Topics in Resource-efficient Kernel Learning

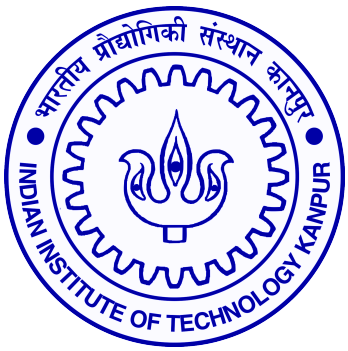
Author:

Purushottam Kar

Supervisors:

Dr. Harish C. Karnick

Dr. Manindra Agrawal



A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

to the

Department of Computer Science and Engineering

Indian Institute of Technology Kanpur, INDIA

November 2013

Statement of Thesis Preparation

I, **Purushottam Kar**, declare that this thesis titled, **"Topics in Resource-efficient Kernel Learning"**, hereby submitted in partial fulfillment of the requirements for the degree of **Doctor of Philosophy** and the work contained herein are my own. I further confirm that:

1. The "Thesis Guide" was referred to for preparing the thesis.
2. Specifications regarding thesis format have been closely followed.
3. The contents of the thesis have been organized based on the guidelines.
4. The thesis has been prepared without resorting to plagiarism.
5. All sources used have been cited appropriately.
6. The thesis has not been submitted elsewhere for a degree.

Name: Purushottam Kar

Roll No.: Y8111062

Department: Computer Science and Engineering

Certificate

It is certified that the work contained in the thesis titled **"Topics in Resource-efficient Kernel Learning"**, by **Purushottam Kar**', has been carried out under our supervision and that this work has not been submitted elsewhere for a degree.

Dr. Harish C. Karnick
Professor
Dept. of Comp. Sci. and Engg.
Indian Institute of Technology Kanpur

Dr. Manindra Agrawal
Professor
Dept. of Comp. Sci. and Engg.
Indian Institute of Technology Kanpur

November 2013

Synopsis of Thesis

Name: Purushottam Kar Roll No.: Y8111062
Degree for which thesis is submitted: Doctor of Philosophy
Department: Computer Science and Engineering
Thesis Title: Topics in Resource-efficient Kernel Learning
Name of Supervisors:
1. Dr. Harish C. Karnick, Professor, Dept. of CSE, IIT Kanpur
2. Dr. Manindra Agrawal, Professor, Dept. of CSE, IIT Kanpur
Month and year of submission: November 2013

Keywords: machine learning, learning with kernels, indefinite kernels, similarity functions, sparse learning, random features, fast learning, online learning, generalization bounds

Kernel learning algorithms have received much attention in the past two decades and occupy a prominent position within machine learning having given state-of-the-art performance in several domains. Much of the power of kernel methods comes from their ability to implicitly represent complex functions in high dimensional spaces that enables algorithms to learn and predict without having to worry about the dreaded "curse of dimensionality".

This, however, comes at a price of increased hypothesis complexity that causes these algorithms to be slow during training as well as at prediction time. With an increase in demand for real time applications, this prevents kernel algorithms from being applied to several domains. A second drawback of kernel-based learning methods is the problem of choosing an appropriate kernel for a given learning task. In several learning situations, much can be gained by appropriately choosing a suitable kernel before applying kernel learning methods. This problem is somewhat more severe for traditional kernel learning methods due to their dependence on so-called "Mercer kernels" that prevents them from fully utilizing rich domain-specific knowledge in the learning process.

Our work makes three contributions in these directions that we briefly describe below. Subsequently, we give a more detailed account of each contribution.

1. We propose a method (Kar and Karnick, 2012) that allows kernel learning algorithms using dot product Mercer kernels to offer fast prediction routines by way of learning approximate predictors. Our method has a desirable side-effect of offering fast training times as well.

2. We develop a learning framework that allows efficient use of non-Mercer kernels in the learning process. We show that our learning framework admits fast training and prediction routines as well as crisp learning theoretic generalization guarantees (Kar and Jain, 2011) and (Kar and Jain, 2012).

3. We develop an online learning framework that allows us to learn multivariate functions (Kar et al., 2013). Among other things, this allows us to perform kernel selection by way of Multiple Kernel Learning as well as learn (indefinite) similarity functions.

Accelerated Learning with Dot Product Kernels

As we have noted, kernel methods owe their power to their ability to implicitly represent points in arbitrarily high dimensional function spaces. This however, comes at a cost of increased complexity of the resulting predictor. A typical kernel predictor (for learning tasks such as classification, regression, principal component analysis etc) has the following form

$$h(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i K(\mathbf{x}, \mathbf{x}_i)$$

where $\mathbf{x}_i$ are the training points. Typically $\alpha_i \neq 0$ for a large fraction of the training set, a fact that has been confirmed theoretically as well. This can cause the predictor to be slow at test time due to the large number of kernel computations involved. However, we notice that this predictor has a simple alternate form. Indeed, applying the kernel trick gives us $K(\mathbf{x}, \mathbf{y}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle$, and we get

$$h(\mathbf{x}) = \langle \mathbf{w}, \Phi(\mathbf{x}) \rangle$$

for some $\mathbf{w} \in \mathcal{H}_K$ where $\mathcal{H}_K$ is the Reproducing Kernel Hilbert Space corresponding to the kernel K and $\Phi : \mathcal{X} \to \mathcal{H}_K$ is the corresponding feature map. The main idea behind our work is the following: the predictor is expressible as a unit operation albeit in a high (possibly infinite) dimensional RKHS. If one could reduce the dimensionality of the RKHS while approximately preserving inner products by devising some map, then one would be able to express the predictor as a single inner product in a small(er) dimensional space. More specifically, our goal is to devise a map $\Psi : \mathcal{H}_K \to \mathbb{R}^D$ for some finite D such that for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, we have $K(\mathbf{x}, \mathbf{y}) \approx \langle \Psi(\Phi(\mathbf{x})), \Psi(\Phi(\mathbf{y})) \rangle$. We shall call such a map an *approximate feature map* for the kernel K.

The proposed scheme is concisely represented in Figure 1. The motivation for the existence of such inner product preserving maps comes from results such as the Johnson-Lindenstrauss Lemma. However, algorithmically, one cannot apply the maps Φ and Ψ in succession as the intermediate step involves working in a high dimensional space $\mathcal{H}_K$. In our work Kar and Karnick (2012), we provide an alternate map $Z : \mathcal{X} \to \mathbb{R}^D$ for some small D such that it implicitly encodes the composition of the maps Φ and Ψ.

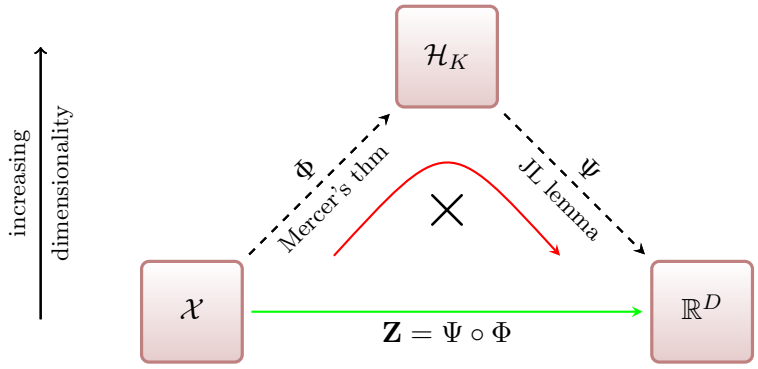


Figure 1: Approximate Feature maps for Kernels

Our maps work for *Dot Product Kernels* which are kernels of the form $K(\mathbf{x}, \mathbf{y}) = f(\langle \mathbf{x}, \mathbf{y} \rangle)$ for some analytic function $f : \mathbb{R} \to \mathbb{R}$. This takes into account a large family of kernels including the Polynomial kernels $K(\mathbf{x}, \mathbf{y}) = (\langle \mathbf{x}, \mathbf{y} \rangle + c)^p$ for some $c \geq 0, p \in \mathbb{Z}^+$, Exponential kernel $K(\mathbf{x}, \mathbf{y}) = \exp\left(\frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\sigma^2}\right)$ for some $\sigma > 0$ (which is simply the non-normalized Gaussian kernel) and Vovk's kernel $K(\mathbf{x}, \mathbf{y}) = \frac{1-\langle \mathbf{x}, \mathbf{y} \rangle^p}{1-\langle \mathbf{x}, \mathbf{y} \rangle}$ for some $p \in \mathbb{Z}^+ \cup \{0\}$.

Our method relies on (an extension of) a result in functional analysis due to Schoenberg Schoenberg (1942). Our basic result ensures that if $\mathcal{X}$ is a compact subset of $\mathbb{R}^d$ and $K(\mathbf{x}, \mathbf{y}) = f(\langle \mathbf{x}, \mathbf{y} \rangle)$, then if $D = \Omega\left(\frac{d}{\varepsilon^2} \log\left(\frac{1}{\varepsilon\delta}\right)\right)$, then the feature map $\mathbf{Z} : \mathcal{X} \to \mathbb{R}^D$ is an ε-approximate feature map for K with probability $1 - \delta$ i.e. for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, $|K(\mathbf{x}, \mathbf{y}) - \langle \mathbf{Z}(\mathbf{x}), \mathbf{Z}(\mathbf{y}) \rangle| < \varepsilon$.

Our experiments show that our technique offers impressive speedups for the SVM classification problem with respect to both training as well as test times.

Supervised Learning with Similarity Functions

Kernel learning has traditionally advocated the use of Mercer (or positive definite) kernels. Although these kernels allow us algorithmic as well as analytic advantages leading to fast training routines and clean generalization guarantees, their strict mathematical form excludes several notions of similarity or distance that are prevalent in various domains. Examples include the Earth-mover's distance in image retrieval, BLAST scores in bio-informatics and a variety of graph-based metrics that can be defined on graphical structures such as social networks.

However, the indefiniteness of these kernels prevents them from being directly used in kernel methods such as the SVM. For the same reason, they lack the nice geometric interpretation that Mercer kernels possess that makes proving generalization guarantees easier. Our work addresses both these problems in a unified manner.

At the heart of our work is the notion of a *good* kernel. More specifically, we devise notions that formally capture the suitability of a given (possibly indefinite) kernel to a given learning task. It is worthwhile noting that such notions of *goodness* also exist for

Mercer kernels where typically, a Mercer kernel is considered good for a given learning task such as classification or regression if there exists a good predictor in the RKHS of the kernel with a large functional margin. We extend this notion to indefinite kernels[1] wherein for any supervised learning task (we address different learning tasks separately below) and any kernel, we are able to state if the kernel is (ε, B)-good for the learning task for some $\varepsilon, B > 0$.

Having established this notion, our work provides two types of guarantees:

1. **Utility Guarantee**: we are able to guarantee that if we are given an (ε, B)-good similarity function K for a learning task, then using $\text{poly}\,(1/\varepsilon_1, \log(1/\delta))$ training samples, we can, with probability at least $1 - \delta$, learn a hypothesis that has generalization error bounded by $\varepsilon + \varepsilon_1$. We show this by giving an algorithm that samples *landmark* points from the domain, constructing a feature map out of them and then solving a linear prediction problem in this new feature space. Since solving a linear prediction problem is computationally simple, this gives our framework fast training routines.

2. **Admissibility Guarantee**: we are able to show that if a Mercer kernel K is *good* for a particular learning task by virtue of having a large margin predictor in its RKHS, then there exist $\varepsilon, B > 0$ such that K is (ε, B)-good as a similarity function as well. This effectively demonstrates that our notions of goodness are not artificial or too restrictive but in fact, extend known notions of Mercer kernel goodness. We prove these results using (extensions of) a technique given by Srebro (2007).

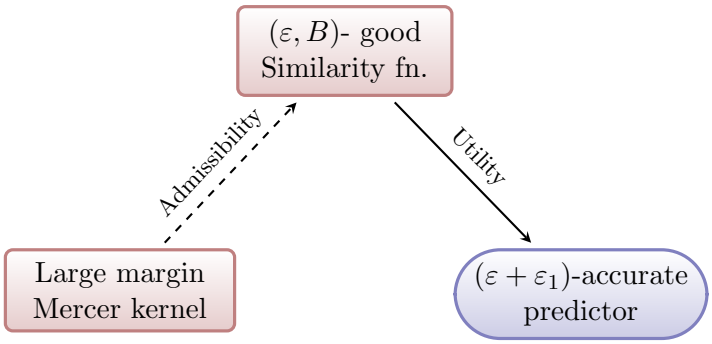


Figure 2: Utility and Admissibility guarantees

Figure 2 concisely presents these guarantees. Such notions of goodness for indefinite kernels (or similarity functions) were first proposed by Balcan and Blum (2006) for the problem of binary classification. In the first part of our work (Kar and Jain, 2011), we extend the notion of goodness proposed by Balcan and Blum in order to make it more flexible to diverse learning scenarios. Our method involves sampling *labeled* landmark points from the domain and creating random classifier *stumps* out of pairs of positive and

[1] ↑ We shall use the terms *indefinite kernel* and *similarity function* interchangeably.

negative landmark points. Our proposed method outperforms the method of Balcan and Blum in experiments as well as enjoys good generalization (utility) properties with respect to the misclassification loss function.

In the next part of the work (Kar and Jain, 2012), we extend the above framework to other supervised learning problems. It turns out that the notions of similarity goodness proposed for classification in (Balcan and Blum, 2006; Kar and Jain, 2011) assume the presence of discrete labels which restricts them to classification-like scenarios. We propose a new notion of goodness that can be used in arbitrary supervised learning situations. We also show that our notion of goodness reduces to that of Balcan and Blum when labels are indeed binary. We also propose learning algorithms that can take a *good* similarity function and a few *unlabeled* landmark points and learn a good predictor.

Using this new framework, we are able to prove utility and admissibility guarantees for several supervised learning problems such as regression, ranking and ordinal regression. In each case, our utility guarantee holds with respect to a widely used loss function. For instance, for regression, our utility guarantee holds with respect to the mean squared error and for ranking, it holds with respect to the NDCG loss function. We are also able to, in each case, show that good Mercer kernels for these problems are also good with respect to our notion of goodness, thus providing an admissibility guarantee.

For the case of real valued regression, we are additionally able to devise a framework that can learn sparse predictors that are faster at test time. Our learning method is a pursuit type algorithm that uses a variant of Forward Greedy Selection (Shalev-Shwartz et al., 2010b) to learn a sparse predictor. We show that this framework still enjoys utility and admissibility properties.

In our experiments, which intend to demonstrate the effectiveness of the proposed framework, our methods outperform a baseline method of Kernel regression on both regression and ordinal regression tasks with the sparse learning formulation giving especially attractive results for the case of real-valued regression.

Online Learning with Pairwise Loss Functions

For learning problems such as regression and classification, the loss functions used in these settings, for example hinge loss, squared loss, ε-insensitive loss, take as input a labeled point and a hypothesis and output how well the hypothesis performs on that point. However, in several learning scenarios such as Mahalanobis metric learning, multiple kernel learning and bipartite ranking, the hypothesis must be evaluated on two input points. For instance, in metric learning, a hypothesis is evaluated on the basis of how well is it able to keep oppositely labeled instances apart while keeping together similarly labeled points Jin et al. (2009). In such situations, the loss function has the following form

$$\ell : \mathcal{H} \times \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$$

where $\mathcal{H}$ is the hypothesis space and $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ is the (labeled) domain. Learning with pairwise loss functions entails several algorithmic and learning theoretic challenges. From an algorithmic point of view, seldom does the learning algorithm receive i.i.d pairs from the domain $\mathcal{Z} \times \mathcal{Z}$. Instead, what one receives is a set of n i.i.d. points from the domain $\mathcal{Z}$. Consequently, the learning algorithm has to create pairs out of these points. Since processing all $\Omega\left(n^2\right)$ pairs is usually prohibitive, this presents a challenge.

From a learning theoretic point of view, such pair creation creates difficulties for generalization error bounds since the pairs constructed out of the training points are no longer i.i.d. owing to the intersection between pairs. This problem is well known in learning theory literature as the *coupling problem*.

Online learning methods are very well suited to domains where training data is too voluminous to be stored in memory. This motivates the idea of extending existing online learning techniques to devise algorithms that can learn from pairwise loss functions. To do so, one would have to overcome the problems of pair creation and coupling in the online setting. Our main contribution in this work (Kar et al., 2013) is an online learning framework for learning with pairwise loss functions. We propose memory efficient algorithms for which we prove crisp learning theoretic guarantees such as regret and online-to-batch conversion guarantees.

Our learning framework consists of a learner that is given a finite memory buffer. The learner observes data points as they arrive in a continuous data stream. However, he can store at most s points in the buffer at any given time. At each time instant t, the learner receives the incoming point $\mathbf{z}_t$, pairs it up with points present in the buffer, and uses these pairs to update its hypothesis. We give simple algorithms for updating the hypothesis as well as the buffer at each time step – the buffer update algorithm is a randomized algorithm. We give the following bounds for our algorithms

1. At any fixed time step $t > s$, the contents of the buffer represent s i.i.d. samples from the set $\{\mathbf{z}_1, \ldots, \mathbf{z}_{t-1}\}$ where s is the buffer size.

2. With probability at least $1 - \delta$ over the randomness used to update the buffer, the learning algorithm incurs regret no more than $\mathcal{O}\left(\sqrt{\frac{\log n}{s}}\right)$ with respect to algorithms that have unbounded sized buffers. Here n is the total number of steps in the online learning process and s is the buffer size. Consequently, to achieve a non-trivial regret bound, we only need $s > \omega\left(\log n\right)$ which demonstrates the space-effectiveness of our framework.

We also prove online-to-batch conversion bounds that can give generalization error bounds for online learning algorithms that learn from pairwise loss functions in terms of their regret bounds. Our bounds are tighter and are applicable to more diverse learning scenarios than those of a recent result in the same area (Wang et al., 2012). We give below some salient points of our bounds:

1. **Dimension independence**: for a large class of learning algorithms, our bounds have no dependence on the input dimensionality. This allows us to apply our bounds to kernelized situations (which operate in infinite dimensional spaces) as well. This is in contrast with the bounds of Wang et al. that have a linear dependence on input dimensionality.

2. **Sparse learning**: we are also able to give bounds for algorithms that try to learn sparse functions such as sparse vectors or trace norm bounded matrices using sparsity promoting regularizers such as the Manhattan norm or the Trace norm.

3. **Bounded memory algorithms**: our results apply to a large class of algorithms that maintain finite buffers to store (a subset of) previously seen points in order to learn within a memory budget. This is in contrast with the bounds of Wang et al. which only address algorithms that have unbounded buffers.

4. **Fast rates**: we are able to guarantee fast $\tilde{\mathcal{O}}\left(\frac{1}{n}\right)$ rates of convergence for algorithms that use strongly convex loss functions.

Our learning techniques are applicable to problems such as two-stage multiple kernel learning, metric learning, maximizing the area under the ROC curve (equivalent to bipartite ranking) and similarity (indefinite kernel) learning.

Contents

List of Figures

List of Tables

List of Algorithms

List of Publications

Large portions of various chapters of this thesis have appeared as the following publications:

1. Purushottam Kar, Bharath Sriperumbudur, Prateek Jain and Harish Karnick, On the Generalization Ability of Online Learning Algorithms for Pairwise Loss Functions. In *30th International Conference on Machine Learning* (ICML), 2013.
   [Chapter 6]

2. Purushottam Kar and Prateek Jain. Supervised Learning with Similarity Functions. In *26th Annual Conference on Neural Information Processing Systems* (NIPS), 2012.
   [Chapter 5]

3. Purushottam Kar and Harish Karnick. Random Feature Maps for Dot Product Kernels. In *15th International Conference on Artificial Intelligence and Statistics* (AISTATS), 2012.
   [Chapter 3]

4. Purushottam Kar and Prateek Jain. Similarity-based Learning via Data Driven Embeddings. In *25th Annual Conference on Neural Information Processing Systems* (NIPS), 2011.
   [Chapter 4]

A list of other publications that were not included in this thesis is given below:

1. Aman Dhesi and Purushottam Kar. Random Projection Trees Revisited. In *24th Annual Conference on Neural Information Processing Systems* (NIPS), 2010.

2. Sumit Ganguly and Purushottam Kar. Estimating the first frequency moment of data streams in nearly optimal space and time. In *12th Italian Conference on Theoretical Computer Science* (ICTCS), 2010.

3. Arnab Bhattacharya, Purushottam Kar and Manjish Pal. On Low Distortion Embeddings of Statistical Distance Measures into Low Dimensional Spaces. In *20th International Conference on Database and Expert Systems Applications* (DEXA), 2009.

Chapter 1

Introduction to Learning

Contents

Abstract *This chapter gives a gentle introduction to machine learning. Various learning frameworks are considered and examples are given for each of them to both motivate the learning methodology as well as establish the notation. The empirical and regularized risk minimization principles are introduced that are embodied in the kernel learning framework in later chapters. The chapter also gives an introduction to basic statistical learning theory that is used in the thesis to provide generalization bounds for various learning algorithms.*

1.1 Introduction

Before presenting a mathematically precise formulation of learning, we motivate the need for machine learning in the present world. This is being included to make the discussion self contained and complete.[1]

Our society has, in the past few decades, seen a shift from economies relying on traditional manufacturing industries to economies based on harnessing information. This

[1] ↑ The ↑ symbol at the beginning of footnotes links back to the occurrence of the footnote in the text.

neo-industrial revolution has ushered in the *Information Age* (Wikipedia) where manipulation of knowledge is crucial to both societal as well as industrial growth. In this new setup however, the gap between *information* and *knowledge* has widened.

Today we are nothing short of flooded with information: mobile devices, online social media and electronic transactions generate terabytes of data every single day, all of which contain valuable insight about consumer trends, market forces and profitable avenues. Online medical records hold the key to identifying outbreaks of epidemics and other socio-medical statistics that can play a critical role in policy making and implementation. Scientific data from astronomical observatories and particle accelerators (for example the Large Hadron Collider) has the potential to pave the way for new scientific discoveries.

This, ironically, is exactly where the gap emerges: it is a challenge to be able to extract useful, actionable knowledge out of this sea of information. More often than not this information is too voluminous for manual processing. In several domains, most notably in the case of scientific data and online user data, the rate at which this information is made available to us is too high to allow all the information to even be stored.

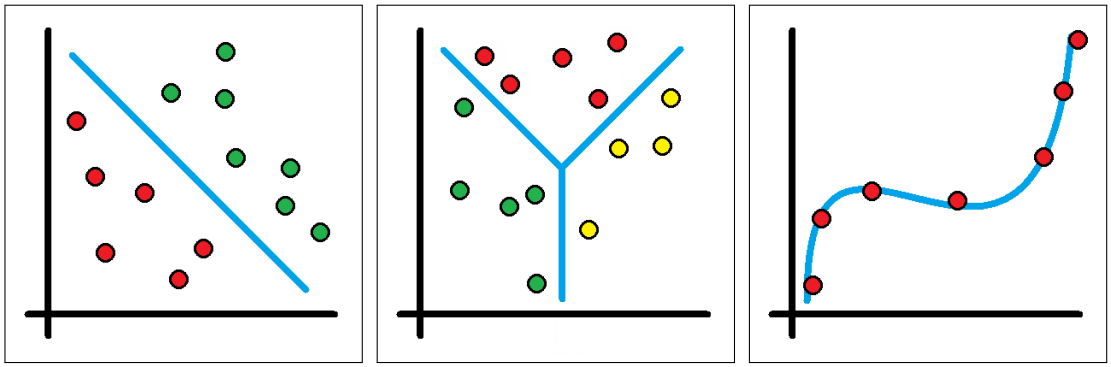
Just to put things in perspective, a study reported in the journal *Science* put the total storage capacity of our planet in 2007 at a figure just shy of 300 exabytes (Hilbert and López, 2011). If one goes by past trends, the figure is expected to enter the zettabyte range in the next 4-5 years. Our combined computation abilities in the same year were estimated at about 6 EIPS (exa or billion billion instructions per second) with Moore's law still holding on, more or less. Compared to this, we generated about 0.8 exabytes of data on the Internet *every day* on an average during 2012 (Mashable). This included posts on blogs, online social fora like Facebook and Twitter, content uploads such as photographs and video and electronic mail. A quick calculation tells us that we would have used up all 300 exabytes available to us within 2012 itself.

1.2 Common Learning Tasks

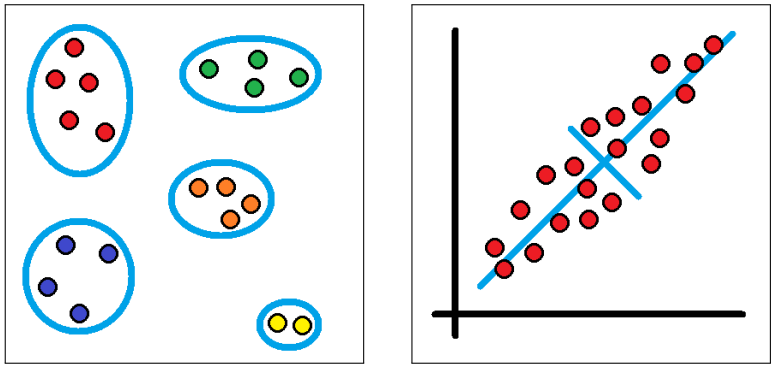
The previous discussion clearly indicates a need for automated routines to process all the data available to us. More specifically, there is a need to efficiently utilize the computational power available to us in order to solve the problem of knowledge extraction.

The area of machine learning concerns itself with precisely this question. Given some raw data, machine learning algorithms try to discover various types of interesting structures or patterns within the data that help us to 1) understand various properties of the data as well as 2) predict future behavior. Thus machine learning techniques can provide us with both deductive as well as inductive abilities that can be utilized for various tasks such as data analytics, mathematical modeling, data mining and prediction. As examples we consider a few popular machine learning tasks below (refer to Figure 1.1 for illustrations):

1. **Binary Classification**: as the name suggests, this task concerns itself with identifying a discrete pattern within the data. For instance take the example of spam



(a) Binary Classification (b) Multi-class Classification (c) Regression



(d) Clustering (e) Principal Component Analysis

Figure 1.1: Illustrations for some common learning tasks.

classification. An average email user gets a fair amount of undesirable emails which may range anywhere from innocuous promotional offers to sinister phishing attempts. It is very desirable to have spam filters that can automatically dump such mails into a specially designated *spam folder*. However, the notion of spam varies from person to person - some people are not offended by promotional offers whereas some are - which is why spam filters have to adapt themselves to the (several) idiosyncrasies of each email user to deliver efficient spam filtering. This poses an interesting binary classification problem where the task is to correctly assign each object (mail) to its class (spam/non-spam).

2. **Multi-class Classification**: whereas the pattern underlying the spam classification example is binary valued (every email is designated either *spam* or *non-spam*), in general one can talk about a multi-valued pattern. For instance, consider the problem of handwriting recognition where we are expected to classify digitized images of handwritten characters into one of 36 classes: 26 alphabets {a,. . . ,z} and 10 numerals {0,. . . ,9}.

3. **Regression**: this is historically one of the most well studied learning tasks and has immense applications in economics and the sciences. The task here is quite similar to

that of curve fitting: given a set of points in some space and a real value associated with each of them, our goal is to figure out some real valued function over the space that can accurately match the values associated with each of the points. This real value is usually some quantity that we are interested in predicting for points in the space. For instance, if we are interested in predicting family expenditure of a certain population, we can do so by encoding each family as a three tuple by considering its monthly income, number of members in the family and their median age and then posing the problem of predicting the average monthly expenditure of the family as a real valued regression problem.

4. **Clustering**: in this task, the aim is to discover some (possibly hierarchical) structure in the data. This can help in performing a taxonomic analysis of a given dataset. For instance, given genome sequences of a set of organisms, an application of clustering can help us club organisms with similar genomic structure together thus aiding in identification of species, genera and families.

5. **Component Analysis**: this task has several applications in signal processing domains such as image and speech processing. Given a set of signals as a time/space sampled series, it may be necessary to extract useful components out of the data. This can help in noise reduction as well as dataset size reduction both of which are desirable from the point of view of increasing accuracy and processing speed. For instance, *Independent Component Analysis* can help identify distinct speech or voice patterns in a recording. *Principal Component Analysis* is very widely used as a data dimensionality reduction technique and in face recognition tasks.

There are several other learning tasks such as multi-label classification, ordinal regression and ranking which we have not discussed above but which are nevertheless well studied as well as useful in modeling practical scenarios.

1.3 The Learning Methodology

The various learning tasks studied in machine learning have been divided into broad learning frameworks for ease of study. Two most commonly studied learning frameworks are *Supervised Learning* and *Unsupervised Learning*. The first three tasks discussed above fall into the supervised category whereas the last two are unsupervised learning tasks. There are several other learning frameworks, some which are modifications of these two such as *Semi-supervised Learning*, *Online Learning*, *Active Learning* and *Reinforcement Learning* which we shall not discuss in detail here. The reader is referred to one of several standard texts in machine learning such as (Duda et al., 2000) and (Mitchell, 1997) for an introduction to other learning frameworks. We shall however, discuss online learning in Chapter 6. In the sections below, we briefly introduce the supervised and unsupervised learning frameworks.

1.3.1 Supervised Learning

This framework involves the abstraction of a teacher-student interaction and the learning process is seen as a one round game. The basic setting consists of a *domain* $\mathcal{X}$ and a *label set* $\mathcal{Y}$: the domain is simply the space from where all data points are arriving whereas the label set is the set of all possible labels that can be assigned to domain elements.

For instance, in the spam filtering case, the domain is simply the set of all possible emails (i.e. texts in the English language). The label set has just two elements {*non-spam,spam*} which can be succinctly represented as the set $\{0, 1\}$: 0 indicating *non-spam* and 1 indicating *spam*. For multi-class classification, the label set contains more than just two elements but is still a discrete and finite set. For regression, the label set is usually some contiguous interval of the real line i.e. $\mathcal{Y} \subset \mathbb{R}$.

The *teacher* holds two quantities with itself which it hides from the *student*. The first quantity is a *Target Function* $f^t : \mathcal{X} \to \mathcal{Y}$ which mathematically captures the notion of "correct" labels. For example, in the spam classification case, this function encodes whether the email user views the emails as spam or not. The second quantity is a *distribution* $\mathcal{D}$ on the domain $\mathcal{X}$ which would be used to sample points from the domain[2]. The *student* has no direct access whatsoever to either of these two objects.

In the first round, the teacher samples n data points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ from the distribution, calculates their true labels $y_i = f^t(\mathbf{x}_i)$ and sends the collection $\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ as *Training Data* to the student. For instance, in the spam classification, the email user might himself act as the teacher by specifically marking certain emails as spam and leaving others as non-spam. The task of the student is to use this training data to come up with a *Hypothesis Function* $h : \mathcal{X} \to \mathcal{Y}$. Usually the student is restricted to choose a hypothesis from a *Hypothesis Class* $\mathcal{H}$. As we shall see later, this is essential since allowing the hypothesis class $\mathcal{H}$ to be unrestricted can mislead the student. The learning game ends when the student has arrived at a hypothesis: the student is said to have *learned* this hypothesis.

The aim of the student is to come up with a hypothesis that "agrees" with the target on "most" domain elements - in some sense, the student's goal is to read the teacher's mind (just as the spam filter's goal is to read the email user's mind). This is formalized by the notion of a *Loss Function* $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ which tells us what price we pay for making a wrong prediction. For instance, in the spam classification case (where $\mathcal{Y} = \{0, 1\}$), a natural loss function is the *zero-one loss function* which is defined as follows:

$$\ell_{\text{zero-one}}(y_1, y_2) = \begin{cases} 0 & \text{if } y_1 = y_2 \\ 1 & \text{if } y_1 \neq y_2 \end{cases}$$

Thus, in case the student correctly classifies an email, he does not encounter any loss. On

[2] ↑ The more informed reader might complain that this model does not address the possibility that the label might be a non-deterministic function of the data point. A more general model would, instead of having a target function and a distribution, simply have a distribution on the product domain $\mathcal{X} \times \mathcal{Y}$. For sake of simplicity we continue to assume noiseless models for now.

the other hand, classifying a spam email as non-spam or the other way round incurs the student a unit loss. In case of problems such as regression, it might be too restrictive to expect the student to output exactly the true value (such as the exact monthly expenditure of a family) - it is much more reasonable to expect the student to output a value close to the true value. The widely used *quadratic loss* or *least squares loss* function encodes this intuition. Not surprisingly, this is also the loss function behind the popular *least squares* learning algorithm for real valued regression.

$$\ell_{\text{quad}}(y_1, y_2) = (y_1 - y_2)^2$$

Note that in certain situations, the student is allowed to output hypotheses with a range set $\mathcal{Y}'$ different from $\mathcal{Y}$. In such cases, the loss function is appropriately modified to look like $\ell : \mathcal{Y}' \times \mathcal{Y} \to \mathbb{R}_+$. Whatever be the case, the student is penalized according to the expected loss (or population risk) incurred by his hypothesis. This is defined using the *Risk Functional*

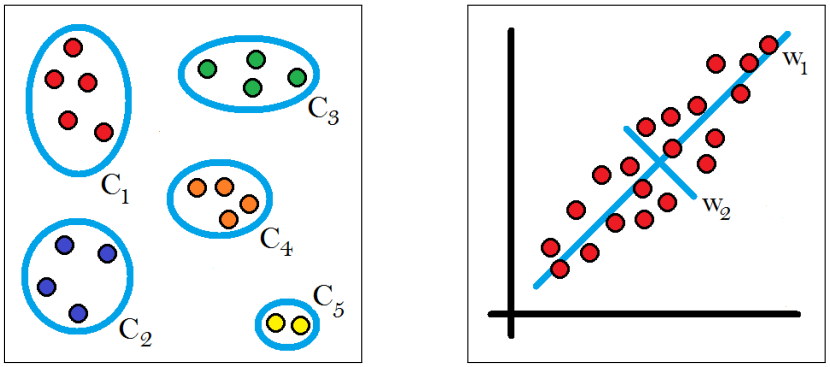
$$\mathcal{R} : h \longmapsto \int_{\mathcal{X}} \ell(h(\mathbf{x}), f^t(\mathbf{x}))\ d\mathcal{D}(\mathbf{x})$$

The risk functional encodes the extent to which the student's hypothesis "agrees" with the target on average - thus the goal of the student is to minimize $\mathcal{R}(h)$ as much as possible. The risk functional satisfies certain interesting properties such as *non-negativity* i.e. for all $h \in \mathcal{H}$, we have $\mathcal{R}(h) \geq 0$ and some sort of an *identity property* i.e. $\mathcal{R}(f^t) = 0$.

Usually, in order to make the learning model realistic, the teacher is given no restrictions on how to choose the target function f^t and neither is the student allowed to make any such assumptions. This is known as the *Agnostic Setting* of learning. There are more restricted settings such as the *Proper Learning* setting where the teacher is constrained to choose the target function from some hypothesis class i.e. $f^t \in \mathcal{H}^t$ and the student is made aware of the class $\mathcal{H}^t$ in advance. In the agnostic case, since the student is not allowed to make any such assumptions, his goal becomes to do as well as his hypothesis class allows him to do i.e. achieve a risk as close to $\mathcal{R}(\mathcal{H}) := \inf_{h \in \mathcal{H}} \mathcal{R}(h)$. Note that in case of proper learning where $\mathcal{H} = \mathcal{H}^t$, we have $\mathcal{R}(\mathcal{H}) = 0$ which is a consequence of the identity property[3].

The techniques that the student uses to learn the hypothesis from the training data is the subject of much research within machine learning. We shall look at some of these in a subsequent section in this chapter. The problem of giving guarantees on the *excess risk* or $\mathcal{R}(h) - \mathcal{R}(\mathcal{H})$ for the hypothesis learned by the student shall also be discussed in this chapter within the broad framework of *Statistical Learning Theory*.

[3] ↑ This is no longer true in the noisy model where one is bound to incur the *Bayes Risk* (see for example Steinwart and Christmann, 2008b, Definition 2.3) no matter what.



(a) Clustering (b) Principal Component Analysis

Figure 1.2: Illustrations for some unsupervised learning tasks.

1.3.2 Unsupervised Learning

In this learning framework, the task is to extract useful structures within the data. This framework is characterized by the absence of "labels" and a well-defined teacher that the supervised learning framework had. The only "supervision" allowed in this learning framework is the specification of the type of structures to be found. A common way of discovering these structures is by establishing a *cost function* or a *potential function* that associates with each possible structure, a real number that tells us how suitable that structure is. One then tries to find the structure that optimizes this cost or potential function or some approximation of it.

To make things concrete, let us take the example of clustering: we are provided with a set of points $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ from some metric space $(\mathcal{X}, d)$ with an associated distance function d. Our job is to partition the set S into k disjoint subsets, each of which is called a *cluster* (refer to Figure 1.2a for an illustration). More specifically, we want a collection of k disjoint subsets $\mathcal{C} = \{C_1, \ldots, C_k\}$ such that the union of the subsets is S itself. The objective behind this is to be able to club together, in the same cluster, points that have a greater affinity to each other than to points outside the cluster. For instance, this can help us cluster organisms with similar genetic material to identify a particular species.

A potential function that encodes this intuition is the well known *k-means* objective. This objective assigns a cost to any clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$ of the set S by checking how spread out are points within each cluster : ideally we would want points in a cluster to be tightly bound together. To do this we first define *cluster centers* as follows

$$\mu_i = \arg\min_{\mathbf{x} \in \mathcal{X}} \sum_{j:\mathbf{x}_j \in C_i} d(\mathbf{x}_j, \mathbf{x})$$

and then calculate the potential of this clustering as follows

$$\text{cost}(\mathcal{C}) = \sum_{i=1}^{k} \sum_{j:\mathbf{x}_j \in C_i} d(\mathbf{x}_j, \mu_i)$$

Although this potential function is known to be NP-hard to optimize (Mahajan et al., 2012), the popular Lloyd's algorithm is able to obtain local minima efficiently.

The other example we discuss here is that of principal component analysis. In this case one is given a set of points in some vector space (say $\mathbb{R}^d$) and the goal is to identify components in the data that correspond to uncorrelated directions. These components are identified such that the first component has the largest possible variance and each subsequent component in turn has the highest variance possible in the subspace orthogonal to the space spanned by the preceding components. This ensures that the components are uncorrelated in a pairwise manner (refer to Figure 1.2b for an illustration).

Thus, if the n data points are arranged in an $d \times n$ matrix $\mathbf{X}$, the goal is to find an orthonormal set of directions $\mathbf{w}_1, \ldots, \mathbf{w}_d$ such that

$$\mathbf{w}_1 = \underset{\|\mathbf{w}\|=1}{\arg\max}\ \mathrm{Var}\left[\!\left[\mathbf{w}^\top \mathbf{X}\right]\!\right]$$

and for each subsequent principal component $\mathbf{w}_k$, the following holds

$$\mathbf{w}_k = \underset{\|\mathbf{w}\|=1}{\arg\max}\ \mathrm{Var}\left[\!\left[\mathbf{w}^\top \mathbf{X}^{(k-1)}\right]\!\right]$$

where $\mathbf{X}^{(0)} = \mathbf{X}$ and $\mathbf{X}^k = \mathbf{X}\left(I - \sum_{i=1}^{k} \mathbf{w}_i \mathbf{w}_i^\top\right)$. The implicit potential function being minimized here is the residual variance in the undiscovered components.

