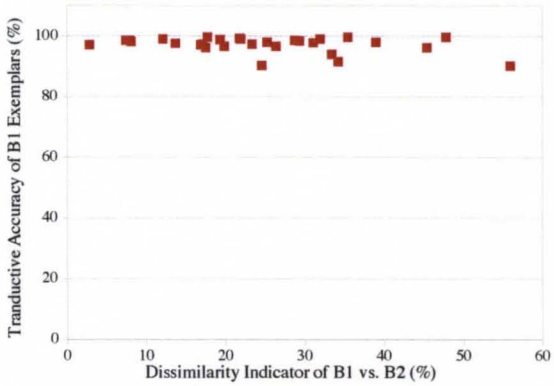


(a) Linear: $|\mathcal{E}1| = 0.005|\mathcal{B}1|$

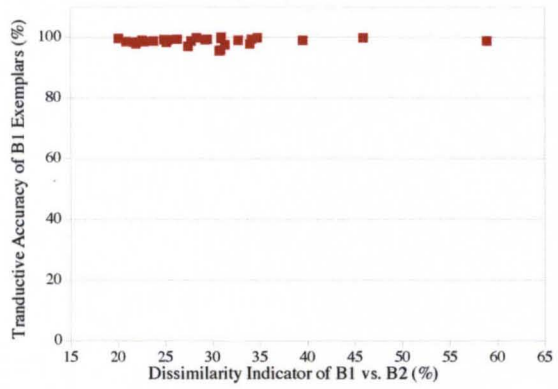
(d) RBF: $|\mathcal{E}1| = 0.005|\mathcal{B}1|$



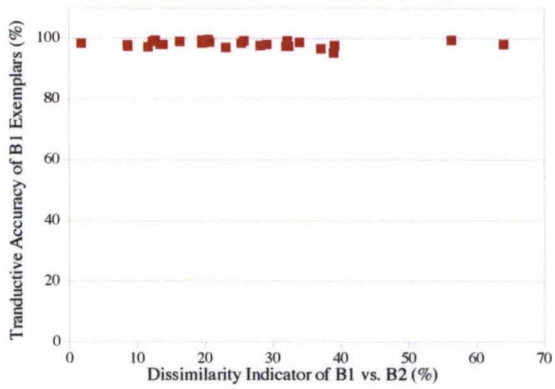
(b) Linear: $|\mathcal{E}1| = 0.0075|\mathcal{B}1|$



(e) RBF: $|\mathcal{E}1| = 0.0075|\mathcal{B}1|$



(c) Linear: $|\mathcal{E}1| = 0.01|\mathcal{B}1|$



(f) RBF: $|\mathcal{E}1| = 0.01|\mathcal{B}1|$

Figure IV.15: QP-EXMP model verification for the **Text** data set using linear and RBF kernels.

Problem 8. *Submodular formulation (SUBMOD-EXMP) of the QP-EXMP in Problem 6:*

$$\max_{|\mathcal{E}| \leq \mathcal{M}_\mathcal{E}} \mathcal{G}(\mathcal{E}) \tag{IV.15}$$

where

$$\begin{aligned} \mathcal{G}(\mathcal{E}) \;=\; & -\lambda_1 \sum_{j,j' \in \mathcal{E}} [\mathbf{K_{uu}}]_{j,j'} + \lambda_2 \sum_{j \in \mathcal{E}, i \in \mathcal{L}} [\mathbf{K_{lu}}]_{i,j} + \lambda_3 \sum_{j \in \mathcal{E}, j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{j,j'} \\ & \underbrace{+ \sum_{j,j' \in \mathcal{E}} \lambda_1 d \left[\delta_{j,j'} \left(4|\mathcal{U}| + 2\right) - 1\right]}_{\mathcal{T}}, \end{aligned} \tag{IV.16}$$

and $\mathcal{G}$ is a submodular set function defined on all subsets $\mathcal{E} \subset \mathcal{U}$ of unlabeled samples eligible to be chosen as exemplars, $0 \leq \mathbf{K}_{.,.} \leq d$, and $\delta_{j,j'} = 1$ for $j = j'$ and 0 otherwise.

Problem 8 maximizes the negative of the discrete version of QP-EXMP in Eq.IV.14. The constant term $\mathcal{T}$ is added to enforce the monotonicity and submodularity of the function $\mathcal{G}(\mathcal{E})$, see Theorem 5.

Theorem 5. *The set function $\mathcal{G}(\mathcal{E})$ in Problem 8 is monotone (non-decreasing), submodular, and $\mathcal{G}(\emptyset) = 0$.*

Proof. First, $\mathcal{G}(\emptyset) = 0$ follows directly from the definition in Eq.(IV.16) where all the summations are on elements in the set $\mathcal{E}$. Therefore if $\mathcal{E} = \emptyset$ then $\mathcal{G}(\emptyset) = 0$. Next we prove the *monotonicity property*. Using the definition of $\mathcal{G}(\mathcal{E})$, we can show that for any $m \in \mathcal{U}$ and $m \notin \mathcal{E}$, the increase in the objective value of $\mathcal{G}$ due to adding m is,

$$\begin{aligned} \mathcal{G}(\mathcal{E} \cup m) - \mathcal{G}(\mathcal{E}) \;=\; & -2\lambda_1 \sum_{j' \in \mathcal{E}} [\mathbf{K_{uu}}]_{m,j'} + \lambda_2 \sum_{i \in \mathcal{L}} [\mathbf{K_{lu}}]_{i,m} + \lambda_3 \sum_{j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{m,j'} \\ & - 2\lambda_1 |\mathcal{E}| d - \lambda_1 \Big([\mathbf{K_{uu}}]_{m,m} + d \Big) + 4\lambda_1 |\mathcal{U}| d + 2\lambda_1 d \end{aligned} \tag{IV.17}$$

For any kernel matrix $\mathbf{K}$, where $0 \leq \mathbf{K}_{i,j} \leq d$, since

$$\begin{aligned} & \lambda_2 \sum_{i \in \mathcal{L}} [\mathbf{K_{lu}}]_{i,m} \geq 0 \\ & \lambda_3 \sum_{j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{m,j'} \geq 0 \\ & 4\lambda_1 |\mathcal{U}| d - 2\lambda_1 \sum_{j' \in \mathcal{E}} \Big(d + [\mathbf{K_{uu}}]_{m,j'} \Big) \geq 0 \\ & 2\lambda_1 d - \lambda_1 \Big([\mathbf{K_{uu}}]_{m,m} + d \Big) \geq 0 \end{aligned} \tag{IV.18}$$

then,

$$\mathcal{G}(\mathcal{E} \cup m) - \mathcal{G}(\mathcal{E}) \geq 0.$$

Thus the monotonicity property of $\mathcal{G}(\mathcal{E})$ holds true.

Now we prove the *submodularity* of $\mathcal{G}(\mathcal{E})$ by assuming the set $\mathcal{F} = \{\mathcal{E} \cup q\}$, where $q \in \mathcal{U}$. Using the same set element m we used earlier, i.e. $m \in \mathcal{U}$ and $m \notin \mathcal{E}$, we need to show that adding m to the set $\mathcal{E}$ has more effect than adding it to the set $\mathcal{F}$ as stated in Def. 2. Since

$$\begin{aligned} \mathcal{G}(\mathcal{F}) \;=\; & -\lambda_1 \sum_{j,j' \in \{\mathcal{E} \cup q\}} [\mathbf{K_{uu}}]_{j,j'} + \lambda_2 \sum_{j \in \{\mathcal{E} \cup q\}, i \in \mathcal{L}} [\mathbf{K_{lu}}]_{i,j} + \lambda_3 \sum_{j \in \{\mathcal{E} \cup q\}, j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{j,j'} \\ & + \sum_{j,j' \in \{\mathcal{E} \cup q\}} \lambda_1 d \left[\delta_{j,j'} \left(4|\mathcal{U}| + 2\right) - 1\right] \end{aligned} \tag{IV.19}$$

then

$$\begin{aligned} \mathcal{G}(\mathcal{F} \cup m) - \mathcal{G}(\mathcal{F}) \;=\; & -2\lambda_1 \sum_{j' \in \mathcal{F}} [\mathbf{K_{uu}}]_{m,j'} + \lambda_2 \sum_{i \in \mathcal{L}} [\mathbf{K_{lu}}]_{i,m} + \lambda_3 \sum_{j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{m,j'} \\ & - 2\lambda_1 |\mathcal{F}| d - \lambda_1 \left([\mathbf{K_{uu}}]_{m,m} + d \right) + 4\lambda_1 |\mathcal{U}| d + 2\lambda_1 d \end{aligned} \tag{IV.20}$$

Therefore

$$[\mathcal{G}(\mathcal{E} \cup m) - \mathcal{G}(\mathcal{E})] - [\mathcal{G}(\mathcal{F} \cup m) - \mathcal{G}(\mathcal{F})] \;=\; 2\lambda_1 \left(d - [\mathbf{K_{uu}}]_{q,m} \right) \geq 0 \tag{IV.21}$$

Hence the set function $\mathcal{G}(\mathcal{E})$ is submodular. □

In summary, we have proved that the proposed SUBMOD-EXMP in Problem 8 is monotonically increasing and submodular. Therefore, we can use a simple efficient greedy algorithm to optimize it and find the exemplars as described in Algorithm 6.

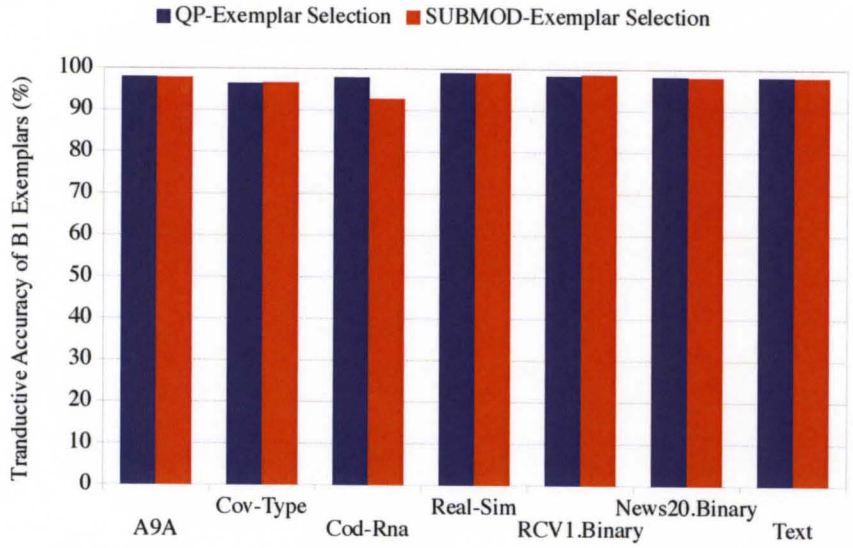
To illustrate the benefits of the proposed SUBMOD-EXMP, we repeated the experiments we conducted for the QP-EXMP model verification in Section IV.2.3, but this time we used both the QP-EXMP and SUBMOD-EXMP algorithms. Figure IV.16 presents a comparison between both algorithms in terms of the transductive accuracy and the computational efficiency. The results shows no statistically significant difference in the transductive accuracies of the QP-EXMP and SUBMOD-EXMP algorithms. Meanwhile, the SUBMOD-EXMP achieves several folds of computational efficiency improvements over the QP-EXMP algorithm as shown in the logarithmically scaled time comparison in Fig.IV.16.

Algorithm 6: Greedy Algorithm to Optimize the Proposed SUBMOD-EXMP

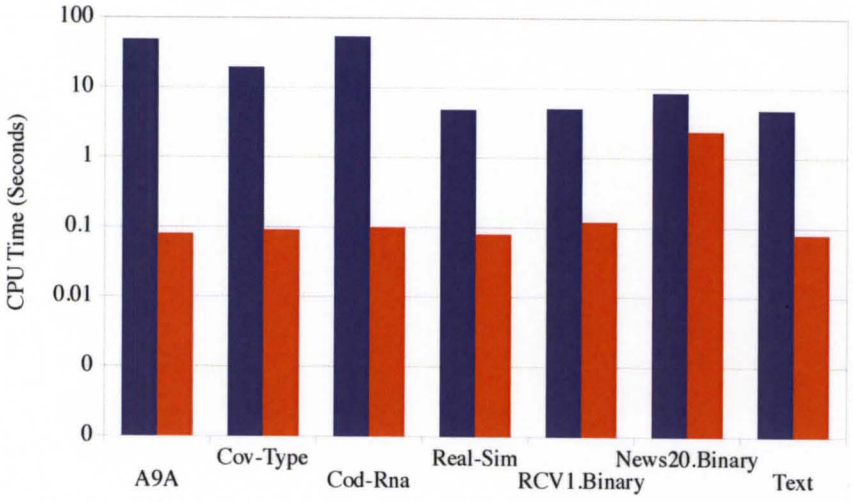
Input : $\mathcal{B} = \{\mathcal{L}, \mathcal{U}\}$: A data batch with labeled and unlabeled samples.
$\mathcal{M}$: Size of the exemplars set to be chosen.

Output: $\mathcal{E}^*$: The best set of exemplars given the size constraint.

begin
  Set $\mathcal{E}_0 := \phi$
  for $i := 1$ **to** $\mathcal{M}$ **do**
    foreach $m \in \mathcal{U} \backslash \mathcal{E}_{i-1}$ **do**
      MarginalBenefit $[m] := \mathcal{G}(\mathcal{E}_{i-1} \cup m) - \mathcal{G}(\mathcal{E}_{i-1})$ (Eq.(IV.17))
    end
    $m^* := \arg\max_m$ MarginalBenefit $[m]$
    $\mathcal{E}_i := \mathcal{E}_{i-1} \cup m^*$
  end
  $\mathcal{E}^* := \mathcal{E}_{\mathcal{M}}$
end



(a)



(b)

Figure IV.16: Comparing the QP-EXMP and SUBMOD-EXMP in terms of the transductive accuracy and computational efficiency.

IV.4 Proposed Incremental/Online SUBMOD-S^{3}VM

We propose an incremental/online SUBMOD-S^{3}VM algorithm to perform on streaming data. The idea is the same as we described for the incremental/online QP-S^{3}VM, where the standard SUBMOD-S^{3}VM is repetitively applied on the newly arriving data batch along with an exemplar representation of the data stream so far, see Fig.IV.17. For labeling a batch $\mathcal{B}_t$, we choose the stream representation to consist of $\mathcal{B}_{t-1}\mathcal{E}_{t-2}$, where $\mathcal{B}_{t-1}$ is the most recently processed batch and $\mathcal{E}_{t-2}$ is the set of exemplars summarizing the stream until the time instant $t-2$. We believe that keeping the last processed batch $\mathcal{B}_{t-1}$ rather summarizing it instantly, gives the algorithm the ability to perform aggressive summarization without the fear that the removed samples might be affect dramatically near future decisions.

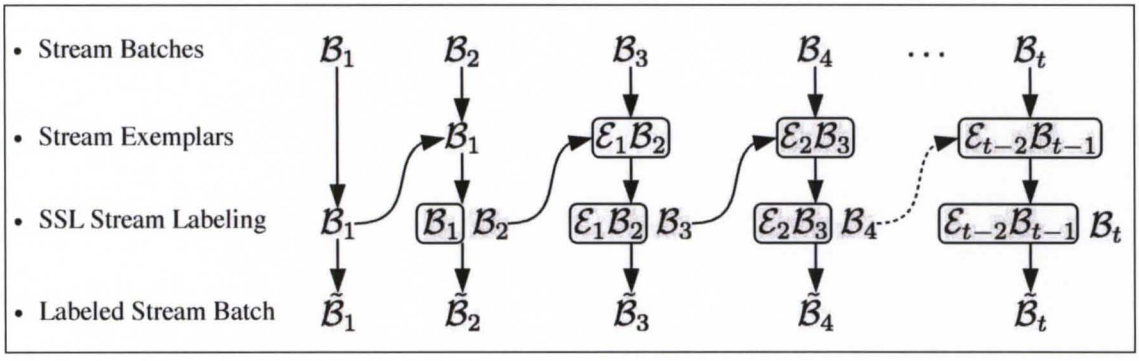


Figure IV.17: Illustration of the incremental/online SUBMOD-S^{3}VM learning procedure using the proposed SUBMOD-EXMP exemplar selection algorithm.

Figure IV.18 illustrates the procedure we propose to automatically decide the size of the exemplar representation of the stream. When a new batch $\mathcal{B}_{t+2}$ arrives, the full available stream representation $\mathcal{E}_t\mathcal{B}_{t+1}$ is used to label it using the SUBMOD-S^{3}VM and resulting in the labels $\tilde{\mathcal{B}}_{t+2}$. Now, to summarize the current stream representation $\mathcal{E}_t\mathcal{B}_{t+1}$, we iteratively use SUBMOD-EXMP with increasing cardinality of the selected exemplars to obtain a new exemplar set $\mathcal{E}_{t+1}$. Increasing the number of exemplars is aimed at achieving a good labeling of the new batch that is the same as the labeling obtained with the whole available stream representation, i.e. $\tilde{\mathcal{B}'}_{t+2} = \tilde{\mathcal{B}}_{t+2}$. Increasing the size of the exemplars stops if an exact match of $\tilde{\mathcal{B}'}_{t+2}$ is obtained or the storage limit amiable for stream representation is

exact match of $\tilde{\mathcal{B}'}_{t+2}$ is obtained or the storage limit amiable for stream representation is reached. In our experiments we set the initial size of the exemplars to be $0.1|\mathcal{E}_t\mathcal{B}_{t+1}|$ and is increased by the same amount at each iteration. We also assume that the exemplar set $\mathcal{E}_t$ at any time instant can not exceed in size a data batch. That is to say that we assume that the main memory has the size of three times a data batch.

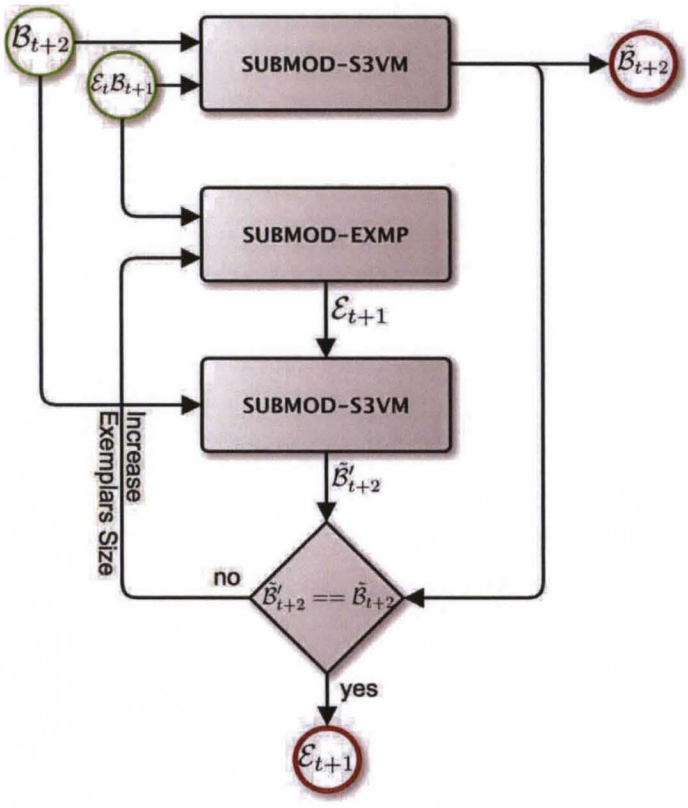


Figure IV.18: Diagram of the proposed incremental/online SUBMOD-S^{3}VM illustrating the use of SUBMOD-EXMP to summarize streaming data.

Finally, upon observation we found that achieving $\tilde{\mathcal{B}'}_{t+2} = \tilde{\mathcal{B}}_{t+2}$ is quite hard and results in unnecessary large storage of exemplars. On the other hand, through experiments we have seen that a tolerance of 5% difference between $\tilde{\mathcal{B}'}_{t+2}$ and $\tilde{\mathcal{B}}_{t+2}$ can achieve almost the same results but with much faster performance and much smaller memory storage. In the experiments we compare these two *strict* and *tolerant* procedures as well as the batch learning where all the data starting at the beginning of the stream is used to label the current batch.

IV.5 Experimental Results

In this section we present the experiments we conducted to examine the performance of the proposed incremental/online SUBMOD-S^{3}VM algorithm. We evaluate the performance of the proposed work in terms of transductive accuracy, time complexity, and memory requirements. As a benchmark for optimal accuracy, we provide the transductive accuracy obtained by batch based learning where all samples in the data stream to the current moment are used to predict the labels for the current batch. Moreover, we compare both procedures, *strict* and *tolerant*, for determining the exemplar size used by SUBMOD-EXMP as described in the previous section. Finally, the exemplars provided by the SUBMOD-EXMP are not perfect and at each time step, using such exemplars rather than the whole data batch will result in some misclassifications. We report these misclassification as the *cost of summarization*, as we believe it is important how such errors might affect the learning process on the long run. Table IV.1 provides the details of the used data sets.

TABLE IV.1

Data sets used in the experiments [6, 38].

Data set	Samples	Features	Labeled	Batch Size
News20.Binary	19,954	1,355,191	50	1,000
CCAT	23,149	47,236	30	1,000
GCAT	23,149	47,236	30	1,000
Aut-Avn	70,166	20,702	9	2,000
Real-Sim	72,201	20,958	8	2,000
RCV1.Binary	697,641	47,236	35	10,000
KDD-99	1,500,000	122	9	8,000

IV.5.1 Experimental Results for the Incremental Learning Scenario

In this section we provide the experiments conducted under the incremental learning scenario, where data batches presented to the algorithm sequentially are partially labeled. In other words, if there exists labeled samples in the data batch, the algorithm see them

instantly and are used in the process of labeling the rest of unlabeled samples. This scenario is more suitable for learning from enormous partially labeled data that can not fit into the main memory and therefore is broken into manageable batches that can be processed sequentially.

Examining Fig.IV.19-IV.25 we observe that there no statistically significant difference between the transductive accuracy of the incremental SUBMOD-S^3VM algorithm and the batch learning algorithm. As the time complexity graphs show the batch learning has linear time complexity in the size of the data, the incremental SUBMOD-S^3VM shows constant, on average, time complexity. The fluctuations in the time complexity around the constant average are due the changes in the size of the stored exemplar set as can be seen in the subplots (b) and (c) in Fig.IV.19-IV.25.

One important observation is that the *cost of summarization* tend to decrease for all data sets over time. This actually indicates that the exemplars selected over time are getting saturated and in fact they represent correctly the core characteristics of the data stream. Moreover, looking at the *cost of summarization* curves for all the data sets, we noticed that it does not exceed 5% of the batch size, and that only occurs for the early data batches at beginning of the learning process. This observation inspired the *tolerant* procedure described earlier. In fact comparing the performance of the standard (*strict*) and the *tolerant* procedures, we see that the *tolerant* procedure preserves the transductive accuracy while achieving much butter time complexity and memory storage.

Figure IV.25 shows the transductive accuracy produced by the incremental SUBMOD-S^3VM with different batch sizes. The instability of the performance for the small batch size and the improvement observed for the large batch size suggests that for the small batches the stored exemplar are too few to hold a stable comprehensive representation of the stream. Therefore, over time as further reductions are performed some important exemplars are removed which affects the performance. A larger batch size allows a richer representation to be stored (remember that we use a stream representation equal to the batch size) and thus the transductive accuracy becomes more stable.

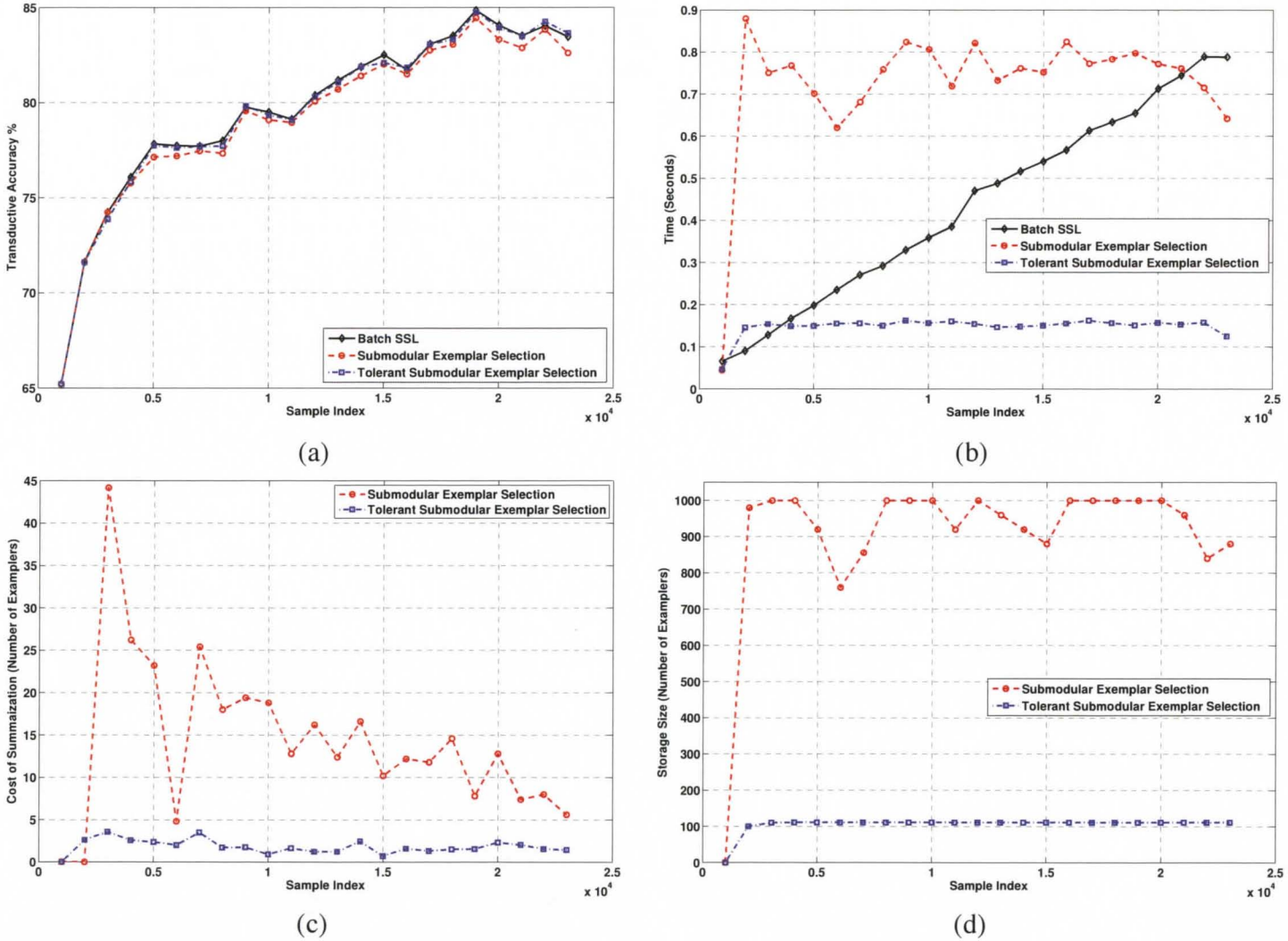


Figure IV.19: Incremental SUBMOD-S^{3}VM results for the **GCAT** data set. (a) Transductive accuracy. (b) Time complexity. (c) Cost of summarization. (d) Storage size.

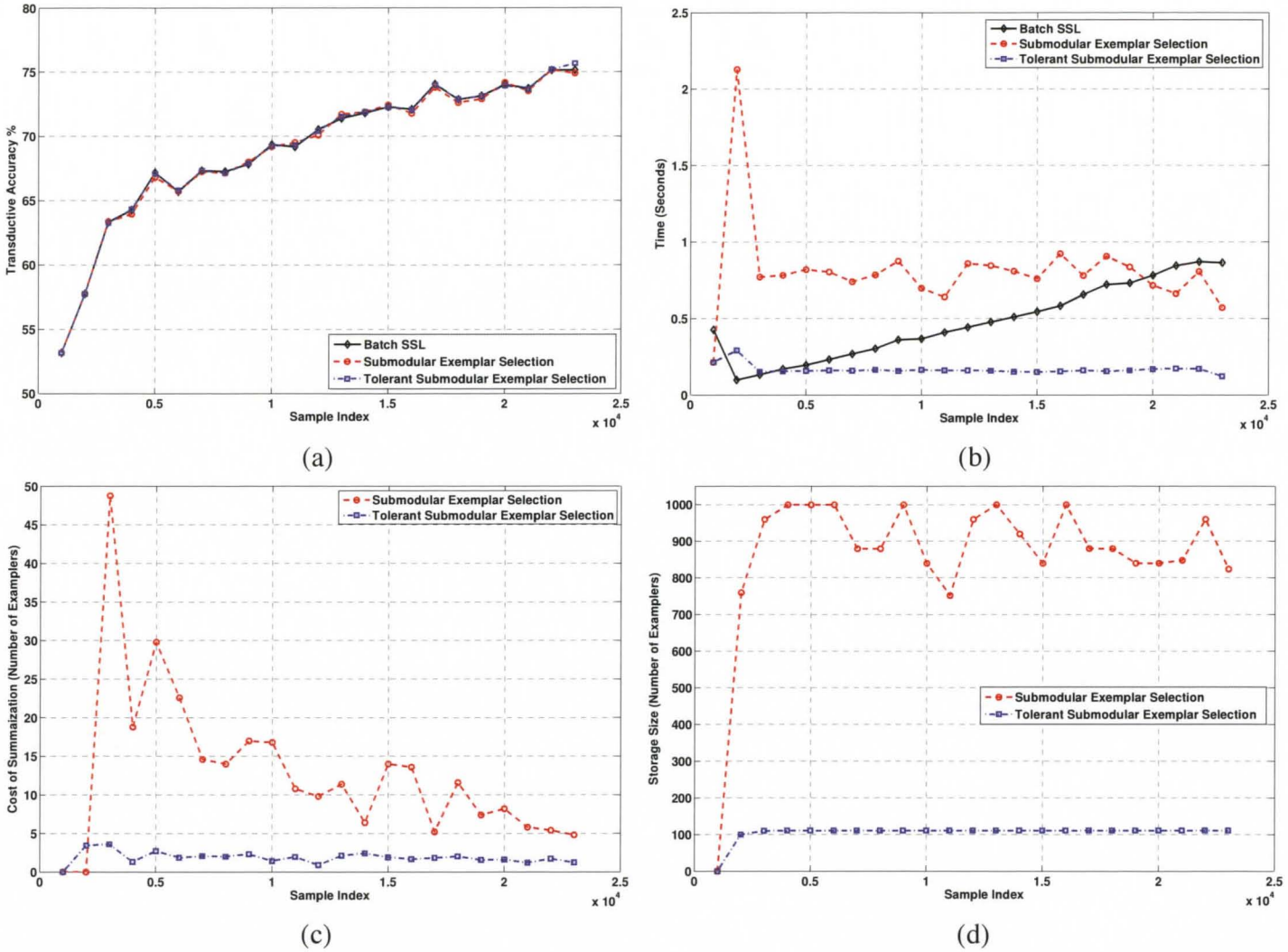


Figure IV.20: Incremental SUBMOD-S^{3}VM results for the **CCAT** data set. (a) Transductive accuracy. (b) Time complexity. (c) Cost of summarization. (d) Storage size.