Having components laid out in decreasing order of variance has several benefits since the first few principal components can be said to encode most of the information in the data. In fact, discarding all but the first few principal components is a very popular method of feature space compression and dimensionality reduction. Principal component analysis can also be used as a preprocessing step to get guaranteed convergence for the Llyod's algorithm for k-means clustering (Kumar and Kannan, 2010) which otherwise does not enjoy any convergence guarantees and can get stuck in local optima.

1.3.3 Other Learning Frameworks

We briefly discuss below some of the other learning frameworks mentioned previously:

1. **Semi-supervised Learning**: this framework is an extension of the supervised learning framework in which the teacher, apart from the labeled training set $\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, also provides the student with *unlabeled data* $\mathcal{U} = \left\{\mathbf{x}_j^u\right\}_{j=1}^m$. The idea behind this is to allow the student to have a better glimpse of the distribution $\mathcal{D}$ by means of having more samples. Typically we have $m \gg n$ which models practical scenarios where obtaining clean and correct labeled data is expensive but obtaining unlabeled data is cheap. For instance, in the handwriting recognition problem it is very easy to assemble a collection of hundreds of thousands of digitized handwritten characters but it is very expensive to manually tag them with the letter

they represent.

2. **Online Learning**: in this framework, the number of rounds of interaction between the teacher and the student are increased. The teacher presents the student with only a single training point in each round and expects the student to revert back with a hypothesis. The teacher then reveals the true label of the point and the student incurs a loss if the label predicted by his hypothesis in this round does not match the true label. The teacher is not restricted to sample points according to any distribution in this framework and can possibly behave in an adversarial manner in an effort to maximize the loss incurred by the student. We will discuss the online learning framework in detail later and refer the reader to Chapter 6 for a more detailed introduction to online learning.

3. **Active Learning**: in this framework, there are multiple rounds of interaction between the teacher and the student as well as the interactions are made bi-directional. The student is now allowed to query the teacher about the true labels of arbitrary points. Frequently, this is implemented by means of first having the teacher provide a large number of unlabeled samples to the student and the student then requesting labels for a small subset of them. Alternatively one can have the student generate the samples on his own. The ability to query the teacher greatly eases the learning problem for the student in several domains such as automata learning (Angluin, 1987) and offers reduced sample complexity in others (Gilad-Bachrach et al., 2006). See (Settles, 2012) for a nice survey of existing work on active learning.

4. **Reinforcement Learning**: this framework is inspired by behaviorist psychology and is concerned with optimizing the behavior of an agent in an interactive environment that responds to the actions of the agent with rewards and punishments. The learning framework is characterized by a *state space* that models the environment, a set of *actions* that the agent can take and a set of (stochastic) rules and polices that govern state transitions, actions taken by the agent at each state and the expected rewards to be gained upon performing a particular action in a particular state. The learning process proceeds in stages with the environment responding to the user's actions by changing it's state and offering various rewards to the user. The aim of the user is to formulate policies so as to maximize the cumulative reward.

At this point we conclude our broad discussion on machine learning. We shall restrict ourselves to supervised learning tasks for the rest of this chapter with the aim of building enough background to introduce kernel learning in Chapter 2.

1.4 Common Learning Strategies

One of the questions we left unanswered in the previous section was how to learn a hypothesis using the training set. Continuing with our focus on the supervised learning model,

we now briefly present certain common learning strategies used in this model. We note that these strategies dictate the learning process at a very abstract level and in practice, learning algorithms have to make several efforts to efficiently implement these strategies.

We recall that the supervised framework involves the student getting a labeled training set $\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ from the teacher. The student is then supposed to use the training data to learn a hypothesis h from some hypothesis class $\mathcal{H}$.

1.4.1 Empirical Risk Minimization

In absence of any other information, it seems natural to choose the hypothesis that seems to fit the training data well. For any hypothesis $h \in \mathcal{H}$, define the *Training Error* or *Empirical Risk* as follows:

$$\hat{\mathcal{R}}(h) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(\mathbf{x}_i), y_i),$$

where ℓ is the loss function being used. Since the empirical risk encodes how well a hypothesis is doing on the training set, it seems natural to choose the following function as the hypothesis:

$$\hat{h} = \arg\min_{h \in \mathcal{H}} \hat{\mathcal{R}}(h).$$

This policy of minimizing the empirical risk is known as the *Empirical Risk Minimization* principle (ERM) and is one of the most popular learning techniques known for its simplicity and efficiency. A crucial choice that one has to make while implementing the ERM principle is that of the hypothesis class $\mathcal{H}$ with respect to its *capacity*.

If the hypothesis class is too weak then the learning process will not be able to offer good performance due to underfitting : the hypothesis class is said to suffer from a large *bias* in this case. On the other hand, if the hypothesis class is too diverse (for instance, if it contains all possible functions $h : \mathcal{X} \to \mathcal{Y}$, then the empirical risk minimizer will overfit terribly (see Section 1.5 for a discussion). Several strategies have been developed in order to address this overfitting problem. We discuss below a few of them.

1.4.2 Structural Risk Minimization

Structural risk minimization (SRM) (Vapnik, 2000) is an effective means of enforcing the *Occam's Razor* principle which states that the simplest hypothesis offering a reasonable explanation of the observed data is usually the "correct" one. To do this, SRM works with a *nested hierarchy* of hypothesis classes of different complexities.

More formally, one considers the classes $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \ldots$ along with a *penalty* function $\mathcal{C}$ associated with each hypothesis class that measures its capacity. For instance, in classification problems, one can use the *VC dimension* of the hypothesis classes as the penalty function. Thus, the SRM principle decides upon the hypothesis as follows:

$$\hat{h} = \arg\min_{h_t \in \mathcal{H}_t, t \geq 1} \hat{\mathcal{R}}(h_t) + \mathcal{C}(\mathcal{H}_t).$$

1.4.3 Regularized Risk Minimization

The SRM principle suffers from the cost of having a multitude of hypothesis classes over which empirical risk minimization must be performed in order to output the hypothesis. Regularized Risk Minimization (RRM) overcomes this by enforcing the penalization on a *per-function* basis rather than on a *per-class* basis. More specifically, RRM works with a single hypothesis class $\mathcal{H}$ and chooses a *Regularizer* $r : \mathcal{H} \to \mathbb{R}_+$ thus assigning a measure of complexity to each function in the hypothesis class. The hypothesis is then chosen as follows:

$$\hat{h} = \arg\min_{h \in \mathcal{H}} \hat{\mathcal{R}}(h) + \lambda r(h),$$

where $\lambda > 0$ is a regularization parameter. This allows the RRM principle to implement the Occam's Razor but with several additional advantages

1. **Efficiency**: RRM requires only a single ERM step as opposed to SRM that requires several. Moreover, in several cases, the regularizers allow one to directly use efficient optimization routines to learn the hypothesis.

2. **Low Bias**: because of its independence to the capacity of the hypothesis class $\mathcal{H}$, RRM allows one to work with classes with very large capacity directly. This is indispensable for kernel-based learning algorithms that typically work in very high dimensional spaces.

3. **Sparsity**: With a proper choice of regularizer, the RRM principle allows one to learn sparse predictors that allow faster predictions at test time.

There are several other learning strategies that overcome the problem of overfitting such as *Minimum Description Length* (MDL), *Bayesian Information Criterion* (BIC) and *Cross Validation* (CV) that we shall not discuss here. Of course one is still faced with the question of how to implement these strategies on various hypothesis classes. In this thesis, we shall mostly focus on regularized risk minimization as it emerges naturally within the framework of large margin predictors. This shall be the topic of discussion in Chapter 2 where we shall use the kernel learning framework to implement this principle.

For now, we move on to look at some properties of these learning strategies. In particular, we shall be interested in knowing if these learning techniques offer any generalization guarantees. In order to do so we introduce in Section 1.5 below, the framework of statistical learning theory using which we will try to prove generalization guarantees for hypotheses learned using the empirical and regularized risk minimization principles.

1.5 Statistical Learning Theory

A learning algorithm is said to offer a *Generalization Guarantee* if it produces a hypothesis $\hat{h} \in \mathcal{H}$ such that the excess risk i.e. $\mathcal{R}(\hat{h}) - \mathcal{R}(\mathcal{H})$ is bounded. Note that since the learning algorithm only gets to see a finite sample of points, it is not trivial to argue about such

bounds. In the following discussion we shall first analyze the ERM principle from the point of view of generalization guarantees and then briefly look at the RRM principle.

Consider a fixed function $h \in \mathcal{H}$. The strong law of large numbers (see for example Dudley, 2002) tells us that in the limit, the empirical risk of h converges to its true risk (or *Population Risk*) with probability one. To see this, suppose we have a training set $\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ of n randomly chosen points. Construct the function $\tilde{h} : x \mapsto \ell(h(x), f^t(x))$. Since the training points are randomly chosen, applying the strong law of large numbers gives us

$$\lim_{n\to\infty} \mathbb{P}\left[\frac{1}{n}\sum_{i=1}^{n} \ell(\tilde{h}(\mathbf{x}_i)) - \mathbb{E}\left[\!\left[\tilde{h}(\mathbf{x})\right]\!\right] = 0\right] = 1,$$

which gives us

$$\lim_{n\to\infty} \mathbb{P}\left[\hat{\mathcal{R}}(h) - \mathcal{R}(h) = 0\right] = 1,$$

which in turn indicates that in the limit of having infinite training data, the empirical risk will converge to the true risk. This seems to indicate that the ERM principle, which selects the hypothesis with the minimum empirical risk, does offer a generalization guarantee. In the following discussion, we see how to formally prove such a generalization bound.

However, this result applies only to the fixed function h. The hypothesis $\hat{h}$, on the other hand depends upon the training points and hence is not a fixed function. To get generalization bounds out of such an argument, we would have to apply such a result *uniformly* over the hypothesis class $\mathcal{H}$.

1.5.1 From Pointwise to Uniform Convergence

The strong law of large numbers guarantees, as we saw, that for any fixed function $h \in \mathcal{H}$, we have the empirical risk converging to the population risk. However, since the empirical risk minimizer $\hat{h}$ is not known a priori and depends upon the training set, it is not a fixed function and hence this result cannot be applied directly to $\hat{h}$. To get around this problem, instead of considering pointwise convergence, we consider uniform convergence. More specifically, instead of looking at the quantity $\hat{\mathcal{R}}(h) - \mathcal{R}(h)$ for a fixed function $h \in \mathcal{H}$, we instead look at the quantity

$$\sup_{h\in\mathcal{H}} \left\{\hat{\mathcal{R}}(h) - \mathcal{R}(h)\right\}.$$

It turns out that proving uniform convergence bounds for hypothesis classes allows us to give generalization guarantees for algorithms implementing the ERM principle over those hypothesis classes. This is formalized in the theorem below:

Theorem 1.1. *Suppose the hypothesis class $\mathcal{H}$ that demonstrates uniform convergence i.e.*

$$\lim_{n\to\infty} \mathbb{P}\left[\sup_{h\in\mathcal{H}} \left\{\hat{\mathcal{R}}(h) - \mathcal{R}(h)\right\} = 0\right] = 1$$

Let $\hat{h} = \arg\min_{h \in \mathcal{H}} \hat{\mathcal{R}}(h)$ *be the empirical risk minimizer over* $\mathcal{H}$*. Then we have*

$$\lim_{n \to \infty} \mathbb{P}\left[\mathcal{R}(\hat{h}) - \mathcal{R}(\mathcal{H}) = 0\right] = 1$$

Proof. The proof proceeds in three steps outlined below:

1. Use the fact that $\hat{\mathcal{R}}(\hat{h}) - \mathcal{R}(\hat{h}) \leq \sup_{h \in \mathcal{H}} \left\{\hat{\mathcal{R}}(h) - \mathcal{R}(h)\right\}$ to conclude that

$$\lim_{n \to \infty} \mathbb{P}\left[\hat{\mathcal{R}}(\hat{h}) - \mathcal{R}(\hat{h}) = 0\right] = 1$$

2. Let $h^* = \arg\min_{h \in \mathcal{H}} \mathcal{R}(h)$. Apply the law of large numbers on the fixed function h^* to conclude

$$\lim_{n \to \infty} \mathbb{P}\left[\hat{\mathcal{R}}(h^*) - \mathcal{R}(h^*) = 0\right] = 1$$

3. Combine the two results obtained above with the facts $\mathcal{R}(\hat{h}) \geq \mathcal{R}(\mathcal{H})$, $\mathcal{R}(h^*) = \mathcal{R}(\mathcal{H})$ and $\hat{\mathcal{R}}(\hat{h}) \leq \hat{\mathcal{R}}(h^*)$ to conclude that

$$\lim_{n \to \infty} \mathbb{P}\left[\mathcal{R}(\hat{h}) - \mathcal{R}(\mathcal{H}) = 0\right] = 1$$

□

What this result tells us is that in the limit of the number of training samples increasing to infinity, we will almost surely learn a hypothesis that achieves the best possible performance within the hypothesis class. Some properties of this result are noteworthy:

1. The result only holds in the limit - this is not very satisfactory since in practice we always have finite training sets. This can be remedied by using *quantitative* or *effective* versions of the strong law of large numbers such as the Hoeffding bound or McDiarmid's inequality (McDiarmid, 1989).

2. Although in its asymptotic form the result holds with probability one, this will rapidly decay to a high probability bound once we invoke any effective version of the strong law of large numbers. This tells us that there may exist training sets which will mislead us into learning a bad hypothesis. However it turns out that the probability of encountering such training sets can be made vanishingly small by choosing large enough training sets.

The preceding discussion shifts the focus onto hypothesis classes that demonstrate uniform convergence. It turns out that only hypothesis classes that are not too powerful can demonstrate uniform convergence. To see this, consider the hypothesis space $\mathcal{H}_{\text{all}}$ that contains all functions mapping points from $\mathcal{X}$ to $\mathcal{Y}$. For any given training set $\mathcal{T}$, there exists a function $h_{\mathcal{T}} \in \mathcal{H}_{\text{all}}$ such that

$$h_{\mathcal{T}}(x) = \begin{cases} f^t(x) & \text{if } x \in \mathcal{T} \\ 0 & \text{if } x \notin \mathcal{T} \end{cases}$$

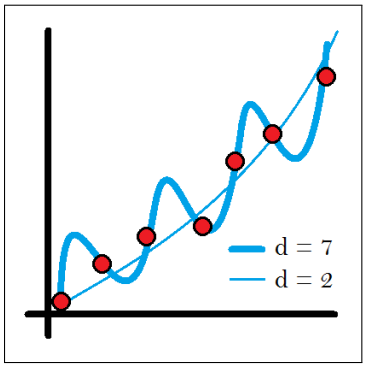


Figure 1.3: Overfitting to the training data.

Clearly $\hat{\mathcal{R}}(h_{\mathcal{T}}) = 0$ but $\mathcal{R}(h_{\mathcal{T}})$ is very large: this happens since the function $h_{\mathcal{T}}$, owing to the power of the hypothesis class $\mathcal{H}$, is simply able to memorize whatever information is present in the training set but fails to acquire any inductive capabilities. Thus, the hypothesis class $\mathcal{H}_{\text{all}}$ fails to demonstrate uniform convergence and empirical risk minimization will fail for this hypothesis class. This example demonstrates some important issues with the ERM principle:

1. **Overfitting**: this happens when the learning algorithm implementing the ERM principle tries to fit the hypothesis too closely to the training data unmindful of its responsibility to generalize to unseen points. This especially hurts the learning algorithm if there is noise in the training data. Figure 1.3 illustrates this for the case of real-valued regression: although a quadratic function is able to provide a close approximation to the training set, a blind pursuit of the ERM principle led the algorithm to fit a degree 7 curve to the data. This will usually lead to poor generalization properties for the learned predictor.

2. **Capacity of Hypothesis Class**: the ERM principle is more prone to overfitting when working with hypothesis classes that have large capacity. For instance, notice that we were able to learn a poor hypothesis such as $h_{\mathcal{T}}$ precisely because the hypothesis class $\mathcal{H}_{\text{all}}$ contained all possible functions from $\mathcal{X}$ to $\mathcal{Y}$. This could not have happened had we restricted $\mathcal{H}$ to be a set of "smooth" functions. The overfitting in Figure 1.3 could also have been prevented if the hypothesis class was constrained to contain only low degree polynomials. Informally, the explaining power of the functions inside a hypothesis class is known as its *Capacity* and having a very high capacity hypothesis class can be counter-productive. This notion will be made formal when we discuss generalization bounds for kernel learning algorithms in Chapter 2.

For sake of completeness, we give below, uniform convergence bounds for a toy hypothesis class to both motivate the notion of capacity a bit more formally as well as to introduce some well known results that shall be used in the rest of the thesis. First of all we introduce the McDiarmid's inequality below.

Theorem 1.2 (McDiarmid's inequality (McDiarmid, 1989)). *Let $X_1, \ldots, X_n$ be independent random variables taking values in some set $\mathcal{X}$. Furthermore, let $f : \mathcal{X}^n \to \mathbb{R}$ be a function of n variables that satisfies, for all $i \in [n]$ and all $x_1, \ldots, x_n, x_i' \in \mathcal{X}$,*

$$\left| f\left(x_1, \ldots, x_i, \ldots, x_n\right) - f\left(x_1, \ldots, x_i', \ldots, x_n\right) \right| \leq c_i$$

then for all $\varepsilon > 0$, we have

$$\mathbb{P}\left[f - \mathbb{E}\left[\!\left[f\right]\!\right] > \varepsilon\right] \leq \exp\left(\frac{-2\varepsilon^2}{\sum_{i=1}^n c_i^2}\right)$$

Taking the special case $f(x_1, \ldots, x_i, \ldots, x_n) = \frac{1}{n}\sum_{i=1}^{n} g(x_i)$, we get the Hoeffding bound.

Theorem 1.3 (Hoeffding's inequality (Hoeffding, 1963)). *Let $X_1, \ldots, X_n$ be independent random variables taking values in some set $\mathcal{X}$. Furthermore, let $g : \mathcal{X} \to \mathbb{R}$ be a function that satisfies, for all $x \in \mathcal{X}$, $|g(x)| \leq c$ then for all $\varepsilon > 0$, we have*

$$\mathbb{P}\left[\frac{1}{n}\sum_{i=1}^{n} g(X_i) - \mathbb{E}\left[\!\left[g\right]\!\right] > \varepsilon\right] \leq \exp\left(\frac{-n\varepsilon^2}{2c^2}\right)$$

Note that both inequalities can be made two sided by an independent invocation of the inequalities with $f = -f$ and $g = -g$ respectively.

1.5.2 A Toy Uniform Convergence Bound

Consider the hypothesis class $\mathcal{H} = \{h_1, \ldots, h_m\}$ with only a finite number of functions i.e. $m < \infty$. Suppose we have n points in the training set. Applying the Hoeffding's inequality to each h_i individually we get

$$\mathbb{P}\left[\mathcal{R}(h_i) - \hat{\mathcal{R}}(h_i) > \varepsilon\right] \leq \exp\left(\frac{-n\varepsilon^2}{B^2}\right)$$

where we have assumed that the loss function ℓ takes values in $[0, B]$. Taking a union bound over the entire hypothesis class and rearranging gives us

$$\mathbb{P}\left[\sup_{h \in \mathcal{H}} \left\{\mathcal{R}(h) - \hat{\mathcal{R}}(h)\right\} > B\sqrt{\frac{\log m + \log \frac{1}{\delta}}{n}}\right] \leq \delta$$

A few observations are in order here:

1. In the limit $n \to \infty$, the hypothesis class $\mathcal{H}$ does indeed demonstrate uniform convergence. The result gives effective finite sample convergence bounds for the same.

2. The excess risk term depends upon the size of the hypothesis class (notice the $\log m$ term). This gives us some indication that the more voluminous the hypothesis class, the greater the risk of learning a bad hypothesis. There exist other notions of

capacity that are able to handle infinite hypothesis classes as well. Some of these will be introduced in Chapter 2.

1.5.3 Generalization Guarantees for Regularized Risk Minimization

We now address the problem of generalization guarantees for algorithms implementing the RRM principle. Recall that the RRM principle chooses the hypothesis as

$$\hat{h} = \arg\min_{h \in \mathcal{H}} \hat{\mathcal{R}}(h) + \lambda r(h).$$

Previously, using uniform convergence, we were able to bound $\mathcal{R}(\hat{h})$ in terms of $\hat{\mathcal{R}}(\hat{h})$ and the capacity of the class $\mathcal{H}$. Since here the notion of a capacity is function dependent and not class dependent, we expect bounds on $\mathcal{R}(\hat{h})$ in terms of $\hat{\mathcal{R}}(\hat{R})$ and $r(\hat{h})$. Since the hypothesis $\hat{h}$ and its complexity $r(\hat{h})$ are not known a priori, this poses a problem. A precise answer to this problem requires a rather delicate argument. Such arguments were first given for the SVM algorithm in (Shawe-Taylor et al., 1998) by constructing data dependent class hierarchies and simulating structural risk minimization upon them. This method and its properties are beyond the scope of this thesis.

However, it turns out that one can still give weaker guarantees using a simple argument. The trick here is to obtain an a priori bound on $r(\hat{h})$ by obtaining an *effective* capacity bound on the class $\mathcal{H}$. To do this, consider a fixed hypothesis $h_0 \in \mathcal{H}$. Due to the minimization step carried out by the RRM principle, we have

$$\hat{\mathcal{R}}(\hat{h}) + \lambda r(\hat{h}) \leq \hat{\mathcal{R}}(h_0) + \lambda r(h_0).$$

Since our loss function is non-negative valued, we have $\hat{\mathcal{R}}(\hat{h}) \geq 0$ which gives us

$$r(\hat{h}) \leq r(h_0) + \frac{\hat{\mathcal{R}}(h_0)}{\lambda}.$$

Thus, one can restrict oneself to the hypothesis class

$$\mathcal{H}_\lambda := \left\{ h \in \mathcal{H} : r(h) \leq r(h_0) + \frac{\hat{\mathcal{R}}(h_0)}{\lambda} \right\}$$

and prove generalization guarantees by showing that the class $\mathcal{H}_\lambda$ demonstrates uniform convergence.

1.6 Concluding Remarks

At first glance, uniform convergence may seem like an overkill and milder notions of convergence might seem possible. However, it is known (see Vapnik, 2000, for example) that for simple learning problems such as classification and regression, uniform convergence is both a necessary as well as sufficient condition to be able to learn from a hypothesis

class $\mathcal{H}$. In such cases, empirical risk minimization is the most natural learning principle as well.

It should also be noted that for more general learning problems such as stochastic convex optimization in Hilbert spaces, empirical risk minimization actually fails (Shalev-Shwartz et al., 2010a) since these learning problems have associated hypothesis classes that do not exhibit uniform convergence. However one can still learn from these hypothesis classes using alternate learning paradigms. The work of Shalev-Shwartz et al. (2010a) provides one such mechanism called Stable Asymptotic Empirical Risk Minimization (stable AERM) which is shown to characterize learnability in the "General Setting of Learning" (introduced in (Vapnik, 2000)) considered by them. However, for our purposes, it would be sufficient to restrict the discussion to hypothesis spaces that do exhibit uniform convergence.

At this point we conclude this introductory chapter with a short glance at the organization of the thesis.

1.7 Organization of the Thesis

This thesis is organized into seven chapters (including the current one) and three appendices. In this chapter we gave a brief introduction to some learning formalisms. In Chapter 2, we shall look at kernel-based learning techniques that utilize the regularized risk minimization principle. We shall also look at techniques to prove excess risk bounds for kernel learning algorithms by proving uniform convergence bounds for kernel hypothesis classes. This chapter will also thrown open some questions that will be answered subsequently as a part of this thesis.

Chapter 3 will introduce a method to accelerate the training and test times of kernel methods. We shall focus on dot product kernels in our work, a family that includes a large number of widely used kernels such as the polynomial kernels and the Gaussian kernel. Our method relies on a technique to achieve uniform approximation of the kernel values over a compact domain via a finite dimensional feature map.

In Chapters 4 and 5 we shall address the problem of learning with indefinite kernels. We shall present a learning framework that admits the use of indefinite kernels and offers fast training and testing routines apart from having learning theoretic generalization guarantees. Chapter 4 shall address the problem of classification whereas Chapter 5 will extend the discussion to other supervised learning problems such as regression and ranking.

In Chapter 6, we shall look at the problem of kernel learning, i.e. the problem of choosing an appropriate kernel for a given task. We shall, however, address a much more general problem of learning with pairwise loss functions that includes problems such as multiple kernel learning, metric learning and bipartite ranking. We shall present an online learning framework for this task, within which we shall describe space-bounded learning algorithms that offer regret and generalization bounds.

Finally, in Chapter 7, we shall conclude the thesis with some directions for future research and open problems.

Throughout this document, theorems and lemmata that were not originally proven as a part of this work cite, as a part of their statement, the work that originally presented the proof.

Chapter 2

Introduction to Kernel Learning

Contents

Abstract *This chapter gives an introduction to kernel-based learning algorithms. The kernel learning method is explained with the help of popular kernel algorithms such as Support Vector Machines and Support Vector Regression. Generalization bounds for kernel learning algorithms are also discussed. Certain issues that shall be addressed as a part of this thesis are motivated in preparation for the forthcoming chapters.*

2.1 Introduction

In the rest of this chapter we shall instantiate the empirical risk minimization principle within the kernel learning framework. We shall also discuss generalization bounds for kernel learning algorithms such as the *Support Vector Machine* by proving uniform convergence bounds for the associated hypothesis classes. We shall also briefly discuss kernel algorithms for a few unsupervised learning problems. In order to present the essentials of kernel learning better, we shall restrict ourselves to supervised learning tasks for most of this chapter. We shall however, toward the end of the chapter, briefly look at how kernel methods are applied to solve unsupervised learning tasks such as clustering and component analysis. We recall that the supervised framework involved the student getting a

training set $\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ from the teacher. The student was then supposed to use the training data to learn a hypothesis h from some hypothesis class $\mathcal{H}$.

2.2 Learning with Kernels

A kernel is a (real valued) bivariate function defined upon our domain of interest

$$K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$$

Typically this kernel would be taken to encode some notion of *similarity* or *affinity* among objects in the domain. Having access to such a notion of affinity, the most obvious learning "algorithm" would be a *nearest neighbor* approach (e.g. see Duda et al. (2000) for example) where a test point is assigned the label of the training point closest to it. However, due to algorithmic and learning theoretic considerations, common kernel based methods choose a more "populist" approach. For instance, if the task at hand is one of classification (i.e. $y_i = \pm 1$), then the student tries to learn a hypothesis of the following form:

$$h(\mathbf{x}) = \begin{cases} 1 & \text{if } \sum_{y_i=1} \alpha_i K(\mathbf{x}, \mathbf{x}_i) \geq \sum_{y_i=-1} \alpha_i K(\mathbf{x}, \mathbf{x}_i) \\ -1 & \text{if } \sum_{y_i=1} \alpha_i K(\mathbf{x}, \mathbf{x}_i) < \sum_{y_i=-1} \alpha_i K(\mathbf{x}, \mathbf{x}_i) \end{cases}$$

We can interpret the student as doing the following: he learns a set of weights $\alpha_i \geq 0$, one for each training point. At the time of testing, he simply takes the test object $\mathbf{x}$ and calculates how similar is it to each of the training objects by calculating the values $K(\mathbf{x}, \mathbf{x}_i)$. This allows him to take a vote among the training points as to which class is similar to the test point the most, with each training point getting α_i votes. The class that wins gets to assign its own label to the test object. The above expression can also be written succinctly as follows

$$h(\mathbf{x}) = \text{sign}\left(\sum_i \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i)\right)$$

A similar form of hypothesis is used in case of real valued regression

$$h(\mathbf{x}) = \sum_i \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i)$$

where $y_i \in \mathbb{R}$ and we are taking a weighted average of the training labels with the kernel (similarity) values acting as weights. In both the examples above, the weights α_i can be interpreted as denoting the *importance* of the training point in the voting process. Thus points far from the decision boundary, that are not expected to be very discriminative, often end up getting values of α_i close to zero.

2.2.1 Learning with Mercer Kernels

Although we chose to interpret the form of the hypothesis as one implementing a vote among the training points, mostly for sake of clarity and motivation, the classical derivation of this form of the hypothesis actually comes from the learning algorithm used with kernels. To investigate these methods, we first need to introduce the notion of *Mercer Kernels*.

Mercer kernels do indeed encode some notion of similarity among the objects in the domain, but they have several other properties. We shall briefly discuss some of them here, choosing to direct the reader to encyclopedic texts such as (Schölkopf and Smola, 2002) for detailed discussions. A kernel is said to be a Mercer or a *Positive semi-definite* kernel if it satisfies the following two properties:

1. Symmetry: for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, we have $K(\mathbf{x}, \mathbf{y}) = K(\mathbf{y}, \mathbf{x})$.
2. Positive semi-definiteness: for all $n \in \mathbb{N}$, all $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathcal{X}$, the *Gram Matrix* $\mathbf{G}_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$ is positive semi-definite i.e. for all $\mathbf{c} \in \mathbb{R}^n, \mathbf{c}^\top \mathbf{G}\mathbf{c} \geq 0$.

Due to the work of Mercer (1909), we know that a kernel K that is continuous on a compact domain $\mathcal{X}$ and satisfies the above property, admits a representation of the following form:

$$K(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{\infty} \lambda_i \psi_i(\mathbf{x}) \psi_i(\mathbf{y}) \tag{2.1}$$

where $\lambda_i \geq 0$ and the functions $\psi_i : \mathcal{X} \to \mathbb{R}$ are the eigenfunctions of the integral operator corresponding to the kernel $T_K : L_2(\mathcal{X}) \to \mathbb{R}$ defined as

$$(T_K f)(.) = \int_{\mathcal{X}} K(\cdot, \mathbf{x}) f(\mathbf{x}) d\mathbf{x}$$

The functions ψ_i form an orthonormal basis of $L_2(\mathcal{X})$, the space of all square integrable functions over $\mathcal{X}$. A noticeable corollary of the form of the kernel presented in Equation 2.1 is that the kernel should be representable in some Hilbert Space as an inner product since we can conceive of a map $\Psi : \mathcal{X} \to \mathcal{H} \subseteq L_2(\mathcal{X})$ such that

$$\Psi(\mathbf{x}) = (\psi_1(\mathbf{x}), \psi_2(\mathbf{x}), \ldots)$$

which would indicate that $K(\mathbf{x}, \mathbf{y}) = \langle \Psi(\mathbf{x}), \Psi(\mathbf{y}) \rangle$. Although it would take some more work to formally prove properties of this map such as convergence to the kernel value and boundedness, the intuition nevertheless is correct in that every Mercer kernel can be realized as an inner product in a Hilbert Space that is referred to as the *Reproducing Kernel Hilbert Space* denoted by $\mathcal{H}_K$ via a map Φ_K. We do not go into the details of the reproducing properties of the kernel and refer the reader to standard texts such as (Schölkopf and Smola, 2002) for the same. Although this map and the corresponding RKHS are not unique in general, there are canonical constructions for the same.

Given these properties of Mercer Kernels, we can now introduce the Support Vector Machine (SVM) and Support Vector Regression (SVR) algorithms. The SVM and SVR algorithms rely on learning linear hypotheses. Thus, if the object of interest $\mathbf{x}$ is represented as a feature vector $\Phi(\mathbf{x}) \in \mathbb{R}^d$ (note that here the feature space is a Euclidean space $\mathbb{R}^d$), the SVM learns a hypothesis that looks like a hyperplane in the ambient space i.e.

$$h(\mathbf{x}) = \text{sign}\left(\langle \mathbf{w}, \Phi(\mathbf{x}) \rangle\right),$$

for some $\mathbf{w} \in \mathbb{R}^d$. Correspondingly, SVR tries to fit a linear function to the regression problem by learning a hypothesis that looks like

$$h(\mathbf{x}) = \langle \mathbf{w}, \Phi(\mathbf{x}) \rangle$$

In order to learn such hypotheses, the SVM and SVR algorithm utilize the Regularized Empirical Risk Minimization principle (see Section 1.4.3). More specifically, if the training set provided is $\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ then the following objective is sought to be minimized:

$$\min_{f \in \mathbb{R}^d} \quad \sum_{i=1}^{n} \ell(f(\mathbf{x}_i), y_i) + \lambda \|f\|$$

where $f(\mathbf{x}_i) = \langle f, \Phi(\mathbf{x}_i) \rangle$[1]. Note that the regularizer used here is a norm. Usually the L2 norm (or its squared version) or the sparsity inducing L1 norm is used. The loss function ℓ depends upon the application. For classification using SVM, the hinge loss function is used. For logistic regression, the logistic loss function is used. For regression using SVR, the squared loss or the ε-insensitive loss function is used.

The accuracy of the learned hypothesis depends on the feature space used. In order to achieve high accuracy, rich feature spaces are required. However, rich and high dimensional feature spaces present a challenge to the learning algorithm by making the problem of solving the optimization problem given above, very expensive. In order to overcome this challenge, we change the optimization problem by looking at the *dual problem* instead. We omit the derivation of the dual problem and simply present the dual of the SVM objective with the hinge loss function

$$\ell_{\text{hinge}}(y_1, y_2) = [1 - y_1 y_2]_+ .$$

For this loss function, the dual problem looks like the following:

$$\begin{aligned} \max_{\boldsymbol{\alpha} \in \mathbb{R}^n} \quad & \sum_{i=1}^{n} \boldsymbol{\alpha}_i - \frac{1}{2} \sum_{i,j} \boldsymbol{\alpha}_i \boldsymbol{\alpha}_j y_i y_j \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle \\ s.t. \quad & \boldsymbol{\alpha}_i \in [0, \lambda] \end{aligned}$$

[1] ↑ We have omitted the bias term for sake of simplicity - this will later simplify the presentation of the dual formulation as well

The resulting classifier can be retrieved after solving the dual as $f(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i y_i \Phi(\mathbf{x}_i)$. Note that the above problem requires access to pairwise inner products among the training points alone and not the feature vectors. This allows us to utilize kernels in the learning process where, in the dual problem, we use the values $K(\mathbf{x}_i, \mathbf{x}_j)$ instead of $\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$. Since kernels realize themselves as inner products in some RKHS, the algorithm will implicitly start using the RKHS as the feature space.

Since this effectively makes the algorithm oblivious to the feature space size, we can use very rich (even infinite dimensional e.g. the Gaussian kernel) feature spaces in the learning process. For instance, using polynomial kernels $K(\mathbf{x}, \mathbf{y}) = (1 + \langle \mathbf{x}, \mathbf{y} \rangle)^p$, we can learn polynomial functions of an arbitrarily high degree as our classifier or regression function. In such cases, the form of the hypothesis changes to the following:

$$h_{\boldsymbol{\alpha}}(\mathbf{x}) = \sum_{i=1}^{n} \boldsymbol{\alpha}_i y_i K(\mathbf{x}, \mathbf{x}_i).$$

Note that this is exactly the form of the classifier we presented at the beginning of this section.

Mercer kernel learning has been studied extensively but we do not pursue it any further. An enormous amount of effort has been put in over the years to make various problems such as classification, regression, ranking, PCA, k-means etc amenable to kernel learning. A parallel line of effort has concentrated on building efficient solvers for the optimization problems that arise in kernel learning formulations. Finally, kernel learning possesses a deep and rich theory that explain why kernel learning algorithms and the techniques used therein offer superior performance. We shall discuss some of these topics later - more specifically, Chapter 3 will look at accelerating the training and test times of kernel algorithms. In the current chapter, we briefly look at how kernel algorithms can be used for some unsupervised problems (Section 2.4). We also touch upon the issue of providing generalization bounds for kernel learning algorithms in Section 2.3.

For now we turn away from Mercer kernels and look at the problem of learning with non-Mercer kernels.

2.2.2 Learning with Indefinite Kernels

If kernels are to be interpreted as measuring some form of similarity or dissimilarity without regard to formal notions such as Mercer kernels then its roots lie in the simplest and earliest form of the nearest neighbor classification algorithms (Duda et al., 2000) that implement the following simple classifier

$$h(\mathbf{x}) = \begin{cases} 1 & \text{if } d(\mathbf{x}, \mathcal{T}_+) \leq d(\mathbf{x}, \mathcal{T}_-) \\ -1 & \text{if } d(\mathbf{x}, \mathcal{T}_-) \leq d(\mathbf{x}, \mathcal{T}_+) \end{cases}$$

where $d(\mathbf{x}, \mathcal{T}_+) = \min_{y_i=+1} d(\mathbf{x}, \mathbf{x}_i)$ is the minimum distance of the test point from a positive point and $d(\mathbf{x}, \mathcal{T}_-)$ being defined analogously. Here d can be an arbitrary notion of distance (or even similarity if the classifier is appropriately defined). This simple classifier encodes the intuitive idea that closeness in the feature space should more often than not, indicate a similarity in labels.

This simple notion can be extended in several ways. For instance, one can, instead of considering all points of a class, consider only a few representative or *landmark points* while deciding upon the label. Such landmarking-based techniques have been extensively explored (Weinshall et al., 1998; Jacobs et al., 2000). The other offshoot from this basic procedure is one in which the distance (or similarity) measure being considered is realized implicitly by some embedding, just as the canonical map realizes the kernel as an inner product in the RKHS.

Such *embedding* based techniques have also been explored widely with work done on properly designing the embeddings so that in the embedding space, a good NN classifier can be learned. Such embedding spaces were used to realize several known algorithms, making them amenable to distance functions such as SVM, quadratic discriminant and Fisher linear discriminant (Pękalska and Duin, 2000; Pękalska et al., 2001; Pękalska and Duin, 2002).

The works of von Luxburg and Bousquet (2004); Gottlieb et al. (2010) developed embedding based approaches that showed how nearest neighbor algorithms can be interpreted as large margin classifiers in appropriate Banach spaces via Lipschitz embeddings, in some sense drawing parallels to large margin classifiers with Mercer kernels (Schölkopf, 2000). Some approaches (for example the work of Goldfarb (1984)) recognized the link between landmarking and embedding approaches and exploited that in the learning process.

However, some of the most focused work in distance-based classification has been in the direction of learning a distance function, most suitable to the classification problem at hand (for example, Mahalanobis metric learning formulations (Weinberger and Saul, 2009)). This is akin to the kernel choice problem in Mercer kernel learning. There have been other approaches that took landmarking based approaches forward, incorporating notions of a "good" distance function into the landmarking step in order to give generalization bounds for the learned classifier.

There has been an equal, if not greater, interest in using similarity functions (that are not necessarily Mercer kernels) in the learning process. There seem to be three main approaches in this direction. In the first approach, an effort is made to make indefinite kernels more suitable for algorithms such as the SVM. Since plugging in an indefinite kernel results in a non-convex QP that is NP-hard to solve, these approaches (for example see the work of Chen et al. (2009a); Luss and d'Aspremont (2007)) try to take the kernel and alter its spectrum (essentially getting rid of all negative eigenvalues) to make them suitable for the SVM formulation.

In the second approach, no care is taken to make the kernel matrix positive semi-definite and it is used directly in the SVM formulation. Such approaches (Haasdonk, 2005;

Ong et al., 2004) face the challenge of solving a non-convex problem at training time. The third approach in some sense, marries the landmarking and embedding approaches and creates an embedding out of landmark points. The work of Balcan and Blum (2006) introduced the notion of a "good" similarity function and used such landmarking techniques to give classifiers with provable generalization guarantees.