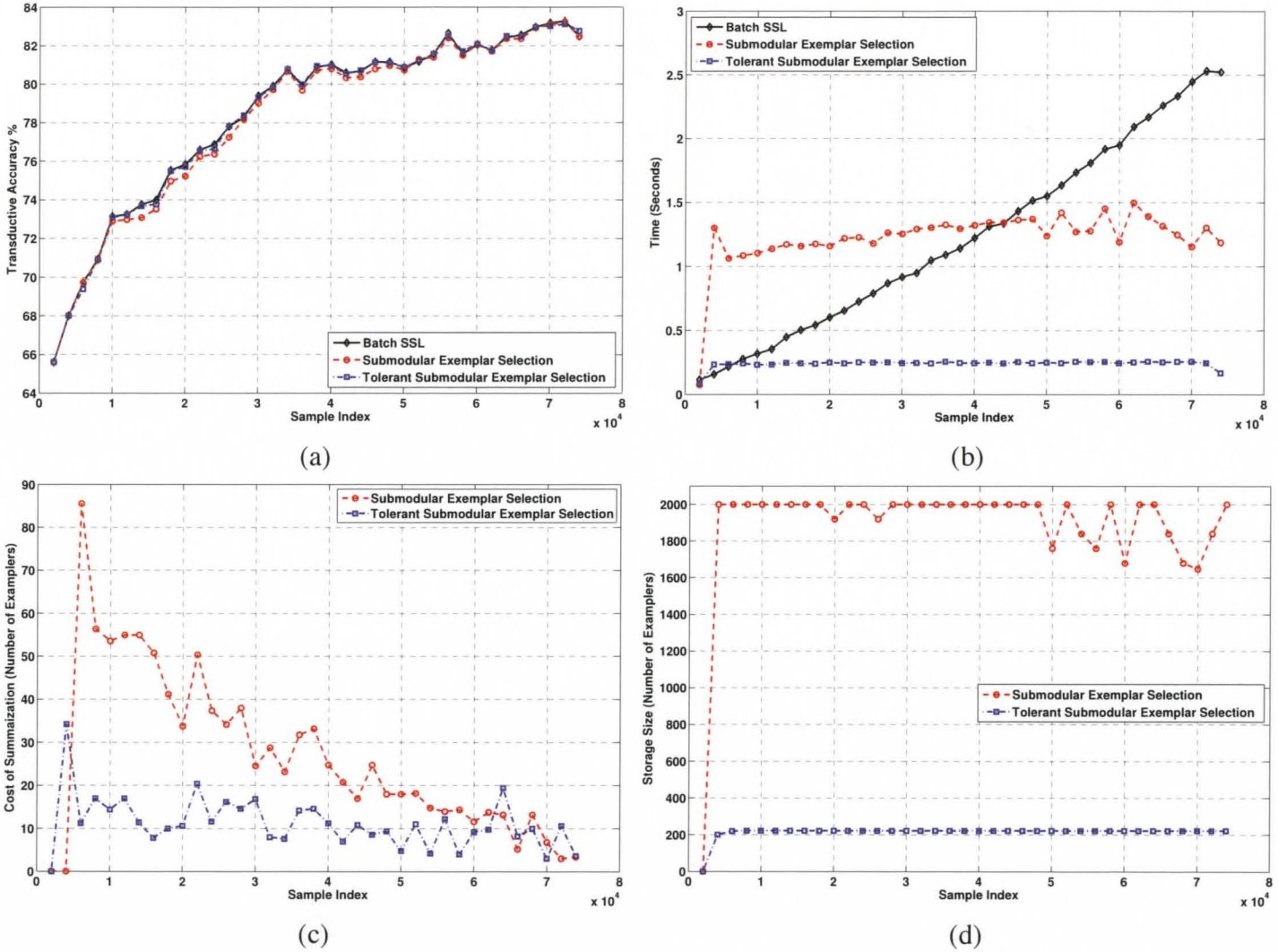


Figure IV.21: Incremental SUBMOD-S^{3}VM results for the **Real-Sim** data set. (a) Transductive accuracy. (b) Time complexity. (c) Cost of summarization. (d) Storage size.

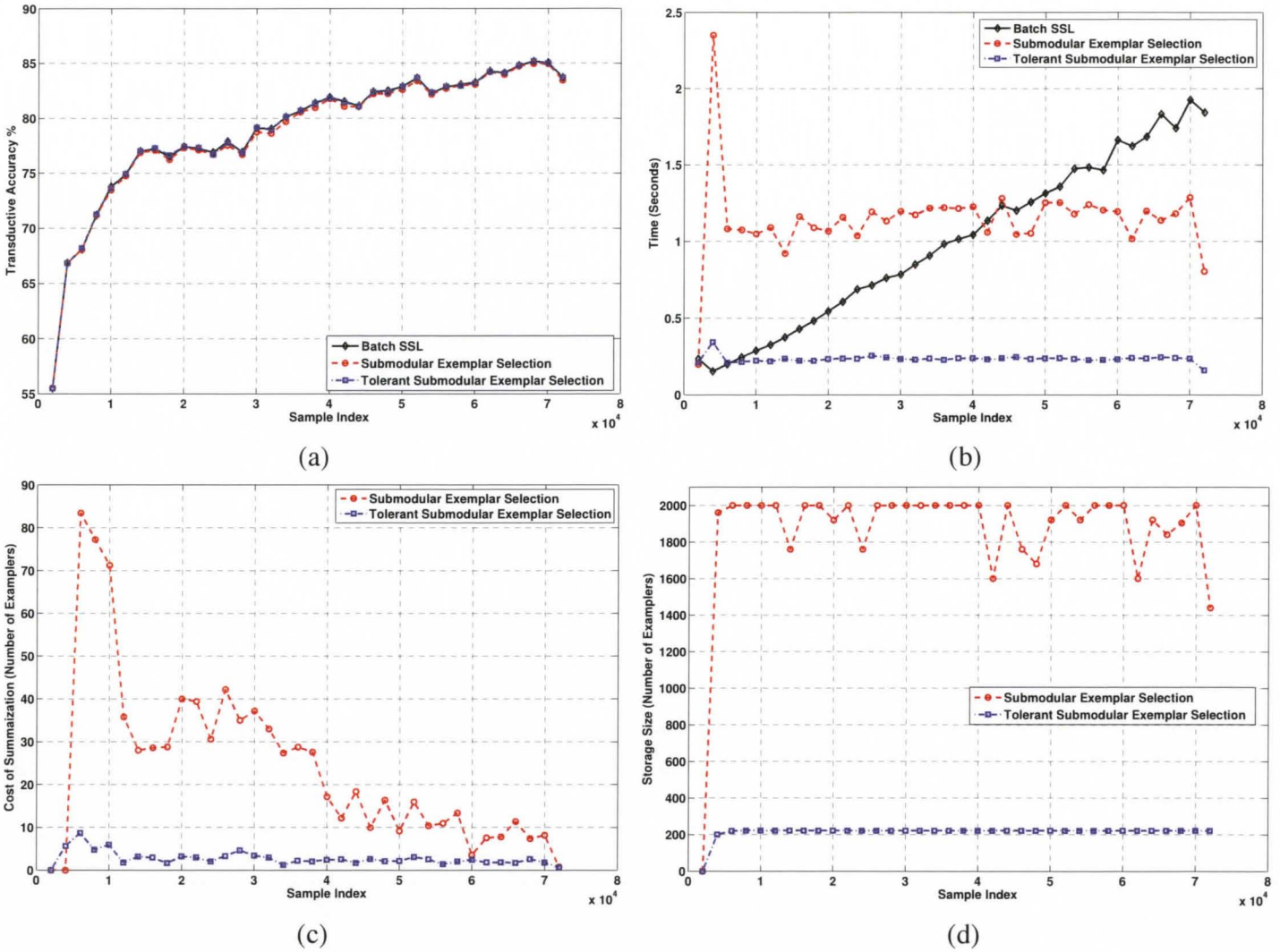


Figure IV.22: Incremental SUBMOD-S^{3}VM results for the **Aut-Avn** data set. (a) Transductive accuracy. (b) Time complexity. (c) Cost of summarization. (d) Storage size.

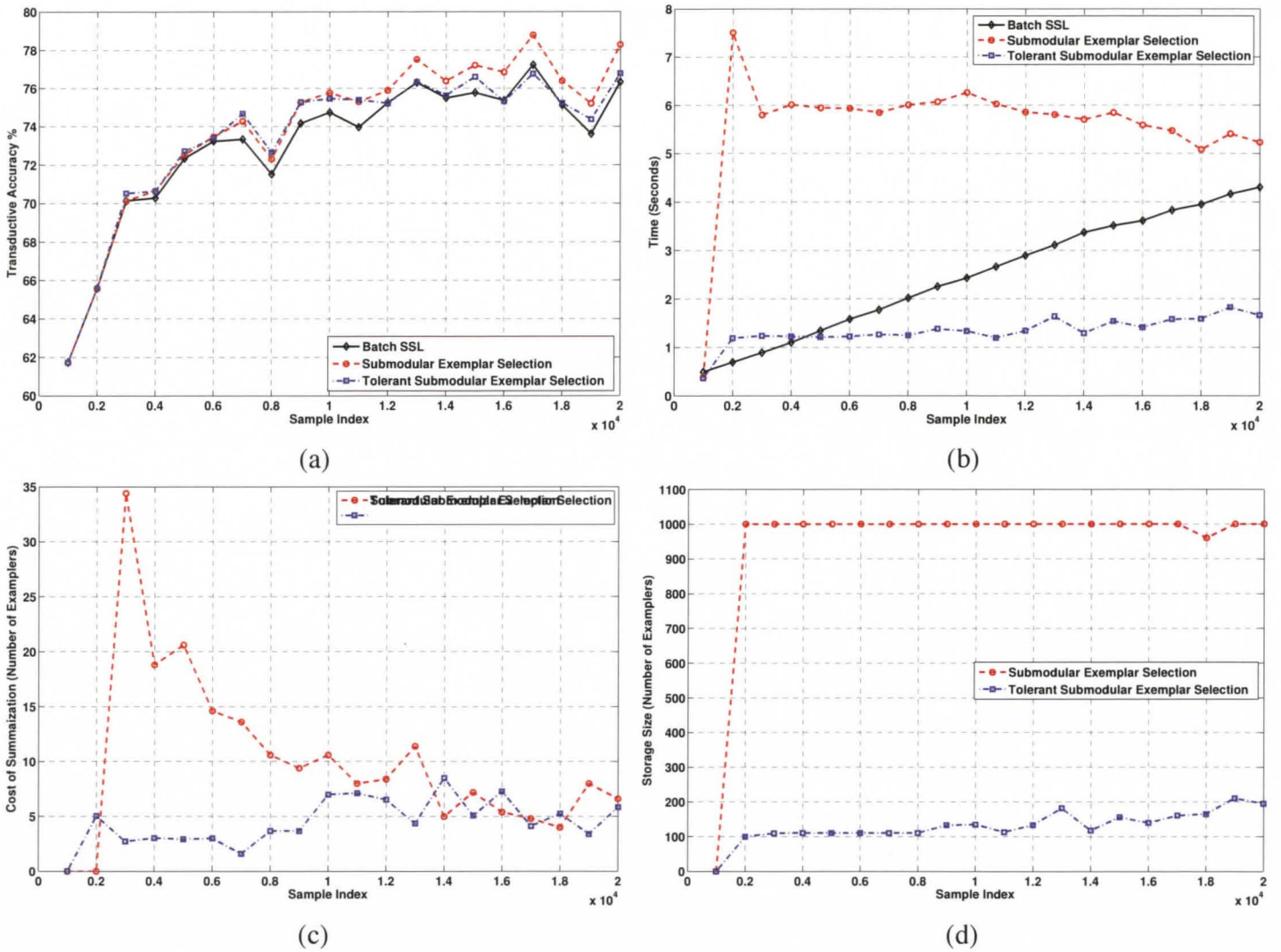


Figure IV.23: Incremental SUBMOD-S^{3}VM results for the **News20.Binary** data set. (a) Transductive accuracy. (b) Time complexity. (c) Cost of summarization. (d) Storage size.

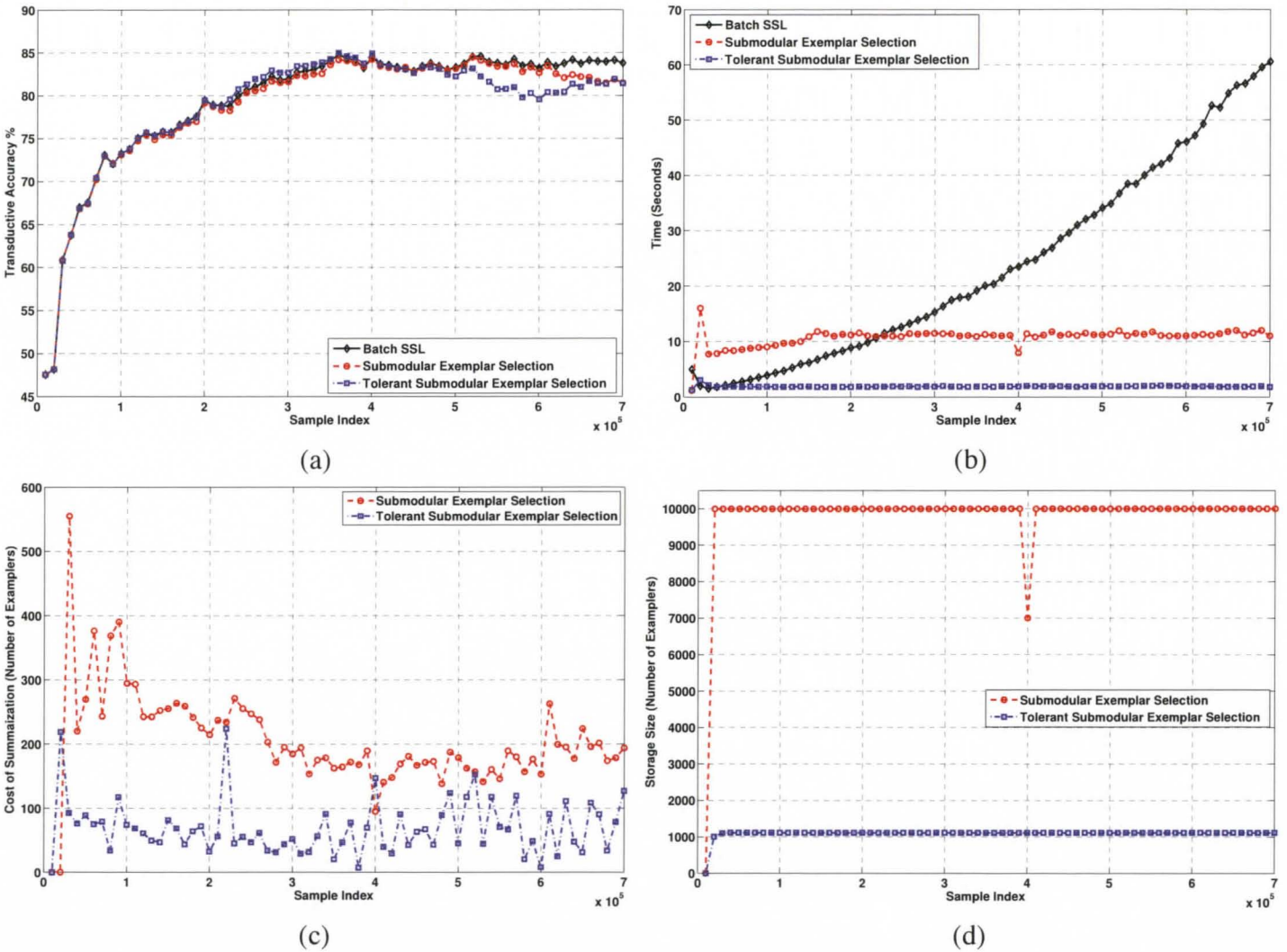


Figure IV.24: Incremental SUBMOD-S^{3}VM results for the **RCV1.Binary** data set. (a) Transductive accuracy. (b) Time complexity. (c) Cost of summarization. (d) Storage size.

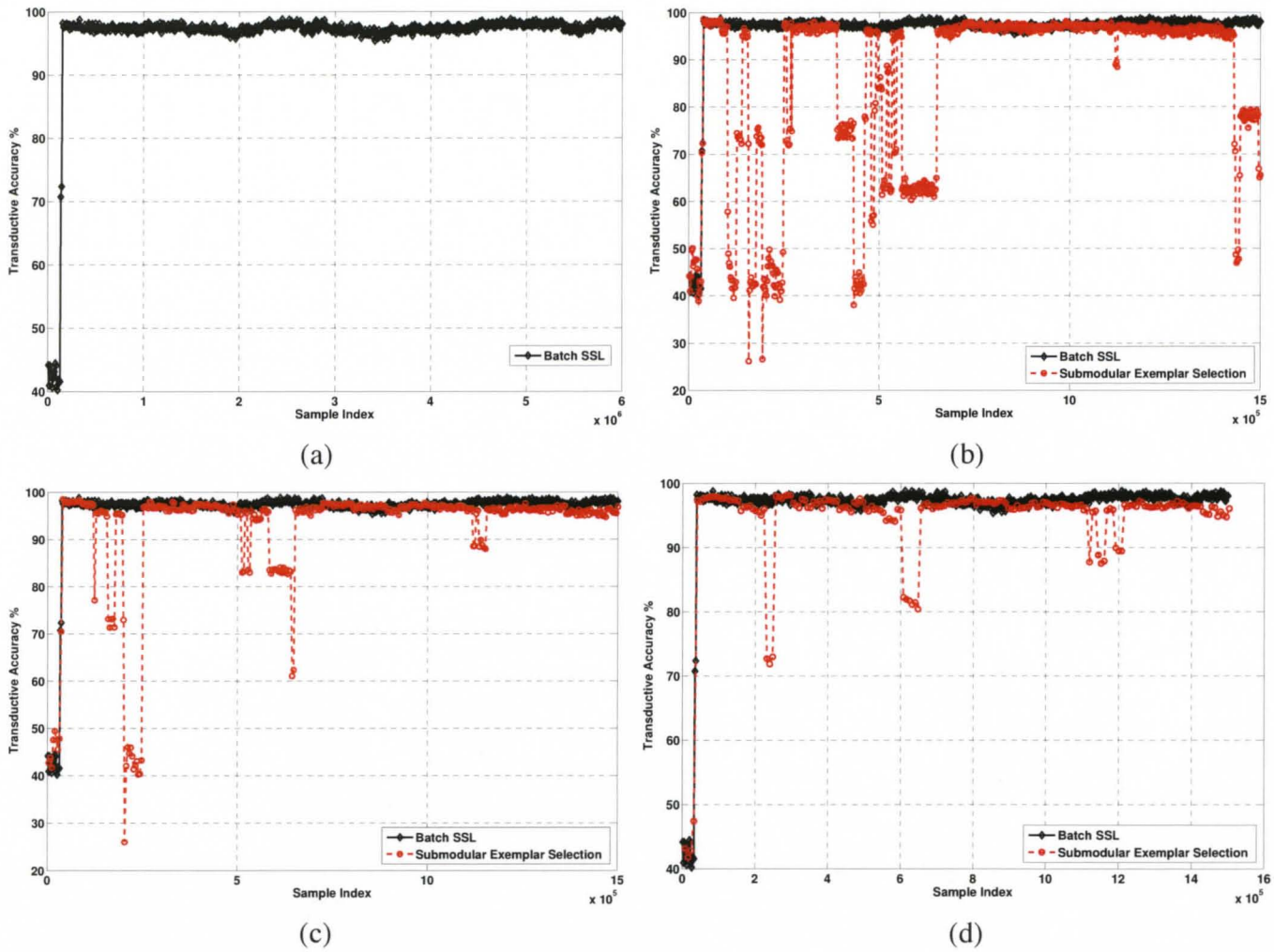


Figure IV.25: Incremental SUBMOD-S^{3}VM results for the **KDD-99** data set. (a) Batch based transductive accuracy. (b) Transductive accuracy using batches of size 2000 samples. (c) Transductive accuracy using batches of size 4000 samples. (d) Transductive accuracy using batches of size 8000 samples.

IV.5.2 Experimental Results for the Online Learning Scenario

In this section we provide the experiments conducted under the online learning scenario, where data batches presented to the algorithm sequentially are always unlabeled and thus have to be labeled. Occasional engagement from the environment is provided in the form for labels for unlabeled samples, however this occurs only as feedback after the algorithm has already finished labeling. This scenario is more suitable for learning from data generated over time.

Figure IV.26-IV.30 show the transductive accuracies of the online SUBMOD-S^{3}VM algorithm, the online batch learning algorithm, and the incremental batch learning algorithm from previous section. The incremental batch learning is observed to initially outperform the other online approaches and the difference diminishes as the stream progresses. This is natural as the incremental batch learning have access to the labeled samples in each new batch, and it uses these labels to find the best labels for the unlabeled samples. However, the online approaches do not have access to the labels and thus their performance is expected to differ in the beginning of the stream. As more batches are processed, the online SUBMOD-S^{3}VM become robust enough to perform as well as the incremental approach.

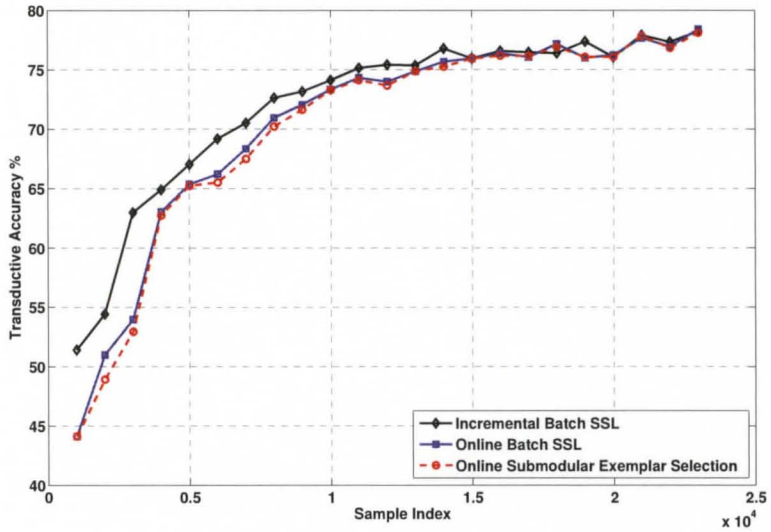


Figure IV.26: Online SUBMOD-S^{3}VM results for the **CCAT** data set.

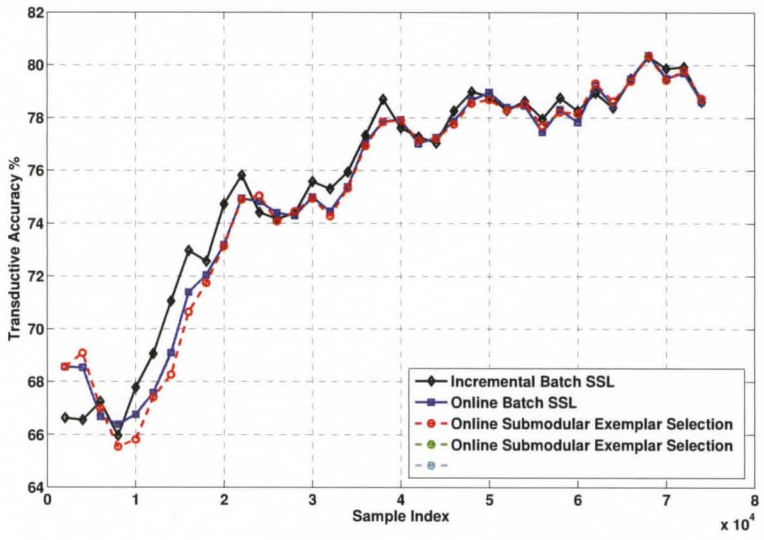


Figure IV.27: Online SUBMOD-S^{3}VM results for the **Real-Sim** data set.

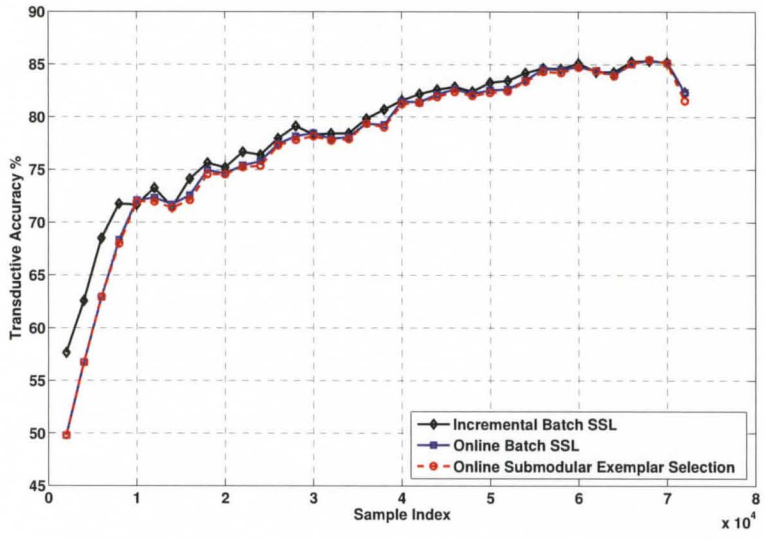


Figure IV.28: Online SUBMOD-S^{3}VM results for the **Aut-Avn** data set.

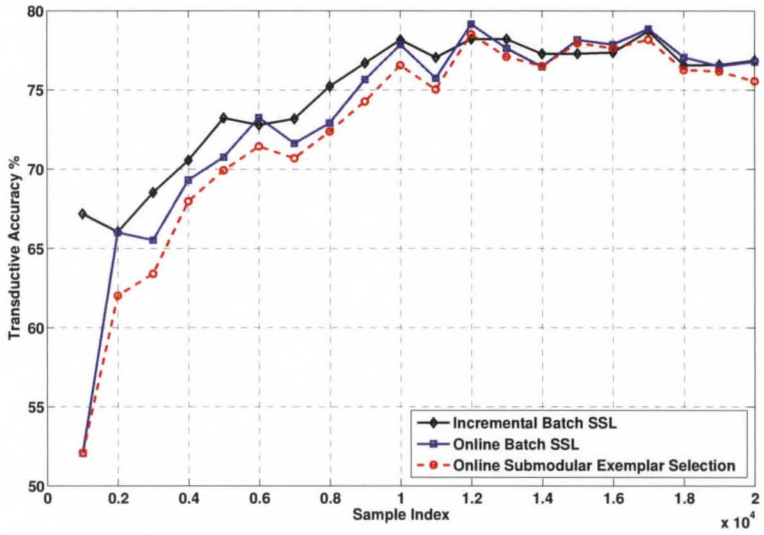


Figure IV.29: Online SUBMOD-S^{3}VM results for the **News20.Binary** data set.

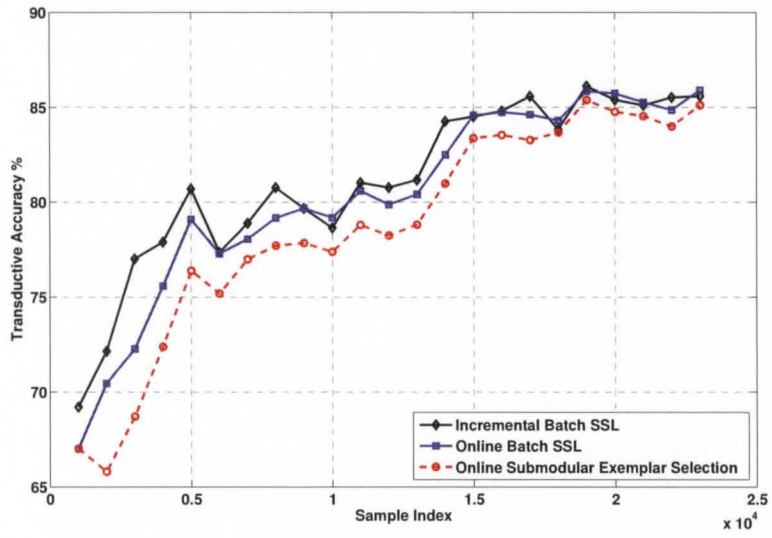


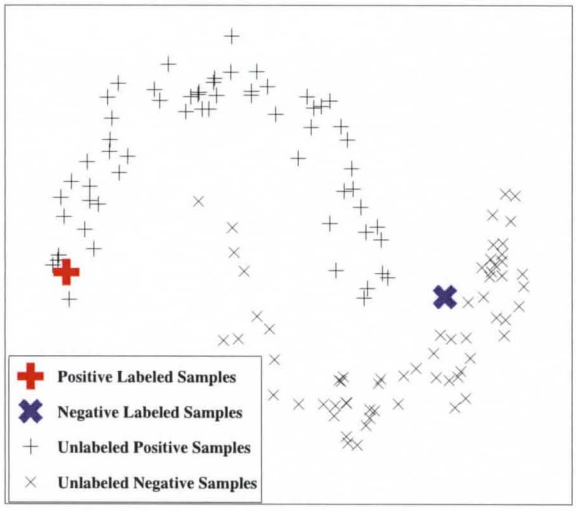
Figure IV.30: Online SUBMOD-S^{3}VM results for the **GCAT** data set.

IV.6 SUBMOD-EXMP Extension for Inter-batch Dependence

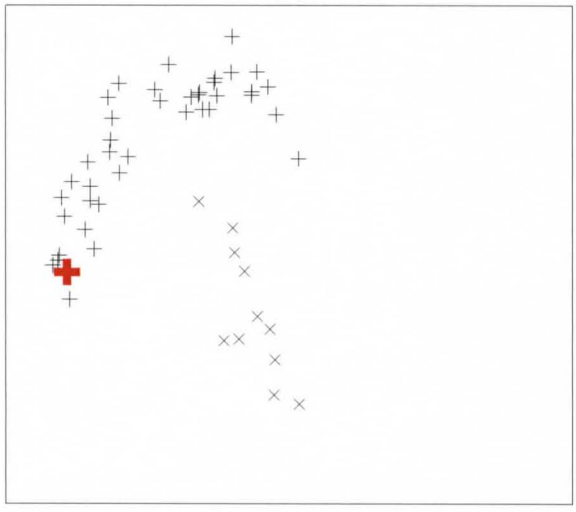
In the proposed SUBMOD–EXMP algorithm, for given two batches $\mathcal{B}_1$ and $\mathcal{B}_2$, and $\mathcal{B}_1$ is required to be reduced via exemplar selection into $\mathcal{E}_1$, the SUBMOD–EXMP uses only to select the exemplars. $\mathcal{B}_2$ on the other hand is only used during the procedure of deciding the proper size for $\mathcal{E}_1$. In this section we propose an improvement over the SUBMOD-EXMP model where $\mathcal{B}_2$ is involved as well in the process of selecting $\mathcal{E}_1$ as it include up-to-date information about the dense regions of the data space that might not be clear in $\mathcal{B}_1$ and thus might get ignored during the exemplar selection process.

Figure IV.31 depicts a special case of the Two Moons data set, where the batch $\mathcal{B}_1$ includes two clusters belonging to two different classes and one of them is much denser than the other. Another disadvantage against the lighter cluster is that it does not have a labeled sample to attract attention to it, whereas the dense cluster has a labeled sample which gives it double the attention from the SUBMOD–EXMP algorithm. As can be seen in Fig.IV.32(Left Column), the SUBMOD-EXMP does not include $\mathcal{B}_2$ in the decision to select $\mathcal{E}_1$ which results in selecting all of $\mathcal{E}_1$ from the dense cluster while considering the light cluster as noise. Had $\mathcal{B}_2$ been included in the decision, it would become clear that the light cluster in $\mathcal{B}_1$ is in fact a primitive sign for a much denser region in $\mathcal{B}_2$ and thus few exemplars would be chosen from the light cluster, see Fig.IV.32(Right Column).

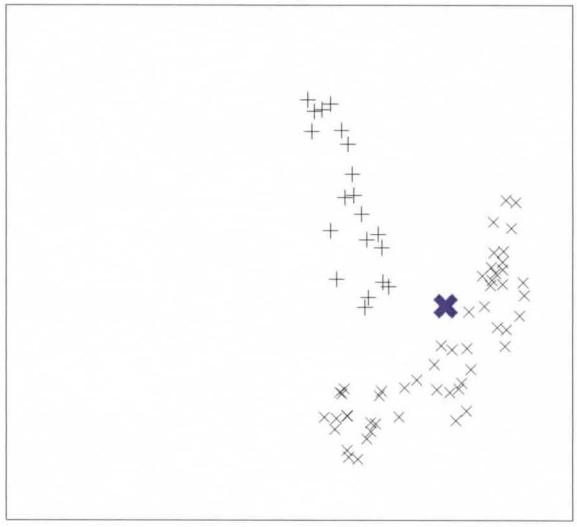
We tested this idea on reveal of the data sets we used earlier and the results and presented in Fig.IV.33-IV.35. It is clear that the idea is valid and that in some cases, such as in Fig.IV.33, it provides obvious improvement over the standard SUBMOD-EXMP and in the rest of data sets we do not decay in the performance due to the use of the inter-batch dependence idea.



(a)



(b)



(c)

Figure IV.31: (a) Two moons partially labeled data set. (b) Batch $\mathcal{B}1$. (c) Batch $\mathcal{B}2$.

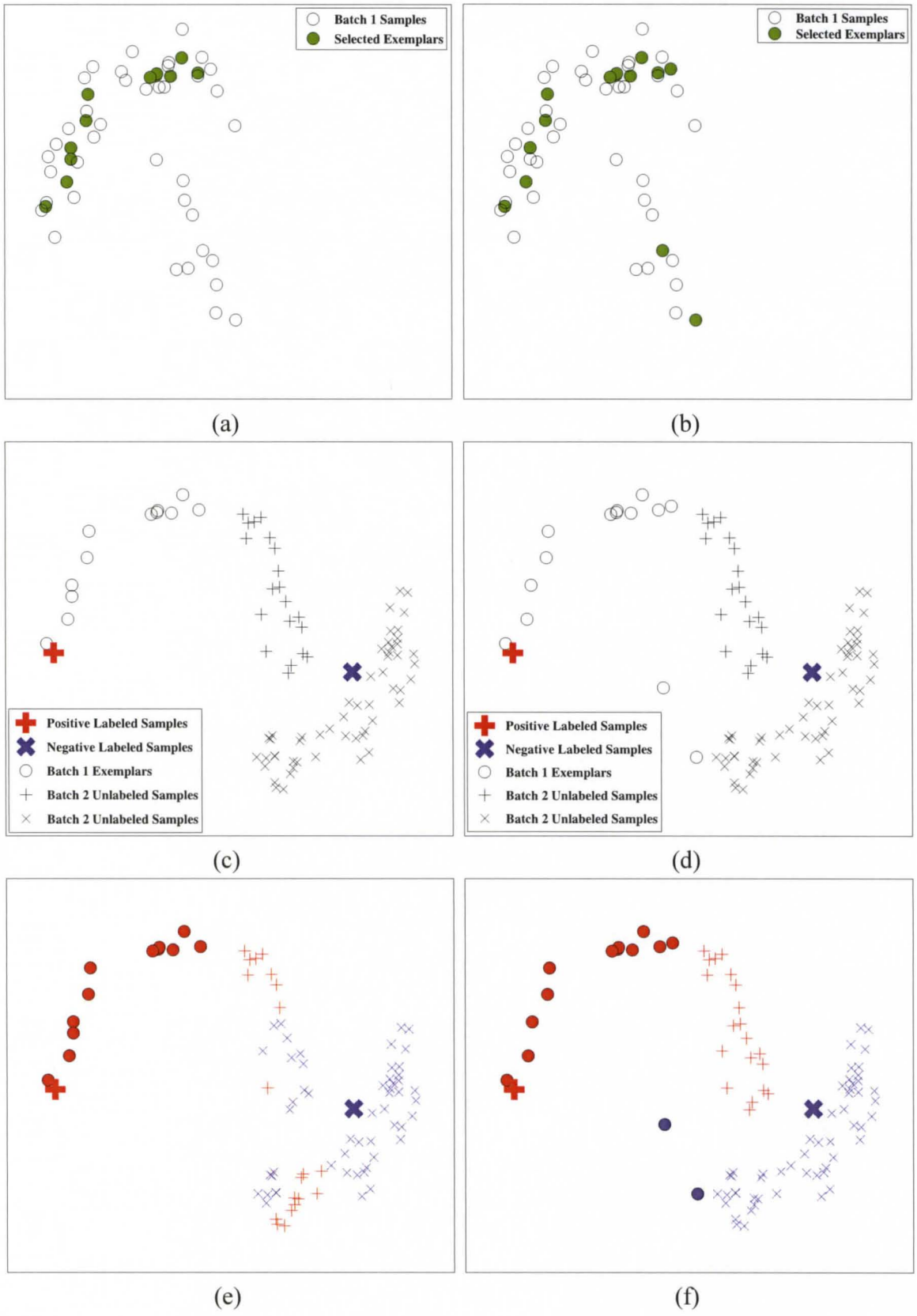
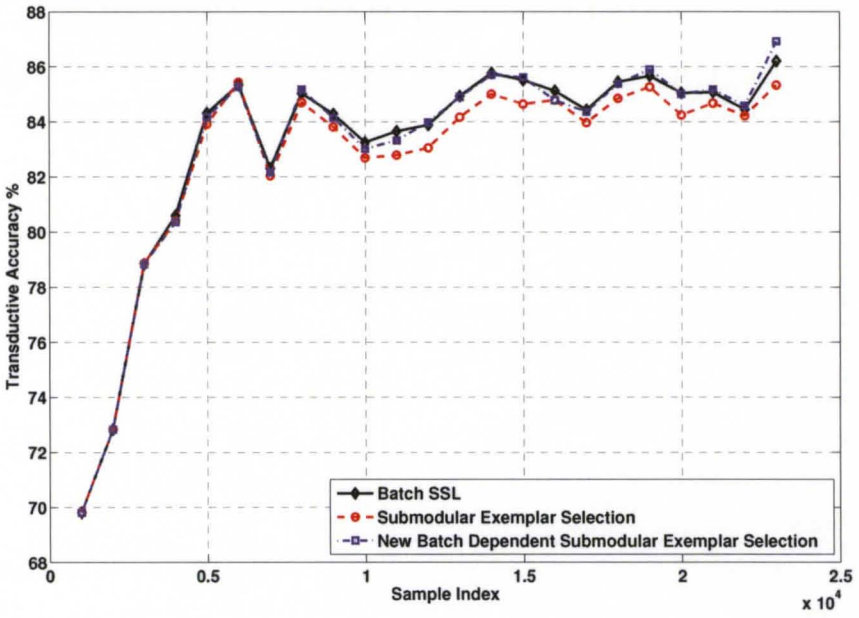
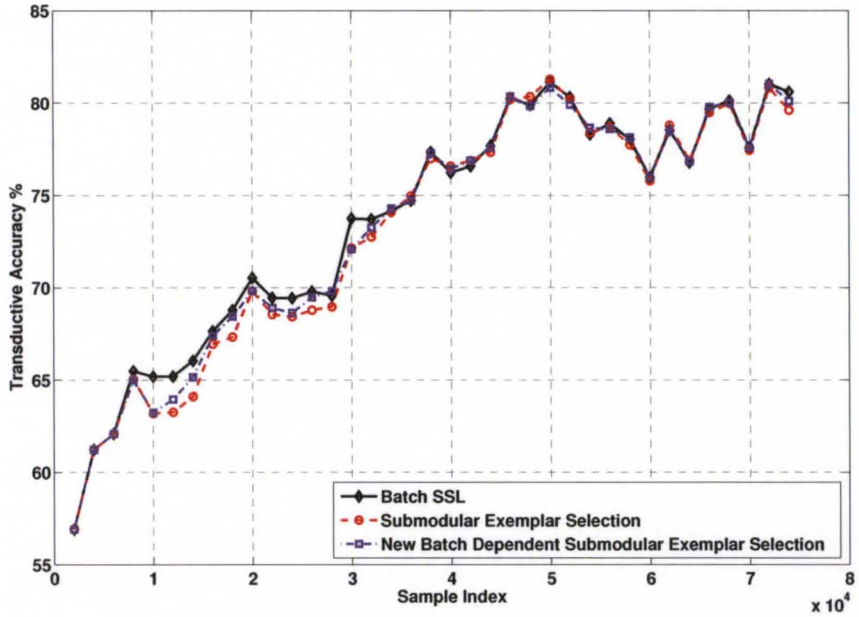


Figure IV.32: Importance of inter-batch dependence. Left Column: Output of SUBMOD-EXMP model. Right Column: Output if inter-batch dependence is enforced. (a-b) Selecting exemplars from batch $\mathcal{B}1$. (c-d) Using exemplars and batch $\mathcal{B}2$ as input for SSL. (e-f) Output of SSL.

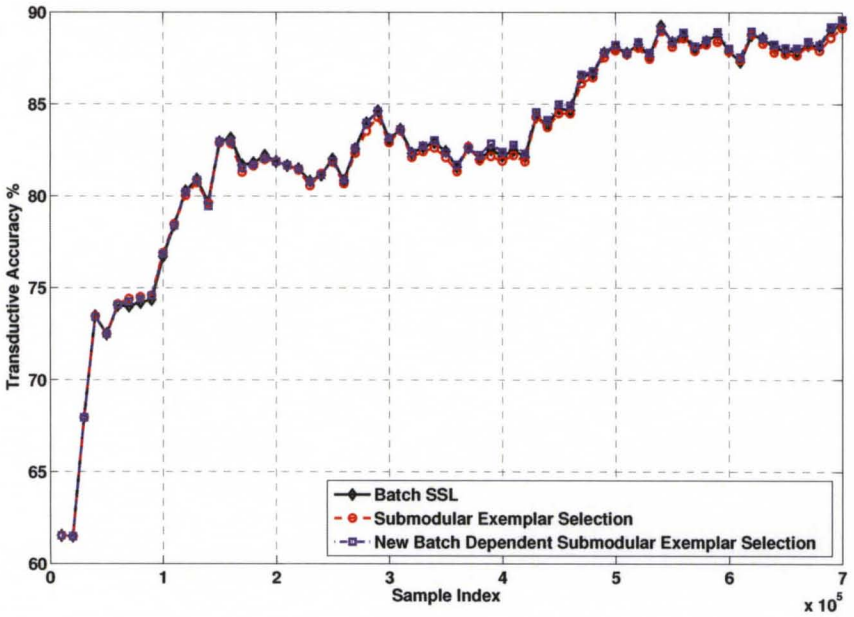


(a) Gcat

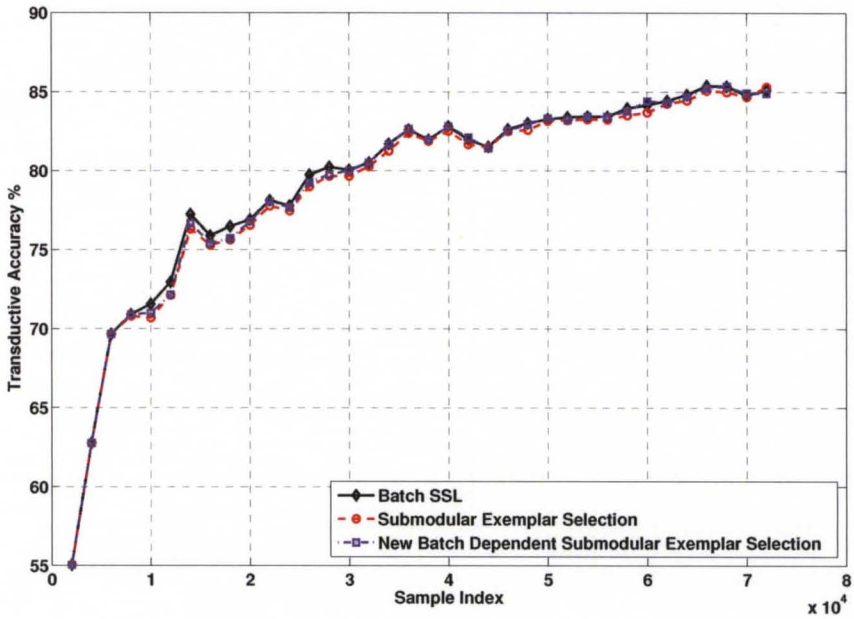


(b) Real Sim

Figure IV.33: Batch Dependent Transductive Accuracy

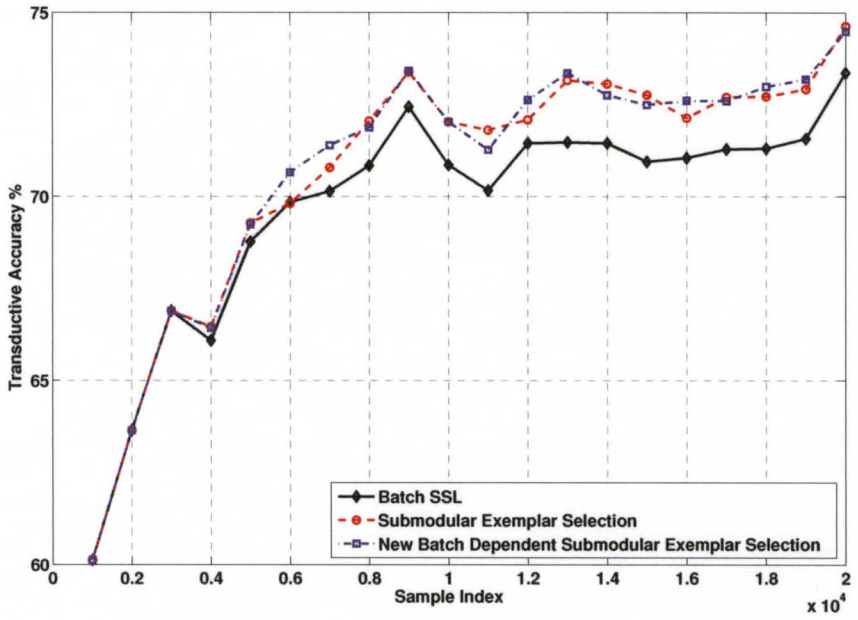


(a) RCV1

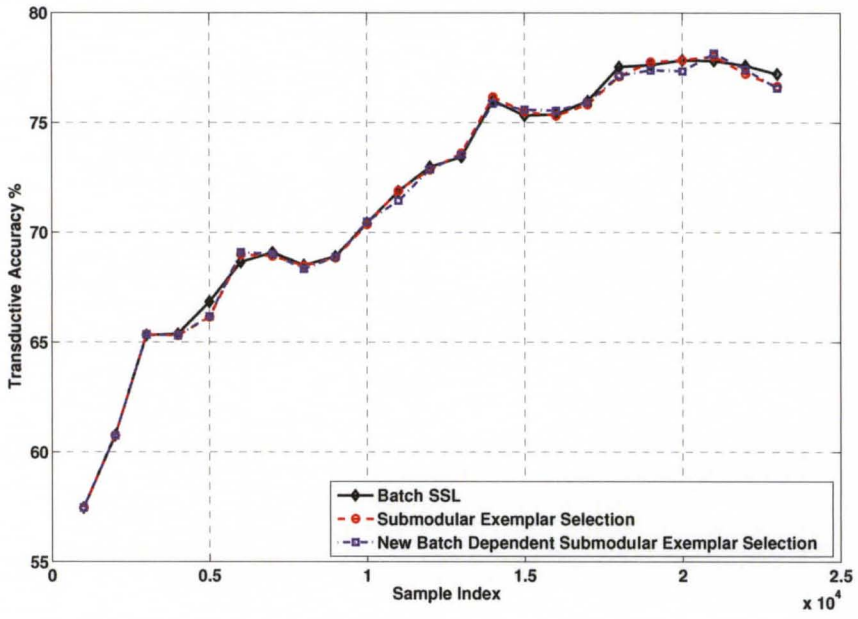


(b) Aut-Avn

Figure IV.34: Batch Dependent Transductive Accuracy



(a) News20Binary



(b) ccat

Figure IV.35: Batch Dependent Transductive Accuracy

CHAPTER V

ACTIVE LEARNING EXTENSION FOR QP/SUBMOD-S^{3}VM

The previous chapters considered two aspects of our proposed *Never-Ending Learning* framework, specifically learning under limited supervision in Chapter III and doing so over time in Chapter IV. In both chapters the learning process is unidirectional. Knowledge flows from the oracle (i.e. environment) to the learning machine in the form of labels or dense regions of unlabeled samples. In this chapter we present an extension of the QP/SUBMOD-S^{3}VM algorithms in Chapter III for *Active Learning*, where the learning machine requests feedback from an oracle in the form of labels for important samples. Active learning serves as the interaction between the learning machine and the oracle. This interaction is necessary for the proposed *Never-Ending Learning* framework as the learning machine is deployed for extended periods of time. Occasionally, the learning machine will come across hard samples that may deteriorate the performance in the long run unless handled properly. Therefore, *active learning* provides a mechanism for requesting feedback from the oracle when needed.

V.1 Proposed Active Learning for QP/SUBMOD-S^{3}VM

The proposed *active learning* technique is formulated in Problem 9.

Problem 9. *Quadratic Programming Active Learning Algorithm for QP-S^{3}VM (QP-ACTV):*

Given a data batch $\mathcal{B} = \{\mathcal{U}, \mathcal{L}\}$, with $\mathcal{U}$ and $\mathcal{L}$ being subsets of unlabeled and labeled samples, respectively. The proposed QP-ACTV selects unlabeled samples from the subset $\mathcal{U}$ in order to be labeled by an oracle using the following optimization problem:

$$\underset{\mathbf{a}'=[a_1,\ldots,a_{|\mathcal{U}|}]}{\arg\min} \quad \lambda_1 \mathbf{a}'\mathbf{K_{uu}}\mathbf{a} + \lambda_2 |\mathbf{y}'\mathbf{K}_{lu}|\mathbf{a} + \lambda_3 \mathbf{1}'_{|\mathcal{U}|}\mathbf{K}_{uu}\mathbf{a} \tag{V.1}$$

subject to

$$\mathbf{a}'\mathbf{1}_{|\mathcal{U}|} = \mathcal{M}_{\mathcal{V}}, \quad \mathbf{0} \leq \mathbf{a} \leq \mathbf{1}_{|\mathcal{U}|}. \tag{V.2}$$

where

$\mathbf{1}_{|\mathcal{L}|}$*: A ones vector of length* $|\mathcal{L}|$*. Similarly is* $\mathbf{1}_{|\mathcal{U}|}$*.*

$\mathbf{a}' = [a_1, \ldots, a_{|\mathcal{U}|}], a_j = 1$ *for selected exemplars and* $a_j = 0$ *otherwise.*

$\mathcal{M}_{\mathcal{V}}$*: The number of samples to be labeled by the oracle.*

$\mathbf{K_{ll}} = \mathbf{K}_{i,i'}\ \forall i, i' \in \mathcal{L}, \quad \mathbf{K_{uu}} = \mathbf{K}_{j,j'}\ \forall j, j' \in \mathcal{U}, \quad \mathbf{K_{lu}} = \mathbf{K}_{i,j}\ \forall i \in \mathcal{L}, j \in \mathcal{U}.$

Note: Equation (V.1) *can be rewritten in the standard quadratic programming form as follows:*

$$\underset{\mathbf{a}'=[a_1,\ldots,a_{|\mathcal{U}|}]}{\arg\min} \ \lambda_1 \mathbf{a}'\mathbf{K_{uu}}\mathbf{a} + (\lambda_2 |\mathbf{y}'\mathbf{K}_{lu}| + \lambda_3 \mathbf{1}'_{|\mathcal{U}|}\mathbf{K}_{uu})\mathbf{a} \tag{V.3}$$

V.2 QP-ACTV Model Interpretation

The first term in Eq.(V.1) is the same as that in the QP-EXMP model in Section IV.2.2 where we showed that it enforces diversity amongst the selected exemplars summarizing a data stream. For the purposes of active learning, requesting feedback from an oracle is expensive and thus in active learning, the goal is to gain the maximum feedback supervision using the minimum number of requests. Therefore, enforcing diversity maximizes the information gain per feedback request made by the active learning algorithm.

To interpret the second in Eq.(V.1) we rewrite it as follows:

$$\lambda_2 |\mathbf{y}'\mathbf{K}_{lu}|\mathbf{a} = \lambda_2 \sum_{j \in \mathcal{U}} a_j \left| \sum_{i \in \mathcal{L}} y_i [\mathbf{K_{lu}}]_{i,j} \right| \tag{V.4}$$

The absolute value expression finds the difference in similarities between all positive/negative labeled samples and each unlabeled sample. As $a_j \in [0, 1]$, minimizing this term entails assigning $a_j = 0$ to samples that have large similarity with either positively or negatively labeled samples. On the other hand, samples that are highly dissimilar or equally similar to both signs of labeled samples, are assigned $a_j = 1$. In fact the second term handles one essential kind of confusion in label propagation semi-supervised learning problems.

An unlabeled sample that is too dissimilar to any labeled sample or equally similar to two labeled samples from opposite classes is considered confusing and error prone. Therefore, it is highly valuable for *active learning*.

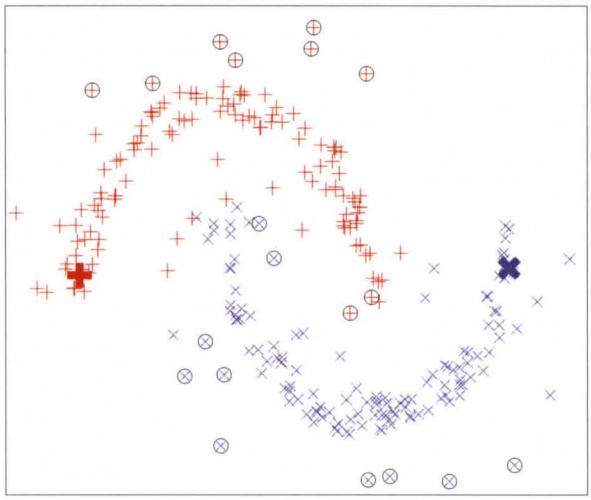
Minimizing the third term as expressed in Eq.(V.5) involves assigning $a_j = 1$ to unlabeled samples with small aggregated similarity with other unlabeled samples and vice versa for $a_j = 0$.

$$\lambda_3 \mathbf{1}'_{|\mathcal{U}|} \mathbf{K}_{uu} \mathbf{a} = \lambda_3 \sum_{j,j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{j,j'} a_j \tag{V.5}$$

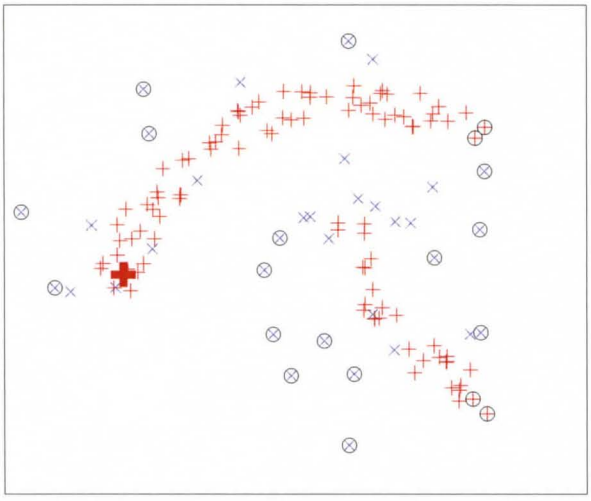
In other words, the third term in Eq.(V.1) encourages selecting samples, for active learning, that are far from dense regions of unlabeled samples. This idea is coherent with the concepts of label propagation in semi-supervised learning where labels propagate mainly through dense regions of unlabeled samples. Therefore stray unlabeled samples are likely to be confusing and appropriate for *active learning*.

In summary, the proposed QP-ACTV formulation in Problem 9 uses three basic ideas for active learning: a) Similarity to labeled samples is small, b) existence far from dense regions, and c) diversity of selections to minimize the number of requests to the oracle.

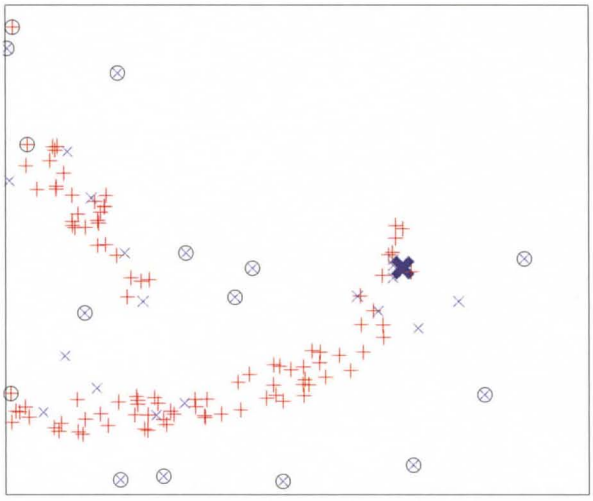
Figure V.1 provides some examples of the samples selected for active labeling by the QP-ACTV model. The depicted data set has only two labeled samples. Figure V.1(a) shows the full data set and the selected samples in circles. It is noticeable that many of the selected samples exist in sparse areas. However, in the center of the Fig. V.1(a), we see samples that are very close to dense regions and yet are selected. Such samples are likely to be far or equally distant from all the labeled samples. In Fig. V.1(b) and Fig. V.1(c) we use QP-ACTV with subsets of the full data. It is more clear in these plots that samples can be in very dense regions and yet be selected for active labeling. The diversity of the selected samples is also clear in the figure, otherwise we would have had most of the samples in a limited area.



(a)



(b)



(c)

Figure V.1: (a) Two Moons data set with two labeled samples. Circled samples are selected for active labeling by QP-ACTV. (b) Batch 1 from the Two Moons data set with samples selected for active labeling. (c) Batch 2 from the Two Moons data set with samples selected for active labeling.

V.3 Submodular Optimization of QP-ACTV (SUBMOD-ACTV)

This section presents a submodular reformulation of the QP-ACTV problem to overcome the computational burden of quadratic programming. Similar to what we have done in previous chapters, we transform the current optimization problem into a set function optimization problem where we select the best subset of samples among all possible sample sets for active learning. Following the procedure we used previously, the submodular formulation is as follows:

Problem 10. *Submodular formulation (SUBMOD-ACTV) of the QP-ACTV in Problem 9:*

$$\max_{|\mathcal{V}| \leq \mathcal{M}_\mathcal{V}} \mathcal{J}(\mathcal{V}) \tag{V.6}$$

where

$$\begin{aligned} \mathcal{J}(\mathcal{V}) \;=\; & -\lambda_1 \sum_{j,j' \in \mathcal{V}} [\mathbf{K_{uu}}]_{j,j'} - \lambda_2 \sum_{j \in \mathcal{V}} \left| \sum_{i \in \mathcal{L}} y_i \, [\mathbf{K_{lu}}]_{i,j} \right| - \lambda_3 \sum_{j \in \mathcal{V}, j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{j,j'} \\ & + \underbrace{\sum_{j \in \mathcal{V}} d \, [\lambda_1 (2|\mathcal{U}| + 1) + \lambda_2 |\mathcal{L}| + \lambda_3 |\mathcal{U}|]}_{\mathcal{T}}, \end{aligned} \tag{V.7}$$

and $\mathcal{J}$ is a submodular set function defined on all subsets $\mathcal{V} \subset \mathcal{U}$ of unlabeled samples eligible to be chosen as exemplars, $0 \leq \mathbf{K}_{:,:} \leq d$.

Once again, the submodular formulation is basically the negative of the discrete of the quadratic programming problem, with the addition of a designed constant that ensures the monotonicity and submodularity of the new objective set function, as explained in Theorem 6.

Theorem 6. *The set function $\mathcal{J}(\mathcal{V})$ in Problem 10 is monotone (non-decreasing), submodular, and $\mathcal{J}(\emptyset) = 0$.*

Proof. First, $\mathcal{J}(\emptyset) = 0$ follows directly from the definition in Eq.(V.7) where all the summations are on elements in the set $\mathcal{V}$. Therefore if $\mathcal{V} = \emptyset$ then $\mathcal{J}(\emptyset) = 0$. Next we prove the *monotonicity property*. Using the definition of $\mathcal{J}(\mathcal{V})$, we can show that for any $m \in \mathcal{U}$

and $m \notin \mathcal{V}$, the increase in the objective value of $\mathcal{J}$ due to adding m is,

$$\begin{aligned}\mathcal{J}(\mathcal{V} \cup m) - \mathcal{J}(\mathcal{V}) \;=\; & -2\lambda_1 \sum_{j' \in \mathcal{V}} [\mathbf{K_{uu}}]_{m,j'} - \lambda_1 [\mathbf{K_{uu}}]_{m,m} - \lambda_2 \left| \sum_{i \in \mathcal{L}} y_i [\mathbf{K_{lu}}]_{i,m} \right| \\ & - \lambda_3 \sum_{j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{m,j'} + 2\lambda_1 |\mathcal{U}| d + \lambda_1 d + \lambda_2 |\mathcal{L}| d + \lambda_3 |\mathcal{U}| d\end{aligned} \tag{V.8}$$

For any kernel matrix $\mathbf{K}$, where $0 \leq \mathbf{K}_{i,j} \leq d$, since

$$\begin{aligned} & 2\lambda_1 |\mathcal{U}| d - 2\lambda_1 \sum_{j \in \mathcal{V}} [\mathbf{K_{uu}}]_{m,j'} \geq 0 \\ & \lambda_1 d - \lambda_1 [\mathbf{K_{uu}}]_{m,m} \geq 0 \\ & \lambda_2 |\mathcal{L}| d - \lambda_2 \left| \sum_{i \in \mathcal{L}} y_i [\mathbf{K_{lu}}]_{i,m} \right| \geq 0 \\ & \lambda_3 |\mathcal{U}| d - \lambda_3 \sum_{j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{m,j'} \geq 0 \end{aligned} \tag{V.9}$$

then,

$$\mathcal{J}(\mathcal{V} \cup m) - \mathcal{J}(\mathcal{V}) \geq 0.$$

Thus the monotonicity property of $\mathcal{J}(\mathcal{V})$ holds true.

Now we prove the *submodularity* of $\mathcal{J}(\mathcal{V})$ by assuming the set $\mathcal{F} = \{\mathcal{V} \cup q\}$, where $q \in \mathcal{U}$. Using the same set element m we used earlier, i.e. $m \in \mathcal{U}$ and $m \notin \mathcal{V}$, we need to show that adding m to the set $\mathcal{V}$ has more effect than adding it to the set $\mathcal{F}$ as stated in Def. 2. Since

$$\begin{aligned}\mathcal{J}(\mathcal{F}) \;=\; & -\lambda_1 \sum_{j,j' \in \{\mathcal{V} \cup q\}} [\mathbf{K_{uu}}]_{j,j'} - \lambda_2 \sum_{j \in \{\mathcal{V} \cup q\}} \left| \sum_{i \in \mathcal{L}} y_i [\mathbf{K_{lu}}]_{i,j} \right| - \lambda_3 \sum_{j \in \{\mathcal{V} \cup q\}, j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{j,j'} \\ & + \sum_{j \in \{\mathcal{V} \cup q\}} d\,[\lambda_1 (2|\mathcal{U}| + 1) + \lambda_2 |\mathcal{L}| + \lambda_3 |\mathcal{U}|],\end{aligned} \tag{V.10}$$

then

$$\begin{aligned}\mathcal{J}(\mathcal{F} \cup m) - \mathcal{J}(\mathcal{F}) \;=\; & -2\lambda_1 \sum_{j' \in \mathcal{F}} [\mathbf{K_{uu}}]_{m,j'} - \lambda_1 [\mathbf{K_{uu}}]_{m,m} - \lambda_2 \left| \sum_{i \in \mathcal{L}} y_i [\mathbf{K_{lu}}]_{i,m} \right| \\ & - \lambda_3 \sum_{j' \in \mathcal{U}} [\mathbf{K_{uu}}]_{m,j'} + 2\lambda_1 |\mathcal{U}| d + \lambda_1 d + \lambda_2 |\mathcal{L}| d + \lambda_3 |\mathcal{U}| d\end{aligned} \tag{V.11}$$

Therefore

$$[\mathcal{J}(\mathcal{V} \cup m) - \mathcal{J}(\mathcal{V})] - [\mathcal{J}(\mathcal{F} \cup m) - \mathcal{J}(\mathcal{F})] \;=\; 2\lambda_1 \left[\mathbf{K_{uu}}\right]_{q,m} \geq 0 \qquad \text{(V.12)}$$

Hence the set function $\mathcal{J}(\mathcal{V})$ is submodular. □

Having formulated the SUBMOD-ACTV, we can use a simple greedy approach to optimize the problem efficiently as we have done previously.

V.4 Experimental Results

To test the proposed active learning model, we follow the experimental set up we used for the incremental SUBMOD-S^{3}VM in Sec.IV.5. Batches of partially labeled samples arrive over time. At each time iteration, the proposed SUBMOD-ACTV model selects a set $\mathcal{V}$ of the unlabeled samples for active labeling by an oracle. Once the labels are provided by the oracle, the labels are reflected on the available partially labeled data and the incremental SUBMOD-S^{3}VM is invoked. In the experiments we set the size of $\mathcal{V}$ to %25 of the batch size.

The goal of the experiments is to examine whether the actively selected samples actually help the transductive accuracy of the incremental SUBMOD-S^{3}VM algorithm. Furthermore, the experiments should verify that any achieved enhancement in the transductive accuracy is not just due to the increase in the number of labeled samples used in the learning process. To that end, we compare the performance of the proposed SUBMOD-ACTV algorithm to randomly selected active samples.

In Figs V.2-V.4 we see that the proposed SUBMOD-ACTV algorithm consistently achieves significantly better transductive accuracy when used with the incremental SUBMOD-S^{3}VM. An important observation from the figures is that random selection of samples for active labeling clearly lags in performance behind the SUBMOD-ACTV algorithm. This shows that the SUBMO-ACTV selects particularly important samples for active labeling and that the improvement over the standard incremental SUBMOD-S^{3}VM is not due to the increased number of labeled samples.