We do not review past work on indefinite kernels any further since Section 2.5 as well as Chapters 4 and 5 will discuss related work in this area. For now we try to briefly look at the question of how can one give generalization bounds for predictors learned using kernels. This shall form the basis of many theoretical guarantees that we give in later chapters.

2.3 Generalization Bounds for Kernel Predictors

The question we try to address in this section is when we can expect kernel hypothesis classes to demonstrate uniform convergence. Recall that a hypothesis class is said to demonstrate uniform convergence if

$$\mathbb{P}\left[\lim_{n\to\infty}\sup_{h\in\mathcal{H}}\left\{\hat{\mathcal{R}}(h)-\mathcal{R}(h)\right\}=0\right]=1$$

Answering the above question will lead us to formal notions of *Capacity* of hypothesis classes - only classes that are not over-powerful in their explaining power will demonstrate uniform convergence (although in the light of the result in Shalev-Shwartz et al. (2010a) and Vapnik (2000), this corresponds to learnability only for simple learning problems such as regression and classification).

For sake of simplicity, we shall restrict ourselves to regression problems in this section. We shall also not consider learning formulations that utilize non-Mercer kernels since those will be addressed in detail in Chapters 4 and 5.

We shall address the problem of uniform convergence by trying to prove *finite sample* bounds instead. More specifically we will show that for any $\varepsilon > 0$,

$$\mathbb{P}\left[\sup_{h\in\mathcal{H}}\left\{\hat{\mathcal{R}}(h)-\mathcal{R}(h)\right\}>\varepsilon\right]\leq\delta_n$$

such that $\delta_n \downarrow 0$.

2.3.1 Covering Numbers

The first technique that we present is a simple technique that works best if the feature spaces are low dimensional and is called the method of *Covering Numbers*. Suppose we are trying to solve a regression problem using SVR. Also suppose for now that there exists a constant $\Lambda > 0$ such that our learning algorithm shall only output hypothesis with $\|f\| \leq \Lambda$ (note that such a bound can easily be obtained using the argument presented in

Section 1.5.3). Thus, our hypothesis class is effectively

$$\mathcal{F} = \left\{ f \in \mathbb{R}^d, \|f\| \leq \Lambda \right\}$$

To use the covering number argument, we need to be able to tell how far from each other are two functions in $\mathcal{F}$. This requires a metric to be established over this set. For sake of simplicity, we present the covering number argument using the L_∞ norm defined as follows:

$$\|f\|_\infty := \sup_{\mathbf{x} \in \mathcal{X}} |f(\mathbf{x})|$$

Thus, the distance between two functions $f_1, f_2 \in \mathcal{F}$ is given by

$$\|f_1 - f_2\| = \sup_{\mathbf{x} \in \mathcal{X}} |f_1(\mathbf{x}) - f_2(\mathbf{x})|$$

which is simply the maximum discrepancy between the value of the two functions on any domain point $\mathbf{x} \in \mathcal{X}$. This allows us to do the following: for any fixed function $f \in \mathcal{F}$ we know by a simple application of the Höffding bound (see Theorem 1.3) that the empirical and population risks are very close to each other i.e. $\mathbb{P}\left[\mathcal{R}(f) - \hat{\mathcal{R}}(f) > \varepsilon/2\right] \leq \delta$ for training set size $n = \Omega\left(\frac{1}{\varepsilon^2} \log \frac{1}{\delta}\right)$. What we do now is select a (finite) set of functions $\mathcal{C} \subset \mathcal{F}$ such that every function $f \in \mathcal{F}$ is at most $\varepsilon/4$ distance away from some function in $\mathcal{C}$. Suppose C has N functions in total.

By a simple union bound we can show that with probability at least $1 - \delta$, for all the functions $f \in \mathcal{C}$, we have $\mathcal{R}(f) - \hat{\mathcal{R}}(f) \leq \varepsilon/2$. This requires the number of training points to be increased to $n = \Omega\left(\frac{1}{\varepsilon^2} \log \frac{N}{\delta}\right)$. However, this allows us to prove uniform convergence guarantees in the following way. For any $f \in \mathcal{F}$, let $\tilde{f} \in \mathcal{C}$ be such that $\left\|f - \tilde{f}\right\|_\infty \leq \varepsilon/4$. Then we have

$$\begin{aligned} \mathcal{R}(f) - \hat{\mathcal{R}}(f) &= \mathcal{R}(f) - \mathcal{R}(\tilde{f}) + \mathcal{R}(\tilde{f}) - \hat{\mathcal{R}}(\tilde{f}) + \hat{\mathcal{R}}(\tilde{f}) - \hat{\mathcal{R}}(f) \\ &\leq \varepsilon/4 + \varepsilon/2 + \varepsilon/4 \leq \varepsilon \end{aligned}$$

where the bounds on the first and the third term hold due to the fact that $\left\|f - \tilde{f}\right\|_\infty \leq \varepsilon/4$ and the bound on the second term holds due to the union bound over functions in $\mathcal{C}$. Thus we have proved that with high probability

$$\mathbb{P}\left[\sup_{h \in \mathcal{H}} \left\{ \hat{\mathcal{R}}(h) - \mathcal{R}(h) \right\} > \varepsilon \right] \leq \delta$$

which concludes the uniform convergence proof. The set $\mathcal{C}$ is referred to as an *ε-cover* or an *ε-net* over the set $\mathcal{F}$ with respect to the L_∞ norm. The minimum number of functions N required to establish such a net or a cover is known as the *Covering number* of the set. The Covering number is a widely used of notion of capacity of a set of hypothesis functions and there are several results (see for example (Cucker and Smale, 2001)) that bound this number for a variety of function classes. For instance, the covering number for the class

of norm bounded linear functions in d dimensions at scale ε is of the order of $\mathcal{O}\left(\frac{1}{\varepsilon}\right)^d$.

This notion of capacity shall be used in Chapters 3 and 4 to give generalization bounds for classifiers learned using indefinite kernels and other algorithms.

2.3.2 Rademacher Complexity

The reader would have noticed that the covering number has a strong (actually exponential) dependence on the input dimensionality. This has an adverse effect on generalization bounds proved using this notion of capacity as they tend to have a linear dependence on the ambient dimensionality of the feature space. This seems to prevent us from using rich feature spaces of the kind kernel based learning algorithms offer. To overcome this we use more refined notions of capacity. We introduce one such notion called the *Rademacher complexity*. As before, we present the outline of a generalization bound proof which will lead us to the definition of Rademacher complexity.

Recall that our aim is to bound the quantity $\mathcal{R}(\hat{h}) - \hat{\mathcal{R}}(\hat{h})$ where $\hat{h}$ is the empirical risk minimizer. For sake of clarity we will discard the notation $\mathcal{R}(\cdot)$ and $\hat{\mathcal{R}}(\cdot)$ in favor of the more explicit $\mathbb{E}\left[\!\left[\ell(\cdot)\right]\!\right]$ to denote the population risk and $\mathbb{E}\left[\!\left[\ell(\cdot, \mathcal{T})\right]\!\right]$ to denote empirical risk with respect to the training sample $\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$. Note that, for a sample $\tilde{\mathcal{T}} = \{(\tilde{\mathbf{x}}_i, \tilde{y}_i)\}_{i=1}^n$ that is independent of $\mathcal{T}$, we have

$$\mathbb{E}\left[\!\left[\ell(\hat{h})\right]\!\right] = \mathop{\mathbb{E}}_{\tilde{\mathcal{T}}}\left[\!\left[\mathbb{E}\left[\!\left[\ell(\hat{h}, \tilde{\mathcal{T}})\right]\!\right]\right]\!\right]$$

where the outer expectation is over the choice of the *ghost sample* $\tilde{\mathcal{T}}$. This allows us to present the following chain of (in)equalities

$$\begin{aligned}
\mathcal{R}(\hat{h}) - \hat{\mathcal{R}}(\hat{h}) &= \mathbb{E}\left[\!\left[\ell(\hat{h})\right]\!\right] - \mathbb{E}\left[\!\left[\ell(\hat{h}, \mathcal{T})\right]\!\right] \\
&= \mathop{\mathbb{E}}_{\tilde{\mathcal{T}}}\left[\!\left[\mathbb{E}\left[\!\left[\ell(\hat{h}, \tilde{\mathcal{T}})\right]\!\right]\right]\!\right] - \mathbb{E}\left[\!\left[\ell(\hat{h}, \mathcal{T})\right]\!\right] \\
&\leq \sup_{h \in \mathcal{H}} \left\{ \mathop{\mathbb{E}}_{\tilde{\mathcal{T}}}\left[\!\left[\mathbb{E}\left[\!\left[\ell(h, \tilde{\mathcal{T}})\right]\!\right]\right]\!\right] - \mathbb{E}\left[\!\left[\ell(h, \mathcal{T})\right]\!\right] \right\}.
\end{aligned}$$

Now it is easy to see that the function $g(\mathcal{T}) = \sup_{h \in \mathcal{H}} \left\{ \mathop{\mathbb{E}}_{\tilde{\mathcal{T}}}\left[\!\left[\mathbb{E}\left[\!\left[\ell(h, \tilde{\mathcal{T}})\right]\!\right]\right]\!\right] - \mathbb{E}\left[\!\left[\ell(h, \mathcal{T})\right]\!\right] \right\}$ is perturbed by at most $\mathcal{O}\left(\frac{1}{n}\right)$ due to the perturbation of any training sample $(\mathbf{x}_i, y_i) \in \mathcal{T}$. Thus by an application of McDiarmid's inequality (see Theorem 1.2), we have, using the above chain of inequalities, with probability at least $1 - \delta$,

$$\mathcal{R}(\hat{h}) - \hat{\mathcal{R}}(\hat{h}) \leq \mathop{\mathbb{E}}_{\mathcal{T}}\left[\!\left[\sup_{h \in \mathcal{H}} \left\{ \mathop{\mathbb{E}}_{\tilde{\mathcal{T}}}\left[\!\left[\mathbb{E}\left[\!\left[\ell(h, \tilde{\mathcal{T}})\right]\!\right]\right]\!\right] - \mathbb{E}\left[\!\left[\ell(h, \mathcal{T})\right]\!\right] \right\}\right]\!\right] + \mathcal{O}\left(\sqrt{\frac{\log \frac{1}{\delta}}{n}}\right).$$

We are thus left with the task of estimating the first term of the right hand side of the

above inequality which we do below:

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathcal{T}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left\{\mathop{\mathbb{E}}_{\tilde{\mathcal{T}}}\left[\!\!\left[\mathbb{E}\left[\!\!\left[\ell(h,\tilde{\mathcal{T}})\right]\!\!\right]\right]\!\!\right]-\mathbb{E}\left[\!\!\left[\ell(h,\mathcal{T})\right]\!\!\right]\right\}\right]\!\!\right] &= \mathop{\mathbb{E}}_{\mathcal{T}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left\{\mathop{\mathbb{E}}_{\tilde{\mathcal{T}}}\left[\!\!\left[\mathbb{E}\left[\!\!\left[\ell(h,\tilde{\mathcal{T}})\right]\!\!\right]-\mathbb{E}\left[\!\!\left[\ell(h,\mathcal{T})\right]\!\!\right]\right]\!\!\right]\right\}\right]\!\!\right] \\
&\leq \mathop{\mathbb{E}}_{\mathcal{T},\tilde{\mathcal{T}}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left\{\mathbb{E}\left[\!\!\left[\ell(h,\tilde{\mathcal{T}})\right]\!\!\right]-\mathbb{E}\left[\!\!\left[\ell(h,\mathcal{T})\right]\!\!\right]\right\}\right]\!\!\right] \\
&= \mathop{\mathbb{E}}_{\mathcal{T},\tilde{\mathcal{T}}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left\{\frac{1}{n}\sum_{i=1}^{n}\left(\ell(h(\tilde{\mathbf{x}}_i),\tilde{y}_i)-\ell(h(\mathbf{x}_i),y_i)\right)\right\}\right]\!\!\right] \\
&= \mathop{\mathbb{E}}_{\mathcal{T},\tilde{\mathcal{T}},\{\varepsilon_i\}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left\{\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i\left(\ell(h(\tilde{\mathbf{x}}_i),\tilde{y}_i)-\ell(h(\mathbf{x}_i),y_i)\right)\right\}\right]\!\!\right] \\
&\leq 2\mathop{\mathbb{E}}_{\mathcal{T},\{\varepsilon_i\}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left\{\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i\ell(h(\mathbf{x}_i),y_i)\right\}\right]\!\!\right].
\end{aligned}$$

The quantity $\mathcal{R}_n(\mathcal{H}) = \mathop{\mathbb{E}}_{\mathcal{T},\{\varepsilon_i\}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left\{\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i\ell(h(\mathbf{x}_i),y_i)\right\}\right]\!\!\right]$ is referred to as the *Rademacher Complexity* of the hypothesis set $\mathcal{H}$. The preceding calculations show that with probability at least $1-\delta$, we have

$$\mathcal{R}(\hat{h}) - \hat{\mathcal{R}}(\hat{h}) \leq 2\mathcal{R}_n(\mathcal{H}) + \mathcal{O}\left(\sqrt{\frac{\log\frac{1}{\delta}}{n}}\right).$$

For several function classes the Rademacher Averages behave as $\mathcal{R}_n(\mathcal{H}) = \mathcal{O}\left(\sqrt{\frac{1}{n}}\right)$. There are several ways of estimating the Rademacher complexity (one of them via calculation of Covering Numbers). However, refined calculations are possible (for instance see the work of Kakade et al. (2008)) that, for certain function classes, yield Rademacher Averages that have no dependence on feature space dimensionality and very mild dependence on feature dimensionality for some other function classes.

There are several other notions of capacity that are widely used to analyze the generalization abilities of learning algorithms for various learning problems. Prominent among these are the *VC dimension* that characterizes learnability for classification problems, *Fat shattering dimension* that generalizes to real valued learning problems, *Dudley's integral* that is widely used to study the convergence of Gaussian processes, *Gaussian complexity* that uses Gaussian random variables instead of Rademacher random variables and *Uniform entropy number*.

These notions of capacity are interlinked and frequently one can bound one in terms of the other. For instance, for a binary function class (for classification) with VC dimension d, it can be shown that the Rademacher averages behave as $\mathcal{R}_n(\mathcal{H}) = \mathcal{O}\left(\sqrt{\frac{d}{n}}\right)$ (Bartlett and Mendelson, 2002). However, these details are beyond the scope of this discussion. We will revisit some of these techniques in the subsequent chapters to prove generalization bounds for the algorithms proposed in this thesis. For now we move on to a brief discussion on some kernel based algorithms for unsupervised learning problems.

2.4 Unsupervised Learning with Kernels

Although we have only discussed supervised learning problems such as classification and regression in this chapter so far, several unsupervised learning algorithms such as Component analysis and clustering also admit kernel based algorithms. Here we briefly discuss some of them, choosing to refer the reader to texts such as (Schölkopf and Smola, 2002) for more details and other algorithms.

2.4.1 Kernel Principal Component Analysis

The classical Principal Component Analysis (PCA) algorithm is very widely used as a dimensionality reduction and noise removal tool with applications to image processing and other signal processing tasks. However, the linear PCA is only available to identify linear components in data. In order to enable the algorithm to identify non linear components, the kernel PCA algorithm was proposed by Schölkopf et al. (1998). Traditional PCA operates on mean centered data by diagonalizing the covariance matrix $C = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top$ via eigenvalue decomposition or singular value decomposition by finding $(\lambda, \mathbf{v})$ pairs such that

$$\lambda \mathbf{v} = C\mathbf{v}.$$

Note that the eigenvectors (corresponding to non-zero eigenvalues) of C necessarily lie in the span of the vectors $\mathbf{x}_i$. Hence the previous condition is equivalent to

$$\lambda \langle \mathbf{x}_i, \mathbf{v} \rangle = \langle C\mathbf{v}, \mathbf{x}_i \rangle , \text{ for all } i.$$

This point is crucial in extending the PCA algorithm to kernel spaces. Consider a kernel with an associated RKHS $\mathcal{H}_K$ and feature map $\Phi_K : \mathcal{X} \to \mathcal{H}_K$. In this case, the covariance matrix looks like

$$C = \frac{1}{n}\sum_{i=1}^{n} \Phi_K(\mathbf{x}_i)\Phi_K(\mathbf{x}_i)^\top.$$

By the above argument, finding the eigenvectors of C is equivalent to finding solutions of

$$\lambda \langle \Phi_K(\mathbf{x}_i), \mathbf{v} \rangle = \langle C\mathbf{v}, \Phi_K(\mathbf{x}_i) \rangle , \text{ for all } i.$$

Since $\mathbf{v}$ lies in the span of $\Phi_K(\mathbf{x}_i)$, we can write $\mathbf{v} = \sum \alpha_j \Phi_K(\mathbf{x}_j)$. This reduces the above problem to

$$\lambda \left\langle \Phi_K(\mathbf{x}_i), \sum \alpha_j \Phi_K(\mathbf{x}_j) \right\rangle = \left\langle \frac{1}{n}\sum_{k=1}^{n} \Phi_K(\mathbf{x}_k)\Phi_K(\mathbf{x}_k)^\top \sum \alpha_j \Phi_K(\mathbf{x}_j), \Phi_K(\mathbf{x}_i) \right\rangle , \text{ for all } i.$$

which simplifies to

$$N\lambda G\boldsymbol{\alpha} = G^2\boldsymbol{\alpha}$$

where G is the Gram matrix of the training points and $\alpha = (\alpha_1, \ldots, \alpha_n)^\top \in \mathbb{R}^n$. G being symmetric, has eigenvectors that span the entire $\mathbb{R}^n$. Consequently, we can get the eigenvectors by solving

$$N\lambda\boldsymbol{\alpha} = G\boldsymbol{\alpha}$$

which simply requires diagonalizing G. There are certain details about normalizing the eigenvectors and mean centering the data that we skip here. However, once we have obtained the principal components, it is easy to compute the projection of any new point onto them using the following steps. Suppose we are considering a principal direction $\mathbf{v} = \sum \alpha_j \Phi_K i(\mathbf{x}_j)$ and are interested in finding the projection of a new point $\mathbf{x}$ onto this principal component. This can be done simply by calculating

$$\langle \Phi_K(\mathbf{x}), \mathbf{v} \rangle = \sum \alpha_j \langle \Phi_K(\mathbf{x}), \Phi_K(\mathbf{x}_j) \rangle = \sum \alpha_j K(\mathbf{x}, \mathbf{x}_j).$$

2.4.2 Kernel Clustering

The widely used k-means clustering algorithm can also be modified easily to be made to work in kernel feature spaces. The algorithm proceeds by alternating between 2 steps:

1. E Step: Cluster centers are fixed and each data point is assigned to its closest cluster center

2. M Step: Cluster assignments are fixed and cluster centers are recalculated to minimize the potential given by the within-cluster-sum-of-squares

To do this, we note that cluster centers are always recalculated as the mean of the data points that belong to the cluster. This can be generalized to note that cluster centers are always found in the span of the data points. Thus we can equivalently represent each cluster center as $c = \sum_{i=1}^n \alpha_i \mathbf{x}_i$ which becomes $c = \sum_{i=1}^n \alpha_i \Phi_K(\mathbf{x}_i)$ in case we wish to work in a kernel feature space. Thus, for each cluster center c, we need only maintain the associated $\boldsymbol{\alpha}$ vector.

Using this, computing the distance between a data point $\mathbf{x}_j$ and a cluster center $c = \sum_{i=1}^n \alpha_i \Phi_K(\mathbf{x}_i)$ is easily done as follows:

$$\begin{aligned} \|\Phi(\mathbf{x}_j) - c\|^2 &= \langle \Phi(\mathbf{x}_j) - c, \Phi(\mathbf{x}_j) - c \rangle \\ &= K(\mathbf{x}_j \mathbf{x}_j) - 2\langle \Phi(\mathbf{x}_j), c \rangle + \langle c, c \rangle \\ &= K(\mathbf{x}_j \mathbf{x}_j) - 2\sum_{i=1}^n \alpha_i K(\mathbf{x}_j, \mathbf{x}_i) + \sum_{i,j=1}^n \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \end{aligned}$$

Once we have found out (say s) data points $\mathbf{x}_{i_1}, \ldots, \mathbf{x}_{i_s}$, we can compute the new cluster center as $c = \frac{1}{s}\sum_{j=1}^n \Phi(\mathbf{x}_{i_j})$ by simply updating the corresponding $\boldsymbol{\alpha}$ vector to $\frac{1}{s}\left(\mathbb{1}_{\mathbf{x}_i \text{ in cluster } c}\right)_{i=1}^n$. To initiate the algorithm, in the beginning, we can set $\boldsymbol{\alpha}$ to random values.

Kernel algorithms have been developed for a hot of other problems such as Kernel canonical correlation analysis and Kernel Fisher discriminant analysis which we do not discuss here. We conclude this chapter with a discussion on some of the open problems that shall be looked at in later chapters as a part of this thesis.

2.5 Open Questions

Having reviewed existing work in kernel learning in some detail, we now present an overview of the broad questions this thesis shall seek to address. This section will also point to portions in subsequent chapters where further details may be found. To simplify the presentation, we observe three broad areas wherein we seek progress:

2.5.1 Accelerated Kernel Learning

A quick look at the form of kernel classifiers

$$h(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i K(\mathbf{x}, \mathbf{x}_i)$$

shows that the complexity of evaluating a kernel classifier depends on the number of "support vectors" i.e. the number of points that have non zero values of α_i. Such a form is common for other kernel learning algorithms such as kernel PCA, kernel k-means etc (see Section 2.4 and Chapter 3). This entails a phenomenon which we dub the *Curse of Support* wherein a complex classifier, ideally designed to offer better accuracy, is very slow to evaluate at test time. Moreover, for similar reasons, kernel classifiers and other algorithms are also slow to train since they always work with implicit representations of the hypotheses, the explicit forms being elements in a high (or infinite dimensional RKHS) and infeasible to represent.

This brings about a need to find alternate ways of training and predicting with kernel functions which reduces training and prediction times. Several approaches have been proposed in this direction (see Section 3.2 for details of related work in this area) which try to reduce the training or test time, we aim for methods that are provably accurate and also demonstrate appreciable speedups in practice. We present one such method for the family of dot product kernels in Chapter 3. Our work shall involve novel use of a classical result in harmonic analysis that allows us to develop fast and provably accurate approximations to the kernel values in a way that allows training and prediction routines to be accelerated.

2.5.2 Indefinite Kernel Learning

We have seen in Section 2.2.2 a plethora of approaches that have been applied to allow learning methods like SVM to use indefinite kernels. The introductory sections of Chapters 4 and 5 give more details about related work in this area. However, most existing

work in this area suffers from either of the two problem

1. Lack of Theoretical Analysis: most approaches to indefinite kernel learning discussed above suffer from the lack of a strong theoretical analysis in the form of generalization guarantees. This comes, partly due to the weak models that underlie these models.

2. Slow Training and Prediction routines: several techniques discussed previously involve expensive operations such as nearest neighbor search in high dimensional spaces, expensive eigenvalue decompositions and other spectral operations or solving non-convex formulations. This greatly inhibits the ability of these models to be scaled to larger and more complex learning problems.

Some recent work in this area (Balcan and Blum, 2006; Wang et al., 2007) has addressed these problems to a certain extent by proposing models that have properties of soundness and that, in a principled way, extend Mercer kernel learning. In our work, that is presented in two parts in Chapters 4 and 5, we propose an extended model of learning with indefinite kernels that embodies these desirable properties as well as admits algorithms that are provably accurate and fast in practice. In particular, for the problem of real valued regression, we are able to extend these models and provide complimentary learning algorithms that, both provably as well as empirically, demonstrate a support vector like effect. Our learning models, in general, are applicable to problems such as binary and multi-class classification, real-valued regression, ordinal regression and ranking.

2.5.3 The Kernel Choice Problem

In all kernel learning formulations, whether they utilize Mercer kernels or indefinite kernels, the choice of the kernel is a crucial one, frequently influencing the accuracy of the learned predictor. Consequently, a lot of effort has been devoted to approaches that try to learn the kernel itself. These range from Multiple kernel learning formulations for Mercer kernels (Cortes et al., 2010a,b; Varma and Babu, 2009) all the way to Mahalanobis metric learning formulations (Weinberger and Saul, 2009) for nearest neighbor classifiers.

In several of these formulations, for instance (Weinberger and Saul, 2009; Cortes et al., 2010b; Kumar et al., 2012; Bellet et al., 2012), the loss functions that are used are *pairwise loss functions* in that they evaluate the performance of a hypothesis on two instead of one domain point. As Chapter 6 shows, this presents specific challenges for, both learning algorithms as well as learning theoretic analyses.

In our work, we seek to address both of these challenges by proposing a model for online learning with pairwise loss functions. Our work, presented in Chapter 6, presents an online learning model, space efficient learning algorithms, and accompanying learning theoretic guarantees. Our learning theoretic analyses improve upon existing work on similar models and address a wider class of algorithms. We are able to instantiate our learning model into diverse learning problems such as Mercer kernel learning, indefinite kernel learning, Mahalanobis metric learning and bipartite ranking.

Chapter 3

Random Feature Maps for Dot Product Kernels

Contents

Abstract *Approximating non-linear kernels using feature maps has gained a lot of interest in recent years due to applications in reducing training and testing times of SVM classifiers and other kernel based learning algorithms. We extend this line of work and present low distortion embeddings for dot product kernels into linear Euclidean spaces. We base our results on a classical result in harmonic analysis characterizing all dot product kernels and use it to define randomized feature maps into explicit low dimensional Euclidean spaces in which the native dot product provides an approximation to the dot product kernel with high confidence.*

3.1 Introduction

Kernel methods have gained much importance in machine learning in recent years due to the ease with which they allow algorithms designed to work in linear feature spaces to be applied to implicit non linear feature spaces. Typically these non linear feature spaces are high (often infinite) dimensional and in order to avoid incurring the cost of explicitly working in these spaces, one invokes the well known *kernel trick* which exploits the fact that the algorithms in question interact with data solely through pairwise inner products. For example, instead of directly learning a hyperplane classifier in $\mathbb{R}^d$, one considers a non linear map $\Phi : \mathbb{R}^d \to \mathcal{H}$ such that for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d, \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle_{\mathcal{H}} = K(\mathbf{x}, \mathbf{y})$ for some easily computable kernel K. One then tries to learn a classifier $H : \mathbf{x} \mapsto \mathbf{w}^\top \Phi(\mathbf{x})$ for some $\mathbf{w} \in \mathcal{H}$.

However, one is faced with the problem of representation in these non linear feature spaces and is at the risk of incurring the curse of dimensionality. The solution to this problem comes in the form of *Representer Theorems* (see (Argyriou et al., 2009) for recent results) which act as an implicit dimensionality reduction step by giving us an assurance that the object(s) of interest, for example the normal vector to the hyperplane $\mathbf{w}$ in the case of classification and non-linear regression, the cluster centers in the case of kernel k-means, or the principal components in the case of kernel PCA, would necessarily lie in the span of the non-linear feature maps of the training vectors in the respective examples (see (Schölkopf and Smola, 2002)). For instance, in case of the SVM algorithm, the result ensures that the maximum margin hyperplane in $\mathcal{H}$ would necessarily be of the form

$$\mathbf{w} = \sum \alpha_i \Phi(\mathbf{x}_i)$$

where $\mathbf{x}_i$ are the training points. In case of SVM regression and classification, such a result is arrived at by application of the Karush-Kuhn-Tucker conditions whereas in the other two applications, the respective formulations themselves yield such a result.

Whereas this appears to solve the problem of the curse of dimensionality, it actually paves the way for an entirely new kind of curse – one that we call the *Curse of Support*. In order to evaluate the output of the algorithms on test data, say in the case of SVM classification, one has to compute the kernel measures of the test point with all the training points that participate in defining the normal vector $\mathbf{w}$. This cost can be prohibitive if the support is large. Unfortunately this is almost surely the case with large datasets as demonstrated by several results (Steinwart, 2003; Steinwart and Christmann, 2008a; Bengio et al., 2005) which predict an unbounded growth in the support sizes with growing training set sizes. A similar fate awaits all other kernel algorithms that use the support vector effect in order to avoid explicit representations.

This presents a dilemma where a large training set is beneficial in obtaining superior generalization properties but is simultaneously responsible in slowing down the algorithms' predictive routines.

3.2 Related Work

The problem of slower prediction times has been most widely studied for the SVM algorithm. Of late some efforts have been made at addressing other kernel based algorithms (for example, see (Chitta et al., 2012)). However the most widely studied problem from the point of view of fast prediction has undoubtedly been the SVM classification algorithm.

Research on faster SVM classifiers has progressed in mainly three directions, that can be (roughly) classified as pre-training, in-training, and post-training based techniques.

1. Post-processing based algorithms: these techniques first learn the SVM classifier and then try to compress the classifier by reducing the number of support vectors. For example, very early work by Burges and Bernhard Scholkopf (1996) suggested searching for a reduced set of support vectors such that the resultant hypothesis closely approximates the original hypothesis. The work of Cossalter et al. (2011) extends this idea by clustering the support vectors and noting that for an appreciable fraction of the test points, the cluster centers are themselves sufficient for classification. For other points, the entire set of support vectors is used for classification. These methods have been shown to offer good performance in experiments although they seldom offer strong theoretical guarantees.

2. Approximate training algorithms: these techniques start addressing the problem at the training phase itself by training on objectives that are (slightly) different from the original SVM objective. For instance the work of Joachims and Yu (2009) tries to learn the support vectors themselves (which now need not be training points) under a budget. This turns out to be a non-convex problem that is approximately solved using cutting plane methods. In another similar work Tsang et al. (2005) proposed solving the SVM objective as a minimum enclosing ball problem resulting in the Core Vector Machine formulation. There has also been work involving the use of sparsity promoting regularizers in the training process Bi et al. (2003). Yet again, these methods have been shown to offer attractive speedups in practice. However, they lack theoretical analysis.

3. Kernel approximations: these techniques try to train directly with approximate kernels that are expected to reduce prediction time. This can include approximations to kernel matrix (Achlioptas et al., 2001; Williams and Seeger, 2000), techniques that try to learn an efficiently computable kernel (Jose et al., 2013) or approximations to the kernel function itself by way of random features. It is this third line of work that we pursue in this chapter as it offers both a nice theoretical understanding behind the approach and consequently, learning theoretic guarantees, as well as demonstrate impressive speedups in practice. We will explain this approach in more detail in the following section.

3.2.1 Random Features for Accelerated Kernel Learning

As we saw in the previous discussion, there has been a lot of research on SVM formulations with accelerated prediction routines, the techniques have neither addressed other kernel algorithms nor approached the question behind the curse in a systematic way.

In a very elegant result, Rahimi and Recht (2007) demonstrated how this curse can be beaten by way of low-distortion embeddings. Their result, building upon a classical result in harmonic analysis called Bochner's Theorem (refer to (Rudin, 1962)), shows how to, in some sense, embed the non-linear feature space (i.e. $\mathcal{H}$, the Reproducing Kernel Hilbert Space associated with the kernel K) into a low dimensional Euclidean space while incurring an arbitrarily small additive distortion in the inner product values. More formally they constructed randomized feature maps $Z : \mathbb{R}^d \to \mathbb{R}^D$ such that for $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d, \langle Z(\mathbf{x}), Z(\mathbf{y}) \rangle \approx K(\mathbf{x}, \mathbf{y})$ with very high probability.

This allows one to overcome the curse of support in a systematic way for all the kernel learning tasks mentioned before since one may now work in the explicit low dimensional space $\mathbb{R}^D$ with explicit representations whose complexity depends only on the dimensionality of the space. There are several results (see for example the work of Cortes et al. (2010c)) that guarantee that using kernel values (given by these random feature construction values) that are perturbations to the original kernel value, introduce bounded perturbations in the solutions of algorithms such as SVM and kernel ridge regression. As a result one can bound the loss in accuracy one faces by using these methods. This is in stark contrast with other methods used to obtain fast prediction algorithms in that they are seldom able to offer such guarantees.

The work of Rahimi and Recht (2007) is also reminiscent of the work of Indyk and Motwani (1998) who perform low distortion embeddings (by invoking the Johnson-Lindenstrauss Lemma) in order to overcome the curse of dimensionality for the nearest neighbor problem. Subsequently there has been an increased interest in the kernel learning community toward results that allow one to use linear kernels over some transformed feature space without having to sacrifice the benefits provided by non-linear ones all the while reducing the prediction time. Rahimi and Recht considered only translation invariant kernels i.e. kernels of the form $K(\mathbf{x}, \mathbf{y}) = f(\mathbf{x} - \mathbf{y})$ for some positive definite function $f : \mathbb{R}^d \to \mathbb{R}$. Subsequently Li et al. (2010) generalized this to a larger class of group invariant kernels while still invoking Bochner's theorem.

Maji and Berg (2009) presented a similar result for the intersection kernel (also known as the min kernel) $K(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{d} \min\{\mathbf{x}_i, \mathbf{y}_i\}$ which was generalized by Vedaldi and Zisserman (2010) to the class of additive homogeneous kernels $K(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{d} k_i(\mathbf{x}_i, \mathbf{y}_i)$ where $k_i(x, y) = (xy)^{\frac{\gamma}{2}} f_i(\log x - \log y)$ for some $\gamma \in \mathbb{R}$ and positive definite functions $f_i : \mathbb{R} \to \mathbb{R}$. Vempati et al. (2010) extended this idea to provide feature maps for RBF kernels of the form $K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{1}{2\sigma^2}\chi^2(\mathbf{x}, \mathbf{y})\right)$ where χ^2 is the Chi-squared distance measure.

There have been approaches that try to perform embeddings in a task dependent

manner (see for example (Perronnin et al., 2010)). The idea of directly considering low-rank approximations to the Gram matrix has also been explored (see for example (Bach and Jordan, 2005)). However, the approaches considered by Rahimi and Recht and Vedaldi and Zisserman are the ones that most directly relate to this work.

3.2.2 Our Contribution

In this work we present feature maps approximating positive definite dot product kernels i.e kernels of the form $K(\mathbf{x}, \mathbf{y}) = f(\langle \mathbf{x}, \mathbf{y} \rangle)$ for some real valued function $f : \mathbb{R} \to \mathbb{R}$. More formally we present feature maps $Z : \mathbb{R}^d \to \mathbb{R}^D$ (where we refer to R^d as the **input space** and $\mathbb{R}^D$ as the **embedding space**) such that for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d, \langle Z(\mathbf{x}), Z(\mathbf{y}) \rangle \approx K(\mathbf{x}, \mathbf{y})$ with very high probability. We base our result on a characterization of real valued functions f that yield such positive definite kernels. We also demonstrate how our methods can be extended to compositional kernels of the form $K_{\text{co}}(\mathbf{x}, \mathbf{y}) = K_{\text{dp}}(K(\mathbf{x}, \mathbf{y}))$ where K_{dp} is some dot product kernel and K is an arbitrary positive definite kernel.

The kernels covered by our approach include homogeneous polynomial kernels which are not covered by Vedaldi and Zisserman's treatment of homogeneous kernels as these are inseparable kernels which their approach cannot handle.

In the following, vectors shall be denoted in boldface. $\mathbf{x}_i$ denotes the i^{th} Cartesian coordinate of a vector $\mathbf{x}$. $\mathcal{B}_p(\mathbf{0}, r)$ denotes the set $\left\{\mathbf{x} \in \mathcal{H} : \|\mathbf{x}\|_p \leq r\right\}$ for some inner product space $\mathcal{H}$ (or some finite dimensional Euclidean space $\mathbb{R}^d$). In particular, $\mathcal{B}_1(\mathbf{0}, 1)$ and $\mathcal{B}_2(\mathbf{0}, 1)$ denote set of points with less than unit 1-norm and 2-norm respectively. $\|\cdot\|$ without any subscripts denotes the 2-norm.

3.3 A Characterization of Positive Definite Dot Product Kernels

The result underlying our feature map constructions is a characterization of real valued functions on the real line that can be used to construct positive definite dot product kernels. This is a classical result in harmonic analysis due to Schoenberg (1942), that characterizes positive definite functions on the unit sphere in a Hilbert space. Our first observation, formalized below, is simply the fact that the restriction to the unit sphere is not crucial.

Theorem 3.1. *A function $f : \mathbb{R} \to \mathbb{R}$ defines a positive definite kernel $K : \mathcal{B}_2(\mathbf{0}, 1) \times \mathcal{B}_2(\mathbf{0}, 1) \to \mathbb{R}$ as $K : (\mathbf{x}, \mathbf{y}) \mapsto f(\langle \mathbf{x}, \mathbf{y} \rangle)$ iff f is an analytic function admitting a Maclaurin expansion with only non-negative coefficients i.e. $f(x) = \sum_{n=0}^{\infty} a_n x^n, a_n \geq 0$, $n = 0, 1, 2, \ldots$. Here $\mathcal{B}_2(\mathbf{0}, 1) \subset \mathcal{H}$ for some Hilbert space $\mathcal{H}$.*

Actually Schoenberg shows that a function f need only have a non-negative expansion in terms of Gegenbauer polynomials in order to yield a positive definite kernel over finite dimensional Euclidean spaces (a condition weaker than that of Theorem 3.1). However,

functions f that do not have non-negative Maclaurin expansions are not very useful because they yield kernels that become indefinite after the dimensionality crosses a certain threshold. This is because a dot product kernel that is positive definite over all finite dimensional Euclidean spaces is also positive definite over Hilbert spaces (see Section 3.3.1 for a simple proof of the same).

Most dot product kernels used in practice (see (Schölkopf and Smola, 2002)) satisfy the stronger condition of the Maclaurin expansion having non-negative coefficients and our results readily apply to these.

We note that, as a corollary of Schoenberg's result, all dot product kernels are necessarily unbounded over non-compact domains. This is in stark contrast with translation invariant kernels that are always bounded (see (Rudin, 1962) for a proof). Hence from now on we shall assume that our data is confined to some compact domain $\Omega \subset \mathbb{R}^d$. In order to study the behavior of our feature maps as this domain grows in size, we shall assume that $\Omega \subseteq \mathcal{B}_1(\mathbf{0}, R)$ for some $R > 0$.

We shall also assume that the function f is defined and differentiable on a closed interval $[-I, I]$. The value of I shall be dictated by the value of R chosen above. If f is defined only on an open interval $(-\gamma, \gamma)$ around zero (as is the case when the Maclaurin series has a finite radius of convergence) then we can choose a scalar $c > \frac{I}{\gamma}$, define $g = f\left(\frac{x}{c}\right)$ and use g to define a new kernel K_g. This has the implicit effect of scaling the data vectors in input space $\mathbb{R}^d$ down by a factor of c.

3.3.1 Positive definite dot product kernels over finite dimensional spaces

As noted earlier, the original result of Schoenberg characterizing functions that yield a positive definite dot product kernel over finite dimensional Euclidean spaces in terms of those admitting positive Gegenbauer expansions is not very useful in practice. This is because of two reasons. Firstly, as we shall show below, functions that have non-negative Gegenbauer expansions include those that yield positive definite kernels only up to a certain dimensionality i.e. these kernels are positive definite up to $\mathbb{R}^{d_0}$ for some fixed d_0 and indefinite on all Euclidean spaces of dimensionality $d > d_0$. Secondly, from an algorithmic perspective, the Gegenbauer expansions do not seem amenable to the type of feature construction methods described in this chapter - this is because Gegenbauer polynomials themselves admit negative coefficients.