Figures V.2-V.4 show that the performance of random active learning eventually coincides with the standard incremental SUBMOD-S^{3}VM. The extra labeled samples, though

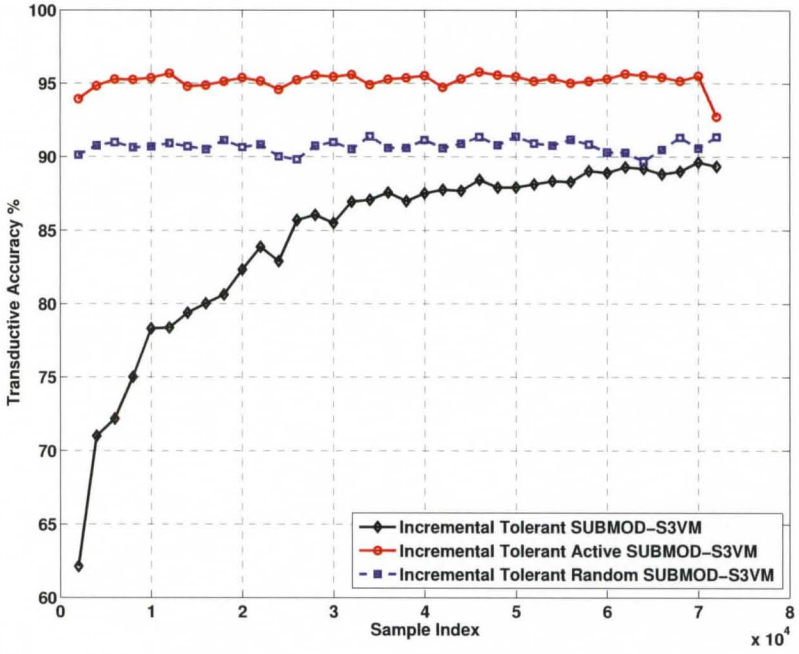


Figure V.2: Active SUBMOD results for the **Aut-Avn** data set.

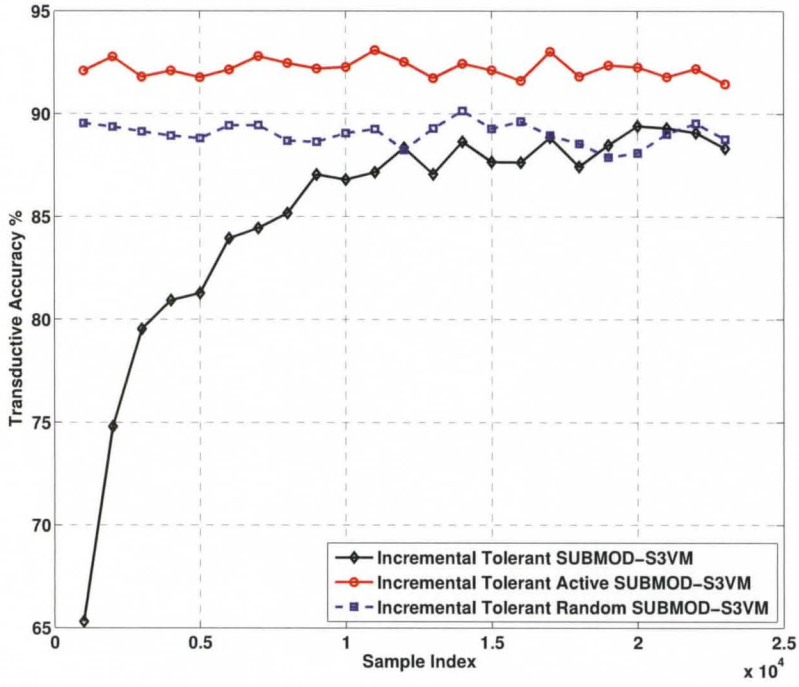


Figure V.3: Active SUBMOD results for the **GCAT** data set.

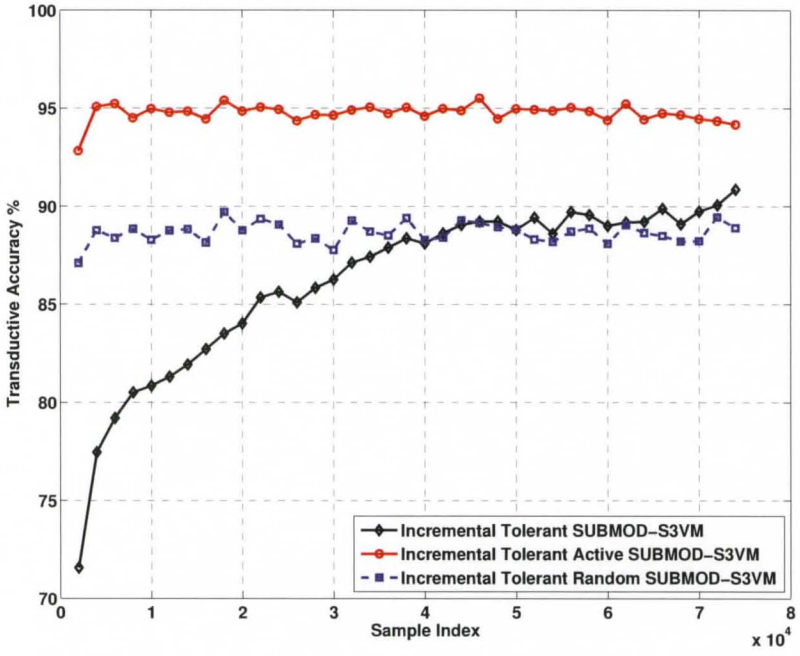


Figure V.4: Active SUBMOD results for the **Real-Sim** data set.

chosen randomly, allows that random active learning to uncover some important structure of the data, which will not be visible for the standard incremental SUBMOD-S^3VM until later on in time.

CHAPTER VI

INCREMENTAL SVM TRAINING VIA LOCAL KERNELS

The dissertation so far presented a framework for SVM-based *Never-Ending Learning* using submodular optimization. The proposed framework is *transductive* in the sense that it only provides labels for the unlabeled samples seen thus far. In other words, there is no *inductive* model that can be used for classifying unseen samples without invoking the semi-supervised learning process. However, a standard framework for learning from streaming data should have an up-to-date inductive model for use at any point in time. In this chapter we provide an algorithm for *incremental supervised SVM* that uses the properties of local kernels (e.g. RBF kernels) to efficiently update an inductive SVM model over time.

Section VI.1 explains the local properties of RBF-SVM during the testing stage, i.e. the local properties of the SVM decision function. Section VI.2 introduces the idea of using local properties of the RBF kernel to perform efficient SVM model updating. Section VI.3 investigates analytically and experimentally whether locality exists during training of RBF-SVM. Section VI.4 proposes a analytical and an experimental estimates for the ultimate neighborhood size. Finally, discussion is provided in Sec.VI.5.

VI.1 Locality of RBF-SVM Decision Function

In this section we will prove the locality of the RBF-SVM decision function. The decision function of SVM has the form,

$$\begin{aligned} f(\mathbf{x}) &= \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b \\ &= \sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b \end{aligned} \tag{VI.1}$$

which can be interpreted as an expert, i.e. voting, system where each expert is represented by a support vector (α_i in (VI.1)) and the final decision on the classification of a data sample $\mathbf{x}$ is basically the summation of weighted decisions of all the experts. The degree of participation of each expert in the final decision [59] is given by

$$\frac{\partial f(\mathbf{x})}{\partial \alpha_i} = y_i K(\mathbf{x}_i, \mathbf{x}) \tag{VI.2}$$

Discarding the sign of the examined support vector we get

$$\left| \frac{\partial f(\mathbf{x})}{\partial \alpha_i} \right| = K(\mathbf{x}_i, \mathbf{x}) \tag{VI.3}$$

In the case of using an RBF kernel,

$$\mathbf{RBF}(\mathbf{x}_i, \mathbf{x}) = \exp(-\frac{\|\mathbf{x}_i - \mathbf{x}\|^2}{2\sigma^2}) \tag{VI.4}$$

equation (VI.3) is interpreted as follows: During the classification phase of an RBF-SVM, the contribution of each support vector in the final classification depends solely on the RBF similarity between the sample to be classified $\mathbf{x}$ and the support vector $\mathbf{x}_i$. The similarity in terms of RBF is basically the $\mathcal{L}^2$ distance between the support vector and the classified sample. This means that the support vectors that are very far from the classified sample, have very small effect on its classification.

VI.2 Incremental Local RBF-SVM Algorithm

The locality of RBF-SVM decision function along with the interpretation of SVM as an expert system made the extension of the locality property to the learning stage an intuitive step [59]. Basically, when viewing RBF-SVM as an expert system, the classification of a data sample depends mostly on the experts closest to it. Therefore, a natural extension of the idea for the learning process is that if a new expert is added to the existing set of experts then the closest experts to it are the ones that will encounter considerable changes. The changes to the rest of the experts will diminish as they get farther from the new expert. Figure VI.1 provides an illustration of the concept of locality during training.

This leads to a natural incremental SVM algorithm that updates an existing SVM^{old} model using only local efficient updates rather than complete retraining from scratch.

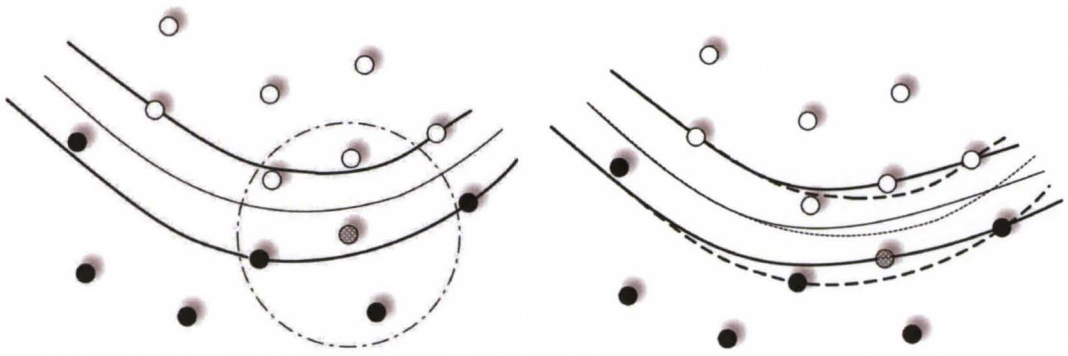


Figure VI.1: Illustration of the locality of RBF-SVM during learning. (a) An established RBF-SVM and a new training sample represented by the dashed sample. (b) The old RBF-SVM is depicted with dashed curves while the updated one is depicted with solid lines.

Specifically, the algorithm starts with SVM^{old} and for each newly arriving sample a neighborhood is constructed as shown in Fig.VI.1. The SMO algorithm, Sec.II.1.4.4, is re-run on the samples inside the neighborhood to obtain SVM^{new}. The neighborhood size is determined in an iterative fashion where the neighboring samples are incrementally added to the neighborhood one at a time. The neighborhood stops growing once the SVM^{new} model stabilized, i.e. no significant changes are observed when adding new samples to the neighborhood.

Despite the appeal of the described local incremental RBF-SVM algorithm, the tranfernsability of the local properties from the testing to the training stage of SVM is still just an intuitive assumption. The locality during the training stage has not been studied analytically or verified experimentally. Moreover, the local incremental SVM algorithm [59] is very iterative as the neighborhood of changed support vectors is constructed by adding one close training sample to the neighborhood at a time and repeating the model updating process.

Therefore, the goal of the rest of this chapter is to formally validate if the locality during training assumption is true. Then using the findings from the validity of the local assumption, we will try to find an estimate of the ultimate neighborhood size for each new sample. This estimate will significantly reduce the number of iterations used to construct

the neighborhood around the new sample [26, 27].

VI.3 RBF-SVM Locality During Training

In this section, we will formally check the validity of the locality assumption during the learning stage with the ultimate goal of finding an estimate of the support vectors neighborhood size that provides the compromise between the computational complexity and the accuracy of the solution without the need for iterations.

To validate the locality during learning premise, we need to measure the changes occurring to an RBF-SVM when a new sample is added to its training data set. As shown in (VI.1), the main parameters of an SVM are the Lagrangian multipliers α_i associated with each support vector. Therefore, throughout our study we will use $\Delta\alpha_i$ as a measure of changes happening to individual support vectors. Hence, the objective of the rest of this section comes down to studying the influence of a new training sample on $\Delta\alpha_i$ for each support vector.

VI.3.1 Exact Formulation of Incremental SVM Training

In order to analytically validate the locality of RBF-SVM during learning, we will use a formulation of $\Delta\alpha_i$ developed for exact incremental training of SVM [16]. To present this formulation we will start from the dual formulation of the SVM training problem where the solution is obtained by minimizing the objective function $\mathcal{W}$ described by

$$\min_{0\leq\alpha_i\leq C} : \mathcal{W} = \frac{1}{2}\sum_{i,j}\alpha_i Q_{ij}\alpha_j - \sum_i \alpha_i + b\sum_i y_i\alpha_i \tag{VI.5}$$

where α_i are the Lagrangian multipliers and $Q_{ij} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$ is a symmetric positive definite kernel matrix. The Karush-Kuhn-Tucker (KKT) conditions result from the first-

order conditions on $\mathcal{W}$ as follows:

$$g_i = \frac{\partial \mathcal{W}}{\partial \alpha_i} = \sum_j Q_{ij}\alpha_j + y_i b - 1 \tag{VI.6}$$

$$= y_i f(\mathbf{x}_i) - 1 \begin{cases} \geq 0 & \alpha_i = 0 & \forall i \in R \\ = 0 & 0 < \alpha_i < C & \forall i \in SV \\ \leq 0 & \alpha_i = C & \forall i \in E \end{cases}$$

$$\frac{\partial \mathcal{W}}{\partial b} = \sum_j y_j \alpha_j = 0 \tag{VI.7}$$

where $SV = \{\mathbf{x}_{s_1}, \mathbf{x}_{s_2}, \ldots, \mathbf{x}_{s_\ell}\}$ is the set of support vectors strictly on the margin, E is the set of support vectors inside the margin, and R is the set of all other data samples. We denote the whole data set by D, where $D = SV \cup R \cup E$.

The basic idea of the incremental SVM learning formulation [16] is to retain the KKT conditions on all the previous samples in the SVM while adding a new sample to the model. To this end, the differences in the KKT conditions, (VI.6) and (VI.7), when a new sample $\mathbf{x}_c$ with label y_c is added to the SVM are expressed as

$$\Delta g_i = Q_{ic}\Delta\alpha_c + \sum_{j \in S} Q_{ij}\Delta\alpha_j + y_i \Delta b, \quad \forall i \in D \cup \{\mathbf{x}_c\} \tag{VI.8}$$

$$0 = y_c \Delta\alpha_c + \sum_{j \in S} y_j \Delta\alpha_j \tag{VI.9}$$

where α_c is the Lagrangian multiplier associated with $\mathbf{x}_c$; it is initialized to zero before being added to the SVM model. Since $g_i = 0$ for the set S, then under the assumption that the new sample $\mathbf{x}_c$ will not change the members of the set SV, (VI.8) and (VI.9) can be formulated $\forall s_i$ as

$$\mathcal{Q}. \begin{pmatrix} \Delta b \\ \Delta\alpha_{s_1} \\ \vdots \\ \Delta\alpha_s \end{pmatrix} = - \begin{pmatrix} y_c \\ Q_{s_1 c} \\ \vdots \\ Q_{s_\ell c} \end{pmatrix} \Delta\alpha_c \tag{VI.10}$$

where

$$\mathcal{Q} = \begin{pmatrix} 0 & y_{s_1} & \cdots & y_{s_\ell} \\ y_{s_1} & Q_{s_1 s_1} & \cdots & Q_{s_1 s_\ell} \\ \vdots & \vdots & \vdots & \vdots \\ y_{s_\ell} & Q_{s_\ell s_1} & \cdots & Q_{s_\ell s_\ell} \end{pmatrix} \tag{VI.11}$$

Hence, the modifications of the SVM model to accommodate the new sample $\mathbf{x}_c$ are described by

$$\begin{aligned} \Delta b &= \beta \Delta \alpha_c \\ \Delta \alpha_j &= \beta_j \Delta \alpha_c, \quad \forall j \in D \end{aligned} \tag{VI.12}$$

where β's are obtained by

$$\begin{pmatrix} \beta \\ \beta_{s_1} \\ \vdots \\ \beta_{s_\ell} \end{pmatrix} = -Q^{-1}. \begin{pmatrix} y_c \\ Q_{s_1 c} \\ \vdots \\ Q_{s_\ell c} \end{pmatrix} \tag{VI.13}$$

and

$$\beta_j = 0 \quad \forall j \notin SV \tag{VI.14}$$

Substituting β's in (VI.8), the changes in the KKT conditions will be

$$\Delta g_i = \gamma_i \Delta \alpha_c \quad \forall i \in D \cup \{\mathbf{x}_c\} \tag{VI.15}$$

where

$$\gamma_i = Q_{ic} + \sum_{j \in S} Q_{ij} \beta_j + y_i \beta \quad \forall i \notin SV \tag{VI.16}$$

and

$$\gamma_i = 0 \quad \forall i \in SV \tag{VI.17}$$

Now that we have presented a closed form formulation of the changes that will occur to an SVM (i.e. $\Delta \alpha_i$) when a new data sample is added to its training data set. We will examine these changes for RBF-SVMs and see whether they are local in nature.

VI.3.2 Analytical Proof of Locality for a Pilot Case

In this section we present a mathematical proof of the locality of RBF-SVM during learning for the pilot case shown in Fig.VI.2. The pilot case consists of an RBF-SVM with two support vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ and a new training data sample $\mathbf{x}_c$. In the following lemma we show that if the $\mathbf{x}_c$ is closer to $\mathbf{x}_1$ than $\mathbf{x}_2$, then the absolute changes in α_1 are larger than the absolute changes in α_2.

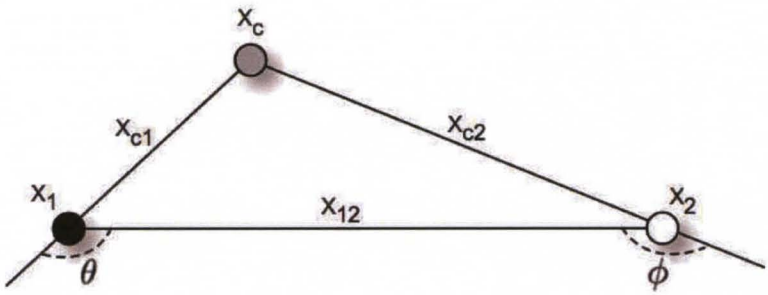


Figure VI.2: Pilot RBF-SVM with two support vectors $\mathbf{x}_1$ and $\mathbf{x}_2$. A new sample $\mathbf{x}_c$ is added to the training data set such that $\mathbf{x}_1$ is closer to $\mathbf{x}_c$ than $\mathbf{x}_2$ is.

Lemma 1. *Let RBF-SVM be any radial-basis-function support vector machine with a training data set* $D = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \dots\}$ *and a set of support vectors* $SV = \{(\mathbf{x}_1, \alpha_1), (\mathbf{x}_2, \alpha_2)\}$, *where* α_i *are the associated Lagrangian multipliers. Assume that a new training data sample* $\mathbf{x}_c$ *is added to* D *and a new support vector machine RBF-SVM*$_{new}$ *is constructed, such that* $D_{new} = D \cup \{\mathbf{x}_c\}$ *and* $SV_{new} = \{(\mathbf{x}_1, \alpha_1'), (\mathbf{x}_2, \alpha_2'), (\mathbf{x}_c, \alpha_c')\}$. *If* $\|X_{c1}\| < \|X_{c2}\|$, *then* $|\Delta\alpha_1| > |\Delta\alpha_2|$, *where* $\Delta\alpha_i = \alpha_i' - \alpha_i$ *are the changes that each support vector will encounter due to the new training sample* $\mathbf{x}_c$ *and* $\|X_{ij}\|$ *is the distance between* $\mathbf{x}_i$ *and* $\mathbf{x}_j$.

Proof. Applying (VI.12) and (VI.13) to the example in Fig.(VI.2), we get the following formulas for $\Delta\alpha_1$ and $\Delta\alpha_2$.

$$\Delta\alpha_1 = \frac{-\Delta\alpha_c y_1 y_c}{2}\left[\frac{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1 - e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}} + e^{-\frac{\|X_{c2}\|^2}{2\sigma^2}}}{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1}\right] \tag{VI.18}$$

$$\Delta\alpha_2 = \frac{-\Delta\alpha_c y_2 y_c}{2}\left[\frac{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1 - e^{-\frac{\|X_{c2}\|^2}{2\sigma^2}} + e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}}}{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1}\right] \tag{VI.19}$$

$$\frac{\Delta\alpha_1}{\Delta\alpha_2} = \frac{y_1\left[e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1 - e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}} + e^{-\frac{\|X_{c2}\|^2}{2\sigma^2}}\right]}{y_2\left[e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1 - e^{-\frac{\|X_{c2}\|^2}{2\sigma^2}} + e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}}\right]} \tag{VI.20}$$

$$\frac{|\Delta\alpha_1|}{|\Delta\alpha_2|} = \frac{\left|e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1 - e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}} + e^{-\frac{\|X_{c2}\|^2}{2\sigma^2}}\right|}{\left|e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1 - e^{-\frac{\|X_{c2}\|^2}{2\sigma^2}} + e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}}\right|} \tag{VI.21}$$

Let

$$\mathcal{C} = e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1 \qquad \text{and} \qquad \mathcal{D} = e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}} - e^{-\frac{n^2\|X_{c1}\|^2}{2\sigma^2}} \tag{VI.22}$$

then (VI.21) becomes,

$$\frac{|\Delta\alpha_1|}{|\Delta\alpha_2|} = \frac{|\mathcal{C} - \mathcal{D}|}{|\mathcal{C} + \mathcal{D}|} \tag{VI.23}$$

Since

$$\|X_{c1}\|^2 > 0 \quad , \quad \sigma > 0 \quad \text{, and} \quad n > 1 \tag{VI.24}$$

then

$$e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}} > e^{-\frac{n^2\|X_{c1}\|^2}{2\sigma^2}} \quad \Rightarrow \boxed{\mathcal{D} > 0} \tag{VI.25}$$

Since

$$\|X_{12}\|^2 > 0 \quad \text{, and} \quad \sigma > 0 \tag{VI.26}$$

then

$$e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} < 1 \quad \Rightarrow \boxed{\mathcal{C} < 0} \tag{VI.27}$$

Since

$$\mathcal{D} > 0 \quad \text{and} \quad \mathcal{C} < 0 \tag{VI.28}$$

then

$$\frac{|\mathcal{C} - \mathcal{D}|}{|\mathcal{C} + \mathcal{D}|} > 1 \tag{VI.29}$$

Hence

$$\frac{|\Delta\alpha_1|}{|\Delta\alpha_2|} > 1 \quad \Rightarrow \boxed{|\Delta\alpha_1| > |\Delta\alpha_2|} \tag{VI.30}$$

□

The introduced lemma has proved the locality property of RBF-SVM during learning for the pilot case in Fig.VI.2. However, the pilot case is rather simple and it is hard to extend the same proof to larger number of support vectors as the formulas get much more involved. Therefore, in the next section we will follow another approach to prove the locality property for RBF-SVMs with large number of support vectors.

VI.3.3 Validation of Locality via Visualization

In this section we follow an approach in which we prove the locality property for problems with large number of support vectors via visualization. To do this we will start with RBF-SVMs trained on real world data sets and then use the closed form formulation in (VI.12) and (VI.13) to calculate the sensitivity β_{s_i} of each support vector in the SVM to the addition of a new sample $\mathbf{x}_c$ to the training set. Then by visualizing β_{s_i} of each support vector for all possible values for $\mathbf{x}_c$ we can investigate whether the locality during learning holds or not. Two data sets have been used for this experiment, FourClass (2D) and Thyroid (5D), see Table VI.1.

In Fig.VI.3, each row depicts the sensitivity $\beta_{s_i} = \frac{\Delta\alpha_{s_i}}{\Delta\alpha_c}$ of a support vector $\mathbf{x}_{s_i}$ to all possible positions of a new training sample $\mathbf{x}_c$ in $\mathbb{R}^2$ within the FourClass data set values range. The left column depicts the values as hight (3D) while the right column depicts them in color (2D). The support vector of interest $\mathbf{x}_{s_i}$ is depicted with a yellow circle on the right column of Fig.VI.3. The rest of the support vectors are represented with red and blue stars. Figure VI.3 illustrates that as the new training sample $\mathbf{x}_c$ get farther from the support vector under analysis $\mathbf{x}_{s_i}$ the 3D surfaces in the left column become flat and the corresponding colored areas in the right column have constant values. This means that when a new training sample $\mathbf{x}_c$ is in these areas the support vector of interest $\mathbf{x}_{s_i}$ will not get affected or in other words $\Delta\alpha_{s_i}$ is negligible. On the other hand, the $|\beta_{s_i}|$ values gets significantly higher when $\mathbf{x}_c$ is close to support vector of interest $\mathbf{x}_{s_i}$.

Moreover, it is shown that when $\mathbf{x}_c$ is in the neighborhood of the support vector of interest then $\beta_{s_i} < 0$ for the most part as shown in Fig.VI.3(a)(c) while $\beta_{s_i} > 0$ in Fig.VI.3(e)(g). This is due the labels of both the new training sample $\mathbf{y}_c$ and the support vector of interest $\mathbf{y}_{s_i}$ along with the negative sign in (VI.13). When $\mathbf{y}_c\mathbf{y}_{s_i} = 1$, the negative sign in (VI.13) results in the output in Fig.VI.3(a)(c) and vice versa when $\mathbf{y}_c\mathbf{y}_{s_i} = -1$ which output is shown in Fig.VI.3(e)(g).

For the Thyroid data set we visualized the values of β_{s_i} using parallel axis visualization where the data dimensions are depicted by parallel axes and each data sample is represented by a line. The values of β_{s_i} are depicted in color; the higher the value the

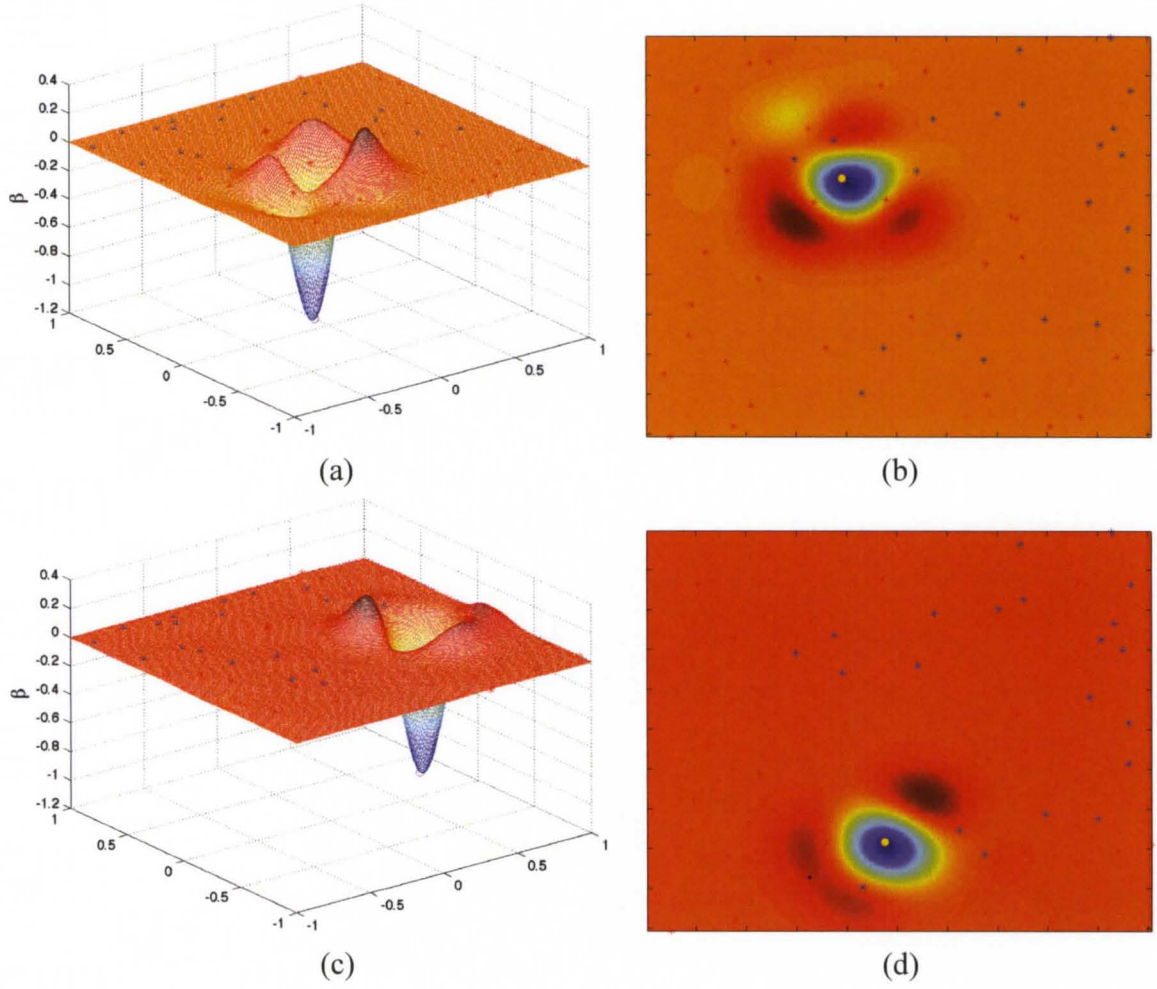


Figure VI.3: β_{s_i} versus $\mathbf{x}_c$ for the FourClass data set. Left column depicts β_{s_i} as hight and right column depicts it as color for the same support vector of interest $\mathbf{x}_{s_i}$. Support vector $\mathbf{x}_{s_i}$ is depicted as a yellow circle in the right column. The rest of support vectors are depicted with red and blue stars depending on their labels.

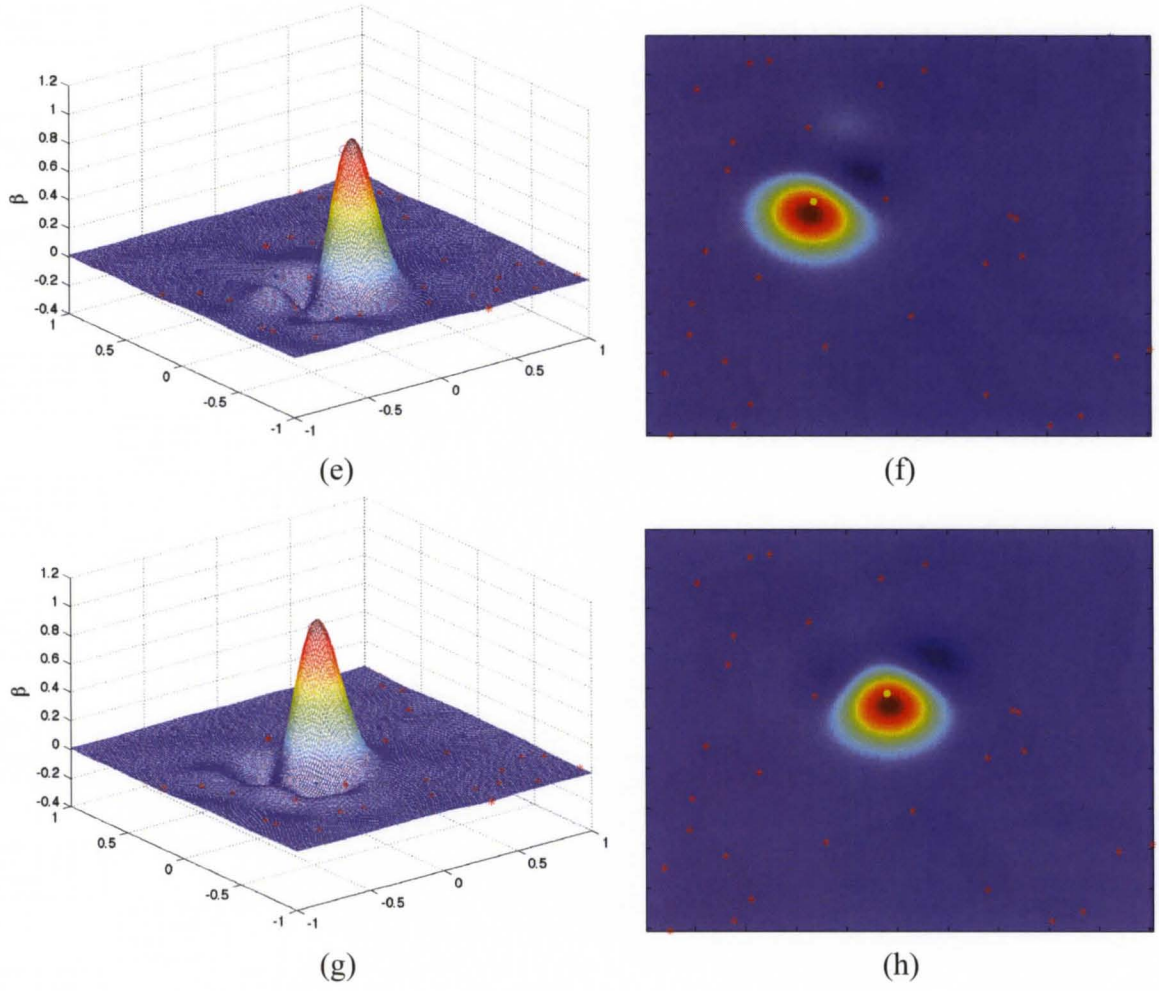


Figure VI.3: β_{s_i} versus $\mathbf{x}_c$ for the FourClass data set. Left column depicts β_{s_i} as hight and right column depicts it as color for the same support vector of interest $\mathbf{x}_{s_i}$. Support vector $\mathbf{x}_{s_i}$ is depicted as a yellow circle in the right column. The rest of support vectors are depicted with red and blue stars depending on their labels.

brighter the color. The support vector of interest $\mathbf{x}_{s_i}$ is depicted as a blue line. In Fig.VI.4, we can see that the lines (high dimensional data samples) surrounding the blue line (support vector of interest) are brighter than those away from it which again means that β_{s_i} is higher when the new sample $\mathbf{x}_c$ is close to the support vector of interest $\mathbf{x}_{s_i}$ and gets lower when $\mathbf{x}_c$ gets farther. This also supports the validity of the locality during learning.

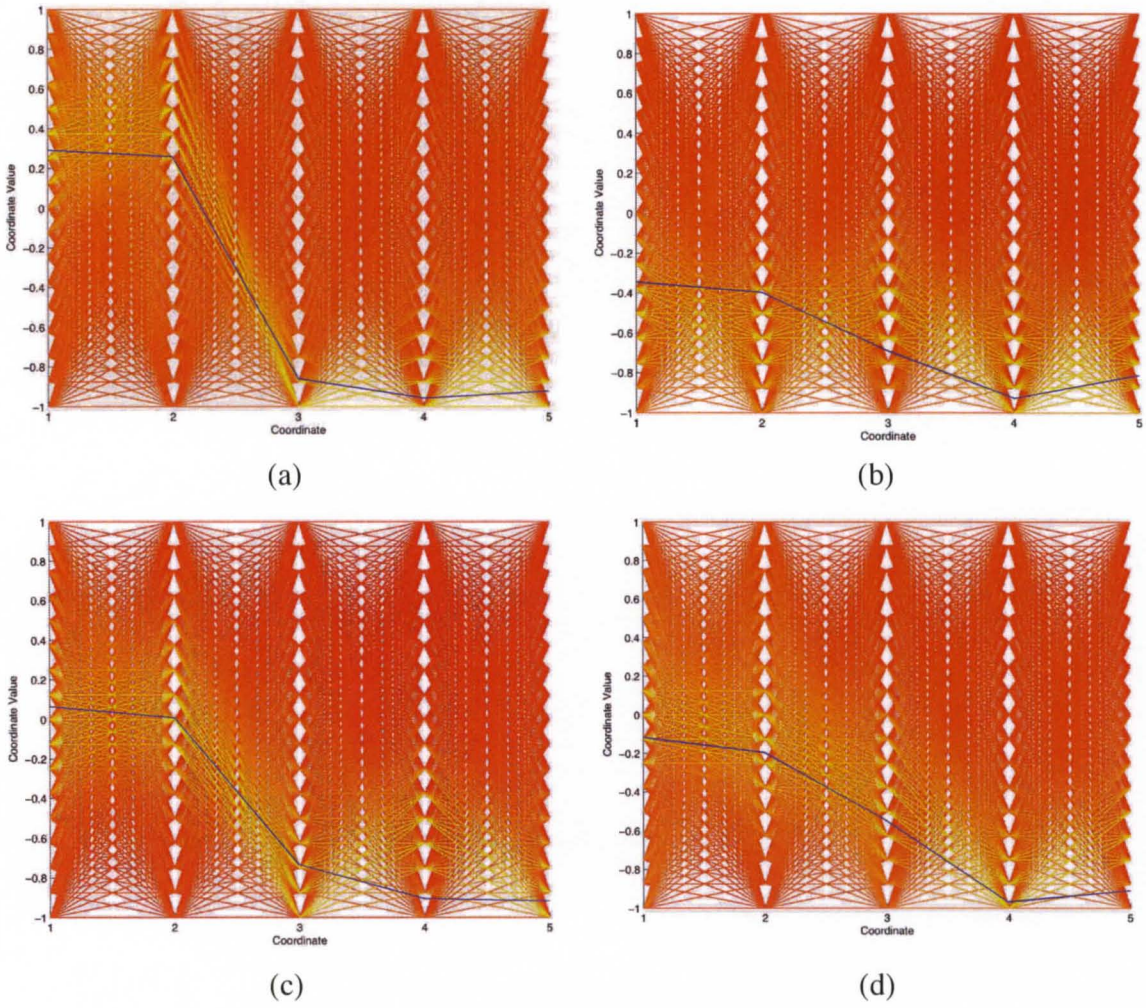


Figure VI.4: β_{s_i} versus $\mathbf{x}_c$ for the Thyroid data set using parallel axes visualization. The support vector of interest $\mathbf{x}_{s_i}$ is depicted as a blue line. The color brightness of each line is proportional to its corresponding value of β_{s_i}.

Having shown analytically that the RBF-SVM is local in nature during the learning stage, we need to quantitatively study the characteristics of the locality property so as to be able to predict the effect of adding a new sample $\mathbf{x}_c$ to the training set of an RBF-SVM.

In the next section we will present this quantitative analysis which will serve; a) as large scale experimental validation of the locality during learning , and b) will help us deduce an estimate for the ultimate neighborhood size.

VI.3.4 Experimental Validation of Locality During Training

In the previous section we focused on examining the sensitivities β_{s_i} of the changes in the support vectors Lagrangian multipliers ($\Delta\alpha_{s_i}$) with respect to the new sample Lagrangian multiplier ($\Delta\alpha_c$),

$$\beta_{s_i} = \frac{\Delta\alpha_{s_i}}{\Delta\alpha_c} \tag{VI.31}$$

This is because $\Delta\alpha_c$ can not be estimated ahead of time and is calculated during the optimization problem involved in the training process. Moreover, the analytical formulation used earlier, Sec.VI.3.1, is based on the assumption that when the new sample $\mathbf{x}_c$ is added to the training set, it will not change the current set of support vectors SV. This assumption is often violated in practice. As such, in this section we consider examining the locality of the RBF-SVM during learning from the experimental perspective where we can measure the exact values of $\Delta\alpha_{s_i}$ while taking into consideration the membership changes in the set of support vectors SV.

VI.3.4.1 Experimental Setup

The experimental setup starts with an RBF-SVM$_{old}$ trained on a data set D_{old}. RBF-SVM$_{old}$ has a set of support vectors SV_{old}. A new sample $\mathbf{x}_c$ is added to D_{old} constructing a new training data set $D_{new} = D_{old} \cup \{\mathbf{x}_c\}$. The training process is repeated on D_{new} and an RBF-SVM$_{new}$ is obtained with a set of support vectors SV_{new}. Then we get $\Delta\alpha_{s_i}$ for all the support vectors $\mathbf{x}_{s_i}$. The obtained $\Delta\alpha_{s_i}$ should takes into account the differences in membership between SV_{new} and SV_{old}; during the transformation from RBF-SVM$_{old}$ to RBF-SVM$_{new}$ some support vectors will stay common in both SV_{new} and SV_{old} while

others will vanish or emerge. Equation (VI.32) shows how $\Delta\alpha_{s_i}$ are obtained.

$$\Delta\alpha_{s_i} = \begin{cases} (\alpha_{s_i})_{new} - (\alpha_{s_i})_{old} & \forall s_i \in SV_{new} \cap SV_{old} \\ (\alpha_{s_i})_{new} & \forall s_i \in SV_{new} \setminus SV_{old} \\ -(\alpha_{s_i})_{old} & \forall s_i \in SV_{old} \setminus SV_{new} \end{cases} \tag{VI.32}$$

This process is repeated for all $\mathbf{x}_c \in SV_{new}$. Throughout the experiments we used 24 data sets, see Table VI.1, that cover a wide range of dimensions.

TABLE VI.1

Data sets used throughout experiments.

Data set	No. Features	No. Samples
Four Class [37]	2	802
Banana [60]	2	5300
Titanic [6]	3	2201
SVMGuide1 [38]	4	7089
Mamographic Mass [6]	5	830
Thyroid [6]	5	215
Diabetes [6]	8	768
Breast Cancer Wisconsin [6]	9	683
Magic Gamma Telescope [6]	10	19020
Heart [6]	13	296
Adult [6]	14	30162
Credit Approval [6]	15	653
Image Segmentation [6]	18	2310
German Credit [6]	20	1000
Twonorm	20	7400
Waveform [6]	21	5000
IJCNN1	22	191681
Ionosphere [6]	34	351
Kr-vs-Kp [6]	36	3196
Spambase [6]	57	4601
Mushrooms [6]	112	8124
Musk2 [6]	166	6598
Internet Advertisement [6]	1558	2358
Gisette [6]	5000	7000

VI.3.4.2 Experiment 1

The purpose of this experiment is to verify experimentally the existence of locality during learning of RBF-SVMs. This is achieved by showing that the changes in the values of $\alpha_{s_i}, \forall s_i \in SV_{old} \cup SV_{new}$, between SV_{old} and SV_{new}, as described in (VI.32), when a new sample $\mathbf{x}_c$ is added to D_{old}, are inversely proportional to the distance $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$, where

$$||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma = \frac{||\mathbf{x}_{s_i} - \mathbf{x}_c||}{\sigma} \tag{VI.33}$$

and σ is the standard deviation parameter of the used RBF kernel.

The choice of $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ in (VI.33) is motivated by the fact that the learning process of RBF-SVM takes place in the RBF feature space not the original Euclidean data space. Furthermore, our main goal in this work is to find an estimate of the region of effect of a new sample $\mathbf{x}_c$ on RBF-SVM$_{old}$. Hence, we ultimately need to find a similarity threshold in the RBF feature space that can be used to decide if a new data sample $\mathbf{x}_c$ has high effect on a support vector $\mathbf{x}_{s_i}$. As we do not know yet the parameters that will affect the similarity measure, we decided to use (VI.33) as it coincides with RBF feature space and meanwhile it may uncover any dependency on σ.

Figure VI.5(a-e) depict $\Delta\alpha_{s_i}$ versus $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ of several new data samples $\mathbf{x}_c$ for an RBF-SVM trained using the Image Segmentation data set. The figure shows that $\Delta\alpha_{s_i}$ decreases with increasing $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$. The large values of $\Delta\alpha_{s_i}$ when $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ is small are due to significant changes in α_{s_i} $\forall s_i \in SV_{old} \cap SV_{new}$, emerging support vectors $\mathbf{x}_{s_i}$ $\forall s_i \in SV_{new} \setminus SV_{old}$, or vanishing support vectors $\mathbf{x}_{s_i}$ $\forall s_i \in SV_{old} \setminus SV_{new}$. On the other hand, the small, almost negligible, values of $\Delta\alpha_{s_i}$ when $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ is large shows that support vectors $\Delta\alpha_{s_i}$ that are far from $\mathbf{x}_c$ are barely affected and also that new or vanishing support vectors exist only near to $\mathbf{x}_c$. In Fig.VI.5(f) we plot $\Delta\alpha_{s_i}$ versus $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ for all new samples $\mathbf{x}_c$. This is actually to show that the behavior experienced in Fig.VI.5(a-e) is not a special case.

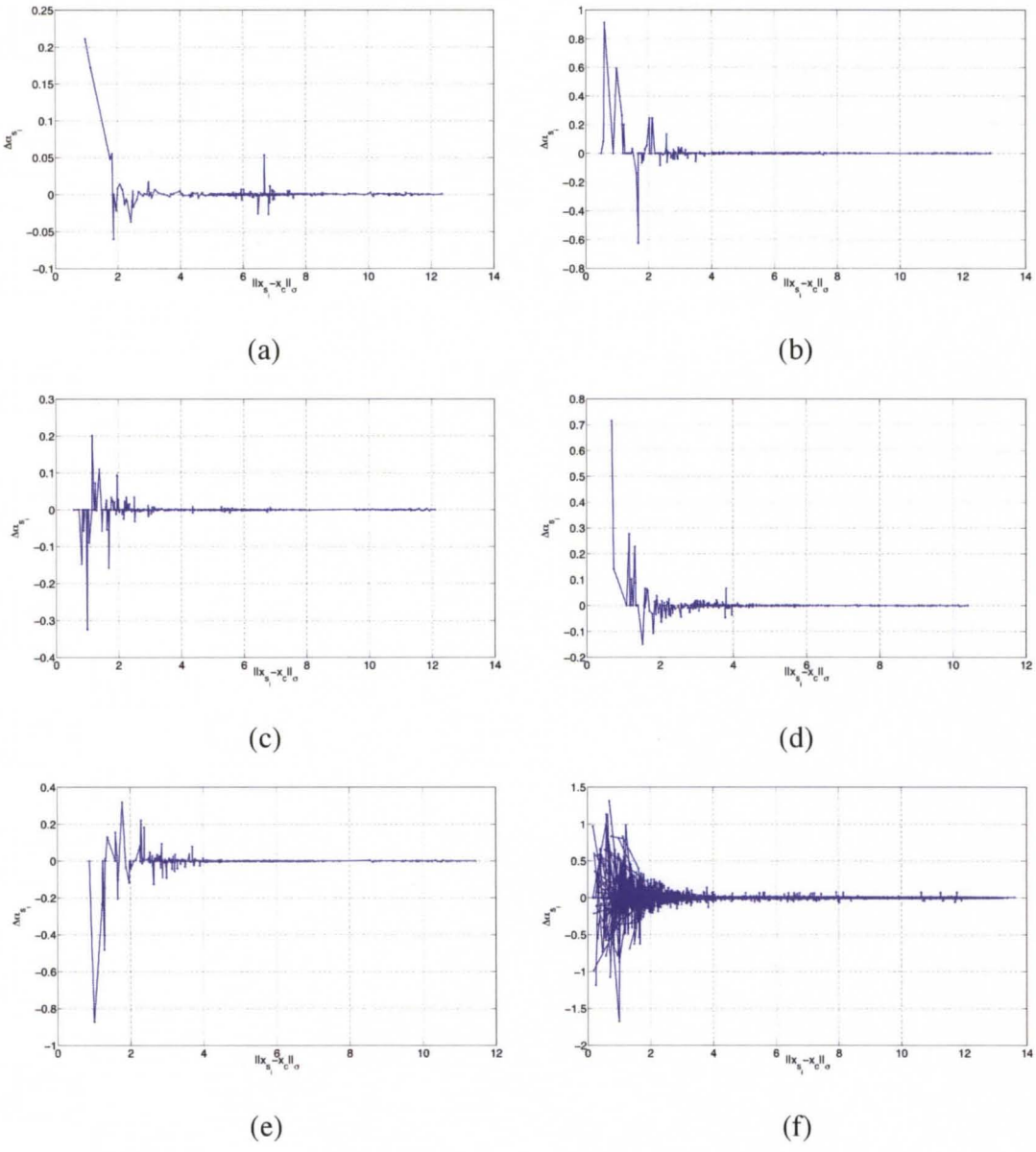


Figure VI.5: Depiction of $\Delta\alpha_{s_i}$ vs $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ for the Image Segmentation data set. (a)-(e) For individual new samples $\mathbf{x}_c$. (f) For all $\mathbf{x}_c$.

After examining Fig.VI.5, we can conclude that the RBF-SVM trained on the Image Segmentation data set shows locality during learning. The same experiment has been repeated for all the data sets in Table VI.1, see Fig.VI.6. As shown in Fig.VI.6, despite the wide variety in the characteristics of the examined data sets in terms of number of dimensions and distributions, all the data sets exhibit the same behavior of decreasing $\Delta\alpha_{s_i}$

age Segmentation data set shows locality during learning. The same experiment has been repeated for all the data sets in Table VI.1, see Fig.VI.6. As shown in Fig.VI.6, despite the wide variety in the characteristics of the examined data sets in terms of number of dimensions and distributions, all the data sets exhibit the same behavior of decreasing $\Delta\alpha_{s_i}$ versus $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$. Therefore, the same conclusion is justified for all the data sets. Hence we have verified experimentally the locality of RBF-SVMs during learning.

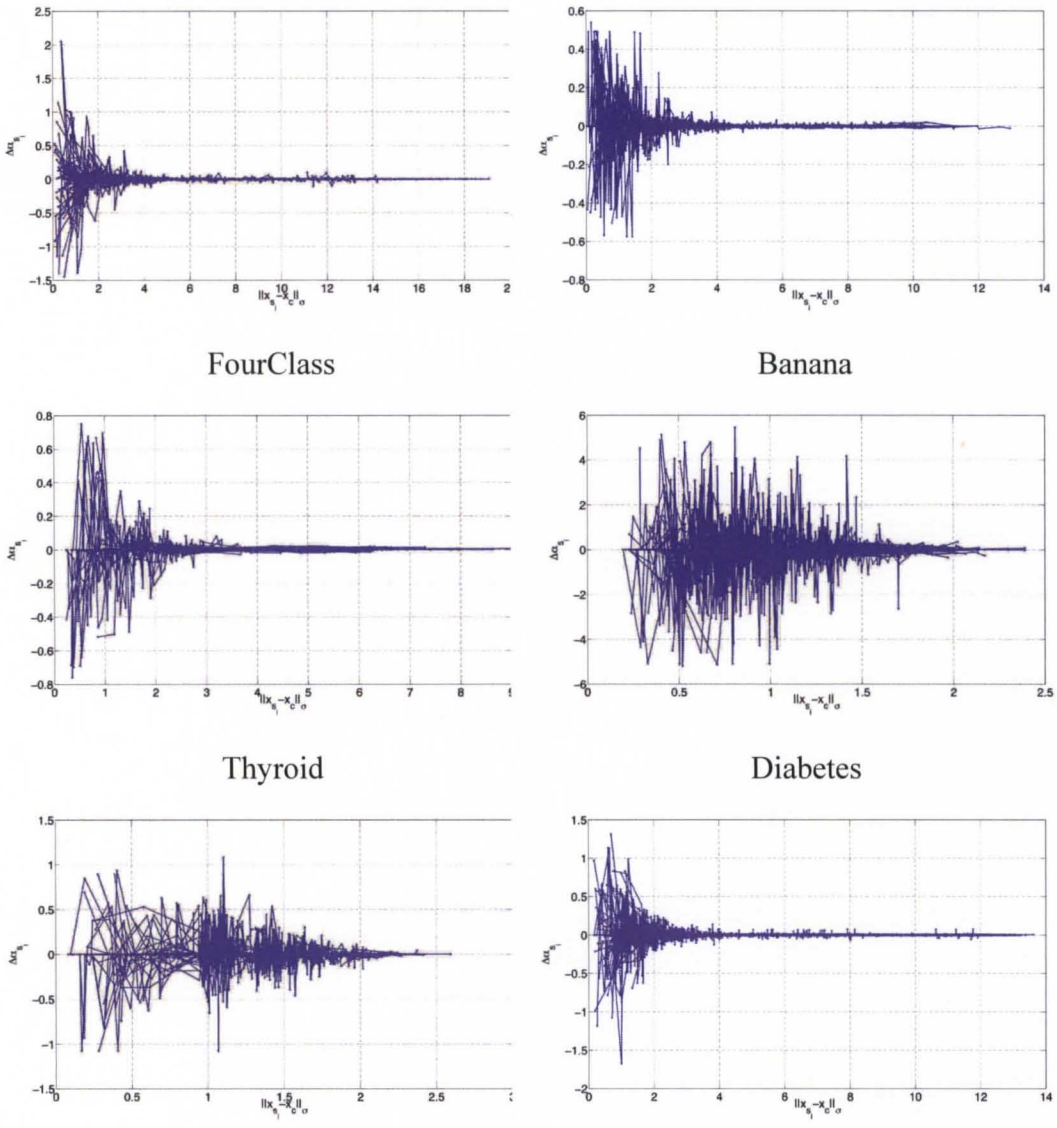
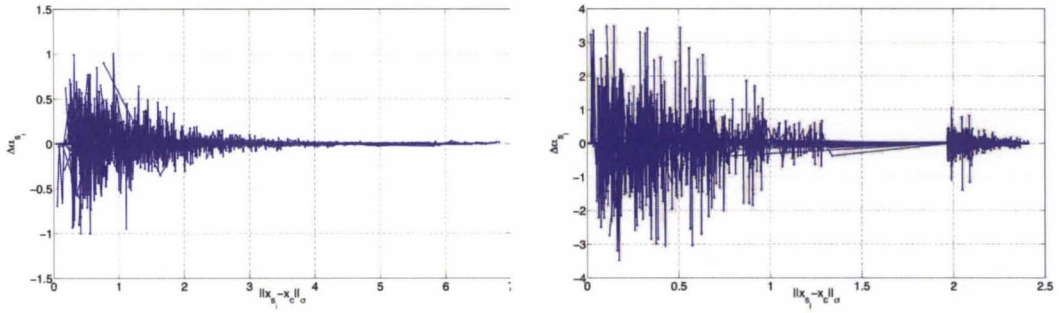
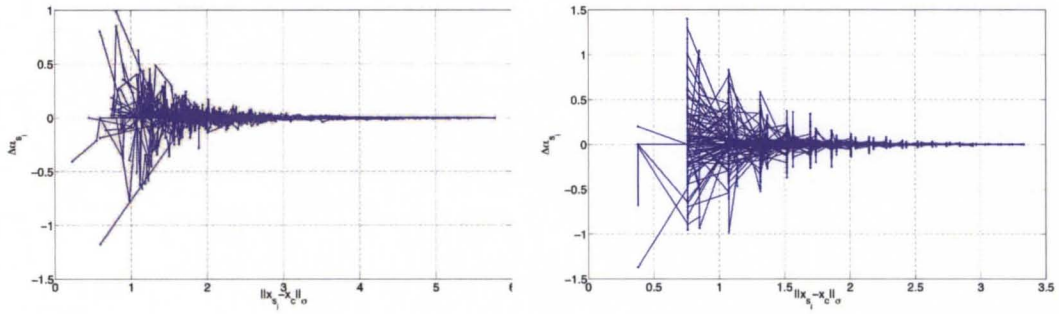


Figure VI.6: $\Delta\alpha_{s_i}$ vs $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ for all $\mathbf{x}_c$ for the data sets in Table VI.1



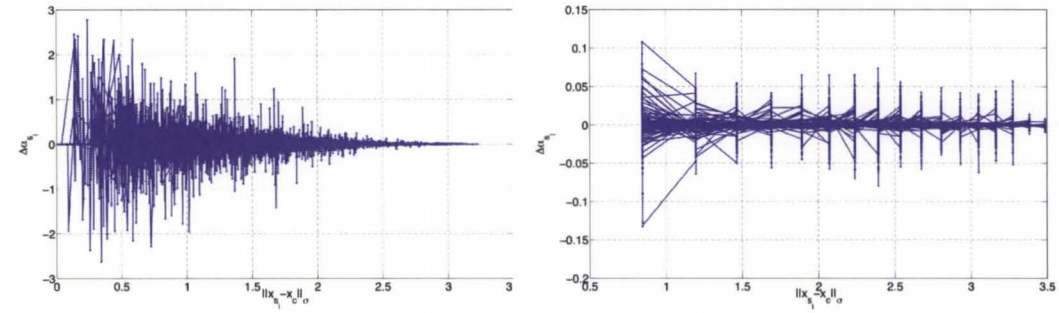
SVMGuide1

IJCNN1



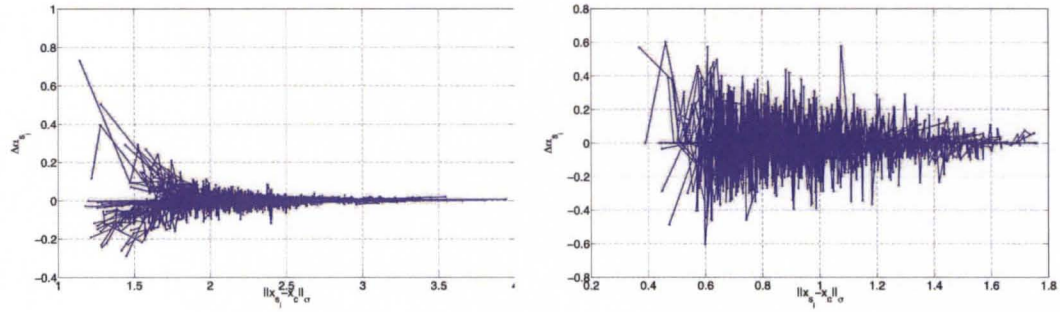
Breast Cancer Wisconsin

Kr-vs-Kp



Adult

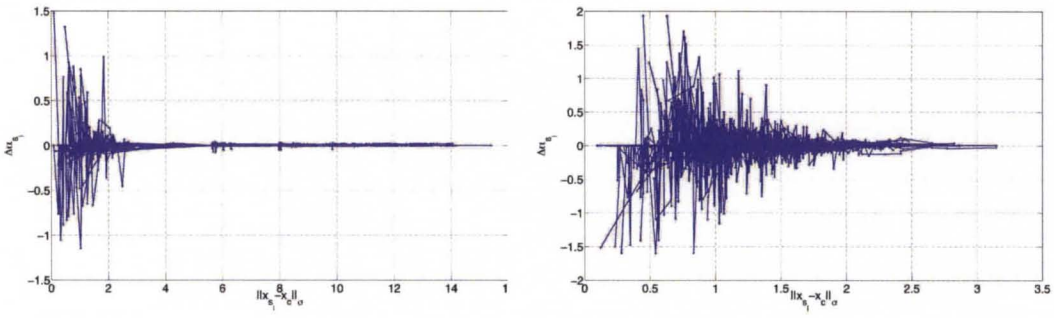
Mushrooms



Ringnorm

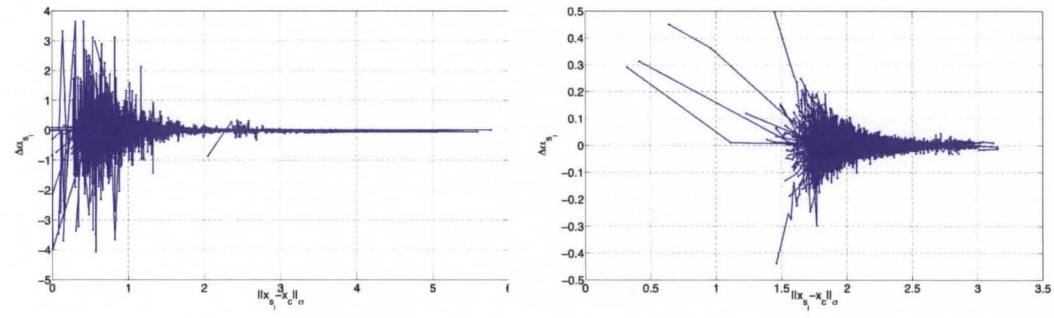
Waveform

Figure VI.6: $\Delta\alpha_{s_i}$ vs $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ for all $\mathbf{x}_c$ for the data sets in Table VI.1



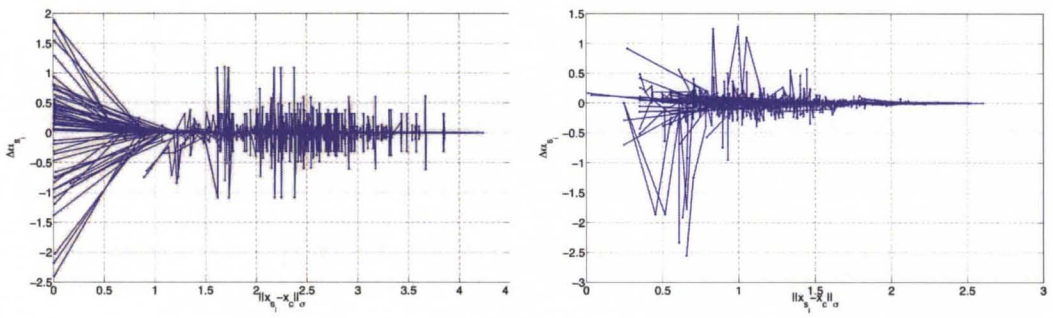
Sick

Ionosphere



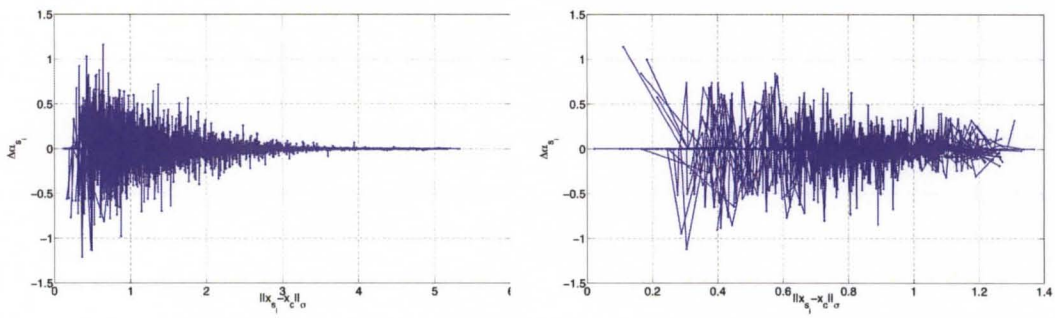
Spambase

Splice



Musk2

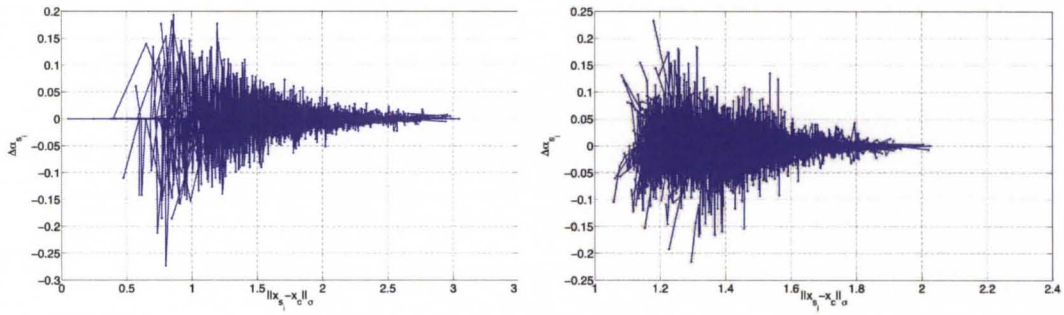
Internet Advertisement



Magic Gamma Telescope

Heart

Figure VI.6: $\Delta\alpha_{s_i}$ vs $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ for all $\mathbf{x}_c$ for the data sets in Table VI.1



German Credit Gisette

Figure VI.6: $\Delta\alpha_{s_i}$ vs $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ for all $\mathbf{x}_c$ for the data sets in Table VI.1

VI.4 Estimating the Ultimate Neighborhood Size

Now that we have analytically and experimentally verified the validity of the assumption that RBF-SVM exhibits local behavior during the learning stage, in this section we investigate also analytically and experimentally the possibility of finding an estimate of the ultimate neighborhood size required for local incremental learning of RBF-SVM.

VI.4.1 Analytical Estimate of the Ultimate Neighborhood Size

In this section we are interested in finding an estimate of the neighborhood size (we will denote it by ν_u) of most significant effect around the new training sample $\mathbf{x}_c$. For this purpose we will extend the definition of locality to be not only characterized by the absolute value of changes in the Lagrangian multipliers values, but also to include the contributions of each support vector to these absolute changes.

To this end we will again consider the simple pilot RBF-SVM in Fig.VI.7 where $\Delta\alpha_1$ and $\Delta\alpha_2$ are given in (VI.18) and (VI.19), but we will repeat them here for convenience.