The result characterizing positive definite functions over Hilbert spaces in terms of positive Maclaurin expansions on the other hand is appealing for the very same reasons - functions satisfying this stronger condition are positive definite over all finite dimensional spaces and the method readily lends itself to feature construction methods.

Lemma 3.2. *A function $f : \mathbb{R} \to \mathbb{R}$ yields positive definite dot product kernels over all finite dimensional Euclidean spaces iff it yields positive definite dot product kernels over Hilbert spaces.*

An easy application of Theorem 3.1 then gives us the following result :

Corollary 3.3. *A function $f : \mathbb{R} \to \mathbb{R}$ yields positive definite kernels over all finite dimensional Euclidean spaces iff it is an analytic function admitting a Maclaurin expansion with only non-negative coefficients.*

However, we note that even functions that have only positive Gegenbauer expansions (and not positive Maclaurin expansions) may admit low dimensional feature maps. This is indicated by the Johnson-Lindenstrauss Lemma (for example see (Indyk and Motwani, 1998)) that predicts the existence of low-distortion embeddings from arbitrary Hilbert spaces (thus, in particular from the reproducing kernel Hilbert spaces of these kernels) to finite dimensional Euclidean spaces. Interestingly, it is very tempting to view the constructions of Rahimi and Recht and Vedaldi and Zisserman (among others) as algorithmic versions of the Johnson-Lindenstrauss Lemma. The challenge in all such cases, however, is to make these constructions explicit, uniform, as well as algorithmically efficient.

3.3.2 Examples of Positive Definite Dot Product Kernels

The most well known dot product kernels are the polynomial kernels which are used in either a homogeneous form ($K(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \mathbf{y} \rangle^p$ for some $p \in \mathbb{N}$) or a non-homogeneous form ($K(\mathbf{x}, \mathbf{y}) = (\langle \mathbf{x}, \mathbf{y} \rangle + r)^p$ for some $p \in \mathbb{N}, r \in \mathbb{R}^+$). Lesser known examples include Vovk's real polynomial kernel ($K(\mathbf{x}, \mathbf{y}) = \frac{1-\langle \mathbf{x}, \mathbf{y} \rangle^p}{1-\langle \mathbf{x}, \mathbf{y} \rangle}$ for some $p \in \mathbb{N}$), Vovk's infinite polynomial kernel ($K(\mathbf{x}, \mathbf{y}) = \frac{1}{1-\langle \mathbf{x}, \mathbf{y} \rangle}$) and the exponential dot product kernel ($K(\mathbf{x}, \mathbf{y}) = \exp\left(\frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\sigma^2}\right)$ for some $\sigma \in \mathbb{R}$).

It is interesting to note that due to a result by Steinwart (2001), the last two kernels (Vovk's infinite kernel and exponential dot product kernel) are universal on any compact subset $S \subset \mathbb{R}^d$ which means that the space of all functions induced by them is dense in $C(S)$, the space of all continuous functions defined on S. The widely used Gaussian kernel is actually a normalized version of the exponential dot product kernel. However Vovk's kernels are seldom used in practice since they are expected to have poor generalization properties due to their flat spectrum as noted by Schölkopf and Smola (2002).

3.4 Random Feature Maps

Schoenberg's result naturally paves the way for a result of the kind presented by Rahimi and Recht in which we can take the Maclaurin's expansion $f(x) = \sum_{n=0}^{\infty} a_n x^n$, view the coefficients a_n as a positive measure defined on $\mathbb{N} \cup \{0\}$ and define estimators for each individual term of the expression. However, as we shall see, estimating higher order terms in our case will require more randomness. Thus, a set of coefficients $\{a_n\}$ defining a heavy tailed distribution would entail huge randomness costs in case the expansion has a large (or infinite) number of terms. For example the sequence $a_n = \frac{1}{n^2}$ has a linear rather than an exponential tail.

To address this issue we do not utilize the coefficients as measure values, rather we impose an external distribution on $\mathbb{N} \cup \{0\}$ having an exponential tail. The distribution

Algorithm 1 Random Maclaurin Feature Maps

Input: A positive definite dot product kernel $K(\mathbf{x}, \mathbf{y}) = f(\langle \mathbf{x}, \mathbf{y} \rangle)$.
Output: A randomized feature map $\mathbf{Z} : \mathbb{R}^d \to \mathbb{R}^D$ such that $\langle \mathbf{Z}(\mathbf{x}), \mathbf{Z}(\mathbf{y}) \rangle \approx K(\mathbf{x}, \mathbf{y})$.

Obtain the Maclaurin expansion of $f(x) = \sum_{n=0}^{\infty} a_n x^n$ by setting $a_n = \frac{f^{(n)}(0)}{n!}$.

Fix a value $p > 1$.

for $i = 1$ **to** D **do**

Choose a non negative integer $N \in \mathbb{N} \cup \{0\}$ with $\mathbb{P}[N = n] = \frac{1}{p^{n+1}}$.

Choose N vectors $\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_N \in \{-1, 1\}^d$ selecting each coordinate using fair coin tosses.

Let feature map $Z_i : \mathbf{x} \mapsto \sqrt{a_N p^{N+1}} \prod_{j=1}^{N} \boldsymbol{\omega}_j^\top \mathbf{x}$.

end for

Output $\mathbf{Z} : \mathbf{x} \mapsto \frac{1}{\sqrt{D}} (Z_1(\mathbf{x}), \ldots, Z_D(\mathbf{x}))$.

that we choose to impose is $\mathbb{P}[N = n] = \frac{1}{p^{n+1}}$ for some fixed $p > 1$. In practice $p = 2$ is a good choice since it establishes a normalized measure over $\mathbb{N} \cup \{0\}$. We will, using this distribution, obtain unbiased estimates for the kernel value and prove corresponding uniform convergence results.

We stress that the positiveness of the coefficients $\{a_n\}$ is still essential for us to be able to provide an embedding into real spaces. If the coefficients are allowed to be negative, the resulting kernels would no longer remain positive definite and we would only be able to provide feature maps that map to pseudo-Euclidean spaces. It turns out that the imposition of an external measure is crucial from a statistical point of view as well. As we shall see later, it allows us to obtain bounded estimators which in turn allow us to use Hoeffding bounds to prove uniform convergence results.

We now move on to describe our feature map : our feature map will essentially be a concatenation of several copies of identical real valued feature maps. These copies will reduce variance and allow us to prove convergence bounds. The following simple fact about random projections is at the core of our feature maps.

Lemma 3.4. *Let $\boldsymbol{\omega} \in \mathbb{R}^d$ be a vector each of whose coordinates have been chosen pairwise independently using fair coin tosses from the set $\{-1, 1\}$ and consider the feature map $Z : \mathbb{R}^d \to \mathbb{R}$, $Z : \mathbf{x} \mapsto \boldsymbol{\omega}^\top \mathbf{x}$. Then for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, $\mathop{\mathbb{E}}_{\boldsymbol{\omega}} [\![Z(\mathbf{x}) Z(\mathbf{y})]\!] = \langle \mathbf{x}, \mathbf{y} \rangle$.*

For a given dot product kernel $K(\mathbf{x}, \mathbf{y}) = f(\langle \mathbf{x}, \mathbf{y} \rangle)$ where $f(x) = \sum_{n=0}^{\infty} a_n x^n$, we construct a single real-valued feature map as follows: first, we randomly pick a number $N \in \mathbb{N} \cup \{0\}$ with $\mathbb{P}[N = n] = \frac{1}{p^{n+1}}$. Next we pick N independent Rademacher vectors $\boldsymbol{\omega}_1 \ldots \boldsymbol{\omega}_N$ and output the feature map $Z : \mathbb{R}^d \to \mathbb{R}$, $Z : \mathbf{x} \mapsto \sqrt{a_N p^{N+1}} \prod_{j=1}^{N} \boldsymbol{\omega}_j^\top \mathbf{x}$.

Below, we establish that the linear kernel obtained by using this feature map gives us an unbiased estimate of the kernel value at each pair of points chosen from the domain Ω.

Lemma 3.5. *Let $Z : \mathbb{R}^d \to \mathbb{R}$ be the feature map constructed above. Then for all $\mathbf{x}, \mathbf{y} \in \Omega$, we have $\mathbb{E}[\![Z(\mathbf{x})Z(\mathbf{y})]\!] = K(\mathbf{x}, \mathbf{y})$ where the expectation is over the choice of the Rademacher vectors.*

Having obtained a feature map giving us an unbiased estimate of the kernel value, we move on to establish bounds on the deviation of the linear kernel given by this map from its expected value. To do this we obtain D such feature maps independently and concatenate them to obtain a multi dimensional feature map $\mathbf{Z} : \mathbb{R}^d \to \mathbb{R}^D, \mathbf{Z} : \mathbf{x} \mapsto \frac{1}{\sqrt{D}} (Z_1(\mathbf{x}), \ldots, Z_D(\mathbf{x}))$. It is easy to see that $\mathbb{E}[\![\langle \mathbf{Z}(\mathbf{x}), \mathbf{Z}(\mathbf{y}) \rangle]\!] = K(\mathbf{x}, \mathbf{y})$. Moreover, such a concatenation is expected to guarantee an exponentially fast convergence to $K(\mathbf{x}, \mathbf{y})$ using Hoeffding bounds. However this requires us to prove that the estimator corresponding to our feature map i.e $Z(\mathbf{x})Z(\mathbf{y})$ is bounded. This we establish below :

Lemma 3.6. *For all $\mathbf{x}, \mathbf{y} \in \Omega, |Z(\mathbf{x})Z(\mathbf{y})| \leq pf(pR^2)$.*

Proof. Since $Z(\mathbf{x})Z(\mathbf{y}) = a_N p^{N+1} \prod_{j=1}^{N} \boldsymbol{\omega}_j^\top \mathbf{x} \prod_{j=1}^{N} \boldsymbol{\omega}_j^\top \mathbf{y}$, by Hölder's inequality we have, for all $j, \left|\boldsymbol{\omega}_j^\top \mathbf{x}\right| \leq \|\boldsymbol{\omega}_j\|_\infty \|\mathbf{x}\|_1 \leq R$ since every coordinate of $\boldsymbol{\omega}_j$ is either 1 or -1 and $\mathbf{x} \in \Omega \subseteq \mathcal{B}_1(\mathbf{0}, R)$. A similar result holds for $\left|\boldsymbol{\omega}_j^\top \mathbf{y}\right|$ as well. Thus we have $|Z(\mathbf{x})Z(\mathbf{y})| \leq a_N p^{N+1} R^{2N} \leq p \cdot \sum_{n=0}^{\infty} a_n p^n R^{2n} = pf(pR^2)$. □

We note here that the imposition of an external measure on $\mathbb{N} \cup \{0\}$ plays a crucial role in the analysis. In absence of the external measure, one is only able to bound the estimator by $\mathcal{O}\left(R^{2N}\right)$ and since N is a potentially unbounded random variable, this makes application of Hoeffding bounds impossible. Although there do exist Hoeffding style bounds for unbounded random variables, they dont seem to work in our case. However, with the simple imposition of an external measure we obtain an estimator that is bounded by a value dependent on the range of values taken by the kernel over the domain, a very desirable quality.

For sake of convenience let us denote $pf(pR^2)$ by C_Ω since it is a constant dependent only on the size of the domain Ω and independent of the dimension of the input space $\mathbb{R}^d$. Note that this constant is proportional to the largest value taken by the kernel in the domain Ω. This immediately tells us that for any $\mathbf{x}, \mathbf{y} \in \Omega, \mathbb{P}\left[|\langle \mathbf{Z}(\mathbf{x}), \mathbf{Z}(\mathbf{y}) \rangle - K(\mathbf{x}, \mathbf{y})| > \varepsilon\right] \leq 2 \exp\left(-\frac{D\varepsilon^2}{8C_\Omega^2}\right)$. However we can give much stronger guarantees than this – we can prove that this loss of confidence need not be incurred over every single pair of points but rather the entire domain at once. More formally, we can show that with very high probability, $\sup_{\mathbf{x},\mathbf{y} \in \Omega} |\langle \mathbf{Z}(\mathbf{x}), \mathbf{Z}(\mathbf{y}) \rangle - K(\mathbf{x}, \mathbf{y})| \leq \varepsilon$.

3.4.1 Uniform Approximation

As stated before, we are able to ensure that the feature map designed above gives an accurate estimate of the kernel value uniformly over the entire domain. For this we exploit the Lipschitz properties of the kernel function and our estimator. A similar approach was

adopted by Rahimi and Recht to provide corresponding uniform convergence properties for their estimator. However it is not possible to import their argument since they were able to exploit the fact that both their kernel as well as their estimator were translation invariant. We, having no such guarantees for our estimator, have to argue differently.

Let $\mathcal{E}(\mathbf{x},\mathbf{y}) = \langle \mathbf{Z}(\mathbf{x}), \mathbf{Z}(\mathbf{y})\rangle - K(\mathbf{x},\mathbf{y})$. We will first show that the function $\mathcal{E}(\cdot,\cdot)$ is Lipschitz over the domain Ω. Since $\mathcal{E}(\cdot,\cdot)$ itself is differentiable (actually analytic), its Lipschitz constant can be bounded by bounding the norms of its gradients i.e. it would suffice to show that $\sup_{\mathbf{x},\mathbf{y}\in\Omega} \|\nabla_{\mathbf{x}}\mathcal{E}(\mathbf{x},\mathbf{y})\| \leq L$ and $\sup_{\mathbf{x},\mathbf{y}\in\Omega} \|\nabla_{\mathbf{y}}\mathcal{E}(\mathbf{x},\mathbf{y})\| \leq L$ for some constant L. This would ensure that if the error incurred by the feature map is small on a pair of vectors then it would also be small on all pairs of vectors that are "close" to these vectors. This is formalized in the following theorem :

Lemma 3.7. *If a bivariate function f defined over $\Omega \subseteq \mathbb{R}^d$ is L-Lipschitz in both its arguments then for every $\mathbf{x},\mathbf{y} \in \Omega$, $\sup_{\substack{\mathbf{x}'\in\mathcal{B}_2(\mathbf{x},r)\cap\Omega \\ \mathbf{y}'\in\mathcal{B}_2(\mathbf{y},r)\cap\Omega}} |f(\mathbf{x},\mathbf{y}) - f(\mathbf{x}',\mathbf{y}')| \leq 2Lr$.*

Proof. We have $|f(\mathbf{x},\mathbf{y}) - f(\mathbf{x}',\mathbf{y}')| \leq |f(\mathbf{x},\mathbf{y}) - f(\mathbf{x},\mathbf{y}')| + |f(\mathbf{x},\mathbf{y}') - f(\mathbf{x}',\mathbf{y}')| \leq L \cdot \|\mathbf{y}-\mathbf{y}'\| + L \cdot \|\mathbf{x}-\mathbf{x}'\| \leq 2Lr$ where in the second step we have used the fact that both $\mathbf{x},\mathbf{y}' \in \Omega$. □

What this allows us to do is choose a set of points $\mathcal{T}$ that set up an ε-net over the domain Ω at some scale ε_1. If we can ensure that the feature maps provide an $(\varepsilon/2)$-close approximation to K at the centers of this net i.e. $\sup_{\mathbf{x},\mathbf{y}\in\mathcal{T}} |\mathcal{E}(\mathbf{x},\mathbf{y})| \leq \varepsilon/2$, then the above result would show us that if the error function $\mathcal{E}(\cdot,\cdot)$ is L-Lipschitz in both its arguments, then $\sup_{\mathbf{x},\mathbf{y}\in\Omega} |\mathcal{E}(\mathbf{x},\mathbf{y})| \leq \varepsilon/2 + 2L\varepsilon_1$ since the ε-net ensures that for all $\mathbf{x},\mathbf{y} \in \Omega$, there exists $\mathbf{x}',\mathbf{y}' \in \mathcal{T}$ such that $\|\mathbf{x}-\mathbf{x}'\|, \|\mathbf{y}-\mathbf{y}'\| \leq \varepsilon_1$. Thus choosing $\varepsilon_1 = \frac{\varepsilon}{4L}$ ensures that $\sup_{\mathbf{x},\mathbf{y}\in\Omega} |\langle \mathbf{Z}(\mathbf{x}), \mathbf{Z}(\mathbf{y})\rangle - K(\mathbf{x},\mathbf{y})| \leq \varepsilon$.

Now ensuring that the feature maps provide a close approximation to the kernel value at all pairs of points taken from $\mathcal{T}$ would cost us a reduction in the confidence parameter by a factor of $|\mathcal{T}|^2$ due to taking a union bound. It is well known (for example see (Cucker and Smale, 2001)) that setting up an ε-net at scale ε_1 in d dimensions over a compact set of diameter Δ takes at most $\left(\frac{4\Delta}{\varepsilon_1}\right)^d$ centers. In our case $\Delta \leq 2R$ since $\Omega \subseteq \mathcal{B}_1(\mathbf{0},R) \subset \mathcal{B}_2(\mathbf{0},R)$ and $\varepsilon_1 = \frac{\varepsilon}{4L}$ i.e. $|\mathcal{T}| \leq \left(\frac{32RL}{\varepsilon}\right)^d$.

We now move on to the task of bounding the Lipschitz constant of the error function. Since $\mathcal{E}(\cdot,\cdot)$ is symmetric in both its arguments, it is sufficient to bound

$$\|\nabla_{\mathbf{x}}\mathcal{E}(\mathbf{x},\mathbf{y})\| \leq \|\nabla_{\mathbf{x}} \langle \mathbf{Z}(\mathbf{x}), \mathbf{Z}(\mathbf{y})\rangle\| + \|\nabla_{\mathbf{x}} K(\mathbf{x},\mathbf{y})\|.$$

We will bound these two quantities separately below.

Lemma 3.8. *We have the following :*

$$\begin{aligned}\sup_{\mathbf{x},\mathbf{y}\in\Omega}\|\nabla_{\mathbf{x}}K(\mathbf{x},\mathbf{y})\| &\leq Rf'(R^2)\\ \sup_{\mathbf{x},\mathbf{y}\in\Omega}\|\nabla_{\mathbf{y}}K(\mathbf{x},\mathbf{y})\| &\leq Rf'(R^2)\end{aligned}$$

Proof. We have, by the definition of K,

$$\nabla_{\mathbf{x}}K(\mathbf{x},\mathbf{y}) = \nabla_{\mathbf{x}}\left(\sum_{n=0}^{\infty}a_n\langle\mathbf{x},\mathbf{y}\rangle^n\right) = \sum_{n=0}^{\infty}a_n\nabla_{\mathbf{x}}\langle\mathbf{x},\mathbf{y}\rangle^n = \mathbf{y}\sum_{n=0}^{\infty}na_n\langle\mathbf{x},\mathbf{y}\rangle^{n-1}.$$

This gives us

$$\|\nabla_{\mathbf{x}}K(\mathbf{x},\mathbf{y})\| = \left\|\mathbf{y}\sum_{n=0}^{\infty}na_n\langle\mathbf{x},\mathbf{y}\rangle^{n-1}\right\| \leq R\sum_{n=0}^{\infty}na_n|\langle\mathbf{x},\mathbf{y}\rangle|^{n-1} \leq R\sum_{n=0}^{\infty}na_n(R^2)^{n-1} = Rf'(R^2)$$

where in the second and the third step we have used the fact that $\mathbf{x},\mathbf{y}\in\Omega\subseteq\mathcal{B}_1(\mathbf{0},R)\subset\mathcal{B}_2(\mathbf{0},R)$. Similarly we can show $\sup_{\mathbf{x},\mathbf{y}\in\Omega}\|\nabla_{\mathbf{y}}K(\mathbf{x},\mathbf{y})\| \leq Rf'(R^2)$. □

Lemma 3.9. *We have the following :*

$$\begin{aligned}\sup_{\mathbf{x},\mathbf{y}\in\Omega}\|\nabla_{\mathbf{x}}\left(Z_1(\mathbf{x})Z_1(\mathbf{y})\right)\| &\leq p^2R\sqrt{d}f'(pR^2)\\ \sup_{\mathbf{x},\mathbf{y}\in\Omega}\|\nabla_{\mathbf{y}}\left(Z_1(\mathbf{x})Z_1(\mathbf{y})\right)\| &\leq p^2R\sqrt{d}f'(pR^2)\end{aligned}$$

Thus we have $L = \sup_{\mathbf{x},\mathbf{y}\in\Omega}\|\nabla_{\mathbf{x}}\mathcal{E}(\mathbf{x},\mathbf{y})\| \leq Rf'(R^2) + p^2R\sqrt{d}f'(pR^2)$. Putting all the results together, we first have by application of union bound that the probability that the feature map will fail at any pair of points chosen from the ε-net is bounded by $2\left(\frac{32RL}{\varepsilon}\right)^{2d}\exp\left(-\frac{D\varepsilon^2}{8C_\Omega^2}\right)$. The covering argument along with the bound on the Lipschitz constant of the error function ensure that with the same confidence, the feature map would provide an ε-accurate estimate on the entire domain Ω. Thus we have the following theorem.

Theorem 3.10. *Let $\Omega\subseteq\mathcal{B}_1(\mathbf{0},R)$ be a compact subset of $\mathbb{R}^d$ and $K(\mathbf{x},\mathbf{y}) = f(\langle\mathbf{x},\mathbf{y}\rangle)$ be a dot product kernel defined on Ω. Then, for the feature map $\mathbf{Z}$ defined in Algorithm 1, we have $\mathbb{P}\left[\sup_{\mathbf{x},\mathbf{y}\in\Omega}|\langle\mathbf{Z}(\mathbf{x}),\mathbf{Z}(\mathbf{y})\rangle - K(\mathbf{x},\mathbf{y})| > \varepsilon\right] \leq 2\left(\frac{32RL}{\varepsilon}\right)^{2d}\exp\left(-\frac{D\varepsilon^2}{8C_\Omega^2}\right)$ where $C_\Omega = pf(pR^2)$ and $L = Rf'(R^2) + p^2R\sqrt{d}f'(pR^2)$ for some small constant $p > 1$. Moreover, with $D = \Omega\left(\frac{dC_\Omega^2}{\varepsilon^2}\log\left(\frac{RL}{\varepsilon\delta}\right)\right)$, one can ensure the same with probability greater than $1-\delta$.*

The behavior of this bound with respect to the dimensionality of the input space, the accuracy parameter and the confidence parameter is of the form $D = \Omega\left(\frac{d}{\varepsilon^2}\log\left(\frac{1}{\varepsilon\delta}\right)\right)$ that matches that of Rahimi and Recht. The bound has a stronger dependence on kernel

specific parameters which appear as non-logarithmic terms due to the unbounded nature of the dot product kernels. Even so, the kernel specific term C_Ω is dependent on the largest value taken by the kernel in the domain Ω, a dependence that is unavoidable for an algorithm giving guarantees on the absolute (rather than relative) deviation from the true value.

3.4.2 An Alternative Feature Map

An alternative method to bounding the amount of randomness being used is to truncate the Maclaurin series after a certain number of terms and use the resulting function to define a new kernel. Since the Maclaurin series of an analytic function defined over a bounded domain converges to it uniformly, we can truncate the series while incurring a uniformly bounded error. A similar approach is used by Vedaldi and Zisserman to present deterministic feature maps. Suppose we have a positive definite dot product kernel K defined on a domain $\Omega \subset \mathcal{B}_1(\mathbf{0}, R)$ in some Euclidean space $\mathbb{R}^d$ by a function $f(x) = \sum_{n=0}^{\infty} a_n x^n$. If we choose $k = k(\varepsilon, R)$ such that $\sum_{n=0}^{k} a_n R^{2n} = f(R^2) - \varepsilon$ (or select some set $S \subset \mathbb{N} \cup \{0\}$ such that $\sum_{n \in S} a_n R^{2n} = f(R^2) - \varepsilon$ and $|S| = k$) and create a new kernel $\tilde{K}(\mathbf{x}, \mathbf{y}) = \sum_{n=0}^{k} a_n \langle \mathbf{x}, \mathbf{y} \rangle^n$, then the residual error

$$R_k = \sup_{\mathbf{x},\mathbf{y} \in \Omega} \left| \tilde{K}(\mathbf{x}, \mathbf{y}) - K(\mathbf{x}, \mathbf{y}) \right| = \sup_{\mathbf{x},\mathbf{y} \in \Omega} \left| \sum_{i=k+1}^{\infty} a_n \langle \mathbf{x}, \mathbf{y} \rangle^n \right| \leq \sum_{i=k+1}^{\infty} a_n R^{2n} \leq \varepsilon$$

since $\Omega \subset \mathcal{B}_1(\mathbf{0}, R) \subset \mathcal{B}_2(\mathbf{0}, R)$ and $\sum_{n=0}^{\infty} a_n R^{2n} = f(R^2)$. Thus for all $\mathbf{x}, \mathbf{y} \in \Omega$, we have $K(\mathbf{x}, \mathbf{y}) - \varepsilon \leq \tilde{K}(\mathbf{x}, \mathbf{y}) \leq K(\mathbf{x}, \mathbf{y}) + \varepsilon$. Since $\tilde{K}$ also satisfies the conditions of Theorem 3.1, one can now obtain ε_1-accurate feature maps for $\tilde{K}$ using the techniques mentioned above and those feature maps would provide an $(\varepsilon + \varepsilon_1)$-accurate estimate to K.

3.5 Generalizing to Compositional Kernels

Given a positive definite dot product kernel K_{dp} and an arbitrary positive definite kernel K, the kernel K_{co} defined as $K_{\text{co}}(\mathbf{x}, \mathbf{y}) = K_{\text{dp}}(K(\mathbf{x}, \mathbf{y}))$ is also positive definite. This fact can be deduced either by directly invoking a result due to (FitzGerald et al., 1995, Theorem 2.1) or by applying Schoenberg's result in conjunction with Mercer's theorem. We now show how to extend the result for dot product kernels to such compositional kernels.

Note that plugging a translation invariant kernel into a dot product kernel yields yet another translation invariant kernel since the set of translation invariant kernels is closed under powering, scalar multiplication and addition. However, a set of homogeneous kernels not sharing the homogeneity parameter is not closed under addition. Hence the

set of homogeneous kernels is not closed under the operations mentioned above and thus, plugging a homogeneous kernel into a dot product kernel in general yields a novel non-homogeneous kernel. We also note that the results obtained in the section above can be now viewed as special cases of the result presented in this section with the dot product being substituted into a dot product kernel.

In order to construct feature maps for the compositional kernel we assume that we have black-box access to a (possibly randomized) feature map selection routine $\mathcal{A}$ which when invoked, returns a feature map $W : \mathbb{R}^d \to \mathbb{R}$ for K. If we assume that the kernel K is bounded and Lipschitz and that the feature map W returned to us is bounded, Lipschitz on expectation and provides an unbiased estimate of K, then one can design (using these feature maps for K) feature maps for K_{co}. The analysis of the final feature map in this case is a bit more involved since we only assume black-box access to $\mathcal{A}$ and only expect the feature map to be Lipschitz on expectation.

We first state our assumptions about the kernel K and the feature maps given by $\mathcal{A}$:

1. K is defined over some domain $\Omega \subset \mathbb{R}^d$.
2. K is bounded i.e. we have $\sup_{\mathbf{x},\mathbf{y}\in\Omega} |K(\mathbf{x},\mathbf{y})| \leq C_K$ for some $C_K \in \mathbb{R}^+$.
3. K is Lipschitz i.e. $\sup_{\mathbf{x},\mathbf{y}\in\Omega} \|\nabla_{\mathbf{x}} K(\mathbf{x},\mathbf{y})\| \leq L_K$ and $\sup_{\mathbf{x},\mathbf{y}\in\Omega} \|\nabla_{\mathbf{y}} K(\mathbf{x},\mathbf{y})\| \leq L_K$, $L_K > 0$.
4. W is an unbiased estimator of K i.e. for all $\mathbf{x},\mathbf{y} \in \Omega$, $\mathbb{E}[\![W(\mathbf{x})W(\mathbf{y})]\!] = K(\mathbf{x},\mathbf{y})$ where the expectation is over the internal randomness of W.
5. W is a bounded feature map i.e. for some $C_W > 0$, $\sup_{\mathbf{x}\in\Omega} |W(\mathbf{x})| \leq \sqrt{C_W}$.
6. W is Lipschitz on expectation i.e. for some $L_W \in \mathbb{R}^+$, $\sup_{\mathbf{x}\in\Omega} \mathbb{E}[\![\|\nabla_{\mathbf{x}} W(\mathbf{x})\|]\!] \leq L_W$.

Our feature map construction algorithm is similar to the one used for dot product kernels. We pick a non-negative integer $N \in \mathbb{N} \cup \{0\}$ with $\mathbb{P}[N = n] = \frac{1}{p^{n+1}}$ for some fixed $p > 1$ and output the feature map $Z : \mathbb{R}^d \to \mathbb{R}$, $Z : \mathbf{x} \mapsto \sqrt{a_N p^{N+1}} \prod_{j=1}^{N} W_j(\mathbf{x})$ where $W_1, \ldots, W_N$ are independent instantiations of the feature map W associated with the kernel K. We concatenate D such feature maps to give our final feature map.

It is clear that on expectation, the product of the feature map values is equal to the value of the kernel i.e. $\mathop{\mathbb{E}}_{N,W_1,\ldots,W_N}[\![\langle \mathbf{Z}(\mathbf{x}), \mathbf{Z}(\mathbf{y})\rangle]\!] = K_{co}(\mathbf{x},\mathbf{y})$ where $\mathbf{Z} : \mathbb{R}^d \to \mathbb{R}^D$, $\mathbf{Z} : \mathbf{x} \mapsto \frac{1}{\sqrt{D}} (Z_1(\mathbf{x}), \ldots, Z_D(\mathbf{x}))$. Yet again we expect that the concatenation of D such feature maps for a large enough D would provide us a close approximation to K_{co} with high probability. For this we first prove that our feature map is bounded.

Lemma 3.11. *For all* $\mathbf{x},\mathbf{y} \in \Omega, |Z(\mathbf{x})Z(\mathbf{y})| \leq pf(pC_W)$.

Proof. $Z(\mathbf{x})Z(\mathbf{y}) = a_N p^{N+1} \prod_{j=1}^{N} W_j(\mathbf{x}) \prod_{j=1}^{N} W_j(\mathbf{x})$. Using the bound on the feature maps we get the inequality $|Z(\mathbf{x})Z(\mathbf{y})| \leq a_N p^{N+1} C_W^N \leq pf(pC_W)$ □

Algorithm 2 Random Maclaurin Feature Maps for Compositional Kernels

Input: A compositional positive definite kernel $K_{\text{co}}(\mathbf{x},\mathbf{y}) = K_{\text{dp}}(K(\mathbf{x},\mathbf{y})) = f(K(\mathbf{x},\mathbf{y}))$.
Output: A randomized feature map $\mathbf{Z} : \mathbb{R}^d \to \mathbb{R}^D$ such that $\langle \mathbf{Z}(\mathbf{x}), \mathbf{Z}(\mathbf{y}) \rangle \approx K_{\text{co}}(\mathbf{x},\mathbf{y})$.

Obtain the Maclaurin expansion of $f(x) = \sum_{n=0}^{\infty} a_n x^n$ by setting $a_n = \frac{f^{(n)}(0)}{n!}$.
Fix a value $p > 1$.
for $i = 1$ **to** D **do**
  Choose a non negative integer $N \in \mathbb{N} \cup \{0\}$ with $\mathbb{P}[N = n] = \frac{1}{p^{n+1}}$.
  Get N independent instantiations of the feature map for K from $\mathcal{A}$ as $W_1, \ldots, W_N$.
  Let feature map $Z_i : \mathbf{x} \mapsto \sqrt{a_N p^{N+1}} \prod_{j=1}^{N} W_j(\mathbf{x})$.
end for
Output $\mathbf{Z} : \mathbf{x} \mapsto \frac{1}{\sqrt{D}} (Z_1(\mathbf{x}), \ldots, Z_D(\mathbf{x}))$.

Thus we have for any $\mathbf{x}, \mathbf{y} \in \Omega$, $\mathbb{P}\left[|\langle \mathbf{Z}(\mathbf{x}), \mathbf{Z}(\mathbf{y}) \rangle - K_{\text{co}}(\mathbf{x},\mathbf{y})| \leq \varepsilon\right]$ with probability at least $1 - 2\exp\left(-\frac{D\varepsilon^2}{8C_1^2}\right)$ where $C_1 = pf(pC_W)$. We now investigate the Lipschitz properties of K_{co} and our feature map.

Lemma 3.12. *We have*

$$\begin{aligned} \sup_{\mathbf{x},\mathbf{y}\in\Omega} \|\nabla_{\mathbf{x}} K_{co}(\mathbf{x},\mathbf{y})\| &\leq L_K f'(C_K) \\ \sup_{\mathbf{x},\mathbf{y}\in\Omega} \|\nabla_{\mathbf{y}} K_{co}(\mathbf{x},\mathbf{y})\| &\leq L_K f'(C_K) \end{aligned}$$

Proof. $K_{\text{comp}}(\mathbf{x},\mathbf{y}) = \sum_{n=0}^{\infty} a_n K(\mathbf{x},\mathbf{y})^n$. Thus we have by linearity

$$\nabla_{\mathbf{x}} K_{\text{comp}}(\mathbf{x},\mathbf{y}) = \sum_{n=0}^{\infty} a_n \nabla_{\mathbf{x}} \left(K(\mathbf{x},\mathbf{y})^n\right) = \sum_{n=0}^{\infty} n a_n K(\mathbf{x},\mathbf{y})^{n-1} \nabla_{\mathbf{x}} K(\mathbf{x},\mathbf{y})$$

which in turn gives us

$$\|\nabla_{\mathbf{x}} K_{\text{comp}}(\mathbf{x},\mathbf{y})\| \leq \|\nabla_{\mathbf{x}} K(\mathbf{x},\mathbf{y})\| \sum_{n=0}^{\infty} n a_n C_K^{n-1} \leq L_K f'(C_K).$$

Similarly we have $\sup_{\mathbf{x},\mathbf{y}\in\Omega} \|\nabla_{\mathbf{y}} K_{\text{co}}(\mathbf{x},\mathbf{y})\| \leq L_K f'(C_K)$. □

We next move on to the Lipschitz properties of $\mathbf{Z}$. Since we have only made assumptions on the expected Lipschitz properties of W, we would only be able to give guarantees on the expected Lipschitz properties of $\mathbf{Z}$. However, as we shall see, these would be sufficient to provide a uniform convergence guarantee over the entire domain Ω. As before, by linearity of expectation, analyzing the expected Lipschitz properties of a single feature map Z are sufficient to guarantee, on expectation, similar properties for $\mathbf{Z}$ as well.

Lemma 3.13. *We have*

$$\begin{aligned}\sup_{\mathbf{x},\mathbf{y}\in\Omega}\|\nabla_{\mathbf{x}}\left(Z(\mathbf{x})Z(\mathbf{y})\right)\| &\leq L_W p^2\sqrt{C_W}f'(pC_W)\\ \sup_{\mathbf{x},\mathbf{y}\in\Omega}\|\nabla_{\mathbf{y}}\left(Z(\mathbf{x})Z(\mathbf{y})\right)\| &\leq L_W p^2\sqrt{C_W}f'(pC_W)\end{aligned}$$

Proof. Since $Z(\mathbf{x})Z(\mathbf{y}) = a_N p^{N+1}\prod_{j=1}^{N} W_j(\mathbf{x})W_j(\mathbf{y})$, by linearity we can write

$$\nabla_{\mathbf{x}}Z(\mathbf{x})Z(\mathbf{y}) = \left(a_N p^{N+1}\prod_{j=1}^{N}W_j(\mathbf{y})\right)\sum_{j=1}^{N}\left(\prod_{i\neq j}W_i(\mathbf{x})\right)\nabla_{\mathbf{x}}W_j(\mathbf{x}).$$

Thus we can then write

$$\begin{aligned}\|\nabla_{\mathbf{x}}Z(\mathbf{x})Z(\mathbf{y})\| &= a_N p^{N+1}\left|\prod_{j=1}^{N}W_j(\mathbf{y})\right|\left\|\sum_{j=1}^{N}\left(\prod_{i\neq j}W_i(\mathbf{x})\right)\nabla_{\mathbf{x}}W_j(\mathbf{x})\right\|\\ &\leq a_N p^{N+1}C_W^{\frac{N}{2}}\sum_{j=1}^{N}C_W^{\frac{N-1}{2}}\left\|\nabla_{\mathbf{x}}W_j(\mathbf{x})\right\|,\end{aligned}$$

which gives us, by linearity of expectation and the bound on the expected Lipschitz properties of the individual estimators,

$$\begin{aligned}\mathbb{E}\left[\!\left[\|\nabla_{\mathbf{x}}Z(\mathbf{x})Z(\mathbf{y})\|\right]\!\right] &\leq Na_N p^{N+1}C_W^{N-\frac{1}{2}}L_W = L_W p^2\sqrt{C_W}\cdot Na_N\left(pC_W\right)^{N-1}\\ &\leq L_W p^2\sqrt{C_W}f'(pC_W)\end{aligned}$$

Similarly we have $\sup_{\mathbf{x},\mathbf{y}\in\Omega}\|\nabla_{\mathbf{y}}\left(Z(\mathbf{x})Z(\mathbf{y})\right)\| \leq L_W p^2\sqrt{C_W}f'(pC_W)$. □

Working as before we find that the error function $\mathcal{E}(\mathbf{x},\mathbf{y}) = \langle\mathbf{Z}(\mathbf{x}),\mathbf{Z}(\mathbf{y})\rangle - K_{\text{co}}(\mathbf{x},\mathbf{y})$ is, on expectation, L_1-Lipschitz for $L_1 = L_K f'(C_K) + L_W p^2\sqrt{C_W}f'(pC_W)$. Hence the probability that the error function will not be $\frac{\varepsilon}{2r}$-Lipschitz is less than $\frac{2L_1 r}{\varepsilon}$ by an application of Markov's inequality. However if this is not the case then constructing an ε-net at scale r over the domain Ω and ensuring that the estimator provides an $\varepsilon/2$-approximation at centers of these points would ensure an ε-accurate estimation to the kernel on the entire domain Ω. Setting up such a net would require at most $\left(\frac{4R}{r}\right)^d$ centers if $\Omega \subseteq \mathcal{B}_1(\mathbf{0},R)$. Adding the failure probabilities of the estimator not being accurate on the ε-net centers to the probability of the error function not being Lipschitz gives us the total error probability of our estimator giving an inaccurate estimate over any point in the domain as $2\left(\frac{4R}{r}\right)^d\exp\left(-\frac{D\varepsilon^2}{8C_1^2}\right) + \frac{2L_1 r}{\varepsilon}$.