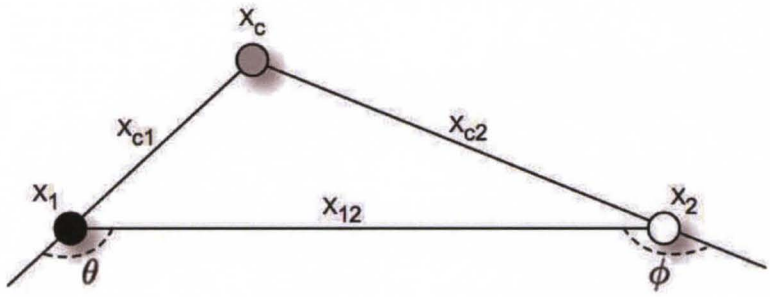


Figure VI.7: Pilot RBF-SVM with two support vectors $\mathbf{x}_1$ and $\mathbf{x}_2$. A new sample $\mathbf{x}_c$ is added to the training data set such that $\mathbf{x}_1$ is closer to $\mathbf{x}_c$ than $\mathbf{x}_2$ is.

$$\Delta\alpha_1 = \frac{-\Delta\alpha_c y_1 y_c}{2}\left[\frac{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1 - e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}} + e^{-\frac{\|X_{c2}\|^2}{2\sigma^2}}}{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1}\right] \tag{VI.34}$$

$$\Delta\alpha_2 = \frac{-\Delta\alpha_c y_2 y_c}{2}\left[\frac{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1 - e^{-\frac{\|X_{c2}\|^2}{2\sigma^2}} + e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}}}{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1}\right] \tag{VI.35}$$

Notably, each $\Delta\alpha_i$ is a function of, $\|X_{ci}\|$, the distances between the new sample $\mathbf{x}_c$ and all the support vectors, that is

$$\Delta\alpha_1 = \mathcal{F}_1(y_c, \Delta\alpha_c, RBF(\|X_{c1}\|), RBF(\|X_{c2}\|)) \tag{VI.36}$$

and

$$\Delta\alpha_2 = \mathcal{F}_2(y_c, \Delta\alpha_c, RBF(\|X_{c1}\|), RBF(\|X_{c2}\|)) \tag{VI.37}$$

Remember that the data samples $\mathbf{x}_1$ and $\mathbf{x}_2$ are not subject to change and hence y_1, y_2, and $\|X_{12}\|$ are not variables.

The basic definition of RBF-SVM locality during learning can be extended in the context of equations (VI.36) and (VI.37) and Fig.VI.7 as follows: Since $\|X_{c1}\|$ and $\|X_{c2}\|$ both contribute to $\Delta\alpha_1$ through an RBF function and as $\|X_{c1}\| < \|X_{c2}\|$, the definition of locality is extended from only,

$$\text{If} \quad \|X_{c1}\| < \|X_{c2}\| \quad \text{then} \quad |\Delta\alpha_1| > |\Delta\alpha_2| \tag{VI.38}$$

to also include a inversely proportional relationship between the closeness of a support vector to the new sample and its contribution to the changes in the values of the Lagrangian

multipliers. In other words, the change of the Lagrangian multiplier of each support vector will be most sensitive to the support vector that is closest to the new sample (in terms of distance). That is, for the case in Fig.VI.7, the following inequality is satisfied,

$$\text{If } \|X_{c1}\| < \|X_{c2}\| \text{ then } \left| \frac{\partial \frac{\Delta\alpha_1}{\Delta\alpha_c}}{\partial \|X_{c1}\|} \right| > \left| \frac{\partial \frac{\Delta\alpha_1}{\Delta\alpha_c}}{\partial \|X_{c2}\|} \right| \tag{VI.39}$$

In (VI.39) we used $\frac{\Delta\alpha_1}{\Delta\alpha_c}$ rather than $\Delta\alpha_1$ as $\Delta\alpha_c$ is only known through solving the SVM optimization problem involved. This however does not affect what we are trying to do.

In the following lemma we show that (VI.39) holds true but with one more constraint on the relationship between $\|X_{c1}\|$ and $\|X_{c2}\|$, that is

$$\text{If } \|X_{c1}\| < m\sigma < \|X_{c2}\| \text{ then } \left| \frac{\partial \frac{\Delta\alpha_1}{\Delta\alpha_c}}{\partial \|X_{c1}\|} \right| > \left| \frac{\partial \frac{\Delta\alpha_1}{\Delta\alpha_c}}{\partial \|X_{c2}\|} \right| \tag{VI.40}$$
$$\text{such that } 1 < m \text{ and } \sigma \text{ is RBF}_\sigma.$$

What the lemma says is that, for the RBF-SVM with two support vectors in Fig.VI.7, when a new sample $\mathbf{x}_c$ is added to training data set, the extended definition of locality in (VI.39) will be violated within a distance of σ around $\mathbf{x}_c$, where σ is the width parameter of the RBF-SVM.

Before introducing the detailed proof, we would like to discuss the logic behind our reasoning: the neighborhood size is defined as the region with significant changes around the new sample $\mathbf{x}_c$. As such we can actually use the violation of the extended definition of locality in (VI.39) as a measure for the significant changes imposed on the RBF-SVM model by the introduction of $\mathbf{x}_c$ to the training data set. This will result in a lower bound for the neighborhood size of value σ (i.e. $\sigma < \nu_u$). The main benefit of establishing a lower bound on the neighborhood size is to reduce the number of iterations which will have an immediate impact on the complexity of the algorithm. Next is the statement and proof of the lemma.

Lemma 2. *Let RBF-SVM be any radial-basis-function support vector machine with a training data set* $D = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \dots\}$ *and a set of support vectors* $SV = \{(\mathbf{x}_1, \alpha_1), (\mathbf{x}_2, \alpha_2)\}$, *where* α_i *are the associated Lagrangian multipliers. Assume that a new training data sample* $\mathbf{x}_c$ *is added to* D *and a new support vector machine RBF-SVM*$_{new}$ *is constructed, such*

that $D_{new} = D \cup \{\mathbf{x}_c\}$ *and* $SV_{new} = \{(\mathbf{x}_1, \alpha_1'), (\mathbf{x}_2, \alpha_2'), (\mathbf{x}_c, \alpha_c')\}$. *If* $\|X_{c1}\| < m\sigma < \|X_{c2}\|$ *and* $m > 1$, *then* $\left|\frac{\partial \frac{\Delta\alpha_1}{\Delta\alpha_c}}{\partial \|X_{c1}\|}\right| > \left|\frac{\partial \frac{\Delta\alpha_1}{\Delta\alpha_c}}{\partial \|X_{c2}\|}\right|$, *where* $\Delta\alpha_i = \alpha_i' - \alpha_i$ *are the changes that each support vector will encounter due to the new training sample* $\mathbf{x}_c$, σ *is the width parameter of the RBF-SVM and* $\|X_{ij}\|$ *is the distance between* $\mathbf{x}_i$ *and* $\mathbf{x}_j$.

Proof. Starting with the closed form formula of $\Delta\alpha_1$,

$$\Delta\alpha_1 = \frac{-\Delta\alpha_c y_1 y_c}{2}\left[\frac{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1 - e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}} + e^{-\frac{\|X_{c2}\|^2}{2\sigma^2}}}{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1}\right] \tag{VI.41}$$

Since $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_c$ lie on a triangle, see Fig.VI.7, the relationships between $\|X_{12}\|$, $\|X_{c1}\|$, and $\|X_{c2}\|$ can be described by the law of cosines as follows,

$$\cos\theta = \frac{\|X_{c2}\|^2 - \|X_{c1}\|^2 - \|X_{12}\|^2}{2\|X_{c1}\|\|X_{12}\|} \tag{VI.42}$$

$$\cos\phi = \frac{\|X_{c1}\|^2 - \|X_{c2}\|^2 - \|X_{12}\|^2}{2\|X_{c2}\|\|X_{12}\|} \tag{VI.43}$$

Next we use (VI.42) to get a version of (VI.41) that is only in terms of $\|X_{c1}\|$ and then we get the derivative with respect to $\|X_{c1}\|$.

$$\Delta\alpha_1 = \frac{-\Delta\alpha_c y_1 y_c}{2}\left[\frac{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1 - e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}} + e^{-\frac{\|X_{12}\|^2 + 2\|X_{12}\|\|X_{c1}\|\cos\theta + \|X_{c1}\|^2}{2\sigma^2}}}{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1}\right] \tag{VI.44}$$

$$\frac{\partial \frac{\Delta\alpha_1}{\Delta\alpha_c}}{\partial \|X_{c1}\|} = \frac{-y_1 y_c}{2}\left[\frac{\frac{\|X_{c1}\|}{\sigma^2} e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}} - \frac{1}{2\sigma^2}\left(2\|X_{12}\|\cos\theta + 2\|X_{c1}\|\right) e^{-\frac{\|X_{12}\|^2 + 2\|X_{12}\|\|X_{c1}\|\cos\theta + \|X_{c1}\|^2}{2\sigma^2}}}{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1}\right] \tag{VI.45}$$

Using (VI.43) to get a version of (VI.41) that is only in terms of $\|X_{c2}\|$ and then we get the derivative with respect to $\|X_{c2}\|$.

$$\Delta\alpha_1 = \frac{-\Delta\alpha_c y_1 y_c}{2}\left[\frac{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1 - e^{-\frac{\|X_{12}\|^2 + 2\|X_{12}\|\|X_{c2}\|\cos\phi + \|X_{c2}\|^2}{2\sigma^2}} + e^{-\frac{\|X_{c2}\|^2}{2\sigma^2}}}{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1}\right] \tag{VI.46}$$

$$\frac{\partial \frac{\Delta\alpha_1}{\Delta\alpha_c}}{\partial \|X_{c2}\|} = \frac{-y_1 y_c}{2}\left[\frac{\frac{1}{2\sigma^2}\left(2\|X_{12}\|\cos\phi + 2\|X_{c2}\|\right) e^{-\frac{\|X_{12}\|^2 + 2\|X_{12}\|\|X_{c2}\|\cos\phi + \|X_{c2}\|^2}{2\sigma^2}} - \frac{\|X_{c2}\|}{\sigma^2} e^{-\frac{\|X_{c2}\|^2}{2\sigma^2}}}{e^{-\frac{\|X_{12}\|^2}{2\sigma^2}} - 1}\right] \tag{VI.47}$$

Let

$$\mathcal{R} = \frac{\frac{\partial \frac{\Delta\alpha_1}{\Delta\alpha_c}}{\partial \|X_{c1}\|}}{\frac{\partial \frac{\Delta\alpha_1}{\Delta\alpha_c}}{\partial \|X_{c2}\|}} \tag{VI.48}$$

$$\mathcal{R} = \frac{\|X_{c1}\| e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}} - \left(\|X_{12}\| \cos\theta + \|X_{c1}\|\right) e^{-\frac{\|X_{12}\|^2 + 2\|X_{12}\|\|X_{c1}\| \cos\theta + \|X_{c1}\|^2}{2\sigma^2}}}{\left(\|X_{12}\| \cos\phi + \|X_{c2}\|\right) e^{-\frac{\|X_{12}\|^2 + 2\|X_{12}\|\|X_{c2}\| \cos\phi + \|X_{c2}\|^2}{2\sigma^2}} - \|X_{c2}\| e^{-\frac{\|X_{c2}\|^2}{2\sigma^2}}} \tag{VI.49}$$

Using (VI.42) and (VI.43) again we get

$$\mathcal{R} = \frac{\|X_{c1}\| e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}} - \left(\|X_{12}\| \cos\theta + \|X_{c1}\|\right) e^{-\frac{\|X_{c2}\|^2}{2\sigma^2}}}{\left(\|X_{12}\| \cos\phi + \|X_{c2}\|\right) e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}} - \|X_{c2}\| e^{-\frac{\|X_{c2}\|^2}{2\sigma^2}}} \tag{VI.50}$$

Since by definition $\|X_{c1}\| < \|X_{c2}\|$, the following are equivalent implementations of this definition

$$\|X_{c1}\| < m\sigma < \|X_{c2}\| \quad \text{where} \quad m > 0 \tag{VI.51}$$

and

$$\|X_{c2}\| = n\|X_{c1}\| \quad \text{where} \quad n > 1 \tag{VI.52}$$

Substituting from (VI.52) in (VI.50) we get,

$$\mathcal{R} = \frac{\|X_{c1}\| e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}} - \left(\frac{n^2+1}{2}\|X_{c1}\| - \frac{1}{2}\frac{\|X_{12}\|^2}{\|X_{c1}\|}\right) e^{-\frac{n^2\|X_{c1}\|^2}{2\sigma^2}}}{\left(\frac{n^2+1}{2n}\|X_{c1}\| - \frac{1}{2n}\frac{\|X_{12}\|^2}{\|X_{c1}\|}\right) e^{-\frac{\|X_{c1}\|^2}{2\sigma^2}} - n\|X_{c1}\| e^{-\frac{n^2\|X_{c1}\|^2}{2\sigma^2}}} \tag{VI.53}$$

Multiplying numerator and denominator of $\mathcal{R}$ by $\frac{e^{\frac{\|X_{c1}\|^2}{2\sigma^2}}}{\|X_{c1}\|}$,

$$\mathcal{R} = \frac{1 - \left(\frac{n^2+1}{2} - \frac{1}{2}\frac{\|X_{12}\|^2}{\|X_{c1}\|^2}\right) e^{-\frac{(n^2-1)\|X_{c1}\|^2}{2\sigma^2}}}{\frac{1}{n}\left(\frac{n^2+1}{2} - \frac{1}{2}\frac{\|X_{12}\|^2}{\|X_{c1}\|^2}\right) - n e^{-\frac{(n^2-1)\|X_{c1}\|^2}{2\sigma^2}}} \tag{VI.54}$$

Taking the absolute and rearranging the terms, $\mathcal{R}$ becomes

$$|\mathcal{R}| = \frac{\left|1 + \left(\frac{1}{2}\frac{\|X_{12}\|^2}{\|X_{c1}\|^2} - \frac{n^2+1}{2}\right) e^{-\frac{(n^2-1)\|X_{c1}\|^2}{2\sigma^2}}\right|}{\left|n e^{-\frac{(n^2-1)\|X_{c1}\|^2}{2\sigma^2}} + \frac{1}{n}\left(\frac{1}{2}\frac{\|X_{12}\|^2}{\|X_{c1}\|^2} - \frac{n^2+1}{2}\right)\right|} \tag{VI.55}$$

Let

$$\mathcal{A} = \frac{1}{2}\frac{\|X_{12}\|^2}{\|X_{c1}\|^2} - \frac{n^2+1}{2} \quad \text{and} \quad \mathcal{B} = e^{-\frac{(n^2-1)\|X_{c1}\|^2}{2\sigma^2}} \tag{VI.56}$$

then

$$|\mathcal{R}| = \frac{|1 + \mathcal{AB}|}{\left|n\mathcal{B} + \frac{\mathcal{A}}{n}\right|} \tag{VI.57}$$

The rest of the proof will investigate the values of $\mathcal{A}$ and $\mathcal{B}$.

Since

$$\mathcal{A} = \frac{1}{2}\frac{\|X_{12}\|^2}{\|X_{c1}\|^2} - \frac{n^2+1}{2} \quad \text{and} \quad \|X_{c2}\| = n\|X_{c1}\| \quad \text{where} \quad n > 1 \tag{VI.58}$$

then

$$\mathcal{A} = \frac{\|X_{12}\|^2 - \|X_{c2}\|^2 - \|X_{c1}\|^2}{2\|X_{c1}\|^2} \tag{VI.59}$$

From Fig.VI.7 we know that

$$\begin{aligned} \|X_{c1}\| + \|X_{c2}\| &\geq \|X_{12}\| \qquad (VI.60)\\ \|X_{c1}\|^2 + \|X_{c2}\|^2 + 2\|X_{c1}\|\|X_{c2}\| &\geq \|X_{12}\|^2 \\ \|X_{c1}\|^2 + \|X_{c2}\|^2 - \|X_{12}\|^2 &\geq -2\|X_{c1}\|\|X_{c2}\| \end{aligned}$$

Since $\|X_{c1}\|^2 > 0$ then

$$\frac{\|X_{c1}\|^2 + \|X_{c2}\|^2 - \|X_{12}\|^2}{2\|X_{c1}\|^2} \geq -\frac{\|X_{c2}\|}{\|X_{c1}\|} \tag{VI.61}$$

$$-\mathcal{A} \geq -n \Rightarrow \boxed{\mathcal{A} < n} \tag{VI.62}$$

Now we will show that $\mathcal{B} < \frac{1}{n}$ is a sufficient condition for $|\mathcal{R}| > 1$ to be true.

If

$$\mathcal{B} < \frac{1}{n} \Rightarrow \boxed{n\mathcal{B} - 1 < 0} \tag{VI.63}$$

and

$$\mathcal{A} \leq n \Rightarrow \boxed{0 \leq n - \mathcal{A}} \tag{VI.64}$$

then

$$\begin{aligned} (n - \mathcal{A})(n\mathcal{B} - 1) &\leq 0 \qquad (VI.65)\\ n^2\mathcal{B} - n - n\mathcal{AB} + \mathcal{A} &\leq 0 \\ n^2\mathcal{B} + \mathcal{A} &\leq n + n\mathcal{AB} \\ n\mathcal{B} + \frac{\mathcal{A}}{n} &\leq 1 + \mathcal{AB} \end{aligned}$$

We will show later that $n\mathcal{B} + \frac{\mathcal{A}}{n} > 0$ but we will use for now as a fact. Since $n\mathcal{B} + \frac{\mathcal{A}}{n} > 0$, then

$$\begin{aligned} 1 &\leq \frac{1 + \mathcal{A}\mathcal{B}}{n\mathcal{B} + \frac{\mathcal{A}}{n}} \\ 1 &\leq \frac{|1 + \mathcal{A}\mathcal{B}|}{\left|n\mathcal{B} + \frac{\mathcal{A}}{n}\right|} \\ 1 &\leq |\mathcal{R}| \end{aligned} \tag{VI.66}$$

Hence we have proved that $\mathcal{B} < \frac{1}{n}$ is a sufficient condition for $|\mathcal{R}| > 1$ to be true. Now we will look at the details of this condition. By definition

$$\begin{aligned} \|X_{c1}\| < \quad m\sigma \quad < \|X_{c2}\| \quad \text{where} \quad m > 0 \\ \frac{\|X_{c1}\|}{m} < \quad \sigma \quad < \frac{\|X_{c2}\|}{m} \\ \frac{\|X_{c1}\|^2}{m^2} < \quad \sigma^2 \quad < \frac{\|X_{c2}\|^2}{m^2} \end{aligned} \tag{VI.67}$$

Using the upper bound on σ^2 and $\mathcal{B}$, we get the following,

$$\begin{aligned} \mathcal{B} = e^{-\frac{(n^2-1)\|X_{c1}\|^2}{2\sigma^2}} < e^{-\frac{(n^2-1)\|X_{c1}\|^2}{2\frac{\|X_{c2}\|^2}{m^2}}} &= e^{-\frac{(n^2-1)\|X_{c1}\|^2}{2\frac{n^2\|X_{c1}\|^2}{m^2}}} \\ &= e^{-\frac{m^2(n^2-1)}{2n^2}} \end{aligned} \tag{VI.68}$$

Since $\mathcal{B} < \frac{1}{n}$ and $n > 1$ then

$$\begin{aligned} e^{-\frac{m^2(n^2-1)}{2n^2}} &\leq \frac{1}{n} \\ \frac{m^2(n^2-1)}{2n^2} &\geq \ln n \\ m &\geq \sqrt{\frac{2n^2 \ln n}{n^2 - 1}} \\ m &> 1 \end{aligned} \tag{VI.69}$$

Hence,

$$\mathcal{B} < \frac{1}{n} \Rightarrow \boxed{m > 1} \tag{VI.70}$$

which concludes the proof.

Now we will introduce the only missing part of the proof, that is proving that $n\mathcal{B} + \frac{\mathcal{A}}{n} > 0$. If

$$n\mathcal{B} + \frac{\mathcal{A}}{n} > 0 \quad \text{and} \quad 0 < \mathcal{B} < \frac{1}{n} \tag{VI.71}$$

then

$$\begin{aligned}
-n^2\mathcal{B} &< \mathcal{A} \qquad (VI.72)\\
-n &< \mathcal{A}\\
-n &< \frac{\|X_{12}\|^2 - \|X_{c2}\|^2 - \|X_{c1}\|^2}{2\|X_{c1}\|^2}\\
-2n\|X_{c1}\|^2 &< \|X_{12}\|^2 - \|X_{c2}\|^2 - \|X_{c1}\|^2\\
-2n\|X_{c1}\|^2 &< \|X_{12}\|^2 - n^2\|X_{c1}\|^2 - \|X_{c1}\|^2\\
\left(n^2 - 2n + 1\right)\|X_{c1}\|^2 &< \|X_{12}\|^2\\
n\|X_{c1}\| - \|X_{c1}\| &< \|X_{12}\|\\
\|X_{c2}\| - \|X_{c1}\| &< \|X_{12}\|\\
\|X_{c2}\| &< \|X_{12}\| + \|X_{c1}\|
\end{aligned}$$

which is actually true by definition from Fig.VI.7. □

In Lemma 2, we have shown that for the pilot case in Fig.VI.7 the ultimate neighborhood size should be larger than σ, where σ is the width parameter of the RBF kernel used in RBF-SVM. This is a big step for our purposes as we now have a lower bound on the size of the neighborhood size which will significantly reduce the number of iterations to find the proper neighborhood size. However, the derived bound is for the simple pilot case and a similar bound for larger RBF-SVM configuration has been found to be very complicated to obtain. Therefore, we will use the result in Lemma 2 as a proof of the concept that it is possible to find an estimate of the ultimate neighborhood size and in the next section we will provide an experimentally derived estimate.

VI.4.2 Experimentally Estimating the Ultimate Neighborhood Size

In this section we will investigate the possibility of experimentally finding an estimate of a threshold on $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ above which the values of $\Delta\alpha_{s_i}$ are of negligible effect on the training of RBF-SVM$_{new}$. Hence, when a new sample $\mathbf{x}_c$ is added to D_{old}, there is no need to calculate all values of $\Delta\alpha_{s_i}$. However, we might just consider the ones that are closest to $\mathbf{x}_c$ which we have verified analytically and experimentally that they encounter significant changes. In other words, we need to find an estimate of the neighborhood size around $\mathbf{x}_c$ which has a significant effect on the quality of RBF-SVM$_{new}$. In the following we will try to compare the output of experiment 1 ($\Delta\alpha_{s_i}$ vs $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$) for all the data sets with the goal of finding a proper value for $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$.

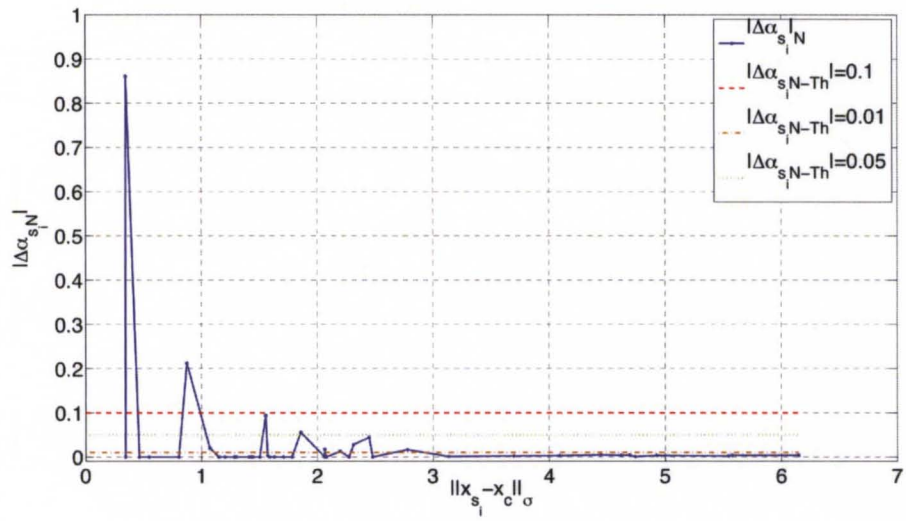
First we will start by normalizing the values of $\Delta\alpha_{s_i}$ so it would be comparable between the different data sets. From (VI.7) we know that $0 < \alpha_{s_i} < C$. Thus we denote the normalized $\Delta\alpha_{s_i}$ by

$$\Delta\alpha_{s_{iN}} = \frac{\Delta\alpha_{s_i}}{C} \tag{VI.73}$$

where $\Delta\alpha_{s_{iN}} \in [-1, 1]$. Next we will try to summarize the output of experiment 1, $\Delta\alpha_{s_{iN}}$ versus $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$, for each data set. Since we are interested in finding a threshold on $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$, we will use several threshold values for $\Delta\alpha_{s_{iN}}$ and find the average corresponding $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ for all $\mathbf{x}_c$ in each data set. Basically, this procedure finds the average $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ after which $\Delta\alpha_{s_{iN}}$ is less than a certain threshold. As the sign of $\Delta\alpha_{s_{iN}}$ is not of interest to us we will just use $|\Delta\alpha_{s_{iN}}|$. We experimented with the threshold values such that $|\Delta\alpha_{s_{iN-Th}}| \in [0, 0.5]$. Remember that as $\Delta\alpha_{s_{iN}} \in [-1, 1]$, $|\Delta\alpha_{s_{iN}}| > 0.5$ corresponds to 50% changes in $|\Delta\alpha_{s_{iN}}|$.

Figure VI.8 illustrates the thresholding procedure. Figure VI.8(a) shows $|\Delta\alpha_{s_{iN}}|$ for a new sample $\mathbf{x}_c$, same as those shown in Fig.VI.5(a-e). For illustration we will use only $|\Delta\alpha_{s_{iN-Th}}| = \{0.01, 0.05, 0.1\}$ to threshold $|\Delta\alpha_{s_{iN}}|$. The thresholding process is depicted in Fig.VI.8(b), where each shaded area represents the region in which $|\Delta\alpha_{s_{iN}}| \leq |\Delta\alpha_{s_{iN-Th}}|$. The corresponding $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ are shown by the arrows and are denoted by ν, where ν_i corresponds to $|\Delta\alpha_{s_{iN-Th}}| = i$. The process is repeated and the average is obtained for all $|\Delta\alpha_{s_{iN}}|$ corresponding to all new samples $\mathbf{x}_c$. Thus each data set has a

ν versus $\left|\Delta\alpha_{s_{iN-Th}}\right|$ that describes the average $\nu_i = ||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ needed so as to have $\left|\Delta\alpha_{s_{iN-Th}}\right| \leq i$.



(a)

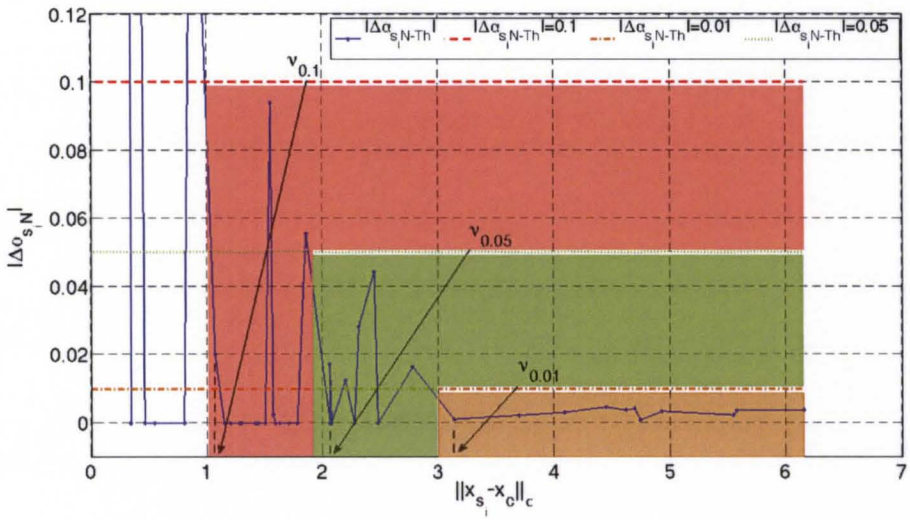


Figure VI.8: Illustration of the thresholding process. (a) $|\Delta\alpha_{s_{iN}}|$ versus $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ thresholded with $\left|\Delta\alpha_{s_{iN-Th}}\right| = \{0.01, 0.05, 0.1\}$. (b) Each shaded area shows the region where $|\Delta\alpha_{s_{iN}}| \leq \left|\Delta\alpha_{s_{iN-Th}}\right|$ and ν is the corresponding $||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$

The results of the thresholding process for all the data sets in Table VI.1 are shown in Fig.VI.9. The results can be interpreted as follows: In order to have small values for $\left|\Delta\alpha_{s_{iN-Th}}\right|$ the value of $\nu = ||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ should be large. As $\left|\Delta\alpha_{s_{iN-Th}}\right|$ gets larger, the

corresponding ν gets smaller. In other words, when a new sample $\mathbf{x}_c$ is added to D_{old} and the RBF-SVM$_{old}$ is retrained to give RBF-SVM$_{new}$, the support vectors $\mathbf{x}_{s_i}$ that will experience significant changes $|\Delta\alpha_{s_{iN}}| \leq |\Delta\alpha_{s_{iN-Th}}|$ are the ones that exist within the close neighborhood of $\mathbf{x}_c$, i.e. the ones with small $\nu = ||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$. Next we will try to make an estimate of ν for which $|\Delta\alpha_{s_{iN-Th}}|$ are of significant effect during the retraining process of RBF-SVM$_{old}$.

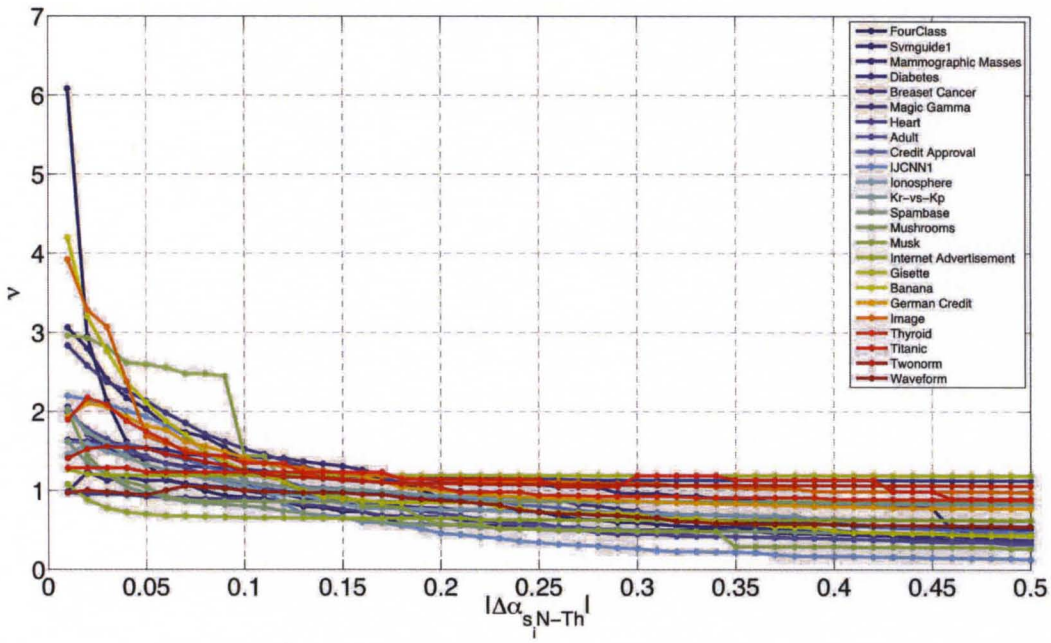


Figure VI.9: ν vs $|\Delta\alpha_{s_{iN-Th}}|$ for all data sets in Table VI.1.

Now we will propose an experimental estimate ν_u of the proper value of $\nu = ||\mathbf{x}_{s_i} - \mathbf{x}_c||_\sigma$ such that all the support vectors $\mathbf{x}_{s_i}$ with $\nu > \nu_u$ will experience negligible changes in their Lagrangian multipliers $|\Delta\alpha_{s_{iN}}| \leq |\Delta\alpha_{s_{iN-Th}}|$. Figure VI.9 illustrates that all the data sets behaves very similarly; the values of ν decay almost exponentially with $|\Delta\alpha_{s_{iN-Th}}|$. This unanimous behavior, despite the wide variety in the characteristics of the data sets with respect to both dimension and distribution, motivated us to propose a global model of ν versus $|\Delta\alpha_{s_{iN-Th}}|$. This model is shown in Fig.VI.10. It is basically the average ν versus $|\Delta\alpha_{s_{iN-Th}}|$ curve for all the curves in Fig.VI.9. Figure VI.10 shows the average curve along with the two standard deviation curves.

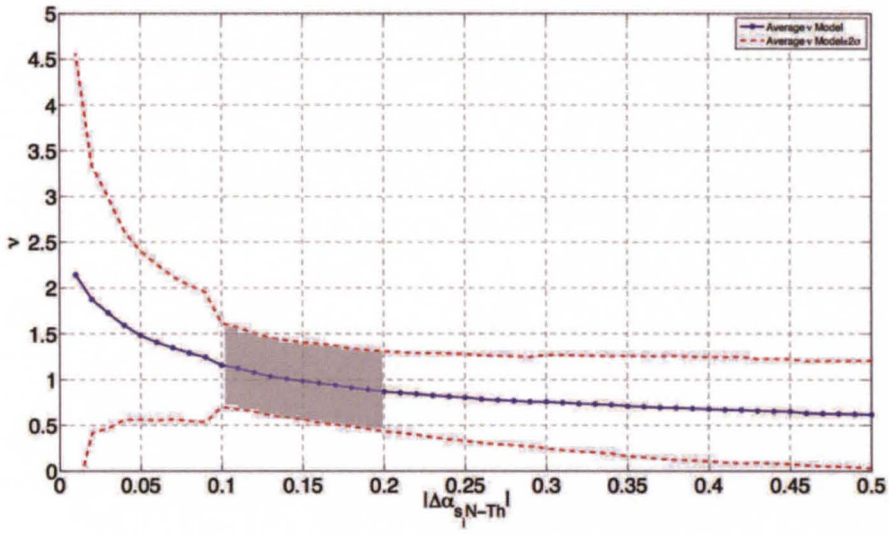


Figure VI.10: Average ν vs $\left|\Delta\alpha_{s_{iN-Th}}\right|$ curve for all data sets along with the two standard deviation curves. Shaded region is the area where the deviation from the average curve is minimal.

Without doubt a universal estimate is not the ultimate way to find the proper values for ν. However, it is always desirable to reach a compromise between complexity and accuracy of an algorithm. In the case of local incremental learning of RBF-SVM [59], performance is the main focus meanwhile complexity is significant due to the iterative trials to find the proper value of ν. We will show that the universal estimate ν_u is a proper way to establish the accuracy/complexity compromise. Furthermore, ν_u can be used as an initialization for local incremental learning of RBF-SVM in [59]. This initialization will decrease significantly the number of iterations in the algorithm.

To estimate ν_u using the model shown in Fig.VI.10, we propose to use the model where it best fits all the data sets. That is where the deviations around the average model are minimal. Figure VI.10 shows that this occurs when $\left|\Delta\alpha_{s_{iN-Th}}\right| \in [0.1, 0.2]$. As the average ν versus $\left|\Delta\alpha_{s_{iN-Th}}\right|$ curve in Fig.VI.10 is almost linear when $\left|\Delta\alpha_{s_{iN-Th}}\right| \in [0.1, 0.2]$, we will choose $\left|\Delta\alpha_{s_{iN-Th}}\right| = 0.15$ which has corresponding $\nu_u = \sigma$, where σ is the standard deviation of the used RBF kernel.

Remembering that $|\Delta\alpha_{s_{iN}}| \leq \left|\Delta\alpha_{s_{iN-Th}}\right|$ and that $\Delta\alpha_{s_{iN}} \in [-1, 1]$, the proposed

universal model is described by $|\Delta\alpha_{s_{iN}}| \leq 0.15$ and $\nu_u = \sigma$ which can be interpreted as follows: When performing incremental learning with RBF-SVM, it is generally common that for all support vectors $\mathbf{x}_{s_i}$ with $\nu > \sigma$ will experience changes in their normalized Lagrangian multipliers that are on average bounded by $|\Delta\alpha_{s_{iN}}| \leq 0.15$. In other words, support vectors outside he neighborhood of size σ from the new training sample will experience less than 15% change in their Lagrangian multipliers. In order for this universal model to be useful for local incremental learning, we need to show that ignoring the changed support vectors outside the σ neighborhood is tolerable. We do that through an experiment that finds the correct classification rate (CCR) of the RBF-SVM versus $\left|\Delta\alpha_{s_{iN-Th}}\right|$. Figure VI.11 shows the outcome of this experiment for the data sets in Fig.VI.10 (using the same color code). We can see that for almost all the depicted data sets the performance of the RBF-SVM does not deteriorate for $\left|\Delta\alpha_{s_{iN-Th}}\right| = 0.15$ and even for the one data set that showed decrease in quality (IJCNN1), the decrease in CCR is within 3% which is relatively insignificant.

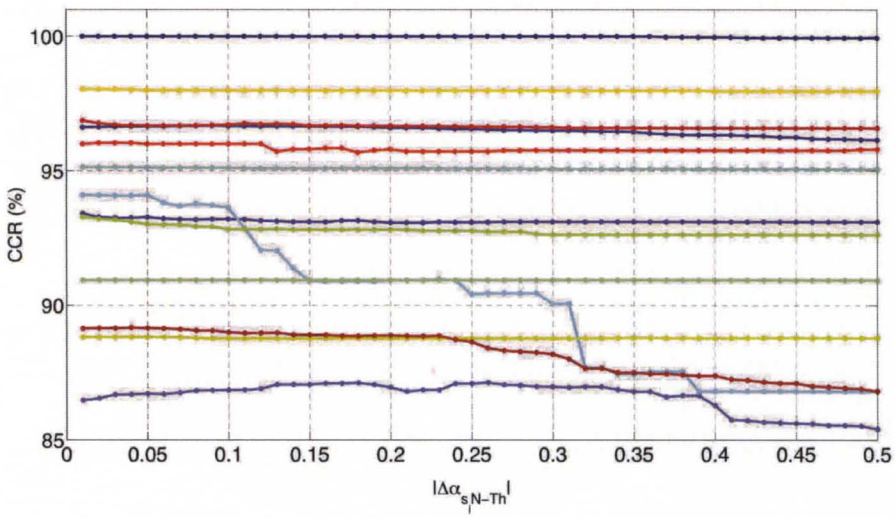


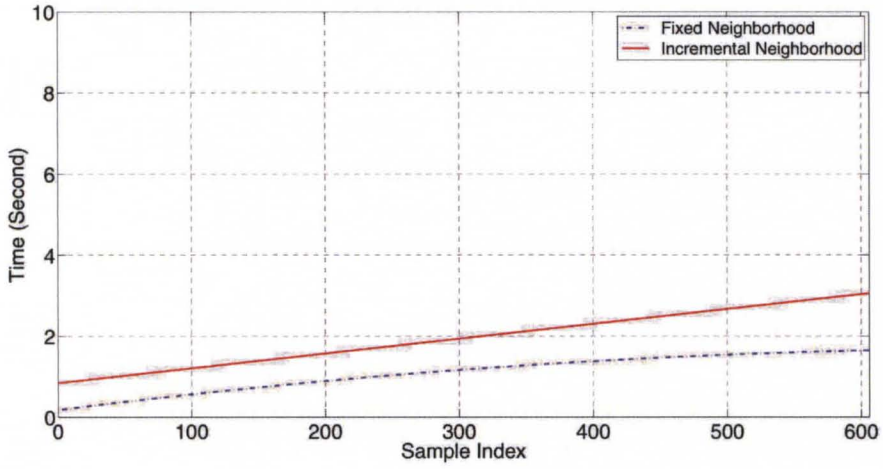
Figure VI.11: Results of Correct classification rate (CCR) vs $\left|\Delta\alpha_{s_{iN-Th}}\right|$.

VI.4.3 Experiments with the Universal Neighborhood Size

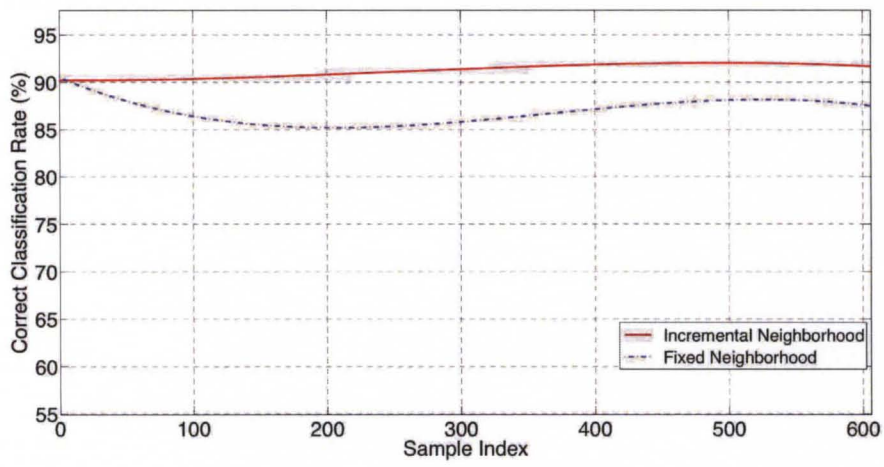
In this section we will examine the applicability of the proposed universal neighborhood size $\nu_u = \sigma$. To this end we will compare the performance of the local incremental RBF-SVM learning algorithm [59] using the proposed universal $\nu_u = \sigma$ as a fixed neighborhood size and compare it with the iterative neighborhood construction. For this experiment we will use the COIL2 [20] data set which was not used during our previous experiments. The algorithm is evaluated in both cases with respect to speed and correct classification rate. Initially an RBF-SVM is trained using 20% of the data set, 40% of the data set is used to estimate the correct classification rate, and finally the remaining 40% are used for incremental learning. The results of the experiments are shown in Fig.VI.12. In Fig.VI.12(a) we see that the proposed universal neighborhood size is more efficient from the speed perspective. Moreover, it is notable that the slope for the iterative neighborhood construction is higher which will make the proposed ν_u even more efficient when the incremental learning continues for long times. On the other hand, Fig.VI.12(b) shows that the iterative neighborhood construction out performs the fixed ν_u with respect to the classification rate. In fact, this is a natural result considering that ν_u is a universal estimate. On the contrary, the results in Fig.VI.12(b) supports the applicability of ν_u as it shows that it maintains a good correct classification rate that is comparable to that of the iterative neighborhood construction. This again suggests using ν_u as an initial neighborhood size for the iterative neighborhood construction to significantly lower the number of iterations.

VI.5 Discussion

In this chapter we verified via analytical analysis and experiments the locality of RBF-SVM during learning which is an important property for incremental learning algorithms as it makes the model updating process during learning increments local rather than global. We also presented a analytical lower bound estimate on the ultimate neighborhood size ($\nu > \sigma$) as well as an experimentally derived one ($\nu = \sigma$). We see that despite the analytical lower bound was derived for a pilot case, its value is consistent with the experimentally derived estimate. This gives high confidence in the correctness of this estimate.



(a)



(b)

Figure VI.12: Results comparison of applying local incremental RBF-SVM [59] with both the proposed universal neighborhood estimate ν_u and the iterative construction with respect to (a) Speed (b) Correct classification rate.

When the universal estimate ν_u of the size of neighborhood for local incremental RBF-SVM was compared with the iterative neighborhood construction, the universal estimate showed superior performance with respect to speed. Meanwhile its performance with respect classification rate was very comparable to the iterative construction.

CHAPTER VII

CONCLUSION AND FUTURE WORK

In this work, we focused on what we believe will be the next generation of learning algorithms. This is where learning machines are set free in an environment to gather data, model it, and occasionally stop for a feedback session with an oracle. The *Never-Ending Learning* framework developed in this dissertation is just the initial step towards such imagination for where the future of machine learning is heading.

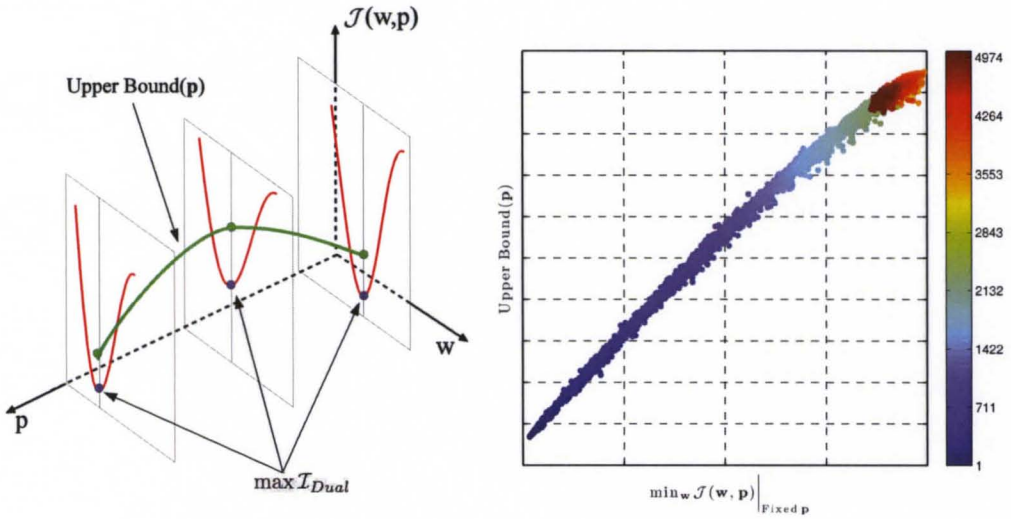
The main contribution of this dissertation is a unified *Never-Ending Learning* framework for classifying streaming data. The developed framework can ideally work on infinite long data streams, with the streaming data partially labeled. Moreover, the developed active learning establishes a mechanism for feedback from a supervising oracle.

To decide on the underlying learning machine for the framework, our criteria was that the chosen algorithm should be powerful in terms of generalization performance and applicability. Moreover, we were eager to make a novel contribution in each component we consider. This is basically why we have chosen SVM as the underlying model of to framework. The generalization performance of SVM is well established and therefore many algorithms are proposed for the semi-supervised learning problem. However, most of them suffer local minima or bad time efficiency.

The dissertation presented the QP-S^3VM algorithm which is a novel formulation of the S^3VM in terms of standard quadratic programming optimization. We showed that this new formulation simplifies the S^3VM problem to a concave quadratic programing problem as illustrated in Fig.VII.1(a). Furthermore, the extensive experiments conduced to validate the QP-S^3VM model, see Fig.VII.1(b), showed the model to be a solid surrogate for the S^3VM problem.

Through the analysis and mathematical interpretation of the QP-S^3VM model, we

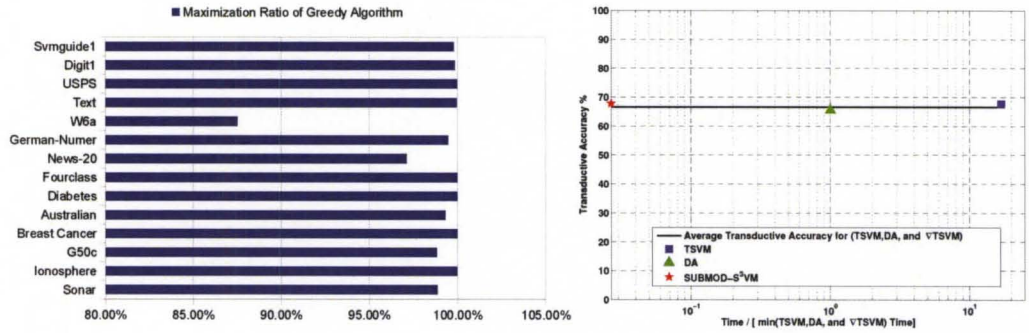
presented an intuitive explanation of the relationship between the low-density separation and graph based methods for semi-supervised learning. What we concluded is that QP-S^{3}VM also performs semi-supervised learning via label propagation, in the same fashion as the graph based algorithms. The difference lies in the insistence of QP-S^{3}VM on assigning district labels in order to avoid creating a decision boundary passing through a data cluster. Whereas, the graph based methods tend to give soft indiscriminate label assignments.



(a) (b)

Figure VII.1: (a) Illustration of the proposed upper bound function with respect to the S^{3}VM. (b) Sample of the upper bound validation for News20.Binary data set.

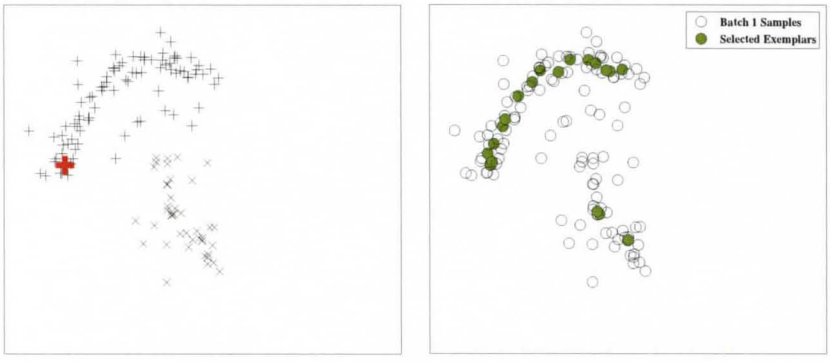
The dissertation also introduced the idea of *submodular set functions optimization* to the problem of semi-supervised learning. Transforming the QP-S^{3}VM into a submodular optimization function (SUBMOD-S^{3}VM) entailed many iterations of careful design to avoid affecting the solution of the original problem. Theoretically, using the greedy approach to maximize a monotone submodular function entails achieving a solution that is at least 63% of the optimal maximum solution. However, in our experiments we mostly achieved above 98% approximation. The lowest approximation achieved was 87.5% of the optimal solution, see Fig.VII.2(a). The transductive accuracy of the developed SUBMOD-S^{3}VM algorithm was shown to be better than the literature state of the art, while achieving up to two orders of magnitude speed up. A sample of such performance is illustrated in Fig.VII.2(b) for the News20.Binary data set.



(a) (b)

Figure VII.2: (a) Sample approximation achieved by SUBMOD-S^{3}VM. (b) Sample accuracy and time efficiency of SUBMOD-S^{3}VM vs the literature state of the art for News20.Binary data set.

The dissertation further introduced a stream summarization algorithms via exemplars selection (QP/SUBMOD-EXMP). These algorithm provide any label propagation semi-supervised learning algorithm with a mechanism for achieving constant time and storage complexity during online learning. The QP/SUBMOD-EXMP algorithms achieve stream summarization by choosing exemplars that preserve the inherent data structures necessary for semi-supervised learning in terms of: a) Outlining dense regions in the data space, and b) establishing label propagation paths from labeled samples to unlabeled ones. Figure VII.3 provides a sample output for the QP/SUBMOD-EXMP algorithms.

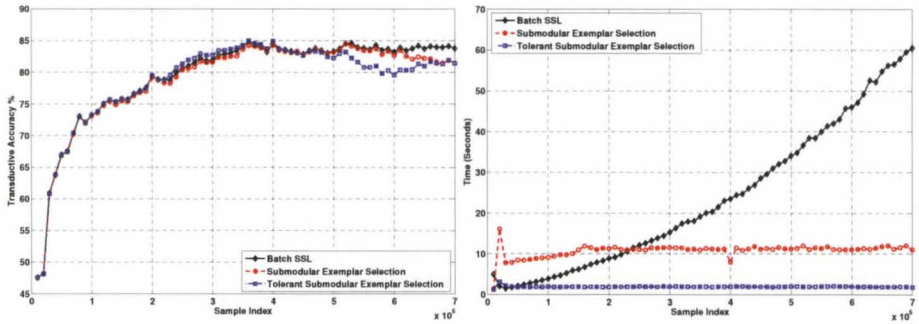


(a) (b)

Figure VII.3: (a) Sample batch input to the QP/SUBMOD-ACTV algorithm. (b) Sample exemplars selected by the QP/SUBMOD-ACTV algorithm.

The experiments on using SUBMOD-EXMP with SUBMOD-S^{3}VM for incremen-

tal learning from streaming data has showed no significant difference in transductive accuracy from the batch learning where all data is provided for the SUBMOD-S^{3}VM, see Fig.VII.4(a) for sample results on the RCV1.Binary data set. However, the use of SUBMOD-EXMP achieved constant storage and time complexity, as can be seen in Fig.VII.4(b).



(a) (b)

Figure VII.4: (a) Sample transductive accuracy achieved by using SUBMOD-ACTV and SUBMOD-S^{3}VM on the RCV1.Binary data set. (b) Corresponding time complexity.

Under the same submodular optimization framework, in Chapter V we provide an active learning algorithm that constitute the feedback between the learning machine and an oracle. The QP/SUBMOD-ACTV algorithms selects samples that QP/SUBMOD-S^{3}VM classify with low confidence. These include samples that are far from dense regions or those existing in dense but are far from any labeled samples. The experiments showed that the SUBMOD-ACTV that significantly improve the transductive accuracy of the SUBMOD-S^{3}VM algorithm.

All the contributions described so far work under the transductive learning paradigm, where only the labels of the unlabeled samples are produced as the output of the learning process. However, no model is available to classify new never seen data. In Chapter VI we presented an inductive incremental learning algorithm for supervised SVM. This algorithm uses the properties of local kernels (e.g. RBF) to perform local and efficient updates to an SVM model. The main contribution of this chapter is that we have proved analytically and illustrated experimentally that the well known locality of the RBF-SVM during the testing stage is actually transferrable to the training stage. The provided contributions in Chapter VI complements the proposed *never-ending learning* framework by providing a

methodology to keep an inductive model of the data stream.

VII.0.1 Directions for Future Work

The *Never-Ending Learning framework* presented has many possibilities for future extensions as summarized below:

- The current framework considers binary classification problems only. Therefore, all the multi-class problems are dealt with in the form of one-vs-one or one-vs-all manner. Extending the framework to handle inherently multi-class problems will entail using the submodular maximization greedy algorithms with multiple classes. This is known in the literature as the *welfare problem*. Establishing the connection between multi-class SUBMOD-S^3VM and the welfare problem is a very promising and exciting line of work. This is especially true as the S^3VM is not known to handle several classes gracefully as the graph-based semi-supervised learning algorithms do.

- Throughout the dissertation we used manual design to transform quadratic programming problems into monotone submodular ones. It will make an interesting extension to investigate techniques that learns the monotone submodular functions directly from the data.

- The presented learning framework considered only a supervised form of feedback from the oracle to the learning machine, where the labels of actively selected samples are revealed to the learning machine. We propose to investigate other forms of unsupervised feedback, where the learning machine requests the relationship between a set of samples not their labels. In this scenario, the oracle will respond by telling if the samples should belong to the same class or different classes without revealing the actual labels.

REFERENCES

[1] Advances in Neural Information Processing Systems 17 [Neural Information Processing Systems, NIPS 2004, December 13-18, 2004, Vancouver, British Columbia, Canada] (2004)

[2] Ageev, A.A., Sviridenko, M.I.: An 0.828–approximation algorithm for the uncapacitated facility location problem. Discrete Appl. Math. 93(2-3), 149–156 (1999)

[3] Amit, Y., Fink, M., Srebro, N., Ullman, S.: Uncovering shared structures in multiclass classification. In: ICML '07: Proceedings of the 24th international conference on Machine learning. pp. 17–24. New York, NY, USA (2007)

[4] Ando, R.K., Zhang, T.: A framework for learning predictive structures from multiple tasks and unlabeled data. Journal of Machine Learning Research 6, 1817–1853 (2005)

[5] Argyriou, A., Evgeniou, T., Pontil, M.: Multi-task feature learning. In: Advances in Neural Information Processing Systems 19, pp. 41–48. MIT Press, Cambridge, MA (2007)

[6] Asuncion, A., Newman, D.: UCI machine learning repository (2007)

[7] Banko, M., Etzioni, O.: Strategies for lifelong knowledge extraction from the web. In: K-CAP '07: Proceedings of the 4th international conference on Knowledge capture. pp. 95–102. New York, NY, USA (2007)

[8] Belkin, M., Matveeva, I., Niyogi, P.: Regularization and semi-supervised learning on large graphs. In: Taylor, J.S., Singer, Y. (eds.) COLT. Lecture Notes in Computer Science, vol. 3120, pp. 624–638. Springer (2004)

[9] Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. Journal of Machine Learning Research 7, 2399–2434 (November 2006)

[10] Bennett, K.P., Demiriz, A.: Semi-supervised support vector machines. In: Proceedings of the 1998 conference on Advances in neural information processing systems II. pp. 368–374. Cambridge, MA, USA (1999)

[11] Bickel, S., Brückner, M., Scheffer, T.: Discriminative learning for differing training and test distributions. In: ICML '07: Proceedings of the 24th international conference on Machine learning. pp. 81–88. New York, NY, USA (2007)

[12] Bickel, S., Scheffer, T.: Dirichlet-enhanced spam filtering based on biased samples. In: Advances in Neural Information Processing Systems 19, pp. 161–168. Cambridge, MA (2007)

[13] Bordes, A., Ertekin, S., Weston, J., Bottou, L.: Fast kernel classifiers with online and active learning. Journal of Machine Learning Research 6, 1579–1619 (2005)

[14] Calinescu, G., Chekuri, C., Pál, M., Vondrák, J.: Maximizing a submodular set function subject to a matroid constraint (extended abstract). In: IPCO '07: Proceedings of the 12th international conference on Integer Programming and Combinatorial Optimization. pp. 182–196. Springer-Verlag, Berlin, Heidelberg (2007)

[15] Caruana, R.: Multitask learning. Machine Learning 28(1), 41–75 (1997)

[16] Cauwenberghs, G., Poggio, T.: Incremental and decremental support vector machine learning. In: Fourteenth Annual Conference on Neural Information Processing Systems (NIPS 2000). pp. 409–415 (2000)

[17] Chapelle, O., Sindhwani, V., Keerthi, S.S.: Optimization techniques for semi-supervised support vector machines. Journal of Machine Learning Research 9, 203–233 (02 2008)

[18] Chapelle, O., Zien, A.: Semi-supervised classification by low density separation. In: Tenth International Workshop on Artificial Intelligence and Statistics. pp. 57–64 (01 2005)

[19] Chapelle, O., Chi, M., Zien, A.: A continuation method for semi-supervised svms. In: ICML '06: Proceedings of the 23rd international conference on Machine learning. pp. 185–192. New York, NY, USA (2006)

[20] Chapelle, O., Schölkopf, B., Zien, A.: Semi-supervised learning. MIT Press, Cambridge, Mass. (2006)

[21] Chapelle, O., Sindhwani, V., Keerthi, S.S.: Branch and bound for semi-supervised support vector machines. In: Twentieth Annual Conference on Neural Information Processing Systems (NIPS 2006). pp. 217–224. Cambridge, MA, USA (09 2007)

[22] Chechetka, A., Guestrin, C.: Efficient principled learning of thin junction trees. In: Platt, J., Koller, D., Singer, Y., Roweis, S. (eds.) Advances in Neural Information Processing Systems 20, pp. 273–280. MIT Press, Cambridge, MA (2008)

[23] Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press, 1 edn. (2000)

[24] Delany, S.J., Cunningham, P., Tsymbal, A., Coyle, L.: A case-based technique for tracking concept drift in spam filtering. Knowledge-Based Systems 18(4–5), 187–195 (2005)

[25] Dhillon, I., Guan, Y., Kulis, B.: A Unified View of Kernel k-means, Spectral Clustering and Graph Cuts. Tech. Rep. TR-04-25, UTCS (Jul 2004)

[26] Emara, W., Kantardzic, M.: An approach for incremental semi-supervised svm. In: ICDMW '07: Proceedings of the Seventh IEEE International Conference on Data Mining Workshops. pp. 539–544. Omaha, NE, USA (2007)

[27] Emara, W., Kantardzic, M.: Incremental learning using partly labeled data. In: CATA 2008: Proceesings of the 23rd International Conference on Computers and Their Applications. pp. 12–17. Cancun, Mexico (2008)

[28] Emara, W., Kantardzic, M.M.: Efficient approximate semi-supervised support vector machines through submodular optimization. In: wen Chen, X., Dillon, T.S., Ishbuchi,

H., Pei, J., Wang, H., Wani, M.A. (eds.) ICMLA (1). pp. 339–345. IEEE Computer Society (2011)

[29] Fan, W., Davidson, I.: On sample selection bias and its efficient correction via model averaging and unlabeled examples. In: SDM07: Proceedings of the Seventh SIAM International Conference on Data Mining. pp. 320–331. Minneapolis, Minnesota, USA (April 2007)

[30] Feige, U.: A threshold of ln n for approximating set cover (preliminary version). In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. pp. 314–318. STOC '96, ACM, New York, NY, USA (1996)

[31] Fisher, M.L., Nemhauser, G.L., Wolsey, L.A.: An analysis of approximations for maximizing submodular set functionsii. In: Cottle, R.W., Dixon, L.C.W., Korte, B., Magnanti, T.L., Todd, M.J., Allgower, E.L., Bartels, R., Chvatal, V., Dennis, J.E., Eaves, B.C., Fletcher, R., Hiriart-Urruty, J.B., Iri, M., Jeroslow, R.G., Johnson, D.S., Lemarechal, C., Lovasz, L., McLinden, L., Padberg, M.W., Powell, M.J.D., Pulleyblank, W.R., Ritter, K., Sargent, R.W.H., Shanno, D.F., Trotter, L.E., Tuy, H., Wets, R.J.B., Witzgall, C., Beale, E.M.L., Dantzig, G.B., Kantorovich, L.V., Koopmans, T.C., Tucker, A.W., Wolfe, P., Balinski, M.L., Hoffman, A.J. (eds.) Polyhedral Combinatorics, Mathematical Programming Studies, vol. 8, pp. 73–87. Springer Berlin Heidelberg (1978), 10.1007/BFb0121195

[32] Fung, G., Mangasarian, O.L.: Incremental support vector machine classification. In: SDM02: Proceedings of the Second SIAM International Conference on Data Mining. pp. 247–260. Arlington, VA, USA (April 2002)

[33] Gantz, J.: The diverse and exploding digital universe: An updated forecast of worldwide information growth through 2011. Tech. Rep. White paper, International Data Corporation (2008)

[34] Grandvalet, Y., Bengio, Y.: Semi-supervised learning by entropy minimization. In: NIPS [1]

[35] Grötschel, M., Lovász, L., Schrijver, A.: Geometric Algorithms and Combinatorial Optimization, Algorithms and Combinatorics, vol. 2. Springer, second corrected edition edn. (1993)

[36] Halperin, E., Zwick, U.: Combinatorial approximation algorithms for the maximum directed cut problem. In: SODA '01: Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms. pp. 1–7. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA (2001)

[37] Ho, T.K., Kleinberg, E.M.: Building projectable classifiers of arbitrary complexity. In: ICPR '96: Proceedings of the 13th International Conference on Pattern Recognition. p. 880. Washington, DC, USA (1996)

[38] Hsu, C.W., Chang, C.C., Lin, C.J.: A practical guide to support vector classification. Tech. rep., Department of Computer Science, National Taiwan University (2003)

[39] Huang, J., Smola, A.J., Gretton, A., Borgwardt, K.M., Schölkopf, B.: Correcting sample selection bias by unlabeled data. In: Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems. pp. 601–608 (2006)

[40] Joachims, T.: Transductive learning via spectral graph partitioning. In: International Conference on Machine Learning (ICML). pp. 290–297 (2003)

[41] Joachims, T.: Transductive inference for text classification using support vector machines. In: Bratko, I., Dzeroski, S. (eds.) Proceedings of ICML-99, 16th International Conference on Machine Learning. pp. 200–209. Morgan Kaufmann Publishers, San Francisco, US, Bled, SL (1999)

[42] Joachims, T.: Transductive inference for text classification using support vector machines. In: ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning. pp. 200–209. San Francisco, CA, USA (1999)

[43] Keerthi, S.S., DeCoste, D.: A modified finite newton method for fast solution of large scale linear svms. J. Mach. Learn. Res. 6, 341–361 (Dec 2005)

[44] Klinkenberg, R., Joachims, T.: Detecting concept drift with support vector machines. In: ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning. pp. 487–494. San Francisco, CA, USA (2000)

[45] Kolter, J.Z., Maloof, M.A.: Dynamic weighted majority: An ensemble method for drifting concepts. Journal of Machine Learning Research 8, 2755–2790 (2007)

[46] Krause, A., Guestrin, C.: Near-optimal observation selection using submodular functions. In: AAAI'07: Proceedings of the 22nd national conference on Artificial intelligence. pp. 1650–1654. AAAI Press (2007)

[47] Krause, A., Guestrin, C.: Nonmyopic active learning of gaussian processes: an exploration-exploitation approach. In: ICML '07: Proceedings of the 24th international conference on Machine learning. pp. 449–456. ACM, New York, NY, USA (2007)

[48] Lawrence, N.D., Jordan, M.I.: Semi-supervised learning via gaussian processes. In: NIPS [1]

[49] Liao, W., Ji, Q., Wallace, W.A.: Approximate nonmyopic sensor selection via submodularity and partitioning. Trans. Sys. Man Cyber. Part A 39(4), 782–794 (2009)

[50] Mitchell, T.M.: The discipline of machine learning. Tech. Rep. CMU-ML-06-108, Carnegie Mellon University - ML Department (July 2006)

[51] Mossel, E., Roch, S.: On the submodularity of influence in social networks. In: STOC '07: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing. pp. 128–134. ACM, New York, NY, USA (2007)

[52] Narasimhan, M., Bilmes, J.: PAC-learning bounded tree-width graphical models. In: Uncertainty in Artificial Intelligence: Proceedings of the Twentieth Conference (UAI-2004). Morgan Kaufmann Publishers (July 2004)

[53] Narasimhan, M., Bilmes, J.: A submodular-supermodular procedure with applications to discriminative structure learning. In: Uncertainty in Artificial Intelligence (UAI). Morgan Kaufmann Publishers, Edinburgh, Scotland (July 2005)

[54] Narasimhan, M., Jojic, N., Bilmes, J.: Q-clustering. In: Weiss, Y., Schölkopf, B., Platt, J. (eds.) Advances in Neural Information Processing Systems 18, pp. 979–986. MIT Press, Cambridge, MA (2006)

[55] Nemhauser, G.L., Wolsey, L.A., Fisher, M.L.: An analysis of approximations for maximizing submodular set functionsi. Mathematical Programming 14, 265–294 (1978)

[56] Osuna, E., Freund, R., Girosi, F.: An improved training algorithm for support vector machines. Proceedings of the 1997 IEEE Workshop Neural Networks for Signal Processing VII. pp. 276–285 (Sep 1997)

[57] Platt, J.C.: Fast training of support vector machines using sequential minimal optimization. Advances in kernel methods: support vector learning pp. 185–208 (1999)

[58] Raina, R., Battle, A., Lee, H., Packer, B., Ng, A.Y.: Self-taught learning: transfer learning from unlabeled data. In: ICML '07: Proceedings of the 24th international conference on Machine learning. pp. 759–766. New York, NY, USA (2007)

[59] Ralaivola, L., d'Alché Buc, F.: Incremental support vector machine learning: A local approach. In: ICANN 2001:Proceedings of International Conference on Artificial Neural Networks, Vienna, Austria, August 21-25, 2001. pp. 322–330 (2001)

[60] Rätsch, G., Onoda, T., Müller, K.R.: Soft margins for AdaBoost. Tech. Rep. NC-TR-1998-021, Department of Computer Science, Royal Holloway, University of London, Egham, UK (Aug 1998), submitted to Machine Learning.