Comparing this with the form $k_1 r^{-d} + k_2 r$ and setting $r = \left(\frac{k_1}{k_2}\right)^{\frac{1}{d+1}}$ gives us the error probability as $2k_1^{\frac{1}{d+1}}k_2^{\frac{d}{d+1}} \leq \left(\frac{32RL_1}{\varepsilon}\right)\exp\left(-\frac{D\varepsilon^2}{8C_1^2 d}\right)$ if $\varepsilon < 8RL_1$. This gives us the following:

Theorem 3.14. *Let $\Omega \subseteq \mathcal{B}_1(\mathbf{0}, R)$ be a compact subset of $\mathbb{R}^d$ and $K_{co}(\mathbf{x}, \mathbf{y}) = K_{dp}(K(\mathbf{x}, \mathbf{y}))$ be a compositional kernel defined on Ω satisfying the necessary boundedness and Lipschitz conditions. Assuming we have black-box access to a feature map selection algorithm for K also satisfying the necessary boundedness and Lipschitz conditions, for the feature map $\mathbf{Z}$ defined in Algorithm 2, we have $\mathbb{P}\left[\sup_{\mathbf{x},\mathbf{y}\in\Omega} |\langle \mathbf{Z}(\mathbf{x}), \mathbf{Z}(\mathbf{y})\rangle - K_{co}(\mathbf{x}, \mathbf{y})| > \varepsilon\right] \leq \left(\frac{32RL_1}{\varepsilon}\right) \exp\left(-\frac{D\varepsilon^2}{8C_1^2 d}\right)$ where $C_1 = pf(pC_W)$ and $L_1 = L_K f'(C_K) + L_W p^2 \sqrt{C_W} f'(pC_W)$ for some small constant $p > 1$. Moreover, with $D = \Omega\left(\frac{dC_1^2}{\varepsilon^2} \log\left(\frac{RL_1}{\varepsilon\delta}\right)\right)$, one can ensure the same with probability greater than $1 - \delta$.*

Yet again the dependence on input space parameters is similar to that in the case of dot product kernel feature maps. The only non-logarithmic kernel specific dependence is on C_1 which encodes the largest possible value taken by the oracle features which is related to the range of values taken by the kernel K.

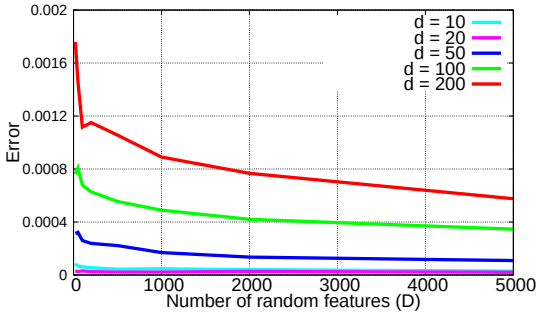
3.6 Experiments

In this section we report results of our feature map construction algorithm on both toy as well as benchmark datasets. In the following, homogeneous kernel refers to the kernel $K_h(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \mathbf{y}\rangle^p$, polynomial kernel refers to $K_p(\mathbf{x}, \mathbf{y}) = (1 + \langle \mathbf{x}, \mathbf{y}\rangle)^p$ and exponential kernel refers to $K_e(\mathbf{x}, \mathbf{y}) = \exp\left(\frac{\langle \mathbf{x}, \mathbf{y}\rangle}{\sigma^2}\right)$. In all our experiments we used $p = 10$ and set the value of the "width" parameter σ to be the mean of all pairwise training data distances, a standard heuristic. We shall denote by d the dimensionality of the original feature space and D to be the number of random feature maps used. Before we move on, we describe a heuristic which when used in conjunction with random feature maps gives attractive results allowing for accelerated training and testing times for the SVM algorithm.

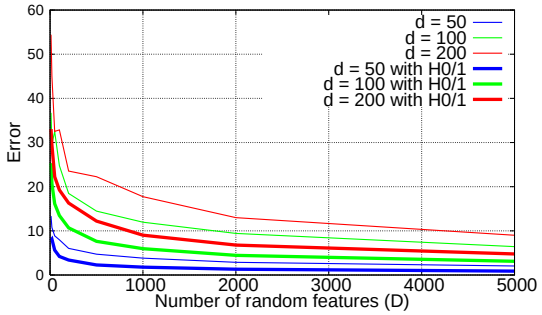
3.6.1 The Heuristic H0/1

Consider a dot product kernel defined by $K(\mathbf{x}, \mathbf{y}) = \sum_{n=0}^{\infty} a_n \langle \mathbf{x}, \mathbf{y}\rangle^n$. This heuristic simply makes an observation that the first two terms of this expansion need not be estimated at all. The first term, being a constant, can be absorbed into the offset parameter of SVM formulations and the second term can be handled by simply adjoining the random features with the original features. This allows us to use all our randomness in estimating higher order terms. We refer to algorithmic formulations that use this heuristic as **H0/1** and those that use only random features as **RF**.

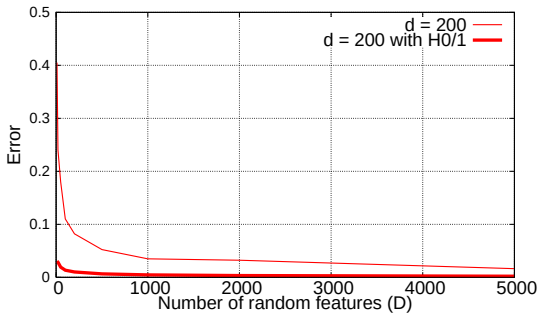
We note some properties of this heuristic. First of all, as we shall see, **H0/1** offers superior accuracies even when using a very small number of random features since we get away with an exact estimate of the leading terms in the Maclaurin expansion. However this is accompanied by two overheads. First of all this offers a small overhead while testing since the test vectors are $(d + D)$-dimensional instead of D-dimensional if we were to use only random features (as is the case with **RF**).



(a) Homogeneous kernel



(b) Polynomial kernel



(c) Exponential kernel

Figure 3.1: Error rates achieved by random feature maps on three dot product kernels. Plots of different colors represent various values of input dimension d. In Figures 3.1b and 3.1c, thin plots represent non-**H0/1** experiments and thick plots of same color represent results for the same value of input dimension d but with **H0/1**.

A more subtle overhead comes at feature map application time since the use of **H0/1** implies that, on an average, each of the D feature maps is estimating a higher order term (as compared to **RF**) which requires more randomness. Moreover, as it takes longer for feature maps estimating higher order terms to be applied (see Algorithm 1), this results in longer feature construction times. Hence, after D is chosen beyond a certain threshold, the benefits offered by **H0/1** are overshadowed by the longer feature construction times and plain **RF** becomes more preferable in terms of lower test times. However, as the experiments will indicate, **H0/1** is an attractive option for ultra fast learning routines for small to moderate values of D which, while increasing feature construction time slightly offer much better classification accuracies than **RF**.

3.6.2 Toy Experiments

In our first experiment, we tested the accuracy of the feature maps on the three dot product kernels K_h, K_p and K_e. We sampled 100 random points from the unit ball in d dimensions (we used various values of d between 10 and 200) and constructed feature maps for various values of D from 10 to 5000. The error incurred by the feature maps was taken to be the average absolute difference between the entries of the kernel matrix

as given by the dot product kernel and that given by the linear kernel on the new feature space given by the feature maps. The results of the experiments, averaged over 5 runs are shown in Figure 3.1. One can see that in each case, the error quickly drops as we increase the value of D.

We also experimented with the effect of **H0/1** on these toy datasets for K_p and K_e (K_h does not have terms corresponding to $n = 0, 1$ and hence **H0/1** cannot be applied). For sake of clarity, the X-axis in all the graphs in Figure 3.1 represent only D and not the final number of features used (which is $d + D$ for **H0/1** experiments). Also, to avoid clutter, we have omitted plots for certain small values of d in Figures 3.1b and 3.1c. Notice how in all cases, **H0/1** registers a sharper drop in error than **RF**.

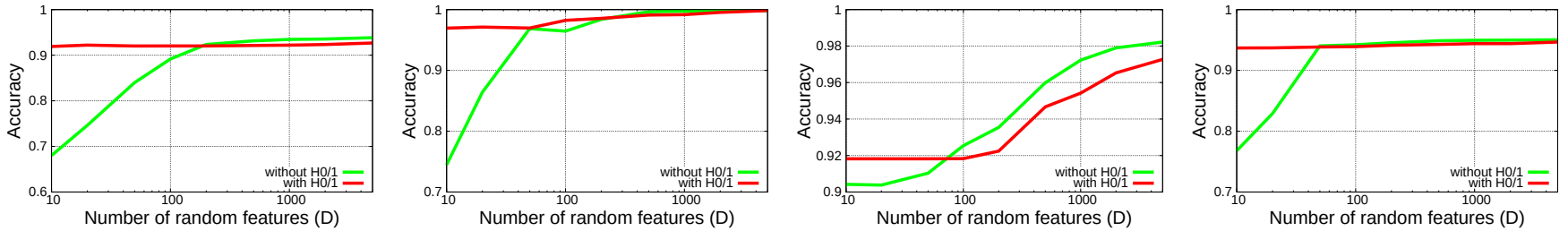
We note that the error rates vary considerably across kernels. This is due to the difference in the range of values taken by these kernels. With the specified values of kernel parameters whereas K_h can only take values in the range $[-1, 1]$ inside $\mathcal{B}_2(\mathbf{0}, 1) \subset \mathbb{R}^d$, K_p can take values up to 1024 and K_e up to 2.73. One notices that the error rates offered by the feature maps also differ in much the same way for these kernels .

3.6.3 Experiments on UCI Datasets

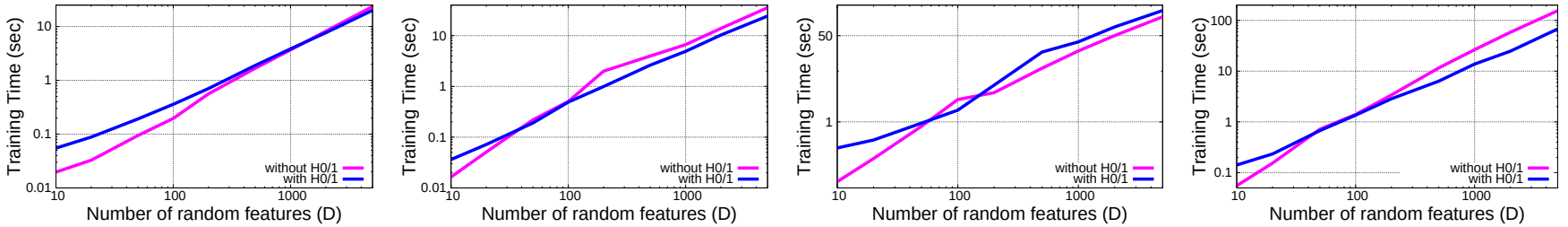
In our second experiment, we tested the performance of our feature map on benchmark datasets (uci). In these experiments we used 60% of the data (subject to a maximum of 20000) for training and the rest as test data. Non-linear kernels were used along with LIBSVM (Chang and Lin, 2011) and random feature routines **RF** and **H0/1** were used alongwith LIBLINEAR (Fan et al., 2008) for the classification tasks. Non-binary problems were binarized randomly for simplicity. Since the kernels are unbounded, the lengths of all vectors were normalized using normalization constants learnt on the training sets. All results presented are averages across five random (but fixed) splits of the datasets.

We first take a look at the performance benefits of **H0/1** on these datasets in Figure 3.2. As before we simply plot D on the X-axis even for **H0/1** experiments for sake of clarity. We observe that in all four cases, **H0/1** offers much higher accuracies as compared to **RF** when used with small number of random features (see Figure 3.2a). Also note that the number of extra features added for **H0/1** is not large (avg. $d = 45$ for the 6 datasets considered). As we increase the number of random features, **H0/1** accuracies move up slowly. However the test feature construction overhead become large after a point and affects test times (see Figure 3.2c). The effect on training times (see Figure 3.2b) is not so clear since the use of **H0/1** also seems to offer greater separability which mitigates the training feature construction overhead in some cases.

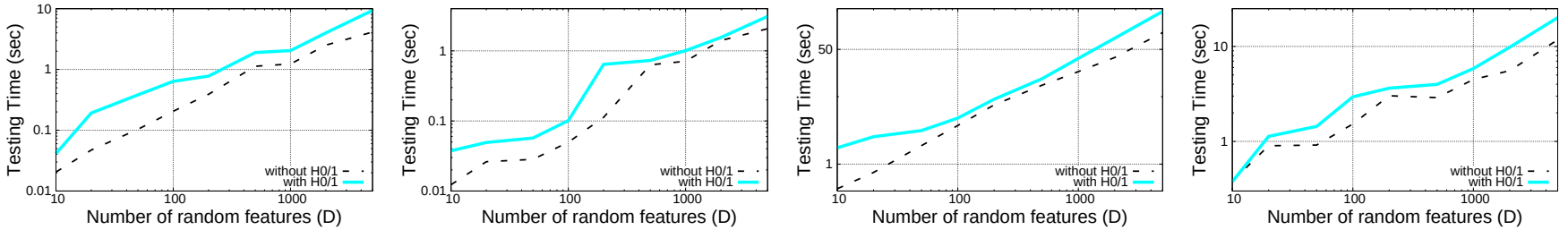
We provide details of the results in Table 3.1. We see that both **RF** and **H0/1** offer significant speedups in both training and test times while offering competitive classification accuracies with **H0/1** doing so at much lower values of D. In some cases the reduction in classification accuracy for **H0/1** is moderate but is almost always accompanied with a spectacular increase in training and test speeds.



(a) Classification accuracies for non-H0/1 (green) and H0/1 (red) routines on 4 datasets



(b) Training times (log-scale) for non-H0/1 (magenta) and H0/1 (blue) routines on the datasets



(c) Testing times (log-scale) for non-H0/1 (gray) and H0/1 (cyan) routines on the datasets

Figure 3.2: Performance of **H0/1** vs non-**H0/1** on four datasets. The first column corresponds to experiments on the `Spambase` dataset with the polynomial kernel. The next three columns correspond to experiments on `Nursery` with the polynomial kernel, `IJCNN` with the exponential kernel and `Cod-RNA` with the exponential kernel.

Dataset	**K** + LIBSVM	**RF** + LIBLINEAR	**H0/1** + LIBLINEAR
`Nursery`	acc = 99.9%	acc = 99.7% *(D = 500)*	acc = 98.2% *(D = 100)*
N = 13000	trn = 18.6s	trn = 3.96s (**4.7**×)	trn = 0.49s (**38**×)
$d = 8$	tst = 3.37s	tst = 0.63s (**5.3**×)	tst = 0.1s (**33**×)
`Spambase`	acc = 93.8%	acc = 93.2% *(D = 500)*	acc = 92.02% *(D = 50)*
N = 4600	trn = 3.64s	trn = 1.67s (**2.2**×)	trn = 0.19s (**19**×)
$d = 57$	tst = 2.84s	tst = 1.13s (**2.5**×)	tst = 0.38s (**7.5**×)
`Cod-RNA`	acc = 95.2%	acc = 94.9% *(D = 500)*	acc = 93.77% *(D = 50)*
N = 60000	trn = 144.1s	trn = 12.1s (**12**×)	trn= 0.63s (**229**×)
$d = 8$	tst = 28.6s	tst = 2.8s (**10**×)	tst = 0.51s (**56**×)
`Adult`	acc = 84.2%	acc = 84.7% *(D = 500)*	acc = 84.7% *(D = 100)*
N = 49000	trn = 179.6s	trn = 21.2s (**8.5**×)	trn = 6.9s (**26**×)
$d = 123$	tst = 60.6s	tst = 15.6s (**3.9**×)	tst = 7.26s (**8.4**×)
`IJCNN`	acc = 98.4%	acc = 97.3% *(D = 1000)*	acc = 92.3% *(D = 200)*
N=141000	trn = 164.1s	trn = 36.5s (**4.5**×)	trn= 4.98s (**33**×)
$d = 22$	tst = 33.4s	tst = 23.3s (**1.4**×)	tst = 7.5s (**4.5**×)
`Covertype`	acc = 77.4%	acc = 77.04% *(D = 1000)*	acc = 75.5% *(D = 100)*
N=581000	trn = 160.95s	trn = 186.1s (—)	trn = 3.9s (**41**×)
$d = 54$	tst = 1653.9s	tst = 236.8s (**7**×)	tst = 70.3s (**23**×)

(a) Polynomial Kernel, $K(\mathbf{x}, \mathbf{y}) = (1 + \langle \mathbf{x}, \mathbf{y} \rangle)^{10}$

Dataset	**K** + LIBSVM	**RF** + LIBLINEAR	**H0/1** + LIBLINEAR
`Nursery`	acc = 99.8%	acc = 99.6% *(D = 500)*	acc = 97.96% *(D = 100)*
N = 13000	trn = 10.8s	trn = 2.52s (**4.3**×)	trn = 0.4s (**27**×)
$d = 8$	tst = 1.7s	tst = 0.6s (**2.8**×)	tst = 0.18s (**9.4**×)
`Spambase`	acc = 93.5%	acc = 92.3% *(D = 500)*	acc = 92.08% *(D = 50)*
N = 4600	trn = 3.19s	trn = 1.9s (**1.7**×)	trn = 0.19s (**17**×)
$d = 57$	tst = 1.89s	tst = 0.6s (**3.1**×)	tst = 0.16s (**74**×)
`Cod-RNA`	acc = 95.2%	acc = 94.9% *(D = 500)*	acc = 93.8% *(D = 50)*
N = 60000	trn = 91.5s	trn = 11.5s (**8**×)	trn= 0.67s (**136**×)
$d = 8$	tst = 17.1s	tst = 2.8s (**6.1**×)	tst = 1.4s (**12**×)
`Adult`	acc = 83.7%	acc = 82.9% *(D = 500)*	acc = 84.8% *(D = 100)*
N = 49000	trn = 263.3s	trn = 39.8s (**6.6**×)	trn = 7.18s (**37**×)
$d = 123$	tst = 33.4s	tst = 14.3s (**2.3**×)	tst = 9.4s (**3.6**×)
`IJCNN`	acc = 98.4%	acc = 97.2% *(D = 1000)*	acc = 92.2% *(D = 200)*
N=141000	trn = 135.8s	trn = 24.9s (**5.5**×)	trn = 5.2s (**26**×)
$d = 22$	tst = 29.98s	tst = 23.4s (**1.3**×)	tst = 9.1s (**3.3**×)
`Covertype`	acc = 80.6%	acc = 76.2% *(D = 1000)*	acc = 75.5% *(D = 100)*
N=581000	trn = 194.1s	trn = 21.4s (**9**×)	trn = 3.7s (**52**×)
$d = 54$	tst = 695.8s	tst = 207s (**3.6**×)	tst = 80.4s (**8.7**×)

(b) Exponential Kernel, $K(\mathbf{x}, \mathbf{y}) = \exp\left(\frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\sigma^2}\right)$

Table 3.1: **RF**, **H0/1** and **K** denote respectively, the use of random features, **H0/1** and actual kernel values. The first columns list the datasets, their sizes (N) and their dimensionalities (d). Subsequent columns list the number of random features used (D), classification accuracies (acc), training/testing times (trn/tst) and speedups (×).

3.7 Proofs

We give missing proofs below.

3.7.1 Proof of Theorem 3.1

We first recollect Schoenberg's result in its original form

Theorem 3.15. *(Schoenberg, 1942, Theorem 2) A function $f : [-1, 1] \to \mathbb{R}$ constitutes a positive definite kernel $K : S_\infty \times S_\infty \to \mathbb{R}$, $K : (\mathbf{x}, \mathbf{y}) \mapsto f(\langle \mathbf{x}, \mathbf{y} \rangle)$ iff f is an analytic function admitting a Maclaurin expansion with only non-negative coefficients i.e. $f(x) = \sum_{n=0}^{\infty} a_n x^n, a_n \geq 0, n = 0, 1, 2, \ldots$. Here $S_\infty = \{\mathbf{x} \in \mathcal{H} : \|\mathbf{x}\|_2 = 1\}$ for some Hilbert space $\mathcal{H}$.*

To see that the non-negativeness of the coefficients of the Maclaurin expansion is necessary just apply Theorem 3.15 to points on S_∞. Since $\{\langle \mathbf{x}, \mathbf{y} \rangle : \mathbf{x}, \mathbf{y} \in \mathcal{B}_2(\mathbf{0}, 1)\} = \{\langle \mathbf{x}, \mathbf{y} \rangle : \mathbf{x}, \mathbf{y} \in S_\infty\}$, the result extends to the general case when the points are coming from $\mathcal{B}_2(\mathbf{0}, 1)$. To see that this suffices we make use of some well known facts regarding positive definite kernels (for example refer to (Schölkopf and Smola, 2002)).

Fact 3.16. *If $K_n, n \in \mathbb{N}$ are positive definite kernels defined on some common domain then the following statements are true*

1. *$c_m K_m + c_n K_n$ is also a positive definite kernel provided $c_m, c_n \geq 0$.*
2. *$K_m K_n$ is also a positive definite kernel.*
3. *If $\lim_{n \to \infty} K_n = K$ and K is continuous then K is also a positive definite kernel.*

Starting with the fact that the dot product kernel is positive definite on any Hilbert space $\mathcal{H}$, applying Fact 3.16.1 and Fact 3.16.2, we get that for every $n \in N$, the kernel $K_n(\mathbf{x}, \mathbf{y}) = \sum_{i=0}^{n} a_i \langle \mathbf{x}, \mathbf{y} \rangle^i$ is positive definite. An application of Fact 3.16.3 along with the fact that the Maclaurin series converges uniformly within its radius of convergence then proves the result.

3.7.2 Proof of Lemma 3.2

We shall first prove this result for the special case of ℓ_2, the Hilbert space of all square summable sequences. Schoenberg's result (Corollary 3.1) will then allow us to extend it to all Hilbert spaces. The *if* part follows readily from the observation that ℓ_2 contains all finite dimensional Euclidean spaces as subspaces and the fact that any kernel that is positive definite over a set is positive definite over all its subsets as well.

For the *only if* part consider any set of n points $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \ell_2$. Clearly there exists an embedding $\Phi : S \to \mathbb{R}^n$ such that for all $i, j \in [n], \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$ (note that the left and the right hand sides are inner products over different spaces). Such

an embedding can be constructed, for example, by taking the Cholesky decomposition of the Gram matrix given by the inner product on ℓ_2 (the entries of the Gram matrix are finite by an application of Cauchy-Schwarz inequality).

Consider the matrix $A = [a_{ij}]$ where $a_{ij} = f\left(\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle\right)$. Since f yields positive definite kernels over all finite dimensional Euclidean spaces, we have $A \succeq 0$. However, by the isometry of the embedding, we have $a_{ij} = f\left(\langle \mathbf{x}_i, \mathbf{x}_j \rangle\right)$. Hence, for any $n < \infty$, for any arbitrary n points, the gram matrix given by $f(\langle \cdot, \cdot \rangle)$ is positive definite (here $\langle \cdot, \cdot \rangle$ is the dot product over ℓ_2). Thus f yields a positive definite kernel over ℓ_2 as well.

To complete the proof we now use Schoenberg's theorem to extend this result to all Hilbert spaces. If a dot product kernel is positive definite over all finite dimensional spaces then the above argument shows it to be positive definite over ℓ_2. Hence, by Corollary 3.1, the function f defining this kernel must have a non-negative Maclaurin's expansion. From here on an argument similar to the one used to prove the sufficiency part of Corollary 3.1 (using Fact 3.16) can be used to show that this kernel is positive definite over all Hilbert spaces.

On the other hand, if a dot product kernel is positive definite over Hilbert spaces, then we use its positive-definiteness over ℓ_2, along with the argument used in showing the *if* part above, to prove that the kernel is positive definite over all finite dimensional Euclidean spaces.

3.7.3 Proof of Lemma 3.4

We have the following chain of equalities:

$$\begin{aligned}
\mathop{\mathbb{E}}_{\boldsymbol{\omega}} \left[\!\left[Z(\mathbf{x}) Z(\mathbf{y}) \right]\!\right] &= \mathop{\mathbb{E}}_{\boldsymbol{\omega}} \left[\!\left[\boldsymbol{\omega}^\top \mathbf{x} \cdot \boldsymbol{\omega}^\top \mathbf{y} \right]\!\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{\omega}} \left[\!\left[\left(\sum_{i=1}^{d} \boldsymbol{\omega}_i \mathbf{x}_i \right) \left(\sum_{i=1}^{d} \boldsymbol{\omega}_i \mathbf{y}_i \right) \right]\!\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{\omega}} \left[\!\left[\sum_{i=1}^{d} \boldsymbol{\omega}_i^2 \mathbf{x}_i \mathbf{y}_i + \sum_{i \neq j}^{d} \boldsymbol{\omega}_i \boldsymbol{\omega}_j \mathbf{x}_i \mathbf{y}_j \right]\!\right] \\
&= \sum_{i=1}^{d} \mathop{\mathbb{E}}_{\boldsymbol{\omega}} \left[\!\left[\boldsymbol{\omega}_i^2 \right]\!\right] \mathbf{x}_i \mathbf{y}_i + \sum_{i \neq j}^{d} \mathop{\mathbb{E}}_{\boldsymbol{\omega}} \left[\!\left[\boldsymbol{\omega}_i \right]\!\right] \mathop{\mathbb{E}}_{\boldsymbol{\omega}} \left[\!\left[\boldsymbol{\omega}_j \right]\!\right] \mathbf{x}_i \mathbf{y}_j \\
&= \sum_{i=1}^{d} \mathbf{x}_i \mathbf{y}_i + 0 = \langle \mathbf{x}, \mathbf{y} \rangle
\end{aligned}$$

where in the third equality we have used linearity of expectation and the pairwise independence of the different coordinates of $\boldsymbol{\omega}$. The fourth equality is arrived at by using properties of the distribution. Notice that any distribution that is symmetric about zero with unit second moment can be used for sampling the coordinates of $\boldsymbol{\omega}$. This particular choice both simplifies the analysis as well as is easy to implement in practice.

3.7.4 Proof of Lemma 3.5

We have the following chain of equalities:

$$\begin{aligned}
\mathbb{E}\left[\!\left[Z(\mathbf{x})Z(\mathbf{y})\right]\!\right] &= \mathop{\mathbb{E}}_{N}\left[\!\left[\mathop{\mathbb{E}}_{\boldsymbol{\omega}_1,\ldots,\boldsymbol{\omega}_N}\left[\!\left[Z(\mathbf{x})Z(\mathbf{y})\right]\!\right]\middle| N\right]\!\right] \\
&= \mathop{\mathbb{E}}_{N}\left[\!\left[a_N p^{N+1}\mathop{\mathbb{E}}_{\boldsymbol{\omega}_1,\ldots,\boldsymbol{\omega}_N}\left[\!\left[\prod_{j=1}^{N}\boldsymbol{\omega}_j^\top\mathbf{x}\prod_{j=1}^{N}\boldsymbol{\omega}_j^\top\mathbf{y}\right]\!\right]\right]\!\right] \\
&= \mathop{\mathbb{E}}_{N}\left[\!\left[a_N p^{N+1}\left(\mathop{\mathbb{E}}_{\boldsymbol{\omega}}\left[\!\left[\boldsymbol{\omega}^\top\mathbf{x}\cdot\boldsymbol{\omega}^\top\mathbf{y}\right]\!\right]\right)^N\right]\!\right] \\
&= \mathop{\mathbb{E}}_{N}\left[\!\left[a_N p^{N+1}\left\langle\mathbf{x},\mathbf{y}\right\rangle^N\right]\!\right] \\
&= \sum_{n=0}^{\infty}\frac{1}{p^{n+1}}\cdot a_n p^{n+1}\left\langle\mathbf{x},\mathbf{y}\right\rangle^n \\
&= K(\mathbf{x},\mathbf{y}).
\end{aligned}$$

where the first step uses the fact that the index N and the vectors $\boldsymbol{\omega}_i$ are chosen independently, the fourth step uses the fact that the vectors $\boldsymbol{\omega}_i$ are chosen independently among themselves and the fifth step uses Lemma 2.

3.7.5 Proof of Lemma 3.9

Since $\left\langle\mathbf{Z}(\mathbf{x}),\mathbf{Z}(\mathbf{y})\right\rangle = \frac{1}{D}\sum_{i=1}^{D} Z_i(\mathbf{x})Z_i(\mathbf{y})$ and $\nabla_{\mathbf{x}}\left\langle\mathbf{Z}(\mathbf{x}),\mathbf{Z}(\mathbf{y})\right\rangle = \frac{1}{D}\sum_{i=1}^{D}\nabla_{\mathbf{x}}\left(Z_i(\mathbf{x})Z_i(\mathbf{y})\right)$ we have by triangle inequality,

$$\left\|\nabla_{\mathbf{x}}\left\langle\mathbf{Z}(\mathbf{x}),\mathbf{Z}(\mathbf{y})\right\rangle\right\| \leq \frac{1}{D}\sum_{i=1}^{D}\left\|\nabla_{\mathbf{x}}\left(Z_i(\mathbf{x})Z_i(\mathbf{y})\right)\right\|.$$

Since all the Z_i feature maps are identical it would be sufficient to bound $\left\|\nabla_{\mathbf{x}}\left(Z_1(\mathbf{x})Z_1(\mathbf{y})\right)\right\|$ and by the above calculation, the same bound would hold for $\left\|\nabla_{\mathbf{x}}\left\langle\mathbf{Z}(\mathbf{x}),\mathbf{Z}(\mathbf{y})\right\rangle\right\|$ as well. Let $Z_1 : \mathbf{x} \mapsto \sqrt{a_N p^{N+1}}\prod_{j=1}^{N}\boldsymbol{\omega}_j^\top\mathbf{x}$ for some $N \leq k$. We then apply the following sequence of simplifications:

$$\begin{aligned}
\nabla_{\mathbf{x}}\left(Z_1(\mathbf{x})Z_1(\mathbf{y})\right) &= \nabla_{\mathbf{x}}\left(a_N p^{N+1}\prod_{j=1}^{N}\boldsymbol{\omega}_j^\top\mathbf{x}\prod_{j=1}^{N}\boldsymbol{\omega}_j^\top\mathbf{y}\right) \\
&= \left(a_N p^{N+1}\prod_{j=1}^{N}\boldsymbol{\omega}_j^\top\mathbf{y}\right)\nabla_{\mathbf{x}}\left(\prod_{j=1}^{N}\boldsymbol{\omega}_j^\top\mathbf{x}\right) \\
&= \left(a_N p^{N+1}\prod_{j=1}^{N}\boldsymbol{\omega}_j^\top\mathbf{y}\right)\sum_{j=1}^{N}\left(\prod_{i\neq j}\boldsymbol{\omega}_i^\top\mathbf{x}\right)\boldsymbol{\omega}_i.
\end{aligned}$$

We note that for any $\boldsymbol{\omega}$ chosen, $\|\boldsymbol{\omega}\| = \sqrt{d}$. Moreover, as we have seen before, for any

$\boldsymbol{\omega}$, we have $\sup_{\mathbf{x}\in\Omega} \left|\boldsymbol{\omega}^\top \mathbf{x}\right| \leq R$ by Hölder's inequality. Thus we can bound $\|\nabla_{\mathbf{x}} \left(Z_1(\mathbf{x})Z_1(\mathbf{y})\right)\|$ as

$$\begin{aligned}
\|\nabla_{\mathbf{x}} \left(Z_1(\mathbf{x})Z_1(\mathbf{y})\right)\| &= \left\| \left(a_N p^{N+1} \prod_{j=1}^{N} \boldsymbol{\omega}_j^\top \mathbf{y} \right) \sum_{j=1}^{N} \left(\prod_{i\neq j} \boldsymbol{\omega}_i^\top \mathbf{x} \right) \boldsymbol{\omega}_i \right\| \\
&= a_N p^{N+1} \left(\prod_{j=1}^{N} \left|\boldsymbol{\omega}_j^\top \mathbf{y}\right| \right) \left\| \sum_{j=1}^{N} \left(\prod_{i\neq j} \boldsymbol{\omega}_i^\top \mathbf{x} \right) \boldsymbol{\omega}_i \right\| \\
&\leq a_N p^{N+1} \left(\prod_{j=1}^{N} \left|\boldsymbol{\omega}_j^\top \mathbf{y}\right| \right) \sum_{j=1}^{N} \left(\prod_{i\neq j} \left|\boldsymbol{\omega}_i^\top \mathbf{x}\right| \right) \|\boldsymbol{\omega}_i\| \\
&\leq a_N p^{N+1} R^N \sum_{j=1}^{N} R^{N-1}\sqrt{d} = N a_N p^{N+1} R^{2N-1}\sqrt{d} \\
&\leq p^2 R\sqrt{d} \sum_{n=0}^{\infty} n a_n (pR^2)^{n-1} = p^2 R\sqrt{d} f'(pR^2)
\end{aligned}$$

where we have used the triangle inequality in the third step. Similarly we can show $\sup_{\mathbf{x},\mathbf{y}\in\Omega} \|\nabla_{\mathbf{y}} \left(Z_1(\mathbf{x})Z_1(\mathbf{y})\right)\| \leq p^2 R\sqrt{d} f'(pR^2)$.

Chapter 4

Classification with Indefinite Kernels

Contents

Abstract *We consider the problem of classification using similarity/distance functions over data. Specifically, we propose a framework for defining the* goodness *of a (dis)similarity function with respect to a given learning task and propose algorithms that have guaranteed generalization properties when working with such* good *functions. Our framework unifies and generalizes the frameworks proposed by Balcan and Blum (2006) and Wang et al. (2007). An attractive feature of our framework is its adaptability to data - we do not promote a fixed notion of goodness but rather let data dictate it. We show, by giving theoretical guarantees that the goodness criterion best suited to a problem can itself be learned which makes our approach applicable to a variety of domains and problems. We propose a* landmarking*-based approach to obtaining a classifier from such learned goodness criteria. We then provide a novel diversity based heuristic to perform task-driven selection of landmark points instead of random selection. We demonstrate the effectiveness of our goodness criteria learning method as well as the landmark selection heuristic on a variety of similarity-based learning datasets and benchmark UCI datasets on which our method consistently outperforms existing approaches by a significant margin.*

4.1 Introduction

Machine learning algorithms have found applications in diverse domains such as computer vision, bio-informatics and speech recognition. Working in such heterogeneous domains often involves handling data that is not presented as explicit features embedded into vector spaces. However in many domains, for example co-authorship graphs, it is natural to devise similarity/distance functions over pairs of points. While classical techniques like decision tree and linear Perceptron cannot handle such data, several modern machine learning algorithms such as support vector machine (SVM) can be *kernelized* and are thereby capable of using kernels or similarity functions.

However, most of these algorithms require the similarity functions to be positive semi-definite (PSD), which essentially implies that the similarity stems from an (implicit) embedding of the data into a Hilbert space. Unfortunately in many domains, the most natural notion of similarity does not satisfy this condition - moreover, verifying this condition is usually a non-trivial exercise. Take for example the case of images on which the most natural notions of distance (Euclidean, Earth-mover) (Indyk and Thaper, 2003) do not form PSD kernels. Co-authorship graphs give another such example.

Consequently, there have been efforts to develop algorithms that do not make assumptions about the PSD-ness of the similarity functions used. One can discern three main approaches in this area. The first approach tries to coerce a given similarity measure into a PSD one by either clipping or shifting the spectrum of the kernel matrix (Pękalska and Duin, 2001; Chen et al., 2009a). However, these approaches are mostly restricted to transductive settings and are not applicable to large scale problems due to eigenvector computation requirements. The second approach consists of algorithms that either adapt classical methods like k-NN to handle non-PSD similarity/distance functions and consequently offer slow test times (Chen et al., 2009a), or are forced to solve non-convex formulations (Ong et al., 2004; Haasdonk, 2005).

The third approach, which has been investigated recently in a series of papers (Balcan and Blum, 2006; Balcan et al., 2008a; Wang et al., 2007) (although the basic idea has been around for quite some time (Graepel et al., 1998)), uses the similarity function to embed the domain into a low dimensional Euclidean space. More specifically, these algorithms choose *landmark* points in the domain which then give the embedding. Assuming a certain "goodness" property (that is formally defined) for the similarity function, these models offer both generalization guarantees in terms of how well-suited the similarity function is to the classification task as well as the ability to use fast algorithmic techniques such as linear SVM (Fan et al., 2008) on the *landmarked space*. The model proposed by Balcan and Blum (2006) give sufficient conditions for a similarity function to be well suited to such a landmarking approach. Wang et al. (2007) on the other hand provide goodness conditions for dissimilarity functions that enable landmarking algorithms.

Informally, a similarity (or distance) function can be said to be good if points of similar labels are closer to each other than points of different labels in some sense. Both the models

described above restrict themselves to a fixed goodness criterion, which need not hold for the underlying data. We observe that this might be too restrictive in many situations and present a framework that allows us to tune the goodness criterion itself to the classification problem at hand. Our framework consequently unifies and generalizes those presented by Balcan and Blum and Wang et al.. We first prove generalization bounds corresponding to landmarked embeddings under a *fixed* goodness criterion. We then provide a uniform-convergence bound that enables us to learn the best goodness criterion for a given problem. We further generalize our framework by giving the ability to incorporate any Lipschitz loss function into our goodness criterion which allows us to give guarantees for the use of various algorithms such as C-SVM and logistic regression on the landmarked space. A desirable side-effect of our model is that our training algorithms require training a linear classifier in the *landmarked space*, for which there exist fast algorithms such as (Fan et al., 2008). This endows our model with fast training as well as testing routines.

Similar to Balcan and Blum and Wang et al., our framework requires random sampling of training points to create the embedding space[1]. However in practice, random sampling is inefficient and requires sampling of a large number of points to form a useful embedding, thereby increasing training and test time. To address this issue, Wang et al. propose a heuristic to select the points that are to be used as landmarks. However their scheme is tied to their optimization algorithm and is computationally inefficient for large scale data. In contrast, we propose a general heuristic for selecting informative landmarks based on a novel notion of diversity which can then be applied to any instantiation of our model.

Finally, we apply our methods to a variety of benchmark datasets for similarity learning as well as ones from the UCI repository. We empirically demonstrate that our learning model and landmark selection heuristic consistently offers significant improvements over the existing approaches. In particular, for a small number of landmark points, which is a practically important scenario as it is expensive to compute similarity function values at test time, our method provides, on an average, accuracy boosts of upto 5% over existing methods. We also note that our methods can be applied on top of any strategy used to learn the similarity measure (eg. MKL techniques (Varma and Babu, 2009)) or the distance measure (eg. (Jain et al., 2012)) itself. Akin to (Balcan and Blum, 2006), our techniques can also be extended to learn a combination of (dis)similarity functions but we do not explore these extensions in this work.

4.2 Methodology

Let $\mathcal{D}$ be a fixed but unknown distribution over the labeled input domain $\mathcal{X}$ and let $\ell : \mathcal{X} \to \{-1, +1\}$ be a labeling over the domain. Given a (potentially non-PSD) similarity function[2] $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, the goal is to learn a classifier $\hat{\ell} : \mathcal{X} \to \{-1, +1\}$ from a finite

[1] ↑ Throughout the chapter, we use the terms *embedding space* and *landmarked space* interchangeably.

[2] ↑ Results described in this section hold for distance functions as well; we present results with respect to similarity functions for sake of simplicity.

number of i.i.d. samples from $\mathcal{D}$ that has bounded generalization error over $\mathcal{D}$.

Now, learning a reasonable classifier seems unlikely if the given similarity function does not have any inherent "goodness" property. Intuitively, the goodness of a similarity function should be its suitability to the classification task at hand. For PSD kernels, the notion of goodness is defined in terms of the margin offered in the RKHS (Balcan et al., 2006). However, a more basic requirement is that the similarity function should preserve affinities among similarly labeled points - that is to say, a good similarity function should not, on an average, assign higher similarity values to dissimilarity labeled points than to similarly labeled points. This intuitive notion of goodness turns out to be rather robust in the sense that all PSD kernels that offer a good margin in their respective RKHSs satisfy some form of this goodness criterion as well (Srebro, 2007).

Recently there has been some interest in studying different realizations of this general notion of goodness and developing corresponding algorithms that allow for efficient learning with similarity/distance functions. Balcan and Blum (2006) present a goodness criteria in which a good similarity function is considered to be one that, for most points, assigns a greater average similarity to similarly labeled points than to dissimilarly labeled points. More specifically, a similarity function is (ε, γ)-good if there exists a weighting function $w : \mathcal{X} \to \mathbb{R}$ such that, at least a $(1 - \varepsilon)$ probability mass of examples $x \sim \mathcal{D}$ satisfies:

$$\mathop{\mathbb{E}}_{x' \sim \mathcal{D}} \left[\!\!\left[w\left(x'\right) K(x, x') | \ell(x') = \ell(x) \right]\!\!\right] \geq \mathop{\mathbb{E}}_{x' \sim \mathcal{D}} \left[\!\!\left[w\left(x'\right) K(x, x') | \ell(x') \neq \ell(x) \right]\!\!\right] + \gamma. \tag{4.1}$$

where instead of average similarity, one considers an average weighted similarity to allow the definition to be more general.

Wang et al. (2007) define a distance function d to be good if a large fraction of the domain is, on an average, closer to similarly labeled points than to dissimilarly labeled points. They allow these averages to be calculated based on some distribution distinct from $\mathcal{D}$, one that may be more suited to the learning problem. However it turns out that their definition is equivalent to one in which one again assigns weights to domain elements, as done by Balcan and Blum (2006), and the following holds

$$\mathop{\mathbb{E}}_{x', x'' \sim \mathcal{D} \times \mathcal{D}} \left[\!\!\left[w(x') w(x'') \operatorname{sgn}\left(d(x, x'') - d(x, x')\right) | \ell(x') = \ell(x), \ell(x'') \neq \ell(x) \right]\!\!\right] > \gamma \tag{4.2}$$

Assuming their respective goodness criteria, Balcan and Blum (2006) and Wang et al. (2007) provide efficient algorithms to learn classifiers with bounded generalization error. However these notions of goodness with a single fixed criterion may be too restrictive in the sense that the data and the (dis)similarity function may not satisfy the underlying criterion. This is, for example, likely in situations with high intra-class variance. Thus there is need to make the goodness criterion more flexible and data-dependent.

To this end, we unify and generalize both the above criteria to give a notion of goodness that is more data dependent. Although the above goodness criteria (4.1) and (4.2) seem

disparate at first, they can be shown to be special cases of a generalized framework where an antisymmetric function is used to compare intra and inter-class affinities. We use this observation to define our novel goodness criterion using arbitrary bounded antisymmetric functions which we refer to as *transfer functions*. This allows us to define a family of goodness criteria of which (4.1) and (4.2) form special cases ((4.1) uses the identity function and (4.2) uses the sign function as transfer function). Moreover, the resulting definition of a good similarity function is more flexible and data dependent. In the rest of the chapter we shall always assume that our similarity functions are normalized i.e. for the domain of interest $\mathcal{X}$, $\sup_{x,y\in\mathcal{X}} K(x,y) \leq 1$.

Definition 4.1 (Good Similarity Function)**.** *A similarity function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is said to be an (ε, γ, B)-good similarity for a learning problem where $\varepsilon, \gamma, B > 0$ if for some antisymmetric transfer function $f : \mathbb{R} \to \mathbb{R}$ and some weighting function $w : \mathcal{X} \times \mathcal{X} \to [-B, B]$, at least a $(1-\varepsilon)$ probability mass of examples $x \sim \mathcal{D}$ satisfies*

$$\mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}} \left[\!\left[w\left(x', x''\right) f\left(K(x,x') - K(x,x'')\right) | \ell(x') = \ell(x), \ell(x'') \neq \ell(x) \right]\!\right] \geq C_f \gamma \quad (4.3)$$

where $C_f = \sup_{x,x'\in\mathcal{X}} f(K(x,x')) - \inf_{x,x'\in\mathcal{X}} f(K(x,x'))$

As mentioned before, the above goodness criterion generalizes the previous notions of goodness (we refer the reader to Appendix A for a discussion) and is adaptive to changes in data as it allows us, as shall be shown later, to learn the best possible criterion for a given classification task by choosing the most appropriate transfer function from a parametrized family of functions. We stress that the property of antisymmetry for the transfer function is crucial to the definition in order to provide a uniform treatment to points of all classes as will be evident in the proof of Theorem 4.2.

As in (Balcan and Blum, 2006; Wang et al., 2007), our goodness criterion lends itself to a simple learning algorithm which consists of choosing a set of d random pairs of points from the domain $\mathcal{P} = \left\{\left(x_i^+, x_i^-\right)\right\}_{i=1}^d$ (which we refer to as *landmark pairs*) and defining an embedding of the domain into a *landmarked space* using these landmarks : $\Phi_L : \mathcal{X} \to \mathbb{R}^d$ as follows

$$\Phi_L(x) = \left(f(K(x, x_i^+) - K(x, x_i^-))\right)_{i=1}^d \in \mathbb{R}^d.$$

The advantage of performing this embedding is the guaranteed existence of a large margin classifier in the landmarked space as shown below.

Theorem 4.2. *If K is an (ε, γ, B)-good similarity with respect to transfer function f and weight function w then for any $\varepsilon_1 > 0$, with probability at least $1-\delta$ over the choice of $d = (8/\gamma^2)\ln(2/\delta\varepsilon_1)$ positive and negative samples, $\left\{x_i^+\right\}_{i=1}^d \subset \mathcal{D}^+$ and $\left\{x_i^-\right\}_{i=1}^d \subset \mathcal{D}^-$ respectively, the classifier $h(x) = sgn[g(x)]$ where*

$$g(x) = \frac{1}{d}\sum_{i=1}^{d} w(x_i^+, x_i^-) f\left(K(x, x_i^+) - K(x, x_i^-)\right)$$

has error no more than $\varepsilon + \varepsilon_1$ *at margin* $\frac{\gamma}{2}$.

However, there are two hurdles to obtaining this large margin classifier. Firstly, the existence of this classifier itself is predicated on the use of the correct transfer function, something which is unknown. Secondly, even if an optimal transfer function is known, the above formulation cannot be converted into an efficient learning algorithm for discovering the (unknown) weights since the formulation seeks to minimize the number of misclassifications which is an intractable problem in general.

We overcome these two hurdles by proposing a nested learning problem. First of all we assume that for some fixed loss function L, given any transfer function and any set of landmark pairs, it is possible to obtain a large margin classifier in the corresponding landmarked space that minimizes L. Having made this assumption, we address below the issue of learning the optimal transfer function for a given learning task. However as we have noted before, this assumption is not valid for arbitrary loss functions. This is why, subsequently in Section 4.2.2, we shall show it to be valid for a large class of loss functions by incorporating surrogate loss functions into our goodness criterion.

4.2.1 Learning the transfer function

In this section we present results that allow us to learn a near optimal transfer function from a family of transfer functions. We shall assume, for some fixed loss function L, the existence of an efficient routine which we refer to as TRAIN that shall return, for any landmarked space indexed by a set of landmark pairs $\mathcal{P}$, a large margin classifier minimizing L. The routine TRAIN is allowed to make use of additional training data to come up with this classifier.

An immediate algorithm for choosing the best transfer function is to simply search the set of possible transfer functions (in an algorithmically efficient manner) and choose the one offering lowest training error. We show here that given enough landmark pairs, this simple technique, which we refer to as **FTUNE** (see Algorithm 4) is guaranteed to return a near-best transfer function. For this we prove a uniform convergence type guarantee on the space of transfer functions.

Let $\mathcal{F} \subset [-1,1]^{\mathbb{R}}$ be a class of antisymmetric functions and $\mathcal{W} = [-B,B]^{\mathcal{X}\times\mathcal{X}}$ be a class of weight functions. For two real valued functions f and g defined on $\mathcal{X}$, let $\|f-g\|_\infty := \sup_{x\in\mathcal{X}} |f(x)-g(x)|$. Let $\mathcal{B}_\infty(f,r) := \{f' \in \mathcal{F} \mid \|f-f'\|_\infty < r\}$. Let L be a C_L-Lipschitz loss function. Let $\mathcal{P} = \left\{\left(x_i^+, x_i^-\right)\right\}_{i=1}^d$ be a set of (random) landmark pairs. For any $f \in \mathcal{F}$, $w \in \mathcal{W}$, define

$$
\begin{aligned}
G_{(f,w)}(x) &= \mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}} \left[\!\left[w\left(x',x''\right) f\left(K(x,x') - K(x,x'')\right) | \ell(x') = \ell(x), \ell(x'') \neq \ell(x) \right]\!\right] \\
g_{(f,w)}(x) &= \frac{1}{d}\sum_{i=1}^{d} w\left(x_i^+, x_i^-\right) f\left(K(x,x_i^+) - K(x,x_i^-)\right)
\end{aligned}
$$

Theorem 4.7 (see Section 4.2.2) guarantees us that for any fixed f and any $\varepsilon_1 > 0$, if d is

large enough then $\mathop{\mathbb{E}}_{x}\left[\!\left[L(g_{(f,w)}(x))\right]\!\right] \leq \mathop{\mathbb{E}}_{x}\left[\!\left[L(G_{(f,w)}(x))\right]\!\right] + \varepsilon_1$. We now show that a similar result holds even if one is allowed to vary f. Before stating the result, we develop some notation.

For any transfer function f and arbitrary choice of landmark pairs $\mathcal{P}$, let $w_{(g,f)}$ be the *best* weighting function for this choice of transfer function and landmark pairs i.e. let $w_{(g,f)} = \mathop{\arg\min}_{w\in[-B,B]^d} \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(g_{(f,w)}(x)\right)\right]\!\right]$ [3]. Similarly, let $w_{(G,f)}$ be the best weighting function corresponding to G i.e. $w_{(G,f)} = \mathop{\arg\min}_{w\in\mathcal{W}} \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w)}(x)\right)\right]\!\right]$. Then we can ensure the following :

Theorem 4.3. *Let $\mathcal{F}$ be a compact class of transfer functions with respect to the infinity norm and $\varepsilon_1, \delta > 0$. Let $\mathcal{N}(\mathcal{F}, r)$ be the size of the smallest ε-net over $\mathcal{F}$ with respect to the infinity norm at scale $r = \frac{\varepsilon_1}{4C_L B}$. Then if one chooses $d = \frac{64B^2C_L^2}{\varepsilon_1^2}\ln\left(\frac{16B\cdot\mathcal{N}(\mathcal{F},r)}{\delta\varepsilon_1}\right)$ random landmark pairs then we have the following with probability greater than $(1-\delta)$*

$$\sup_{f\in\mathcal{F}}\left[\left|\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(g_{(f,w_{(g,f)})}(x)\right)\right]\!\right] - \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w_{(G,f)})}(x)\right)\right]\!\right]\right|\right] \leq \varepsilon_1$$

We shall prove the theorem in two parts. As we shall see, one of the parts is fairly simple to prove. To prove the other part, we shall exploit the Lipschitz properties of the loss function as well as the fact that the class of transfer functions chosen form a compact set. Let us call a given set of landmark pairs to be *good* with respect to a fixed transfer function $f \in \mathcal{F}$ if for the corresponding g, $\mathop{\mathbb{E}}_{x}\left[\!\left[L(g(x))\right]\!\right] \leq \mathop{\mathbb{E}}_{x}\left[\!\left[L(G(x))\right]\!\right] + \varepsilon_1$ for some small fixed $\varepsilon_1 > 0$.

We will first prove, using Lipschitz properties of the loss function that if a given set of landmarks is good with respect to a given transfer function, then it is also good with respect to all transfer functions in its neighborhood. Having proved this, we will apply a standard covering number argument in which we will ensure that a large enough set of landmarks is good with respect to a set of transfer functions that form an ε-net over $\mathcal{F}$ and use the previous result to complete the proof.

We first prove a series of simple results which will be used in the first part of the proof. In the following f and f' are two transfer functions such that $f' \in \mathcal{B}_\infty(f, r) \cap \mathcal{F}$.

Lemma 4.4. *The following results are true*

1. *For any fixed $f \in F$, $\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w_{(G,f)})}(x)\right)\right]\!\right] \leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w)}(x)\right)\right]\!\right]$ for all $w \in \mathcal{W}$.*
2. *For any fixed $f \in F$, any fixed g obtained by an arbitrary choice of landmark pairs, $\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(g_{(f,w_{(g,f)})}(x)\right)\right]\!\right] \leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(g_{(f,w)}(x)\right)\right]\!\right]$ for all $w \in \mathcal{W}$.*
3. *For any $f' \in \mathcal{B}_\infty(f, r) \cap \mathcal{F}$,*

$$\left|\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w_{(G,f)})}(x)\right)\right]\!\right] - \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f',w_{(G,f')})}(x)\right)\right]\!\right]\right| \leq C_L r B.$$

[3] ↑ Note that the function $g_{(f,w)}(x)$ is dictated by the choice of the set of landmark pairs $\mathcal{P}$

4. *For any fixed g obtained by an arbitrary choice of landmark pairs, $f' \in \mathcal{B}_\infty(f, r) \cap \mathcal{F}$,* $\left| \mathop{\mathbb{E}}_{x\sim\mathcal{D}} \left[\!\left[L\left(g_{(f,w_{(g,f)})}(x)\right)\right]\!\right] - \mathop{\mathbb{E}}_{x\sim\mathcal{D}} \left[\!\left[L\left(g_{(f',w_{(g,f')})}(x)\right)\right]\!\right] \right| \leq C_L r B.$

Using the above results we get a preliminary form of the first part of our proof as follows :

Lemma 4.5. *Suppose a set of landmarks is $(\varepsilon_1/2)$-good for a particular classifier $f \in \mathcal{F}$ (i.e. $\mathop{\mathbb{E}}_{x\sim\mathcal{D}} \left[\!\left[L\left(g_{(f,w_{(G,f)})}(x)\right)\right]\!\right] < \mathop{\mathbb{E}}_{x\sim\mathcal{D}} \left[\!\left[L\left(G_{(f,w_{(G,f)})}(x)\right)\right]\!\right] + \varepsilon_1/2$), then the same set of landmarks is also ε_1-good for any $f' \in \mathcal{B}_\infty(f, r) \cap \mathcal{F}$ (i.e. for all $f' \in \mathcal{B}_\infty(f, r) \cap \mathcal{F}$, $\mathop{\mathbb{E}}_{x\sim\mathcal{D}} \left[\!\left[L\left(g_{(f',w_{(g,f')})}(x)\right)\right]\!\right] \leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}} \left[\!\left[L\left(G_{(f',w_{(G,f')})}(x)\right)\right]\!\right] + \varepsilon_1$) for some $r = r(\varepsilon_1)$.*

The full proof of Theorem 4.3 is given in Section 4.4. This result tells us that in a large enough landmarked space, we shall, for each function $f \in \mathcal{F}$, recover close to the best classifier possible for that transfer function. Thus, if we iterate over the set of transfer functions (or use some gradient-descent based optimization routine), we are bound to select a transfer function that is capable of giving a classifier that is close to the best.

4.2.2 Working with surrogate loss functions

The formulation of a good similarity function suggests a simple learning algorithm that involves the construction of an embedding of the domain into a landmarked space on which the existence of a large margin classifier having low misclassification rate is guaranteed. However, in order to exploit this guarantee we would have to learn the weights $w\left(x_i^+, x_i^-\right)$ associated with this classifier by minimizing the empirical misclassification rate on some training set.

Unfortunately, not only is this problem intractable but also hard to solve approximately (Garey and Johnson, 1979; Arora et al., 1997). Thus what we require is for the landmarked space to admit a classifier that has low error with respect to a loss function that can also be efficiently minimized on any training set. In such a situation, minimizing the loss on a random training set would, with very high probability, give us weights that give similar performance guarantees as the ones used in the goodness criterion.

With a similar objective in mind, Balcan and Blum (2006) offer variants of its goodness criterion tailored to the hinge loss function which can be efficiently optimized on large training sets (for example LIBSVM (Chang and Lin, 2011)). Here we give a general notion of goodness that can be tailored to any arbitrary Lipschitz loss function.

Definition 4.6. *A similarity function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is said to be an (ε, B)-good similarity for a learning problem with respect to a loss function $L : \mathbb{R} \to \mathbb{R}^+$ where $\varepsilon > 0$ if for some transfer function $f : \mathbb{R} \to \mathbb{R}$ and some weighting function $w : \mathcal{X} \times \mathcal{X} \to [-B, B]$, $\mathop{\mathbb{E}}_{x\sim\mathcal{D}} [\![L(G(x))]\!] \leq \varepsilon$ where*

$$G(x) = \mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}} \left[\!\left[w\left(x', x''\right) f\left(K(x, x') - K(x, x'')\right) | \ell(x') = \ell(x), \ell(x'') \neq \ell(x)\right]\!\right].$$

One can see that taking the loss functions as $L(x) = \mathbb{1}_{x<C_f\gamma}$ gives us Equation 4.3 which defines a good similarity under the $0-1$ loss function. It turns out that we can, for any Lipschitz loss function, give similar guarantees on the performance of the classifier in the landmarked space.

Theorem 4.7. *If K is an (ε, B)-good similarity function with respect to a C_L-Lipschitz loss function L then for any $\varepsilon_1 > 0$, with probability at least $1-\delta$ over the choice of $d = (16B^2C_L^2/\varepsilon_1^2)\ln(4B/\delta\varepsilon_1)$ positive and negative samples from $\mathcal{D}^+$ and $\mathcal{D}^-$ respectively, the expected loss of the classifier $g(x)$ with respect to L satisfies $\mathop{\mathbb{E}}_{x}[\![L(g(x))]\!] \leq \varepsilon+\varepsilon_1$ where $g(x) = \frac{1}{d}\sum_{i=1}^{d} w\left(x_i^+, x_i^-\right) f\left(K(x, x_i^+) - K(x, x_i^-)\right)$.*

If the loss function is hinge loss at margin γ then $C_L = \frac{1}{\gamma}$. The $0-1$ loss function and the loss function $L(x) = \mathbb{1}_{x<\gamma}$ (implicitly used in Definition 4.1 and Theorem 4.2) are not Lipschitz and hence this proof technique does not apply to them.

4.2.3 Selecting informative landmarks

Recall that the generalization guarantees we described in the previous section rely on random selection of landmark pairs from a fixed distribution over the domain. However, in practice, a totally random selection might require one to select a large number of landmarks, thereby leading to an inefficient classifier in terms of training as well as test times. For typical domains such as computer vision, similarity function computation is an expensive task and hence selection of a small number of landmarks should lead to a significant improvement in the test times. For this reason, we propose a landmark pair selection heuristic which we call **DSELECT** (see Algorithm 3). The heuristic generalizes naturally to multi-class problems and can also be applied to the classification model of Balcan-Blum that uses landmark singletons instead of pairs.

At the core of our heuristic is a novel notion of *diversity* among landmarks. Assuming K is a normalized similarity kernel, we call a set of points $S \subset \mathcal{X}$ *diverse* if the average inter-point similarity is small i.e $\frac{1}{|S|(|S|-1)}\sum_{x,y\in S, x\neq y} K(x,y) \ll 1$ (in case we are working with a distance kernel we would require large inter-point distances). The key observation behind **DSELECT** is that a non-diverse set of landmarks would cause all data points to receive identical embeddings and linear separation would be impossible. Small inter-landmark similarity, on the other hand would imply that the landmarks are well-spread in the domain and can capture novel patterns in the data.

Similar notions of diversity have been used in the past for ensemble classifiers (Venkataramani and Kumar, 2009) and k-NN classifiers (Chen et al., 2009a). Here we use this notion to achieve a better embedding into the landmarked space. Experimental results demonstrate that the heuristic offers significant performance improvements over random landmark selection (see Figure 4.1). One can easily extend Algorithm 3 to multi-class problems by selecting a fixed number of landmarks from each class.

Algorithm 3 DSELECT: Selecting Landmark (Pairs)

Input: A training set T, landmarking size d.

Output: A set of d landmark pairs/singletons.

$\mathcal{L} \leftarrow$ get-random-element(T), $\mathcal{P}_{\mathbf{FTUNE}} \leftarrow \emptyset$
for $j = 2$ **to** d **do**
  $z \leftarrow \arg\min_{x \in T} \sum_{x' \in \mathcal{L}} K(x, x')$.
  $\mathcal{L} \leftarrow \mathcal{L} \cup \{z\}, \quad T \leftarrow T \backslash \{z\}$
end for
for $j = 1$ **to** d **do**
  Sample z_1, z_2, s.t., $\ell(z_1) = 1$, $\ell(z_2) = -1$ randomly from $\mathcal{L}$ with replacement
  $\mathcal{P}_{\mathbf{FTUNE}} \leftarrow \mathcal{P}_{\mathbf{FTUNE}} \cup \{(z_1, z_2)\}$
end for
return $\mathcal{L}$ (for **BBS**), $\mathcal{P}_{\mathbf{FTUNE}}$ (for **FTUNE**)

Algorithm 4 FTUNE: Learning the Best Transfer Function

Input: A family of transfer functions $\mathcal{F}$, a similarity function K and a loss function L

Output: An optimal transfer function $f^* \in \mathcal{F}$.

Select d landmark pairs $\mathcal{P}$.
for all $f \in \mathcal{F}$ **do**
  $w_f \leftarrow \text{TRAIN}(\mathcal{P}, L)$, $L_f \leftarrow L(w_f)$
end for
$f^* \leftarrow \arg\min_{f \in \mathcal{F}} L_f$
return (f^*, w_{f^*}).

4.3 Empirical results

In this section, we empirically study the performance of our proposed methods on a variety of benchmark datasets. We refer to the algorithmic formulation presented in Balcan and Blum (2006) as **BBS** and its augmentation using **DSELECT** as **BBS+D**. We refer to the formulation presented in Wang et al. (2007) as **DBOOST**. We refer to our transfer function learning based formulation as **FTUNE** and its augmentation using **DSELECT** as **FTUNE+D**. In multi-class classification scenarios we will use a *one-vs-all* formulation which presents us with an opportunity to further exploit the transfer function by learning separate transfer function per class (i.e. per one-vs-all problem). We shall refer to our formulation using a single (resp. multiple) transfer function as **FTUNE+D**-S (resp. **FTUNE+D**-M). We take the class of *ramp* functions indexed by a slope parameter as our set of transfer functions. We use 6 different values of the slope parameter $\{1, 5, 10, 50, 100, 1000\}$. Note that these functions (approximately) include both the identity function (used by Balcan and Blum (2006)) and the sign function (used by Wang

et al. (2007)).

Our goal in this section is two-fold: 1) to show that our **FTUNE** method is able to learn a more suitable transfer function for the underlying data than the existing methods **BBS** and **DBOOST** and 2) to show that our diversity based heuristic for landmark selection performs better than random selection. To this end, we perform experiments on a few benchmark datasets for learning with similarity (non-PSD) functions (Chen et al., 2009a) as well as on a variety of standard UCI datasets where the similarity function used is the Gaussian kernel function.

For our experiments, we implemented our methods **FTUNE** and **FTUNE+D** as well as **BBS** and **BBS+D** using MATLAB while using LIBLINEAR (Fan et al., 2008) for SVM classification. For **DBOOST**, we use the C++ code provided by Wang et al.. On all the datasets we randomly selected a fixed percentage of data for training, validation and testing. Except for **DBOOST** , we selected the SVM penalty constant C from the set $\{1, 10, 100, 1000\}$ using validation. For each method and dataset, we report classification accuracies averaged over 20 runs. We compare accuracies obtained by different methods using t-test at 95% significance level.

4.3.1 Similarity learning datasets

First, we conduct experiments on a few similarity learning datasets Chen et al. (2009a); these datasets provide a (non-PSD) similarity matrix along with class labels. For each of the datasets, we randomly select 70% of the data for training, 10% for validation and the remaining for testing purposes. We then apply our **FTUNE**-S, **FTUNE+D**-S, **BBS+D** methods along with **BBS** and **DBOOST** with varying number of landmark pairs. Note that we do not apply our **FTUNE**-M method to these datasets as it overfits heavily to these datasets as typically they are small in size.

We first compare the accuracy achieved by **FTUNE+D**-S with the existing methods. Table 4.1 compares the accuracies achieved by our **FTUNE+D**-S method with those of **BBS** and **DBOOST** over different datasets when using landmark sets of sizes 30 and 300. Numbers in brackets denote standard deviation over different runs. Note that in both the tables **FTUNE+D**-S is one of the best methods (upto 95% significance level) on all but one dataset. Furthermore, for datasets with large number of classes such as Amazon47 and FaceRec our method outperforms **BBS** and **DBOOST** by at least 20% percent. Also, note that some of the datasets have multiple bold faced methods, which means that the two sample t-test (at 95% level) rejects the hypothesis that their mean is different.

Next, we evaluate the effectiveness of our landmark selection criteria for both **BBS** and our method. Figure 4.1 shows the accuracies achieved by various methods on four different datasets with increasing number of landmarks. Note that in all the datasets, our diversity based landmark selection criteria increases the classification accuracy by around $5 - 6\%$ for small number of landmarks.

Dataset/Method	**BBS**	**DBOOST**	**FTUNE+D**-S
AmazonBinary	0.73(0.13)	0.77(0.10)	**0.84**(0.12)
AuralSonar	**0.82**(0.08)	**0.81**(0.08)	**0.80**(0.08)
Patrol	0.51(0.06)	0.34(0.11)	**0.58**(0.06)
Voting	**0.95**(0.03)	**0.94**(0.03)	**0.94**(0.04)
Protein	0.98(0.02)	**1.00**(0.01)	0.98(0.02)
Mirex07	0.12(0.01)	0.21(0.03)	**0.28**(0.03)
Amazon47	0.39(0.06)	0.07(0.04)	**0.61**(0.08)
FaceRec	0.20(0.04)	0.12(0.03)	**0.63**(0.04)

(a) 30 Landmarks.

Dataset/Method	**BBS**	**DBOOST**	**FTUNE+D**-S
AmazonBinary	0.78(0.11)	0.82(0.10)	**0.88**(0.07)
AuralSonar	**0.88**(0.06)	0.85(0.07)	0.85(0.07)
Patrol	**0.79**(0.05)	0.55(0.12)	**0.79**(0.07)
Voting	**0.97**(0.02)	**0.97**(0.01)	**0.97**(0.02)
Protein	**0.98**(0.02)	**0.99**(0.02)	**0.98**(0.02)
Mirex07	0.17(0.02)	0.31(0.04)	**0.35**(0.02)
Amazon47	0.40(0.13)	0.07(0.05)	**0.66**(0.07)
FaceRec	0.27(0.05)	0.19(0.03)	**0.64**(0.04)

(b) 300 Landmarks.

Table 4.1: Accuracies for Benchmark Similarity Learning Datasets for Embedding Dimensionality=30, 300. Bold numbers indicate the best performance with 95% confidence level.

4.3.2 UCI benchmark datasets

We now compare our **FTUNE** method against existing methods on a variety of UCI datasets (uci). We ran experiments with **FTUNE** and **FTUNE+D** but the latter did not provide any advantage. So for lack of space we drop it from our presentation and only show results for **FTUNE**-S (**FTUNE** with a single transfer function) and **FTUNE**-M (**FTUNE** with one transfer function per class). Similar to Wang et al. (2007), we use the Gaussian kernel function as the similarity function for evaluating our method. We set the "width" parameter in the Gaussian kernel to be the mean of all pair-wise training data distances, a standard heuristic. For all the datasets, we randomly select 50% data for training, 20% for validation and the remaining for testing. We report accuracy values averaged over 20 runs for each method with varying number of landmark pairs.

Table 4.2 compares the accuracies obtained by our **FTUNE**-S and **FTUNE**-M methods with those of **BBS** and **DBOOST** when applied to different UCI benchmark datasets. Note that **FTUNE**-S is one of the best on most of the datasets for both the landmarking sizes. Also, **BBS** performs reasonably well for small landmarking sizes while **DBOOST** performs well for large landmarking sizes. In contrast, our methods, especially **FTUNE**-S, consistently outperform the existing methods in both the scenarios.

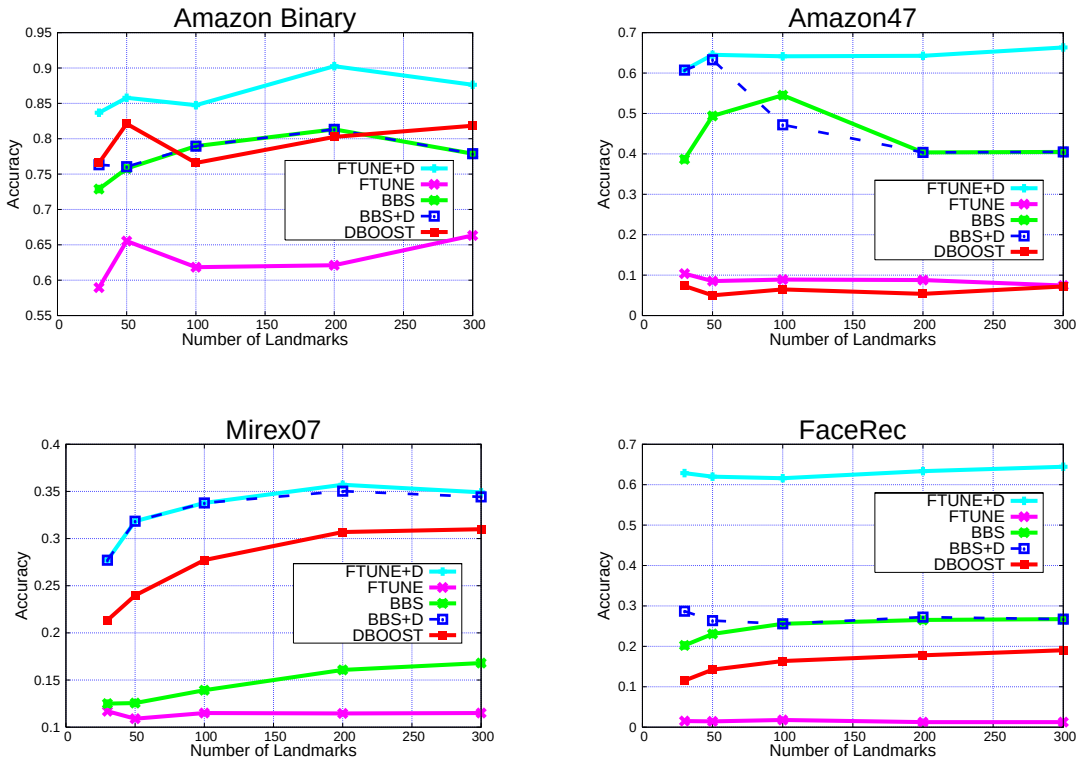


Figure 4.1: Accuracy obtained by various methods on four similarity-based learning datasets as the number of landmarks used increases.

Next, we study accuracies obtained by our method for different landmarking sizes. Figure 4.2 shows accuracies obtained by various methods as the number of landmarks selected increases. Note that the accuracy curve of our method dominates the accuracy curves of all the other methods, i.e. our method is consistently better than the existing methods for all the landmarking sizes considered. Also note that both **FTUNE**-S and **FTUNE**-M perform significantly better than **BBS** and **DBOOST** for small number of landmarks $(30, 50)$.

4.3.3 Discussion

We note that since **FTUNE** selects its output by way of validation, it is susceptible to over-fitting on small datasets but at the same time, capable of giving performance boosts on large ones. We observe a similar trend in our experiments – on smaller datasets (such as those in Table 4.1 with average dataset size 660), **FTUNE** over-fits and performs worse than **BBS** and **DBOOST**. However, even in these cases, **DSELECT** (intuitively) removes redundancies in the landmark points thus allowing **FTUNE** to recover the best transfer function. In contrast, for larger datasets like those in Table 4.2 (average size 13200), **FTUNE** is itself able to recover better transfer functions than the baseline methods and hence both **FTUNE**-S and **FTUNE**-M perform significantly better than the

Dataset/Method	**BBS**	**DBOOST**	**FTUNE**-S	**FTUNE**-M
Cod-rna	0.93(0.01)	0.89(0.01)	**0.93**(0.01)	**0.93**(0.01)
Isolet	0.81(0.01)	0.67(0.01)	**0.84**(0.01)	0.83(0.01)
Letters	0.67(0.02)	0.58(0.01)	**0.69**(0.01)	0.68(0.02)
Magic	0.82(0.01)	0.81(0.01)	**0.84**(0.01)	0.84(0.01)
Pen-digits	0.94(0.01)	0.93(0.01)	**0.97**(0.01)	**0.97**(0.00)
Nursery	**0.91**(0.01)	**0.91**(0.01)	**0.90**(0.01)	0.90(0.00)
Faults	**0.70**(0.01)	0.68(0.02)	**0.70**(0.02)	**0.71**(0.02)
Mfeat-pixel	**0.94**(0.01)	0.91(0.01)	**0.95**(0.01)	**0.94**(0.01)
Mfeat-zernike	**0.79**(0.02)	0.72(0.02)	**0.79**(0.02)	**0.79**(0.02)
Opt-digits	0.92(0.01)	0.89(0.01)	**0.94**(0.01)	**0.94**(0.01)
Satellite	0.85(0.01)	0.86(0.01)	0.86(0.01)	**0.87**(0.01)
Segment	0.90(0.01)	**0.93**(0.01)	0.92(0.01)	**0.92**(0.01)

(a) 30 Landmarks.

Dataset/Method	**BBS**	**DBOOST**	**FTUNE**-S	**FTUNE**-M
Cod-rna	**0.94**(0.00)	0.93(0.00)	**0.94**(0.00)	**0.94**(0.00)
Isolet	0.91(0.01)	0.89(0.01)	**0.93**(0.01)	**0.93**(0.00)
Letters	0.72(0.01)	**0.84**(0.01)	0.83(0.01)	0.83(0.01)
Magic	0.84(0.01)	0.84(0.00)	**0.85**(0.01)	**0.85**(0.01)
Pen-digits	0.96(0.00)	**0.99**(0.00)	**0.99**(0.00)	**0.99**(0.00)
Nursery	0.93(0.01)	**0.97**(0.00)	0.96(0.00)	**0.97**(0.00)
Faults	0.72(0.02)	**0.74**(0.02)	0.73(0.02)	0.73(0.02)
Mfeat-pixel	0.96(0.01)	**0.97**(0.01)	**0.97**(0.01)	**0.97**(0.01)
Mfeat-zernike	0.81(0.01)	0.79(0.01)	**0.82**(0.02)	**0.82**(0.01)
Opt-digits	0.95(0.01)	0.97(0.00)	**0.98**(0.00)	**0.98**(0.00)
Satellite	0.85(0.01)	**0.90**(0.01)	0.89(0.01)	**0.89**(0.01)
Segment	0.90(0.01)	**0.96**(0.01)	0.96(0.01)	0.96(0.01)

(b) 300 Landmarks.

Table 4.2: Accuracies for Gaussian Kernel on UCI Datasets for Embedding Dimensionality=30, 300. Bold numbers indicate the best performance with 95% confidence level.

baselines. Note that **DSELECT** is not able to provide any advantage here since the datasets sizes being large, greedy selection actually ends up hurting the accuracy.

4.4 Proofs

We give missing proofs below.

4.4.1 Proof of Theorem 4.2

We shall prove that with probability at least $1-\delta$, at least a $1-\varepsilon_1$ fraction of points x that satisfy Equation 4.3 are classified correctly by the classifier $h(x)$. Overestimating the error by treating the points that do not satisfy Equation 4.3 as always being misclassified

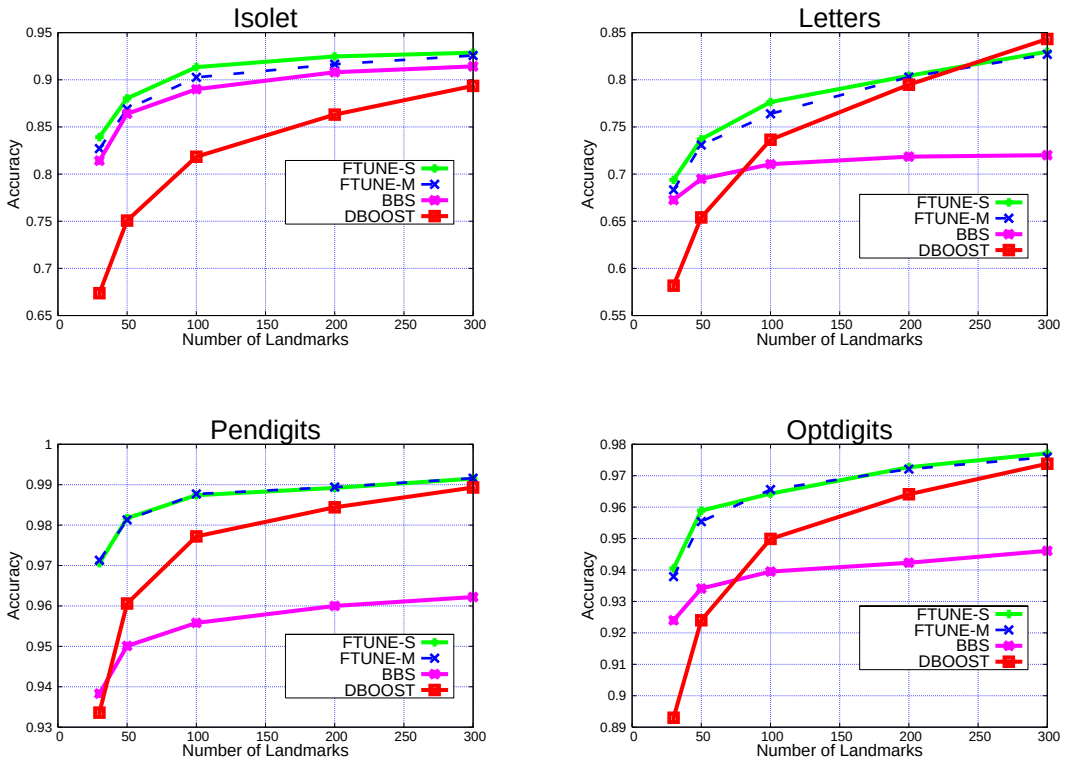


Figure 4.2: Accuracy achieved by various methods on four different UCI repository datasets as the number of landmarks used increases.

will give us the desired result.

For any fixed $x \in \mathcal{X}^+$ that satisfies Equation 4.3, we have

$$\mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}} \left[\!\left[w(x',x'') f\left(K(x,x') - K(x,x'')\right) | \ell(x') = 1, \ell(x'') = -1 \right]\!\right] \geq C_f \gamma$$

hence the Hoeffding Bounds give us

$$\mathbb{P}\left[g(x) < \frac{\gamma}{2}\right] = \mathbb{P}\left[\frac{1}{d}\sum_{i=1}^{d} w(x_i^+, x_i^-) f\left(K(x,x_i^+) - K(x,x_i^-)\right) < \frac{\gamma}{2}\right] \leq 2\exp\left(-\frac{\gamma^2 d}{8}\right)$$

Similarly, for any fixed $x \in \mathcal{X}^-$ that satisfies Equation 4.3, we have

$$\mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}} \left[\!\left[w(x',x'') f\left(K(x,x') - K(x,x'')\right) | \ell(x') = -1, \ell(x'') = 1 \right]\!\right] \geq C_f \gamma$$

hence the Hoeffding Bounds give us

$$\mathbb{P}\left[g(x) > \frac{\gamma}{2}\right] = \mathbb{P}\left[\frac{1}{d}\sum_{i=1}^{d} w(x_i^+, x_i^-) f\left(K(x,x_i^+) - K(x,x_i^-)\right) > \frac{\gamma}{2}\right]$$

$$= \mathbb{P}\left[\frac{1}{d}\sum_{i=1}^{d} w(x_i^+, x_i^-) f\left(K(x, x_i^-) - K(x, x_i^+)\right) < \frac{\gamma}{2}\right] \leq 2\exp\left(-\frac{\gamma^2 d}{8}\right)$$

where in the second step we have used antisymmetry of f.