[61] Seeger, M.: Learning with labeled and unlabeled data. Tech. rep., Institute for ANC, Edinburgh, UK (2000)

[62] Sindhwani, V., Keerthi, S.S., Chapelle, O.: Deterministic annealing for semi-supervised kernel machines. In: Cohen, W. W., A.M. (ed.) 23rd International Conference on Machine Learning. pp. 841–848. ACM Press, New York, NY, USA (06 2006)

[63] Sindhwani, V., Keerthi, S.S., Chapelle, O.: Deterministic annealing for semi-supervised kernel machines. In: ICML '06: Proceedings of the 23rd international conference on Machine learning. pp. 841–848. New York, NY, USA (2006)

[64] Sviridenko, M.: A note on maximizing a submodular set function subject to a knapsack constraint. Operations Research Letters 32(1), 41 – 43 (2004)

[65] Szummer, M., Jaakkola, T.: Information regularization with partially labeled data. In: Becker, S., Thrun, S., Obermayer, K. (eds.) NIPS. pp. 1025–1032. MIT Press (2002)

[66] Thrun, S.: A lifelong learning perspective for mobile robot control. Intelligent Robots and Systems '94. 'Advanced Robotic Systems and the Real World', IROS '94. Proceedings of the IEEE/RSJ/GI International Conference on 1, 23–30 vol.1 (Sep 1994)

[67] Thrun, S.: Is learning the n-th thing any easier than learning the first? In: NIPS. pp. 640–646. MIT Press (1995)

[68] Thrun, S.: Lifelong learning: A case study. Tech. Rep. CMU-CS-95-208, Computer Science Department, Pittsburgh, PA (1995)

[69] Thrun, S., Mitchell, T.M.: Lifelong robot learning. Robotics and Autonomous Systems 15(1-2), 25–46 (1995)

[70] Thrun, S., O'Sullivan, J.: Learning more from less data: Experiment in lifelong learning. In: Seminar Digest (1996)

[71] Vapnik, V.: Estimation of Dependences Based on Empirical Data: Springer Series in Statistics (Springer Series in Statistics). Springer-Verlag New York, Inc., Secaucus, NJ, USA (1982)

[72] Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer-Verlag New York, Inc., New York, NY, USA (1995)

[73] Vapnik, V.N.: Statistical Learning Theory. Wiley (September 1998)

[74] Wang, F., Zhang, C.: Label propagation through linear neighborhoods. IEEE Transactions on Knowledge and Data Engineering 20(1), 55–67 (2008)

[75] Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 226–235. New York, NY, USA (2003)

[76] Wang, J., Shen, X., Pan, W.: On efficient large margin semisupervised learning: Method and theory. J. Mach. Learn. Res. 10, 719–742 (June 2009)

[77] Wu, M., Schölkopf, B.: Transductive classification via local learning regularization. In: 11th International Conference on Artificial Intelligence and Statistics. pp. 628–635. Brookline, MA, USA (03 2007)

[78] Wu, P., Dietterich, T.G.: Improving svm accuracy by training on auxiliary data sources. In: ICML '04: Proceedings of the twenty-first international conference on Machine learning. p. 110. New York, NY, USA (2004)

[79] Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: Thrun, S., Saul, L., Schölkopf, B. (eds.) Advances in Neural Information Processing Systems 16. MIT Press, Cambridge, MA (2004)

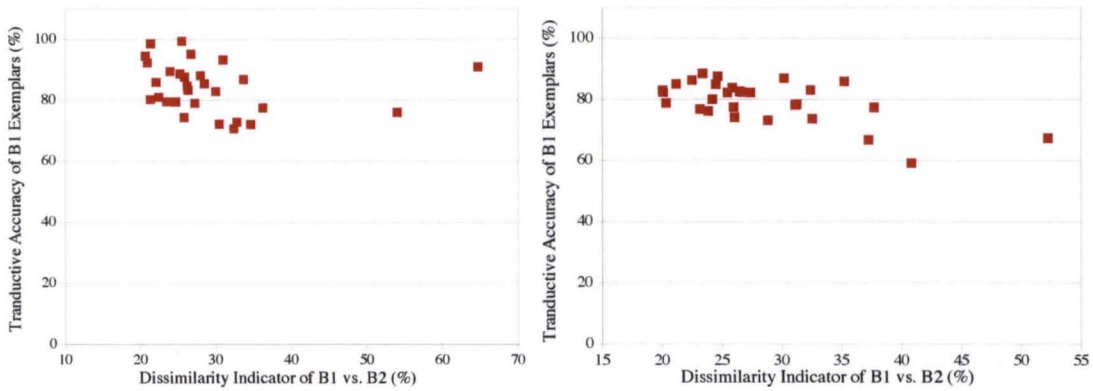
[80] Zhu, X., Ghahramani, Z., Lafferty, J.: Semi–supervised learning using gaussian fields and harmonic functions. In: Proceedings of the International Conference on Machine Learning (2003)

[81] Zhu, X.: Semi-supervised learning literature survey. Tech. Rep. 1530, Computer Sciences, University of Wisconsin-Madison (2005)

[82] Zhu, X., Rogers, T.J., Qian, R., Kalish, C.: Humans perform semi-supervised classification too. In: AAAI. pp. 864–. AAAI Press (2007)

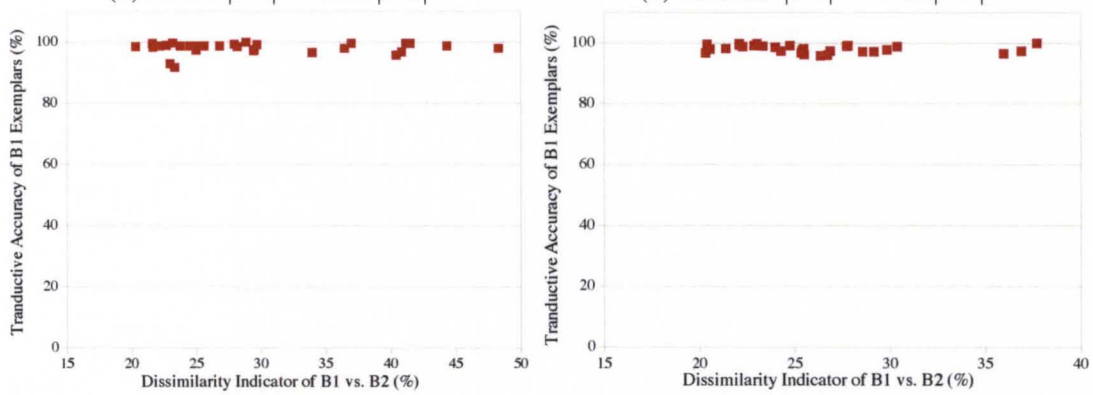
APPENDIX

This following is the rest of the experiments performed in Sec.IV.2.3. verifying the QP-EXMP model.



(a) RBF: $|\mathcal{E}1| = 0.01|\mathcal{B}1|$

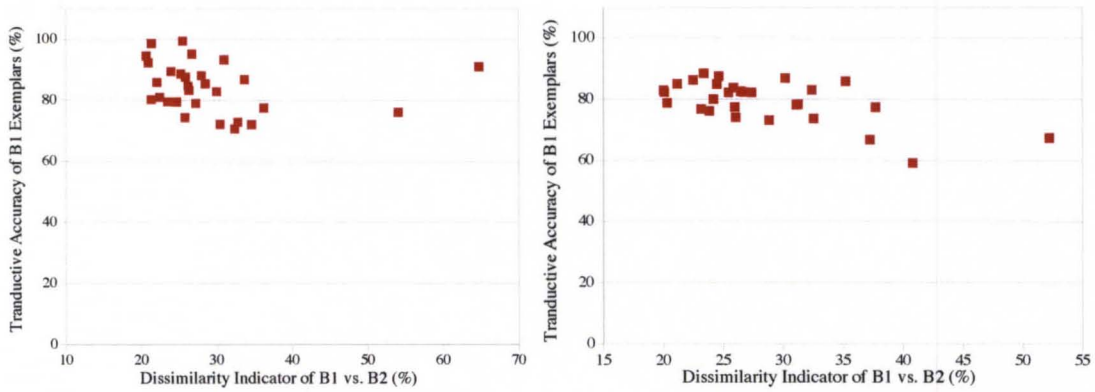
(c) Linear: $|\mathcal{E}1| = 0.02|\mathcal{B}1|$



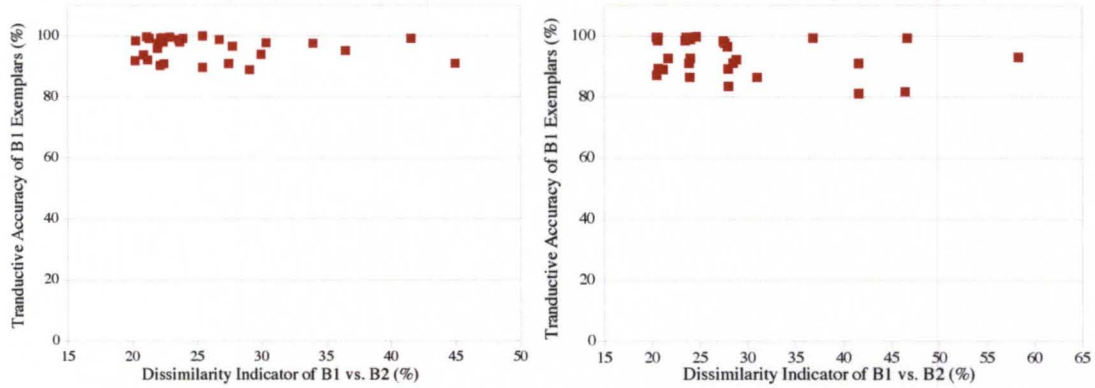
(b) RBF: $|\mathcal{E}1| = 0.1|\mathcal{B}1|$

(d) Linear: $|\mathcal{E}1| = 0.5|\mathcal{B}1|$

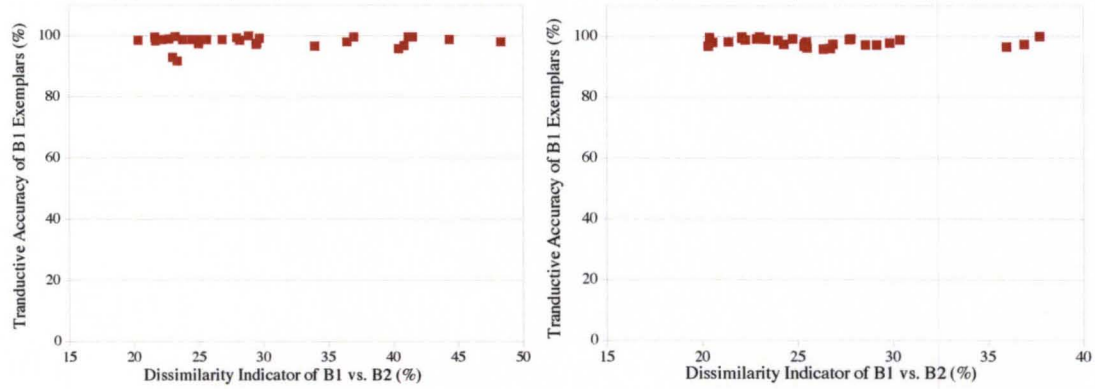
Figure VII.5: QP-EXMP model verification for the **A9A** data set using linear and RBF kernels.



(a) RBF: $|\mathcal{E}1| = 0.01|\mathcal{B}1|$



(b) RBF: $|\mathcal{E}1| = 0.05|\mathcal{B}1|$



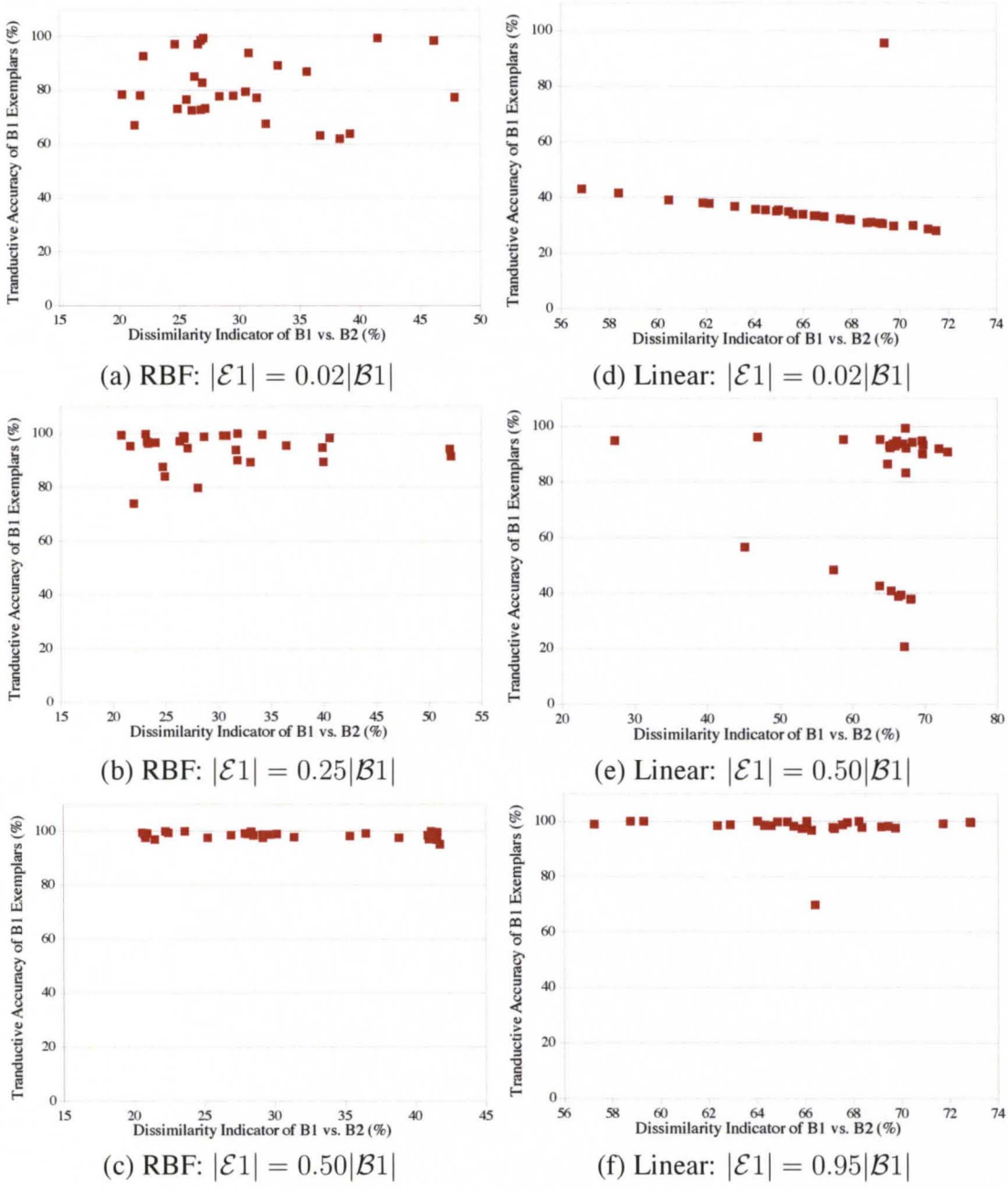
(c) RBF: $|\mathcal{E}1| = 0.1|\mathcal{B}1|$

(d) Linear: $|\mathcal{E}1| = 0.02|\mathcal{B}1|$

(e) Linear: $|\mathcal{E}1| = 0.1|\mathcal{B}1|$

(f) Linear: $|\mathcal{E}1| = 0.5|\mathcal{B}1|$

Figure VII.6: QP-EXMP model verification for the **A9A** data set using linear and RBF kernels.



(a) RBF: $|\mathcal{E}1| = 0.02|\mathcal{B}1|$

(d) Linear: $|\mathcal{E}1| = 0.02|\mathcal{B}1|$

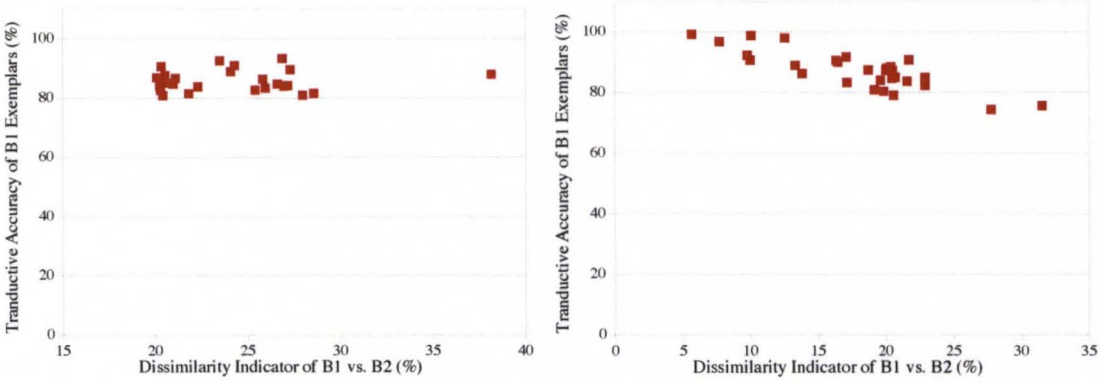
(b) RBF: $|\mathcal{E}1| = 0.25|\mathcal{B}1|$

(e) Linear: $|\mathcal{E}1| = 0.50|\mathcal{B}1|$

(c) RBF: $|\mathcal{E}1| = 0.50|\mathcal{B}1|$

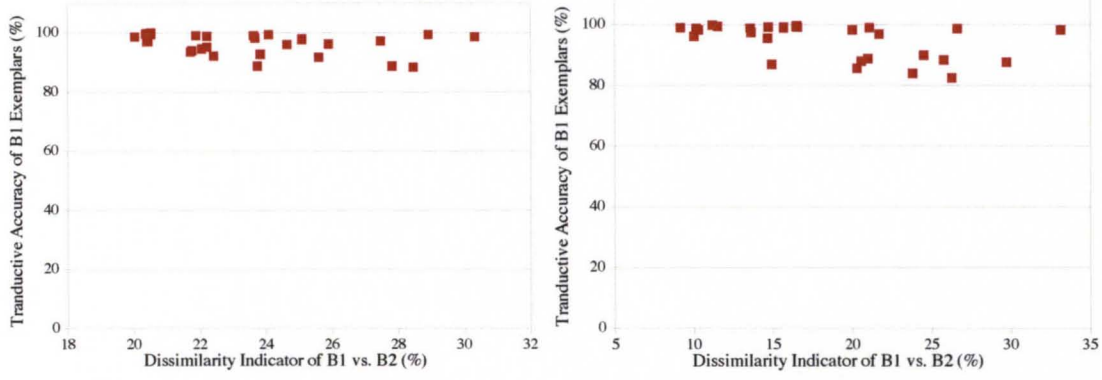
(f) Linear: $|\mathcal{E}1| = 0.95|\mathcal{B}1|$

Figure VII.7: QP-EXMP model verification for the **Cod-Rna** data set using linear and RBF kernels.



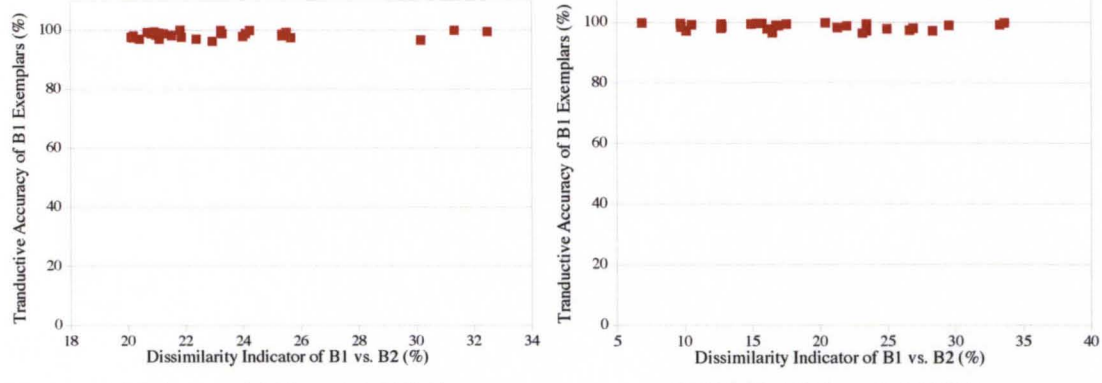
(a) Linear: $|\mathcal{E}1| = 0.005|\mathcal{B}1|$

(d) RBF: $|\mathcal{E}1| = 0.005|\mathcal{B}1|$



(b) Linear: $|\mathcal{E}1| = 0.01|\mathcal{B}1|$

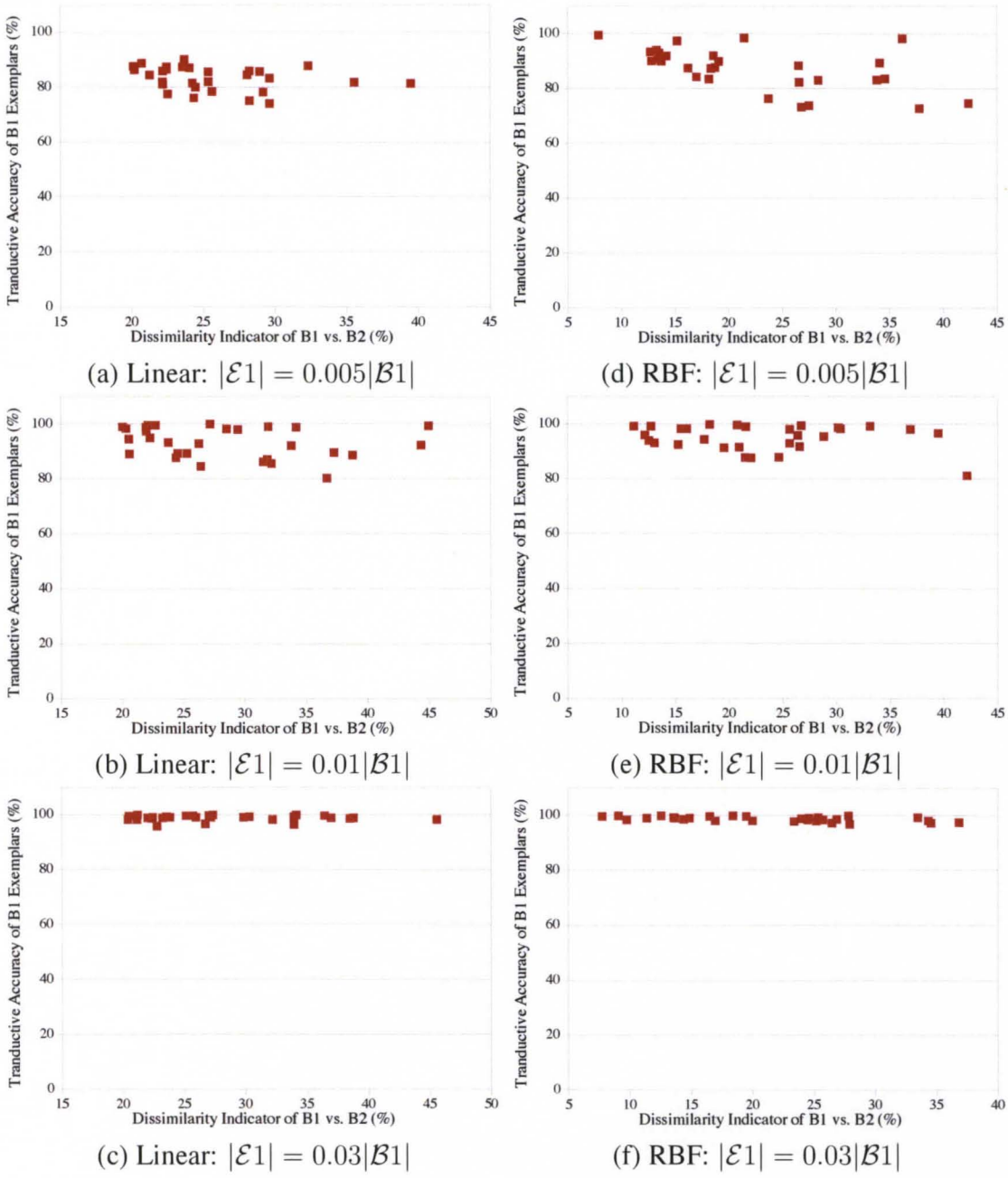
(e) RBF: $|\mathcal{E}1| = 0.01|\mathcal{B}1|$



(c) Linear: $|\mathcal{E}1| = 0.02|\mathcal{B}1|$

(f) RBF: $|\mathcal{E}1| = 0.02|\mathcal{B}1|$

Figure VII.8: QP-EXMP model verification for the **Real-Sim** data set using linear and RBF kernels.



(a) Linear: $|\mathcal{E}1| = 0.005|\mathcal{B}1|$

(d) RBF: $|\mathcal{E}1| = 0.005|\mathcal{B}1|$

(b) Linear: $|\mathcal{E}1| = 0.01|\mathcal{B}1|$

(e) RBF: $|\mathcal{E}1| = 0.01|\mathcal{B}1|$

(c) Linear: $|\mathcal{E}1| = 0.03|\mathcal{B}1|$

(f) RBF: $|\mathcal{E}1| = 0.03|\mathcal{B}1|$

Figure VII.9: QP-EXMP model verification for the **RCV1.Binary** data set using linear and RBF kernels.

CURRICULUM VITAE

Wael Emara

CONTACT INFORMATION

Data Mining Laboratory
Duthie Center for Engineering
University of Louisville
Louisville, KY 40292
waemar01@cardmail.louisville.edu

EDUCATION

University of Louisville, Louisville, Kentucky, USA

Ph.D. of Computer Engineering and Computer Science **2006 - 2012**

- Dissertation: A Submodular Optimization Framework for Never-Ending Learning: Semi-supervised, Online, and Active Learning.
- Advisor: Professor Mehmed Kantardzic.
- Area of Study: Machine Learning and Data Mining.

M.Sc. of Electrical and Computer Engineering **2004 - 2006**

- Area of Study: Computer Vision, Image Processing, and Pattern Recognition.

Mansoura University, Mansoura, Egypt

M.Sc. of Electronics and Electrical Communications Engineering **2001 - 2003**

- Thesis: Medical Images Compression using Wavelets.
- Area of Study: Image Processing.

B.Sc. of Electronics and Electrical Communications Engineering **1995 - 2000**

- Graduation Project: A Computer Based System for the Analysis and Classification of Esophageal Motility Records.

WORK EXPERIENCE

University of Louisville, Louisville, Kentucky USA

Instructor / Research Assistant, CECS Department **2008 - 2011**

Teaching Assistant, CECS Department **2006 - 2008**

Research Assistant, ECE Department **2004 - 2006**

Electronics Research Institute, Cairo, Egypt

Research Assistant, CECS Department **2002 - 2004**

Healthy Information Technology, Cairo, Egypt

R&D Technical Staff **2002 - 2004**

International Control Systems Co., Cairo, Egypt

Electronics Engineer **2000 - 2001**

PUBLICATIONS

Wael Emara and Mehmed Kantardzic. *Mortal Multi-Armed Bandits for Online Learning of Submodular Semi-supervised Support Vector Machine*. In Preparation, 2012.

Wael Emara and Mehmed Kantardzic. *Online Learning of Semi-supervised Support Vector Machines via Submodular Exemplar Model Reduction*. Under Review, 2012.

Wael Emara and Mehmed Kantardzic. *Efficient Approximate Semi-supervised Support Vector Machines Through Submodular Optimization*. In ICMLA 2011: The 10^{th} International Conference on Machine Learning and Applications, Honolulu, Hawaii, USA, 2011.

Wael Emara, Chamila Walgampaya, and Mehmed Kantardzic. *Validation of Click Fraud Detection Models*. In MLDM 2011: The 7^{th} International Conference on Machine Learning and Data Mining in Pattern Recognition, New York, USA, 2011.

Mehmed Kantardzic, Chamila Walgampaya, and Wael Emara, *Click Fraud Prevention in Pay-Per-Click Model: Learning through Multimodal Evidence Fusion*. In ICMWI 2010: International Conference on Machine and Web Intelligence, USTHB University, Algiers, 2010.

Wael Emara and Mehmed Kantardzic. *How Much Locality is Needed forIncremental RBF-SVM Leaning?* In IC-AI 2009: The 2009 International Conference on Artificial Intelligence, Las Vegas, Nevada, USA, 2009.

Wael Emara and Mehmed Kantardzic. *Local Properties of RBF-SVM During Training for Incremental Learning*. In IJCNN 2009: The 2009 International Joint Conference on Neural Networks, Atlanta, Georgia, USA, 2009.

Wael Emara and Mehmed Kantardzic. *The Locality of RBF-SVM for Incremental Learning*. In CIDM 2009: IEEE Symposium on Computational Intelligence in Data Mining, Nashville, Tennessee, USA, 2009.

Wael Emara and Mehmed Kantardzic. *Incremental Learning Using Partly Labeled Data*. In CATA 2008: The 23^{rd} International Conference on Computers and Their Applications, Cancun, Mexico, 2008.

Wael Emara and Mehmed Kantardzic. *An Approach for Incremental Semi-supervised SVM*. In ICDMW 07: The 7^{th} IEEE International Conference on Data Mining Workshops, Washington, DC, USA, 2007.

Research Interests

- **Machine Learning:** Online, Active, Semi-supervised Learning Techniques for Streaming Data.
- **Mathematical Optimization:** Submodular Optimization and Dual Coordinate Descent Methods.
- **Computer Vision:** Object Detection, Recognition, and Tracking.

Awards and Fellowships

- NSF Travel Award to Participate in the Graduate Summer School: Deep Learning, Feature Learning, 2012.
- PhD Dissertation Completion Award, University of Louisville, 2011.
- IEEE Outstanding CECS Student Award, University of Louisville, 2009.