Since we have shown that this result holds true individually for any point x that satisfies Equation 4.3, the expected error (where the expectation is both over the choice of domain points as well as choice of the landmark points) itself turns out to be less than $2\exp\left(-\frac{\gamma^2 d}{8}\right) \leq \varepsilon_1\delta$. Applying Markov's inequality gives us that the probability of obtaining a set of landmarks such that the error on points satisfying Equation 4.3 is greater than ε_1 is at most δ.

Assuming, as mentioned earlier, that the points not satisfying Equation 4.3 can always be misclassified proves our desired result.

4.4.2 Proof of Lemma 4.4

We prove the results in order,

1. Immediate from the definition of $w_{(G,f)}$.

2. Immediate from the definition of $w_{(g,f)}$.

3. We have $\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f',w_{(G,f')})}(x)\right)\right]\!\right] \leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f',w_{(G,f)})}(x)\right)\right]\!\right]$ by an application of Lemma 4.4.1 proven above. For sake of simplicity let us denote $w_{(G,f)} = w$ for the next set of calculations. Now we have

$$\begin{aligned}
G_{(f',w)}(x) &= \mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}}\left[\!\left[w\left(x',x''\right) f'\left(K(x,x') - K(x,x'')\right) | \ell(x') = \ell(x), \ell(x'') \neq \ell(x)\right]\!\right] \\
&\leq \mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}}\left[\!\left[w\left(x',x''\right)\left(f\left(K(x,x') - K(x,x'')\right) + r\right) | \ell(x') = \ell(x), \ell(x'') \neq \ell(x)\right]\!\right] \\
&= \mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}}\left[\!\left[w\left(x',x''\right) f\left(K(x,x') - K(x,x'')\right) | \ell(x') = \ell(x), \ell(x'') \neq \ell(x)\right]\!\right] \\
&\quad + \; r \cdot \mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}}\left[\!\left[w\left(x',x''\right) | \ell(x') = \ell(x), \ell(x'') \neq \ell(x)\right]\!\right] \\
&\leq G_{(f,w)}(x) + rB
\end{aligned}$$

where in the second inequality we have used the fact that $\|f - f'\|_\infty \leq r$ and in the fourth inequality we have used the fact that $w \in \mathcal{W}$. Thus we have $G_{(f',w)}(x) \leq G_{(f,w)}(x) + rB$. Using the Lipschitz properties of L we can now get

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f',w)}(x)\right)\right]\!\right] \leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w)}(x)\right)\right]\!\right] + C_L r B.$$

Thus we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f',w_{(G,f')})}(x)\right)\right]\!\right] &\leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f',w_{(G,f)})}(x)\right)\right]\!\right] \\
&\leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w_{(G,f)})}(x)\right)\right]\!\right] + C_L r B.
\end{aligned}$$

Similarly we can also prove

$$\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w_{(G,f)})}(x)\right)\right]\!\right] \leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f',w_{(G,f')})}(x)\right)\right]\!\right] + C_L r B.$$

This gives us the desired result.

4. The proof follows in a manner similar to the one for Lemma 4.4.3 proven above.

4.4.3 Proof of Lemma 4.5

Theorem 4.7 proven below guarantees that for any fixed $f \in \mathcal{F}$, with probability $1-\delta$ that $\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(g_{(f,w_{(G,f)})}(x)\right)\right]\!\right] < \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w_{(G,f)})}(x)\right)\right]\!\right] + \varepsilon_1/2$. This can be achieved with $d = (64B^2C_L^2/\varepsilon_1^2)\ln(8B/\delta\varepsilon_1)$. Now assuming that the above holds, using the above results we can get the following for any $f' \in \mathcal{B}_\infty(f,r)\cap\mathcal{F}$.

$$\begin{aligned}
\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(g_{(f',w_{(g,f')})}(x)\right)\right]\!\right] &\leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(g_{(f,w_{(g,f)})}(x)\right)\right]\!\right] + C_L r B \\
&\quad \text{(using Lemma 4.4.4)} \\
&\leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(g_{(f,w_{(G,f)})}(x)\right)\right]\!\right] + C_L r B \\
&\quad \text{(using Lemma 4.4.2)} \\
&\leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w_{(G,f)})}(x)\right)\right]\!\right] + \varepsilon_1/2 + C_L r B \\
&\quad \text{(using Theorem 4.7)} \\
&\leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f',w_{(G,f')})}(x)\right)\right]\!\right] + \varepsilon_1/2 + 2C_L r B \\
&\quad \text{(using Lemma 4.4.3)}
\end{aligned}$$

Setting $r = \frac{\varepsilon_1}{4C_L B}$ gives us the desired result.

4.4.4 Proof of Theorem 4.3

As mentioned earlier we shall prove the theorem in two parts as follows :

1. (Part I) In this part we shall prove the following :

$$\sup_{f\in\mathcal{F}}\left[\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(g_{(f,w_{(g,f)})}(x)\right)\right]\!\right] - \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w_{(G,f)})}(x)\right)\right]\!\right]\right] \leq \varepsilon_1$$

We first set up an ε-net over $\mathcal{F}$ at scale $r = \frac{\varepsilon_1}{4C_L B}$. Let there be $\mathcal{N}(\mathcal{F},r)$ elements in this net. Taking $d = (64B^2C_L^2/\varepsilon_1^2)\ln(8B\cdot\mathcal{N}(\mathcal{F},r)/\delta\varepsilon_1)$ landmarks should ensure that the landmarks, with very high probability, are good for all functions in the net by an application of union bound. Since every function in $\mathcal{F}$ is at least r-close to some function in the net, Lemma 4.5 tells us that the same set of landmarks are, with very high probability, good for all the functions in $\mathcal{F}$. This proves the first part of our result.

2. (Part II) In this part we shall prove the following :

$$\sup_{f\in\mathcal{F}}\left[\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w_{(G,f)})}(x)\right)\right]\!\right]-\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(g_{(f,w_{(g,f)})}(x)\right)\right]\!\right]\right]\leq\varepsilon_1$$

This part is actually fairly simple to prove. Intuitively, since one can imagine G as being the output of an algorithm that is allowed to take the entire domain as its landmark set, we should expect $\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w_{(G,f)})}(x)\right)\right]\!\right]\leq\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(g_{(f,w_{(g,f)})}(x)\right)\right]\!\right]$ to hold unconditionally for every f. For a formal argument, let us build up some more notation. As we have said before, for any transfer function f and arbitrary choice of d landmark pairs $\mathcal{P}$, we let $w_{(g,f)}\in[-B,B]^d$ be the best weighing function for this choice of transfer function and landmark pairs. Now let $\overline{w_{(g,f)}}$ be the best possible extension of $w_{(g,f)}$ to the entire domain. More formally, for any $w^*\in[-B,B]^d$ let $\overline{w^*}=\mathop{\arg\min}_{w\in\mathcal{W},w_{|\mathcal{P}}=w^*}\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w)}(x)\right)\right]\!\right]$.

Now Lemma 4.4.1 tells us that for any $f\in\mathcal{F}$ and any choice of landmark pairs $\mathcal{P}$, $\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,w_{(G,f)})}(x)\right)\right]\!\right]\leq\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,\overline{w_{(g,f)}})}(x)\right)\right]\!\right]$. Furthermore, since $\overline{w_{(g,f)}}$ is chosen to be the most beneficial extension of $w_{(g,f)}$, we also have $\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(G_{(f,\overline{w_{(g,f)}})}(x)\right)\right]\!\right]\leq\mathop{\mathbb{E}}_{x\sim\mathcal{D}}\left[\!\left[L\left(g_{(f,w_{(g,f)})}(x)\right)\right]\!\right]$. Together, these two inequalities give us the second part of the proof.

4.4.5 Proof of Theorem 4.7

For any $x\in\mathcal{X}$, we have, by an application of Hoeffding bounds

$$\mathop{\mathbb{P}}_{g}\left[|G(x)-g(x)|>\varepsilon_1\right]<2\exp\left(-\frac{\varepsilon_1^2 d}{2B^2}\right)$$

since $|g(x)|\leq B$. Here the notation $\mathop{\mathbb{P}}_{g}[\cdot]$ signifies that the probability is over the choice of the landmark points. Thus for $d>\frac{4B^2}{\varepsilon_1^2}\ln\left(\frac{2}{\delta}\right)$, we have

$$\mathop{\mathbb{P}}_{g}\left[|G(x)-g(x)|>\varepsilon_1\right]<\delta^2.$$

For sake of simplicity let us denote by $\mathsf{BAD}(x)$ the event $|G(x)-g(x)|>\varepsilon_1$. Thus we have, for every $x\in\mathcal{X}$, $\mathop{\mathbb{E}}_{g}\left[\!\left[\mathbb{1}_{\mathsf{BAD}(x)}\right]\!\right]<\delta^2$. Since this is true for every $x\in\mathcal{X}$, this also holds in expectation i.e. $\mathop{\mathbb{E}}_{x}\mathop{\mathbb{E}}_{g}\left[\!\left[\mathbb{1}_{\mathsf{BAD}(x)}\right]\!\right]<\delta^2$. The expectation over x is with respect to the problem distribution $\mathcal{D}$. Applying Fubini's Theorem gives us $\mathop{\mathbb{E}}_{g}\mathop{\mathbb{E}}_{x}\left[\!\left[\mathbb{1}_{\mathsf{BAD}(x)}\right]\!\right]<\delta^2$ which upon application of Markov's inequality gives us $\mathop{\mathbb{P}}_{g}\left[\mathop{\mathbb{E}}_{x}\left[\!\left[\mathbb{1}_{\mathsf{BAD}(x)}\right]\!\right]>\delta\right]<\delta$. Thus, with very high probability we would always choose landmarks such that $\mathop{\mathbb{P}}_{x}[BAD(x)]<\delta$. Thus we have, in such a situation,

$$\mathop{\mathbb{E}}_{x}\left[\!\left[|G(x)-g(x)|\right]\!\right]\leq(1-\delta)\varepsilon_1+\delta\cdot 2B$$

since $\sup_{x\in\mathcal{X}} |G(x) - g(x)| \leq 2B$. For small enough δ we have $\mathop{\mathbb{E}}_{x}[\![|G(x) - g(x)|]\!] \leq 2\varepsilon_1$.

Thus we have

$$\begin{aligned}\mathop{\mathbb{E}}_{x}[\![L(g(x))]\!] - \mathop{\mathbb{E}}_{x}[\![L(G(x))]\!] &= \mathop{\mathbb{E}}_{x}[\![L(g(x)) - L(G(x))]\!] \\ &\leq \mathop{\mathbb{E}}_{x}[\![C_L \cdot |g(x) - G(x)|]\!] \\ &= C_L \cdot \mathop{\mathbb{E}}_{x}[\![|g(x) - G(x)|]\!] \leq 2C_L\varepsilon_1\end{aligned}$$

where we used the Lipschitz properties of the loss function L to arrive at the second inequality. Putting $\varepsilon_1 = \frac{\varepsilon_1'}{2C_L}$ we have $\mathop{\mathbb{E}}_{x}[\![L(g(x))]\!] \leq \mathop{\mathbb{E}}_{x}[\![L(G(x))]\!] + \varepsilon_1' \leq \varepsilon + \varepsilon_1'$ which gives us our desired result.

Actually we can prove something stronger since

$$\begin{aligned}\left|\mathop{\mathbb{E}}_{x}[\![L(g(x))]\!] - \mathop{\mathbb{E}}_{x}[\![L(G(x))]\!]\right| &= \left|\mathop{\mathbb{E}}_{x}[\![L(g(x)) - L(G(x))]\!]\right| \\ &\leq \mathop{\mathbb{E}}_{x}[\![|L(g(x)) - L(G(x))|]\!] \\ &\leq \mathop{\mathbb{E}}_{x}[\![C_L \cdot |g(x) - G(x)|]\!] \leq \varepsilon_1'.\end{aligned}$$

Thus we have $\varepsilon - \varepsilon_1' \leq \mathop{\mathbb{E}}_{x}[\![L(g(x))]\!] \leq \varepsilon + \varepsilon_1'$.

Chapter 5

Supervised Learning with Indefinite Kernels

Contents

Abstract *In this chapter we extend the learning models discussed for classification in Chapter 4 to arbitrary supervised learning problems. We propose a model that is generic enough to handle any supervised learning task and also subsumes models previously proposed for classification. We give a "goodness" criterion for similarity functions w.r.t.*

a given supervised learning task and then adapt a well-known landmarking technique to provide efficient algorithms for supervised learning using "good" similarity functions. We demonstrate the effectiveness of our model on three important supervised learning problems: a) real-valued regression, b) ordinal regression and c) ranking. Furthermore, for the case of real-valued regression, we show how to obtain a sparse predictor with bounded generalization error. For this case, our model is able to demonstrate a support vector*-like effect for indefinite kernel learning. Finally, we report results of our learning algorithms on regression and ordinal regression tasks using non-PSD similarity functions and demonstrate the effectiveness of our algorithms, especially that of the sparse learning algorithm that achieves significantly higher accuracies than the baseline methods while offering reduced computational costs.*

5.1 Introduction

The goal of this chapter is to develop an extended framework for supervised learning with similarity functions. As we have noted earlier, kernel learning algorithms (Schölkopf and Smola, 2002) have become the mainstay of discriminative learning with an incredible amount of effort having been put in, both from the theoretician's as well as the practitioner's side. Kernel learning has found application in several learning tasks such as classification, regression, ranking, clustering and principal component analysis (Schölkopf and Smola, 2002). There exist several theoretical results providing generalization guarantees on the performance of kernel learning algorithms as well several *off-the-shelf* toolkits available for various kernel learning problems.

However, existing algorithms for most of these tasks require the kernel function to be positive semi-definite (PSD), which can be a limiting factor for several applications. Reasons being: 1) the Mercer's condition is a formal statement that is hard to verify, 2) several natural notions of similarity that arise in practical scenarios are not PSD, and 3) it is not clear as to why an artificial constraint like PSD-ness should limit the usability of a kernel.

Several recent papers have demonstrated that indefinite similarity functions can indeed be successfully used for learning (Haasdonk, 2005; Ong et al., 2004; Chen et al., 2009b; Luss and d'Aspremont, 2007). However, most of the existing work focuses on classification tasks and provides specialized techniques for the same, albeit with little or no theoretical guarantees. A notable exception is the line of work by Balcan and Blum (2006); Wang et al. (2007); Kar and Jain (2011) that defines a goodness criterion for a similarity function and then provides an algorithm that can exploit this goodness criterion to obtain provably accurate classifiers. However, their definitions are yet again restricted to the problem of classification as they take a "margin" based view of the problem that requires positive points to be more similar to positive points than to negative points by at least a constant margin.

In this work, we instead take a "target-value" point of view and require that target

values of similar points be similar. Using this view, we propose a generic goodness definition that also admits the goodness definition of Balcan and Blum (2006) for classification as a special case. Furthermore, our definition can be seen as imposing the existence of a smooth function over a generic space defined by similarity functions, rather than over a Hilbert space as required by typical goodness definitions of PSD kernels.

We then adapt the landmarking technique of Balcan and Blum (2006) to provide an efficient algorithm that reduces learning tasks to corresponding learning problems over a linear space. The main technical challenge at this stage is to show that such reductions are able to provide good generalization error bounds for the learning tasks at hand. To this end, we consider three specific problems: a) regression, b) ordinal regression, and c) ranking. For each problem, we define appropriate surrogate loss functions, and show that our algorithm is able to, for each specific learning task, guarantee bounded generalization error with polynomial sample complexity. Moreover, by adapting a general framework given by Srebro (2007), we show that these guarantees do not require the goodness definition to be overly restrictive by showing that our definitions admit all good PSD kernels as well.

For the problem of real-valued regression, we additionally provide a goodness definition that captures the intuition that usually, only a small number of landmarks are influential w.r.t. the learning task. However, to recover these landmarks, the uniform sampling technique would require sampling a large number of landmarks thus increasing the training/test time of the predictor. We address this issue by applying a sparse vector recovery algorithm given by Shalev-Shwartz et al. (2010b) and show that the resulting sparse predictor still has bounded generalization error.

This allows us to present a model for indefinite kernel learning that is able to provably demonstrate a support vector effect. It is notable that previous work in this domain (Balcan and Blum, 2006; Wang et al., 2007; Kar and Jain, 2011) has only been able to present "dense" models. Our sparse learning formulations result in hypotheses that offer accelerated prediction times, a very desirable quality.

We also address an important issue faced by algorithms that use landmarking as a feature construction step viz (Balcan and Blum, 2006; Wang et al., 2007; Kar and Jain, 2011), namely that they typically assume separate landmark and training sets for ease of analysis. In practice however, one usually tries to overcome paucity of training data by reusing training data as landmark points as well. We use an argument outlined by Ben-David et al. (2008) to theoretically justify such "double dipping" in our case. The details of the argument are given in Appendix B.

We perform several experiments on benchmark datasets that demonstrate significant performance gains for our methods over the baseline of kernel regression. Our sparse landmark selection technique provides significantly better predictors that are also more efficient at test time.

5.1.1 Related Work

We briefly recapitulate some related work for sake of completeness. Existing approaches to extend kernel learning algorithms to indefinite kernels can be classified into three broad categories: a) those that use indefinite kernels directly with existing kernel learning algorithms, resulting in non-convex formulations (Haasdonk, 2005; Ong et al., 2004). b) those that convert a given indefinite kernel into a PSD one by either projecting onto the PSD-cone (Chen et al., 2009b; Luss and d'Aspremont, 2007) or performing other spectral operations (Chen et al., 2009a). The second approach is usually expensive due to the spectral operations involved apart from making the method inherently transductive. Moreover, any domain knowledge stored in the original kernel is lost due to these task oblivious operations and consequently, no generalization guarantees can be given. c) those that use notions of "task-kernel alignment" or equivalently, notions of "goodness" of a kernel, to give learning algorithms (Balcan and Blum, 2006; Wang et al., 2007; Kar and Jain, 2011). This approach enjoys several advantages over the other approaches listed above. These models are able to use the indefinite kernel directly with existing PSD kernel learning techniques; all the while retaining the ability to give generalization bounds that quantitatively parallel those of PSD kernel learning models. In this chapter, we adopt the third approach for general supervised learning problem.

5.2 Problem formulation and Preliminaries

The goal in similarity-based supervised learning is to closely approximate a *target* predictor $y : \mathcal{X} \to \mathcal{Y}$ over some domain $\mathcal{X}$ using a *hypothesis* $\hat{f}(\,\cdot\,; K) : \mathcal{X} \to \mathcal{Y}$ that restricts its interaction with data points to computing similarity values given by K. Now, if the similarity function K is not discriminative enough for the given task then we cannot hope to construct a predictor out of it that enjoys good generalization properties. Hence, it is natural to define the "goodness" of a given similarity function with respect to the learning task at hand.

Definition 5.1 (Good similarity function: preliminary)**.** *Given a learning task $y : \mathcal{X} \to \mathcal{Y}$ over some distribution $\mathcal{D}$, a similarity function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is said to be (ε_0, B)-good with respect to this task if there exists some bounded weighing function $w : \mathcal{X} \to [-B, B]$ such that for at least a $(1 - \varepsilon_0)$ $\mathcal{D}$-fraction of the domain, we have* $y(\mathbf{x}) = \mathop{\mathbb{E}}_{\mathbf{x}' \sim \mathcal{D}} [\![w(\mathbf{x}') y(\mathbf{x}') K(\mathbf{x}, \mathbf{x}')]\!]$.

The above definition is inspired by the definition of a "good" similarity function with respect to classification tasks given by Balcan and Blum (2006). However, their definition is tied to class labels and thus applies only to classification tasks. Similar to Balcan and Blum, the above definition calls a similarity function K "good" if the target value $y(\mathbf{x})$ of a given point $\mathbf{x}$ can be approximated in terms of (a weighted combination of) the target values of the K-"neighbors" of $\mathbf{x}$. Also, note that this definition automatically enforces a smoothness prior on the framework.

However the above definition is too rigid. Moreover, it defines goodness in terms of violations, a non-convex loss function. To remedy this, we propose an alternative definition that incorporates an arbitrary (but in practice always convex) loss function.

Definition 5.2 (Good similarity function: final)**.** *Given a learning task $y : \mathcal{X} \to \mathcal{Y}$ over some distribution $\mathcal{D}$, a similarity function K is said to be (ε_0, B)-good with respect to a loss function $\ell_S : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}$ if there exists some bounded weighing function $w : \mathcal{X} \to [-B, B]$ such that if we define a predictor as $f(\mathbf{x}) := \mathop{\mathbb{E}}_{\mathbf{x}' \sim \mathcal{D}} [\![w(\mathbf{x}') K(\mathbf{x}, \mathbf{x}')]\!]$, then we have $\mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} [\![\ell_S(f(\mathbf{x}), y(\mathbf{x}))]\!] \leq \varepsilon_0$.*

Note that Definition 5.2 reduces to Definition 5.1 for $\ell_S(a, b) = \mathbb{1}_{\{a \neq b\}}$. Moreover, for the case of binary classification where $y \in \{-1, +1\}$, if we take $\ell_S(a, b) = \mathbb{1}_{\{ab \leq B\gamma\}}$, then we recover the (ε_0, γ)-goodness definition of a similarity function, given in Definition 3 of Balcan and Blum. Also note that, assuming $\sup_{\mathbf{x} \in \mathcal{X}} \{|y(\mathbf{x})|\} < \infty$ we can w.l.o.g. merge $w(\mathbf{x}')y(\mathbf{x}')$ into a single term $w(\mathbf{x}')$.

Having given this definition we must make sure that "good" similarity functions allow the construction of effective predictors (Utility property). Moreover, we must make sure that the definition does not exclude commonly used PSD kernels (Admissibility property). Below, we formally define these two properties and in later sections, show that for each of the learning tasks considered, our goodness definition satisfies these two properties.

5.2.1 Utility

Definition 5.3 (Utility)**.** *A similarity function K is said to be ε_0-useful w.r.t. a loss function $\ell_{actual}(\cdot, \cdot)$ if the following holds: there exists a learning algorithm $\mathcal{A}$ that, for any $\varepsilon_1, \delta > 0$, when given poly$(1/\varepsilon_1, \log(1/\delta))$ "labeled" and "unlabeled" samples from the input distribution $\mathcal{D}$, with probability at least $1 - \delta$, generates a hypothesis $\hat{f}(\mathbf{x}; K)$ s.t. $\mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\ell_{actual}\left(\hat{f}(\mathbf{x}), y(\mathbf{x})\right) \right]\!\!\right] \leq \varepsilon_0 + \varepsilon_1$. Note that $\hat{f}(\mathbf{x}; K)$ is restricted to access the data solely through K.*

Here, the ε_0 term captures the *misfit* or the bias of the similarity function with respect to the learning problem. Notice that the above utility definition allows for learning from unlabeled data points and thus puts our approach in the semi-supervised learning framework.

All our utility guarantees proceed by first using unlabeled samples as *landmarks* to construct a landmarked space. Next, using the goodness definition, we show the existence of a good linear predictor in the landmarked space. This guarantee is obtained in two steps as outlined in Algorithm 5: first of all we choose d unlabeled landmark points and construct a map $\Psi : \mathcal{X} \to \mathbb{R}^d$ (see Step 1 of Algorithm 5) and show that there exists a linear predictor over $\mathbb{R}^d$ that closely approximates the predictor f used in Definition 5.2 (see Lemma 5.22 in Section 5.6). In the second step, we learn a predictor (over the landmarked space) using ERM over a fresh labeled training set (see Step 3 of Algorithm 5). We then use

Algorithm 5 Supervised learning with Similarity functions

Input: A target predictor $y : \mathcal{X} \to \mathcal{Y}$ over a distribution $\mathcal{D}$, an (ε_0, B)-good similarity function K, labeled training points sampled from $\mathcal{D}$: $\mathcal{T} = \left\{(\mathbf{x}_1^t, y_1), \ldots, (\mathbf{x}_n^t, y_n)\right\}$, loss function $\ell_S : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}^+$.

Output: A predictor $\hat{f} : \mathcal{X} \to \mathbb{R}$ with bounded true loss over $\mathcal{D}$

1: Sample d unlabeled landmarks from $\mathcal{D}$: $\mathcal{L} = \left\{\mathbf{x}_1^l, \ldots, \mathbf{x}_d^l\right\}$
`// Else subsample` d `landmarks from` $\mathcal{T}$ `(see B for details)`
2: $\Psi_{\mathcal{L}} : \mathbf{x} \mapsto \frac{1}{\sqrt{d}} \left(K(\mathbf{x}, \mathbf{x}_1^l), \ldots, K(\mathbf{x}, \mathbf{x}_d^l)\right) \in \mathbb{R}^d$
3: $\hat{\mathbf{w}} = \underset{\mathbf{w} \in \mathbb{R}^d : \|\mathbf{w}\|_2 \leq B}{\arg\min} \sum_{i=1}^n \ell_S \left(\left\langle \mathbf{w}, \Psi_{\mathcal{L}}(\mathbf{x}_i^t) \right\rangle, y_i\right)$
4: **return** $\hat{f} : \mathbf{x} \mapsto \langle \hat{\mathbf{w}}, \Psi_{\mathcal{L}}(\mathbf{x}) \rangle$

individual task-specific arguments and Rademacher average-based generalization bounds (Kakade et al., 2008) thus proving the utility of the similarity function.

5.2.2 Admissibility

In order to show that our models are not too rigid, we prove that they admit good PSD kernels.

The notion of a good PSD kernel for us will be one that corresponds to a prevalent large margin technique for the given problem. In general, most notions correspond to the existence of a linear operator in the RKHS of the kernel that has small loss at large margin. More formally,

Definition 5.4 (Good PSD Kernel)**.** *Given a learning task $y : \mathcal{X} \to \mathcal{Y}$ over some distribution $\mathcal{D}$, a PSD kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ with associated RKHS $\mathcal{H}_K$ and canonical feature map $\Phi_K : \mathcal{X} \to \mathcal{H}_K$ is said to be (ε_0, γ)-good with respect to a loss function $\ell_K : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}$ if there exists $\mathbf{W}^* \in \mathcal{H}_K$ such that $\|\mathbf{W}^*\| = 1$ and*

$$\mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\ell_K \left(\frac{\langle \mathbf{W}^*, \Phi_K(\mathbf{x}) \rangle}{\gamma}, y(\mathbf{x}) \right) \right]\!\!\right] < \varepsilon_0$$

We will show, for all the learning tasks considered, that every (ε_0, γ)-good PSD kernel, when treated as simply a similarity function with no consideration of its RKHS, is also $(\varepsilon + \varepsilon_1, B)$-good for arbitrarily small ε_1 with $B = h(\gamma, \varepsilon_1)$ for some function h. To prove these results we will adapt techniques introduced by Srebro (2007) with certain modifications and task-dependent arguments.

5.3 Applications

We will now instantiate the general learning model described above to real-valued regression, ordinal regression and ranking by providing utility and admissibility guarantees. To improve clarity, we relegate all proofs to a dedicated section presented later in the chapter (Section 5.7).

5.3.1 Real-valued Regression

Real-valued regression is a quintessential learning problem (Schölkopf and Smola, 2002) that has received a lot of attention in the learning literature. In the following we shall present algorithms for performing real-valued regression using non-PSD similarity measures. We consider the problem with $\ell_{\text{actual}}(a, b) = |a - b|$ as the true loss function. For the surrogates ℓ_S and ℓ_K, we choose the ε-insensitive loss function (Schölkopf and Smola, 2002) defined as follows:

$$\ell_\varepsilon(a, b) = \ell_\varepsilon(a - b) = \begin{cases} 0, & \text{if } |a - b| < \varepsilon, \\ |a - b| - \varepsilon, & \text{otherwise.} \end{cases}$$

The above loss function automatically gives us notions of good kernels and similarity functions by appealing to Definitions 5.4 and 5.2 respectively. It is easy to transfer error bounds in terms of absolute error to those in terms of mean squared error (MSE), a commonly used performance measure for real-valued regression.

Using the landmarking strategy described in Section 5.2.1, we can reduce the problem of real regression to that of a linear regression problem in the landmarked space. More specifically, the ERM step in Algorithm 5 becomes the following:

$$\operatorname*{arg\,min}_{\mathbf{w} \in \mathbb{R}^d : \|\mathbf{w}\|_2 \leq B} \sum_i^n \ell_\varepsilon\left(\langle \mathbf{w}, \Psi_{\mathcal{L}}(\mathbf{x}_i) \rangle - y_i\right).$$

There exist solvers (for instance (Ho and Lin, 2012)) to efficiently solve the above problem on linear spaces. Using proof techniques sketched in Section 5.2.1 along with specific arguments for the ε-insensitive loss, we can prove generalization guarantees and hence utility guarantees for the similarity function.

Theorem 5.5. *Every similarity function that is (ε_0, B)-good for a regression problem with respect to the insensitive loss function $\ell_\varepsilon(\cdot, \cdot)$ is $(\varepsilon_0 + \varepsilon)$-useful with respect to absolute loss as well as $(B\varepsilon_0 + B\varepsilon)$-useful with respect to mean squared error. Moreover, both the dimensionality of the landmarked space as well as the labeled sample complexity can be bounded by $\mathcal{O}\left(\frac{B^2}{\varepsilon_1^2} \log \frac{1}{\delta}\right)$.*

We are also able to prove the following (tight) admissibility result:

Theorem 5.6. *Every PSD kernel that is (ε_0, γ)-good for a regression problem is, for any $\varepsilon_1 > 0$, $\left(\varepsilon_0 + \varepsilon_1, \mathcal{O}\left(\frac{1}{\varepsilon_1 \gamma^2}\right)\right)$-good as a similarity function as well. Moreover, for any $\varepsilon_1 < 1/2$ and any $\gamma < 1$, there exists a regression instance and a corresponding kernel that is $(0, \gamma)$-good for the regression problem but only (ε_1, B)-good as a similarity function for $B = \Omega\left(\frac{1}{\varepsilon_1 \gamma^2}\right)$.*

5.3.2 Sparse Regression

An artifact of a random choice of landmarks is that very few of them might turn out to be "informative" with respect to the prediction problem at hand. For instance, in a

$$\begin{array}{ll} \max\limits_{\boldsymbol{\alpha}} & \boldsymbol{\alpha}^\top \mathbb{1} - \frac{1}{2}\sum\limits_{i,j=1}^{n} \boldsymbol{\alpha}_i \boldsymbol{\alpha}_j y_i y_j K(\mathbf{x}_i \mathbf{x}_j) \\ \text{s.t.} & 0 \leq \boldsymbol{\alpha}_i \leq \lambda \end{array} \qquad \begin{array}{ll} \min\limits_{\boldsymbol{\alpha}} & \frac{1}{2}\left\|\boldsymbol{\alpha}\right\|_2^2 + \sum\limits_{i=1}^{n} \xi_i \\ \text{s.t.} & \langle \boldsymbol{\alpha}, \Psi_{\mathcal{L}}(\mathbf{x}_i) \rangle \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{array}$$

Mercer Kernel Learning | Indefinite Kernel Learning

Figure 5.1: Comparison of Mercer and indefinite kernel learning formulations.

network, there might exist *hubs* or *authoritative* nodes that yield rich information about the learning problem. If the relative abundance of such nodes is low then random selection would compel us to choose a large number of landmarks before enough "informative" ones have been collected.

However this greatly increases training and testing times due to the increased costs of constructing the landmarked space. Thus, the ability to prune away irrelevant landmarks would speed up training and test routines. To this end, we note certain similarities between PSD kernel learning and the indefinite kernel learning model presented here. Both learning models (see Section 2.2.1) propose hypotheses of the form

$$h(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i K(\mathbf{x}, \mathbf{x}_i), \ \alpha_i \in \mathbb{R}$$

However, the formulations that are used to arrive at these hypotheses have a subtle but important difference as demonstrated in Figure 5.1. Note that for simplicity, we have used the formulations for the classification problem considered by the SVM algorithm. However, the arguments we make hold true for the case of regression as well.

Note that in the Mercer kernel formulation, the combination vector $\boldsymbol{\alpha}$ is L_1 regularized and that frequently leads to sparse combinations leading to the *support vector effect*. In contrast, the indefinite kernel formulation imposes an L_2 regularization on the combination vector that does not promote sparsity. Due to this reason, this formulation frequently results in dense combinations and landmark selection heuristics (e.g diversity heuristic used in (Kar and Jain, 2011; Chen et al., 2009a)) are required to introduce sparsity.

In contrast, we present a model below that guarantees that our predictor will select a small number of landmarks while incurring bounded generalization error. This has a desirable effect of producing hypotheses with accelerated prediction times. However this requires a careful restructuring of the learning model to incorporate the "informativeness" of landmarks.

Definition 5.7. *A similarity function K is said to be (ε_0, B, τ)-good for a real-valued regression problem $y : \mathcal{X} \to \mathbb{R}$ if for some bounded weight function $w : \mathcal{X} \to [-B, B]$ and* choice function *$R : \mathcal{X} \to \{0, 1\}$ with $\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}[\![R(\mathbf{x})]\!] = \tau$, the predictor $f : \mathbf{x} \mapsto \mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}[\![w(\mathbf{x}')K(\mathbf{x}, \mathbf{x}')|R(\mathbf{x}')]\!]$ has bounded ε-insensitive loss i.e. $\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}[\![\ell_\varepsilon\left(f(\mathbf{x}), y(\mathbf{x})\right)]\!] < \varepsilon_0$.*

The role of the choice function is to single out informative landmarks, while τ specifies the relative density of informative landmarks. Note that the above definition is similar in spirit to the goodness definition presented in Balcan et al. (2008a). While the motivation behind the work of Balcan et al. was to give an improved admissibility result for binary classification, we squarely focus on the utility guarantees; with the aim of accelerating our learning algorithms via landmark pruning.

We prove the utility guarantee in three steps as outlined below. In the first step we project our learning problem, via the landmarking step given in Step 1 of Algorithm 5, to a linear landmarked space and show that the landmarked space admits a sparse linear predictor with bounded ε-insensitive loss. This is formalized in Theorem 5.8 given below.

Theorem 5.8. *Given a similarity function that is (ε_0, B, τ)-good for a regression problem, there exists a randomized map $\Psi : \mathcal{X} \to \mathbb{R}^d$ for $d = \mathcal{O}\left(\frac{B^2}{\tau\varepsilon_1^2} \log \frac{1}{\delta}\right)$ such that with probability at least $1 - \delta$, there exists a linear operator $\tilde{f} : \mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x} \rangle$ over $\mathbb{R}^d$ such that $\|\mathbf{w}\|_1 \leq B$ with ε-insensitive loss bounded by $\varepsilon_0 + \varepsilon_1$. Moreover, with the same confidence we have $\|\mathbf{w}\|_0 \leq \frac{3d\tau}{2}$.*

Our proof follows that of Balcan et al. (2008a), however we additionally prove sparsity of $\mathbf{w}$ as well. Note that the number of landmarks required here is a $\Omega(1/\tau)$ fraction greater than that required by Theorem 5.5. This formally captures the intuition presented earlier of a small fraction of dimensions (read landmarks) being actually relevant to the learning problem, thus necessitating sampling of a large number of landmarks to get enough good ones. So, in the second step, we use the *Forward Greedy Selection* algorithm given in (Shalev-Shwartz et al., 2010b) to learn a sparse predictor. The use of this learning algorithm necessitates the use of a different generalization bound in the final step to complete the utility guarantee given below. We refer the reader to Section 5.3.2.1 for the details of the algorithm and its utility analysis.

Theorem 5.9. *Every similarity function that is (ε_0, B, τ)-good for a regression problem with respect to the insensitive loss function $\ell_\varepsilon(\cdot, \cdot)$ is $(\varepsilon_0 + \varepsilon)$-useful with respect to absolute loss as well; with the dimensionality of the landmarked space being bounded by $\mathcal{O}\left(\frac{B^2}{\tau\varepsilon_1^2} \log \frac{1}{\delta}\right)$ and the labeled sampled complexity being bounded by $\mathcal{O}\left(\frac{B^2}{\varepsilon_1^2} \log \frac{B}{\varepsilon_1 \delta}\right)$. Moreover, this utility can be achieved by an $\mathcal{O}(\tau)$-sparse predictor on the landmarked space.*

We note that the improvements obtained here by using the sparse learning methods of Shalev-Shwartz et al. provide $\Omega(\tau)$ increase in sparsity.

We now prove admissibility results for this sparse learning model. We do this by showing that the dense model analyzed in Theorem 5.5 and that given in Definition 5.7 are interpretable in each other for an appropriate selection of parameters. The guarantees in Theorem 5.6 can then be invoked to conclude the admissibility proof.

Theorem 5.10. *Every (ε_0, B)-good similarity function K is also $\left(\varepsilon_0, B, \frac{\bar{w}}{B}\right)$-good where $\bar{w} = \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} [\![|w(\mathbf{x})|]\!]$. Moreover, every (ε_0, B, τ)-good similarity function K is also $(\varepsilon_0, B/\tau)$-good.*

Algorithm 6 Sparse regression (Shalev-Shwartz et al., 2010b)

Input: A β-smooth loss function $\ell(\cdot,\cdot)$, regularization parameter C_W used in Equation 5.2, error tolerance ε
Output: A sparse predictor $\hat{\mathbf{w}}$ with bounded loss

$k \leftarrow \left\lceil \frac{8C_W^2}{\varepsilon^2} \right\rceil$, $\mathbf{w}^{(0)} = \mathbf{0}$
for $t = 1$ **to** k **do**
  $\boldsymbol{\theta}^{(t)} \leftarrow \nabla_{\mathbf{w}} \mathcal{R}(\mathbf{w}^{(t)}) = \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}} \left[\!\!\left[\frac{\partial}{\partial \mathbf{w}} \ell\left(\left\langle \mathbf{w}^{(t)}, \mathbf{x} \right\rangle, y(\mathbf{x})\right) \right]\!\!\right]$
  $r_t = \arg\max_{j \in d} \left| \boldsymbol{\theta}_j^{(t)} \right|$
  $\delta_t = \left\langle \boldsymbol{\theta}^{(t)}, \mathbf{w}^{(t)} \right\rangle + C_W \left\| \boldsymbol{\theta}^{(t)} \right\|_\infty$
  $\eta_t = \min\left\{1, \frac{\delta_t}{4C_W^2 \beta}\right\}$
  $\mathbf{w}^{(t+1)} \leftarrow (1-\eta_t)\,\mathbf{w}^{(t)} + \eta_t \mathrm{sign}\left(-\boldsymbol{\theta}_{r_t}^{(t)}\right) C_W \mathbf{e}^{r_t}$
  if $\delta_t \le \varepsilon$ **then**
    return $\mathbf{w}^{(t)}$
  end if
end for
return $\mathbf{w}^{(k)}$

Using Theorem 5.6, we immediately have the following corollary:

Corollary 5.11. *Every PSD kernel that is (ε_0, γ)-good for a regression problem is, for any $\varepsilon_1 > 0$, $\left(\varepsilon_0 + \varepsilon_1, \mathcal{O}\left(\frac{1}{\varepsilon_1 \gamma^2}\right), 1\right)$-good as a similarity function as well.*

In the following, we shall see how to extract a sparse predictor in the landmarked space with good generalization properties. The following analysis shall assume the the existence of a good predictor on the landmarked space and hence all subsequent results shall be conditioned on the guarantees given by Theorem 5.8.

5.3.2.1 Learning Sparse Predictors in the Landmarked Space

We use the *Forward Greedy Selection* algorithm of Shalev-Shwartz et al. to extract a sparse predictor in the landmarked space. The algorithm is presented in pseudo code form in Algorithm 6. The algorithm can be seen as a (modified) form of orthogonal matching pursuit wherein at each step we add a coordinate to the support of the weight vector. The coordinate is added in a greedy manner so as to provide maximum incremental benefit in terms of lowering the loss. Thus the sparsity of the resulting predictor is bounded by the number of steps for which this algorithm is allowed to run. The algorithm requires that it be used with a *smooth* loss function. A loss function $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^+$ is said to be β-smooth if, for all $y, a, b \in \mathbb{R}$, we have

$$\ell(a, y) - \ell(b, y) \le \frac{\partial}{\partial x} \ell(x, y)\Big|_{x=b} (a - b) + \frac{\beta (a-b)^2}{2}$$

Unfortunately, this excludes the ε-insensitive loss commonly used for regression tasks. However it is possible to run the algorithm with a smooth surrogate whose loss can be transferred to ε-insensitive loss. Following Shalev-Shwartz et al., we choose the following

loss function:

$$\tilde{\ell}_\beta(a,b) = \inf_{v\in\mathbb{R}} \left[\frac{\beta}{2}v^2 + \ell_\varepsilon\left(a-v,b\right)\right]$$

One can, by a mildly tedious case-by-case analysis, arrive at an explicit form for this loss function

$$\tilde{\ell}_\beta(a,b) = \begin{cases} 0 & |a-b| \le \varepsilon \\ \frac{\beta}{2}\left(|a-b|-\varepsilon\right)^2 & \varepsilon < |a-b| < \varepsilon + \frac{1}{\beta} \\ |a-b| - \varepsilon - \frac{1}{2\beta} & |a-b| \ge \varepsilon + \frac{1}{\beta} \end{cases}$$

Note that this loss function is convex as well as differentiable (actually β-smooth) which will be crucial in the following analysis. Moreover, for any a, b we have

$$0 \le \ell_\varepsilon\left(a,b\right) - \tilde{\ell}_\beta(a,b) \le \frac{1}{2\beta} \tag{5.1}$$

Analysis of Forward Greedy Selection: We need to setup some notation before we can describe the guarantees given for the predictor learned using the Forward Greedy Selection algorithm. Consider a domain $\mathcal{X} \subset \mathbb{R}^d$ for some $d > 0$ and the class of functions $\mathcal{F} = \{\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x}\rangle : \|\mathbf{w}\|_1 \le C_W\}$. For any distribution $\mathcal{D}$ on $\mathcal{X}$ and any predictor from $\mathcal{F}$, define $\mathcal{R}_\mathcal{D}(\mathbf{w}) := \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}[\![\ell_\varepsilon\left(\langle \mathbf{w}, \mathbf{x}\rangle, y(\mathbf{x})\right)]\!]$ and $\tilde{\mathcal{R}}_\mathcal{D}(\mathbf{w}) := \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\!\left[\tilde{\ell}_\beta(\langle \mathbf{w}, \mathbf{x}\rangle, y(\mathbf{x}))\right]\!\!\right]$. Also let $\bar{\mathbf{w}}$ be the minimizer of the following program

$$\bar{\mathbf{w}} = \mathop{\arg\min}_{\mathbf{w}:\|\mathbf{w}\|_1 \le C_W} \tilde{\mathcal{R}}_\mathcal{D}(\mathbf{w}) \tag{5.2}$$

Then (Shalev-Shwartz et al., 2010b, Theorem 2.4), when specialized to our case, guarantees that Algorithm 6, when executed with $\tilde{\ell}_\beta(\cdot,\cdot)$ as the loss function for $\beta = \frac{1}{\varepsilon_2}$, produces a k-sparse predictor $\hat{\mathbf{x}}$, for $k = \left\lceil \frac{8C_W^2}{\varepsilon_2^2} \right\rceil$, with $\|\hat{\mathbf{w}}\|_1 \le C_W$ such that

$$\tilde{\mathcal{R}}_\mathcal{D}(\hat{\mathbf{w}}) - \tilde{\mathcal{R}}_\mathcal{D}(\bar{\mathbf{w}}) \le \varepsilon_2$$

Thus, if we can show the existence of a good predictor in our space with bounded L_1 norm then this would upper bound the loss incurred by the minimizer of Equation 5.2 and using (Shalev-Shwartz et al., 2010b, Theorem 2.4) we would be done. Note that Theorem 5.8 does indeed give us such a guarantee which allows us to make the following argument: we are guaranteed the existence of a predictor $\tilde{f}$ with L_1 norm bounded by B that has ε-insensitive loss bounded by $(\varepsilon_0 + \varepsilon_1)$. Thus if we take $C_W = B$ in Equation 5.2 and use the left inequality of Equation 5.1, we get $\tilde{\mathcal{R}}_\mathcal{D}(\bar{\mathbf{w}}) \le \varepsilon_0 + \varepsilon_1$. Thus we have $\tilde{\mathcal{R}}_\mathcal{D}(\hat{\mathbf{w}}) \le \varepsilon_0 + \varepsilon_1 + \varepsilon_2$. Using Equation 5.1 (right inequality) with $\beta = \frac{1}{\varepsilon_2}$, we get $\mathcal{R}_\mathcal{D}(\hat{\mathbf{w}}) \le \varepsilon_0 + \varepsilon_1 + 3\varepsilon_2/2$.

However it is not possible to give utility guarantees with bounded sample complexities using the above analysis, the reason being that Algorithm 6 requires us to calculate, for any given vector $\mathbf{w}$, the vector $\nabla_\mathbf{w}\tilde{\mathcal{R}}(\mathbf{w}) = \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\!\left[\frac{\partial}{\partial \mathbf{w}}\tilde{\ell}_\beta(\langle \mathbf{w}, \mathbf{x}\rangle, y(\mathbf{x}))\right]\!\!\right]$ which is infeasible to calculate for a distribution with infinite support since it requires unbounded sample complexities. To remedy we shall, as suggested by Shalev-Shwartz et al., take $\mathcal{D}$ not to

be the true distribution over the entire domain $\mathcal{X}$, but rather the *empirical distribution* $\mathcal{D}_{\text{emp}} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{\{\mathbf{x}=\mathbf{x}_i\}}$ for a given sample of training points $\mathbf{x}_1, \ldots, \mathbf{x}_n$. Note that the result of Shalev-Shwartz et al. holds for any distribution which allows us to proceed as before.

Notice however, that we are yet again faced with the challenge of proving an upper bound on the loss incurred by the minimizer of Equation 5.2. This we do as follows: the predictor $\tilde{f}$ defined in Theorem 5.8 has expected ε-insensitive loss over the entire domain bounded by $\varepsilon_0 + \varepsilon_1$. Hence it will, with probability greater than $(1-\delta)$, have at most $\varepsilon_0 + \varepsilon_1 + \mathcal{O}\left(\frac{B}{\sqrt{n}}\right)$ loss on a random sample of n points by an application of Hoeffding's inequality. Thus we have $\tilde{\mathcal{R}}_{\mathcal{D}_{\text{emp}}}(\bar{\mathbf{w}}) \leq \varepsilon_0 + \varepsilon_1 + \mathcal{O}\left(\frac{B}{\sqrt{n}}\right)$ with high probability.

The main difference in this analysis shall be that the guarantee on $\hat{\mathbf{w}}$ we get will be on its *training loss* rather than its true loss, i.e. we will have $\mathcal{R}_{\mathcal{D}_{\text{emp}}}(\hat{\mathbf{w}}) \leq \varepsilon_0 + \varepsilon_1 + \mathcal{O}\left(\frac{B}{\sqrt{n}}\right) + \varepsilon_2$. However since Algorithm 6 guarantees $\|\hat{\mathbf{w}}\|_1 \leq C_W = B$, we can still hope to bound its generalization error. More specifically, Lemma 5.12, given below, shows that with probability greater than $(1-\delta)$ over the choice of training points we will have, for all $\mathbf{w} \in \mathbb{R}^d$, $\mathcal{R}_{\mathcal{D}}(\mathbf{w}) - \mathcal{R}_{\mathcal{D}_{\text{emp}}}(\mathbf{w}) \leq \tilde{\mathcal{O}}\left(\frac{B}{\sqrt{n}}\right)$ where the $\tilde{\mathcal{O}}(\cdot)$ notation hides certain log factors.

Lemma 5.12 (Risk bounds for sparse linear predictors (Kakade et al., 2008))**.** *Consider a real-valued prediction problem y over a domain $\mathcal{X} = \{\mathbf{x} : \|\mathbf{x}\|_\infty \leq C_X\} \subset \mathbb{R}^d$ and a linear learning model $\mathcal{F} : \{\mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x}\rangle : \|\mathbf{w}\|_0 \leq k, \|\mathbf{w}\|_1 \leq C_W\}$ under some fixed loss function $\ell(\cdot,\cdot)$ that is C_L-Lipschitz in its second argument. For any $f \in \mathcal{F}$, let $\mathcal{L}_f = \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}[\![\ell(f(\mathbf{x}), y(\mathbf{x}))]\!]$ and $\hat{\mathcal{L}}_f^n$ be the empirical loss on a set of n i.i.d. chosen points, then we have, with probability greater than $(1-\delta)$,*

$$\sup_{f\in\mathcal{F}} \left(\mathcal{L}_f - \hat{\mathcal{L}}_f^n\right) \leq 2C_L C_X C_W \sqrt{\frac{2\log(2d)}{n}} + C_L C_X C_W \sqrt{\frac{\log(1/\delta)}{2n}}.$$

Thus, by applying a union bound, with probability at least $(1-2\delta)$, we will choose a training set such that $\tilde{f}$, and consequently $\bar{\mathbf{w}}$, has bounded loss on that set as well as the uniform convergence guarantee of Lemma 5.12 will hold. Then we can bound the true loss of the predictor returned by Algorithm 6 as

$$\mathcal{R}_{\mathcal{D}}(\hat{\mathbf{w}}) \leq \mathcal{R}_{\mathcal{D}_{\text{emp}}}(\hat{\mathbf{w}}) + \tilde{\mathcal{O}}\left(\frac{B}{\sqrt{n}}\right) \leq \varepsilon_0 + \varepsilon_1 + \varepsilon_2 + \tilde{\mathcal{O}}\left(\frac{B}{\sqrt{n}}\right)$$

where the first inequality uses the uniform convergence guarantee and the second inequality holds conditional on $\tilde{f}$ having bounded loss on a given training set. The final guarantee is formally given in Theorem 5.9.

Note that using Lemma 5.23 here would at best guarantee a decay of $\mathcal{O}\left(\sqrt{\frac{d}{n}}\right)$. Transferring ε-insensitive loss to absolute loss requires an addition of ε. Using all the results given above, we can now give a proof for Theorem 5.9 (see Section 5.7.5).

We note that Forward Greedy Selection gives $\mathcal{O}\left(\frac{1}{k}\right)$ error rates, which are much better, if the loss function being used is smooth. This can be achieved by using squared

loss $\ell_{\text{sq}}(a,b) = (a-b)^2$ as the surrogate. However we note that assuming goodness of the similarity function in terms of squared loss would impose strictly stronger conditions on the learning problem. This is because $\mathbb{E}[\![\ell_{\text{sq}}(a,b)]\!] = \sup(a-b) \cdot \mathbb{E}[\![|a-b|]\!]$ and thus, under boundedness conditions, squared loss is bounded by a constant times the absolute loss but it is not possible to bound absolute loss (or ε-insensitive loss) as a constant multiple of the squared loss since there exist distributions such that $\mathbb{E}[\![|a-b|]\!] = \Omega\left(\frac{1}{\inf(|a-b|)} \cdot \mathbb{E}[\![\ell_{\text{sq}}(a,b)]\!]\right)$ and $\frac{1}{\inf(|a-b|)}$ can diverge.

5.3.3 Ordinal Regression

The problem of ordinal regression requires an accurate prediction of (discrete) labels coming from a finite ordered set $[r] = \{1, 2, \ldots, r\}$. The problem is similar to both classification and regression, but has some distinct features due to which it has received independent attention (Chu and Keerthi, 2007; Agarwal, 2008) in domains such as product ratings etc. The most popular performance measure for this problem is the absolute loss which is the absolute difference between the predicted and the true labels.

A natural and rather tempting way to solve this problem is to relax the problem to real-valued regression and threshold the output of the learned real-valued predictor using predefined thresholds $b_1, \ldots, b_r$ to get discrete labels. Although this approach has been prevalent in literature (Agarwal, 2008), as the discussion in the next section shows, this leads to poor generalization guarantees in our model. More specifically, a goodness definition constructed around such a direct reduction is only able to ensure $(1+\varepsilon_0)$-utility i.e. the absolute error rate is always greater than 1.

5.3.3.1 Reductions to real valued regression

One of the simplest learning algorithms for the problem of ordinal regression involves a reduction to real-valued regression (Agarwal, 2008; Chu and Keerthi, 2007) where we modify our goal to that of learning a real valued function f which we then threshold using a set of thresholds $\{b_i\}_{i=1}^r$ with $b_1 = -\infty$ to get discrete labels as shown below

$$y_f(\mathbf{x}) = \arg\max_{i \in [r]} \{b_i : f(\mathbf{x}) \geq b_i\}$$

These thresholds may themselves be learned or fixed a priori. A simple choice for these thresholds is $b_i = i-1$ for $i > 1$. It is easy to show (using a result of Agarwal (2008)) that for the fixed thresholds specified above, we have for all $f : \mathcal{X} \to \mathbb{R}$,

$$\begin{aligned}
\ell_{\text{ord}}(y_f(\mathbf{x}), y(\mathbf{x})) &\leq \min\left\{2|f(\mathbf{x}) - y(\mathbf{x})|, |f(\mathbf{x}) - y(\mathbf{x})| + \frac{1}{2}\right\} \\
&\leq \min\left\{2\ell_\varepsilon(f(\mathbf{x}) - y(\mathbf{x})) + 2\varepsilon, \ell_\varepsilon(f(\mathbf{x}) - y(\mathbf{x})) + \varepsilon + \frac{1}{2}\right\}
\end{aligned}$$

where in the last step we use the fact that $|x| - \varepsilon \leq \ell_\varepsilon(x) \leq |x|$.

It is tempting to use this reduction along with guarantees given for real-valued regression to directly give generalization bounds for ordinal regression. To pursue this further, we need a notion of a good similarity function which we give below:

Definition 5.13. *A similarity function K is said to be (ε_0, B)-good for an ordinal regression problem $y : \mathcal{X} \to [r]$ if for some bounded weight function $w : \mathcal{X} \to [-B, B]$, the following predictor, when subjected to fixed thresholds, has expected ordinal regression error at most ε_0*

$$f : \mathbf{x} \mapsto \mathop{\mathbb{E}}_{\mathbf{x}' \sim \mathcal{D}} \left[\!\left[w(\mathbf{x}') K(\mathbf{x}, \mathbf{x}') \right]\!\right]$$

i.e. $\mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\left[|y_f(\mathbf{x}) - y(\mathbf{x})| \right]\!\right] < \varepsilon_0$.

From the definition of the thresholding scheme used to define y_f from f, it is clear that $|f(\mathbf{x}) - y(\mathbf{x})| \leq |y_f(\mathbf{x}) - y(\mathbf{x})| + \frac{1}{2}$. Since we have $\ell_\varepsilon(x) \leq |x|$ for any $\varepsilon \geq 0$, we have $\ell_\varepsilon(f(\mathbf{x}) - y(\mathbf{x})) \leq |y(\mathbf{x}) - y_f(\mathbf{x})| + \frac{1}{2}$ and thus we have $\mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\left[\ell_\varepsilon(f(\mathbf{x}), y(\mathbf{x})) \right]\!\right] < \varepsilon_0 + \frac{1}{2}$.

Thus, starting with goodness guarantee of the similarity function with respect to ordinal regression, we obtain a guarantee of the goodness of the similarity function K with respect to real-valued regression that satisfies the requirements of Theorem 5.5. Thus we have the existence of a linear predictor over a low dimensional space with ε-insensitive error at most $\varepsilon_0 + \frac{1}{2} + \varepsilon_1$. We can now argue (using results from (Agarwal, 2008)) that this real-valued predictor, when subjected to the fixed thresholds, would yield a predictor with ordinal regression error at most

$$\min\left\{2\left(\varepsilon_0 + \frac{1}{2} + \varepsilon_1\right) + 2\varepsilon, \left(\varepsilon_0 + \frac{1}{2} + \varepsilon_1\right) + \varepsilon + \frac{1}{2}\right\} = 1 + \varepsilon_0 + \varepsilon_1 + \varepsilon.$$

However, this is disappointing since this implies that the resulting predictor would, on an average, give out labels that are at least one step away from the true label. This forms the intuition behind introducing (soft) margins in the goodness formulation that gives us Definition 5.14. Below we give proofs for utility and admissibility guarantees for our model for similarity-based ordinal regression.

5.3.3.2 A Soft Margin Approach to Ordinal Regression

One of the reasons for the bad performance of the previous approach is the presence of the thresholding operation that makes it impossible to distinguish between instances that would not be affected by small perturbations to the underlying real-valued predictor and those that would be affected. To remedy this, we enforce a (soft) margin with respect to thresholding that makes the formulation more robust to noise. More formally, we expect that if a point belongs to the label i, then in addition to being sandwiched between the thresholds b_i and b_{i+1}, it should be separated from these by a margin as well i.e. $b_i + \gamma \leq f(\mathbf{x}) \leq b_{i+1} - \gamma$.

This is a direct generalization of the margin principle in classification where we expect $\mathbf{w}^\top \mathbf{x} > b + \gamma$ for positively labeled points and $\mathbf{w}^\top \mathbf{x} < b - \gamma$ for negatively labeled points.

Of course, wherein classification requires a single threshold, we require several, depending upon the number of labels. For any $x \in \mathbb{R}$, let $[x]_+ = \max\{x, 0\}$. Thus, if we define the γ-margin loss function to be $[x]_\gamma := [\gamma - x]_+$ (note that this is simply the well known hinge loss function scaled by a factor of γ), we can define our goodness criterion as follows:

Definition 5.14. *A similarity function K is said to be (ε_0, B)-good for an ordinal regression problem $y : \mathcal{X} \to [r]$ if for some bounded weight function $w : \mathcal{X} \to [-B, B]$ and some (unknown but fixed) set of thresholds $\{b_i\}_{i=1}^r$ with $b_1 = -\infty$, the predictor $f : \mathbf{x} \mapsto \mathop{\mathbb{E}}_{\mathbf{x}' \sim \mathcal{D}} [\![w(\mathbf{x}')K(\mathbf{x}, \mathbf{x}')]\!]$ satisfies $\mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\left[f(\mathbf{x}) - b_{y(\mathbf{x})}\right]_\gamma + \left[b_{y(\mathbf{x})+1} - f(\mathbf{x})\right]_\gamma \right]\!\!\right] < \varepsilon_0$.*

5.3.3.3 Admissibility Guarantees

We begin by giving the kernel goodness criterion which we adapt from existing literature on large margin approaches to ordinal regression. More specifically we use the framework described by Chu and Keerthi (2007) for which generalization guarantees are given by Agarwal (2008).

Definition 5.15. *Call a PSD kernel K (ε_0, γ)-good for an ordinal regression problem $y : \mathcal{X} \to [r]$ if there exists $\mathbf{W}^* \in \mathcal{H}_K$, $\|\mathbf{W}^*\| = 1$ and a fixed set of thresholds $\{b_i\}_{i=1}^r$ such that*

$$\mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\left[b_{y(\mathbf{x})} + 1 - \frac{\langle \mathbf{W}^*, \Phi_K(\mathbf{x}) \rangle}{\gamma} \right]_+ + \left[\frac{\langle \mathbf{W}^*, \Phi_K(\mathbf{x}) \rangle}{\gamma} - b_{y(\mathbf{x})+1} + 1 \right]_+ \right]\!\!\right] < \varepsilon_0$$

The above definition exactly corresponds to the EXC formulation put forward by Chu and Keerthi except for the fact that during actual optimization, a strict ordering on the thresholds is imposed explicitly. Chu and Keerthi present yet another model called IMC which does not impose any explicit orderings, rather the ordering emerges out of the minimization process itself. Our model can be easily extended to the IMC formulation as well.

We now give utility guarantees for our learning model. We shall give guarantees on both the misclassification error as well as the absolute error of our learned predictor. We say that a set of points $x_1, \ldots, x_i \ldots$ is Δ-spaced if $\min_{i \neq j} \{|x_i - x_j|\} \geq \Delta$. Define the function $\psi_\Delta(x) = \frac{x + \Delta - 1}{\Delta}$.

Theorem 5.16. *Let K be a similarity function that is (ε_0, B)-good for an ordinal regression problem with respect to Δ-spaced thresholds and γ-margin loss. Let $\bar{\gamma} = \max\{\gamma, 1\}$. Then K is $\psi_{(\Delta/\bar{\gamma})}\left(\frac{\varepsilon_0}{\bar{\gamma}}\right)$-useful with respect to ordinal regression error (absolute loss). Moreover, K is $\left(\frac{\varepsilon_0}{\bar{\gamma}}\right)$-useful with respect to the zero-one mislabeling error as well.*

We require spaced thresholds since the margin losses themselves do not present any bound on the ordinal regression error. This is because, if the thresholds are closely spaced together, then even an instance of gross ordinal regression loss could correspond to very small margin loss. To remedy this, we introduce the *spacing* parameter Δ into the model.

The exact manner in which this parameter is utilized can be seen in the proof of Theorem 5.16 in Section 5.7.7. Such a condition can easily be incorporated into the model of Chu and Keerthi (2007) as a constraint in the optimization formulation.

Note that we can bound, both dimensionality of the landmarked space as well as labeled sampled complexity, by $\mathcal{O}\left(\frac{B^2}{\varepsilon_1^2}\log\frac{1}{\delta}\right)$. Notice also, that for $\varepsilon_0 < 1$ and large enough d, n, we can ensure that the ordinal regression error rate is also bounded above by 1 since $\sup_{x\in[0,1],\Delta>0}(\psi_\Delta(x)) = 1$. This is in contrast with the direct reduction to real valued regression which has ordinal regression error rate bounded *below* by 1. In particular, a constraint of $\Delta = 1$ put into an optimization framework ensures that the bounds on mislabeling loss and ordinal regression loss match since $\psi_1(x) = x$ for all x. In general, the cases where the above framework yields a non-trivial bound for the mislabeling error rate, i.e. $\ell_{01} < 1$ (which can always be ensured if $\varepsilon_0 < 1$ by taking large enough d and n), also correspond to those where the ordinal regression error rate is also bounded above by 1 since $\sup_{x\in[0,1],\Delta>0}(\psi_\Delta(x)) = 1$.

This indicates the advantage of the present model over a naive reduction to regression. We can show that our definition of a good similarity function admits all good PSD kernels as well. The kernel goodness criterion we adopt corresponds to the large margin framework proposed by Chu and Keerthi (2007). We refer the reader to Section 5.3.3.3 for the definition and give the admissibility result below.

Theorem 5.17. *Every PSD kernel that is (ε_0, γ)-good for an ordinal regression problem is also $\left(\gamma_1\varepsilon_0 + \varepsilon_1, \mathcal{O}\left(\frac{\gamma_1^2}{\varepsilon_1\gamma^2}\right)\right)$-good as a similarity function with respect to the γ_1-margin loss for any $\gamma_1, \varepsilon_1 > 0$. Moreover, for any $\varepsilon_1 < \gamma_1/2$, there exists an ordinal regression instance and a corresponding kernel that is $(0, \gamma)$-good for the ordinal regression problem but only (ε_1, B)-good as a similarity function with respect to the γ_1-margin loss function for $B = \Omega\left(\frac{\gamma_1^2}{\varepsilon_1\gamma^2}\right)$.*

5.3.4 Ranking

The problem of ranking stems from the need to sort a set of items based on their relevance. In the model considered here, each ranking instance is composed of m documents (pages) $(p_1, \ldots, p_m)$ from some universe $\mathcal{P}$ along with their relevance to some particular query $q \in \mathcal{Q}$ that are given as relevance scores from some set $\mathcal{R} \subset \mathbb{R}$. Thus we have $\mathcal{X} = \mathcal{Q} \times \mathcal{P}^m$ with each instance $\mathbf{x} \in \mathcal{X}$ being provided with a relevance vector $r(\mathbf{x}) = \mathcal{R}^m$. Let the i^{th} query-document pair of a ranking instance $\mathbf{x}$ be denoted by $\mathbf{z}_i \in \mathcal{Q} \times \mathcal{P}$. For any $\mathbf{z} = (p, q) \in \mathcal{P} \times \mathcal{Q}$, let $r(\mathbf{z}) \in \mathbb{R}$ denote the true relevance of document p to query q.

For any relevance vector $\mathbf{r} \in \mathcal{R}^m$, let $\bar{\mathbf{r}}$ be the vector with elements of $\mathbf{r}$ sorted in descending order and $\pi_{\mathbf{r}}$ be the permutation that this sorting induces. For any permutation π, $\pi(i)$ shall denote the index given to the index i under π. Although the desired output of a ranking problem is a permutation, we shall follow the standard simplification (Ravikumar et al., 2011) of requiring the output to be yet another relevance vector $\mathbf{s}$ with the

permutation $\pi_{\mathbf{s}}$ being considered as the actual output. This converts the ranking problem into a vector-valued regression problem.

We will take the true loss function $\ell_{\text{actual}}(\cdot,\cdot)$ to be the popular NDCG loss function (Järvelin and Kekäläinen, 2000) defined below

$$\ell_{\text{NDCG}}(\mathbf{s},\mathbf{r}) = -\frac{1}{\|G(\mathbf{r})\|_D}\sum_{i=1}^{m}\frac{G(\mathbf{r}(i))}{F(\pi_{\mathbf{s}}(i))}$$

where $\|\mathbf{r}\|_D = \max_{\pi\in S_m}\sum_{i=1}^{m}\frac{\mathbf{r}(i)}{F(\pi(i))}$, $G(r) = 2^r - 1$ is the growth function and $F(t) = \log(1+t)$ is the decay function.

For the surrogate loss functions ℓ_K and ℓ_S, we shall use the squared loss function $\ell_{\text{sq}}(\mathbf{s},\mathbf{r}) = \|\mathbf{s}-\mathbf{r}\|_2^2$. We shall overload notation to use $\ell_{\text{sq}}(\cdot,\cdot)$ upon reals as well. For any vector $\mathbf{r}\in\mathcal{R}^m$, let $\eta(\mathbf{r}) := \frac{G(\mathbf{r})}{\|G(\mathbf{r})\|_D}$ and let $\mathbf{r}_i$ denote its i^{th} coordinate.

Due to the decomposable nature of the surrogate loss function, we shall require kernels and similarity functions to act over query-document pairs i.e. $K : (\mathcal{P}\times\mathcal{Q})\times(\mathcal{P}\times\mathcal{Q}) \to \mathbb{R}$. This also coincides with a common feature extraction methodology (see for example (Ravikumar et al., 2011; Agarwal and Niyogi, 2009)) where every query-document pair is processed to yield a feature vector. Consequently, all our goodness definitions shall loosely correspond to the ability of a kernel/similarity to accurately predict the true relevance scores for a given query-document pair. We shall assume ranking instances to be generated by the sampling of a query $q\sim\mathcal{D}_{\mathcal{Q}}$ followed by m independent samples of documents from the (conditional) distribution $\mathcal{D}_{\mathcal{P}|q}$. The distribution over ranking instances is then a product distribution $\mathcal{D} = \mathcal{D}_{\mathcal{X}} = \mathcal{D}_{\mathcal{Q}}\times\underbrace{\mathcal{D}_{\mathcal{P}|q}\times\mathcal{D}_{\mathcal{P}|q}\times\ldots\times\mathcal{D}_{\mathcal{P}|q}}_{m\text{ times}}$. A key consequence of this generative mechanism is that the i^{th} query-document pair of a random ranking instance, for any fixed i, is a random query-document instance selected from the distribution $\mu := \mathcal{D}_{\mathcal{Q}}\times\mathcal{D}_{\mathcal{P}|q}$.

Definition 5.18. *A similarity function K is said to be (ε_0, B)-good for a ranking problem $y : \mathcal{X}\to S_m$ if for some bounded weight function $w : \mathcal{P}\times\mathcal{Q}\to[-B,B]$, for any ranking instance $\mathbf{x} = (q, p_1, p_2, \ldots, p_m)$, if we define $f : \mathcal{X}\to\mathbb{R}^m$ as*

$$f_i := \mathop{\mathbb{E}}_{\mathbf{z}\sim\mu}[\![w(\mathbf{z})K(\mathbf{z}_i,\mathbf{z})]\!]$$

where $\mathbf{z}_i = (p_i, q)$, then we have $\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}[\![\ell_{sq}(f(\mathbf{x}),\eta(r(\mathbf{z})))]\!] < \varepsilon_0$.

Definition 5.19. *A PSD kernel K is said to be (ε_0,γ)-good for a ranking problem $y : \mathcal{X}\to S_m$ if there exists $\mathbf{W}^*\in\mathcal{H}_K$, $\|\mathbf{W}^*\| = 1$ such that if for any ranking instance $\mathbf{x} = (q, p_1, p_2, \ldots, p_m)$, if, for any $\mathbf{W}\in\mathcal{H}_K$, when we define $f(\,\cdot\,;\mathbf{W}) : \mathcal{X}\to\mathbb{R}^m$ as*

$$f_i(\mathbf{x};\mathbf{W}) = \frac{\langle\mathbf{W},\Phi_K(\mathbf{z}_i)\rangle}{\gamma}$$

where f_i is the i^{th} coordinate of the output of f and $\mathbf{z}_i = (p_i, q)$, then we have

$$\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}} [\![\ell_{sq}\left(f(\mathbf{x};\mathbf{W}^*), \eta(r(\mathbf{z}))\right)]\!] < \varepsilon_0.$$

The choice of this surrogate is motivated by consistency considerations. We would ideally like a minimizer of the surrogate loss to have bounded actual loss as well. Using results from (Ravikumar et al., 2011), it can be shown that the above defined surrogate is not only consistent, but that excess loss in terms of this surrogate can be transferred to excess loss in terms of $\ell_{\text{NDCG}}(\cdot,\cdot)$, a very desirable property. Although Ravikumar et al. show this to be true for a whole family of surrogates, we chose $\ell_{\text{sq}}(\cdot,\cdot)$ for its simplicity. All our utility arguments carry forward to other surrogates defined by Ravikumar et al. with minimal changes. We move on to prove utility guarantees for the given similarity learning model.

Theorem 5.20. *Every similarity function that is (ε_0, B)-good for a ranking problem for m-documents with respect to squared loss is $\mathcal{O}\left(\sqrt{\frac{m}{\log m}} \cdot \sqrt{\varepsilon_0}\right)$-useful with respect to NDCG loss.*

We note that the $\mathcal{O}(\sqrt{m})$ dependence of the final utility guarantee on m is because the decay function $F(t) = \log(1+t)$ chosen here (which seems to be a standard in literature but with little theoretical justification) is a very slowly growing function (it might sound a bit incongruous to have an increasing function as our *decay* function - however since this function appears in the denominator in the definition of NDCG, it effectively induces a decay). Using decay functions that grow super-linearly (or rather those that induce super-linear decays), we can ensure $\mathcal{O}\left(\sqrt{\varepsilon_0}\right)$-usefulness since in those cases, $C_F = \mathcal{O}(1)$.

We next prove admissibility bounds for the ranking problem. The learning setting as well as the proof is different for ranking due to presence of multiple entities in a single ranking instance. We refer the reader to Section 5.7.10 for the complete proof.

Theorem 5.21. *Every PSD kernel that is (ε_0, γ)-good for a ranking problem is also $\left(\varepsilon_0 + \varepsilon_1, \mathcal{O}\left(\frac{m\sqrt{m}}{\varepsilon_1\sqrt{\varepsilon_1}\gamma^3}\right)\right)$-good as a similarity function for any $\varepsilon_1 > 0$.*

5.4 Experimental Results

In this section we present an empirical evaluation of our learning models for the problems of real-valued regression and ordinal regression on benchmark datasets taken from a variety of sources (uci; sta; del). We compare our algorithms against kernel regression (KR), a well known technique (Weinberger and Tesauro, 2007) for non-linear regression, whose predictor is of the form:

$$f : \mathbf{x} \mapsto \frac{\sum_{\mathbf{x}_i \in \mathcal{T}} y(\mathbf{x}_i) K(\mathbf{x}, \mathbf{x}_i)}{\sum_{\mathbf{x}_i \in \mathcal{T}} K(\mathbf{x}, \mathbf{x}_i)}.$$

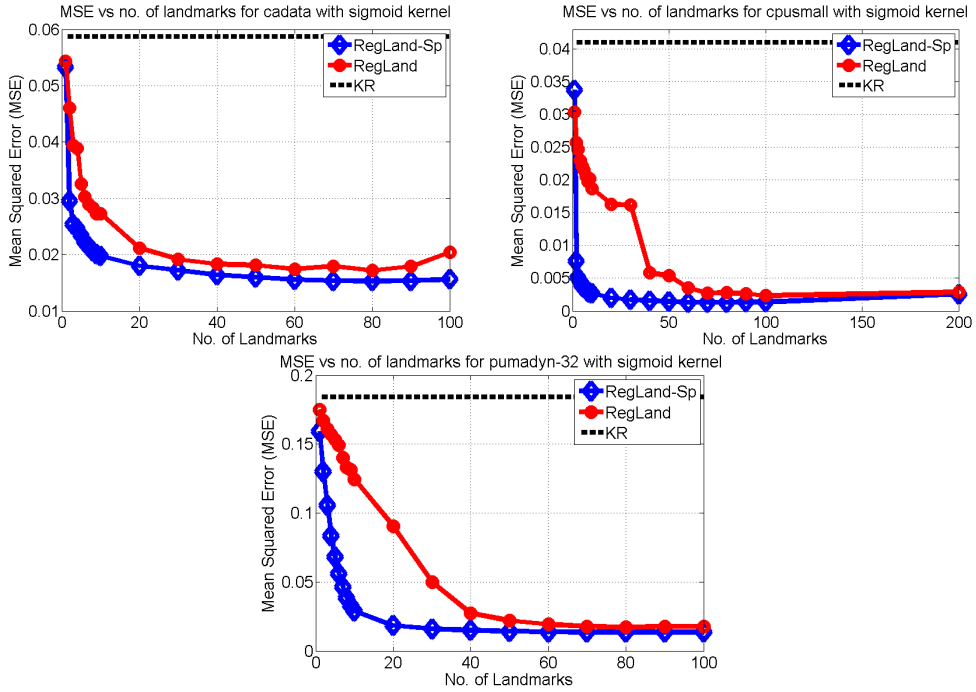
where $\mathcal{T}$ is the training set. We selected KR as the baseline as it is a popular regression method that does not require similarity functions to be PSD. For ordinal regression

problems, we rounded off the result of the KR predictor to get a discrete label. We implemented all our algorithms as well as the baseline KR method in Matlab. In all our experiments we report results across 5 random splits on the (indefinite) Sigmoid: $K(\mathbf{x}, \mathbf{y}) = \tanh(a \langle \mathbf{x}, \mathbf{y} \rangle + r)$ and Manhattan: $K(\mathbf{x}, \mathbf{y}) = -\|\mathbf{x} - \mathbf{y}\|_1$ kernels. Following standard practice, we fixed $r = -1$ and $a = 1/d_{\text{orig}}$ for the Sigmoid kernel where d_{orig} is the dimensionality of the dataset.

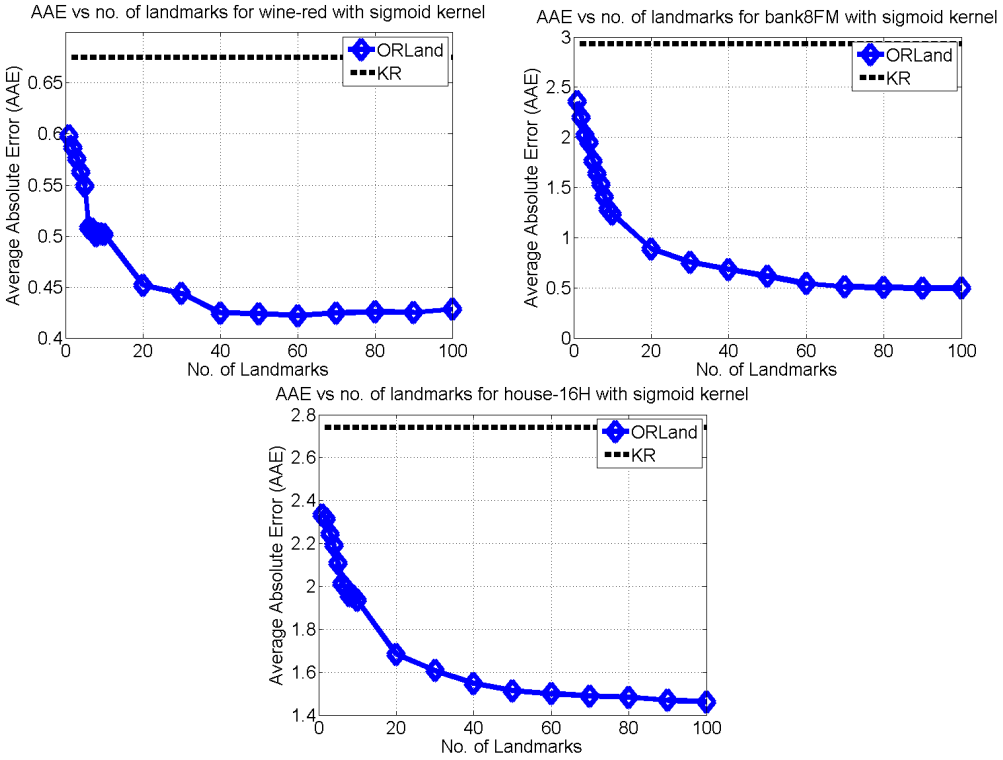
Real valued regression: For this experiment, we compare our methods (RegLand and RegLand-Sp) with the KR method. For RegLand, we constructed the landmarked space as specified in Algorithm 5 and learned a linear predictor using the LIBLINEAR package (Ho and Lin, 2012) that minimizes ε-insensitive loss. In the second algorithm (RegLand-Sp), we used the sparse learning algorithm of Shalev-Shwartz et al. (2010b) on the landmarked space to learn the best predictor for a given sparsity level. Due to its simplicity and good convergence properties, we implemented the *Fully Corrective* version of the Forward Greedy Selection algorithm with squared loss as the surrogate.

We evaluated all methods using Mean Squared Error (MSE) on the test set. Figure 5.2a shows the MSE incurred by our methods along with reference values of accuracies obtained by KR as landmark sizes increase. The plots clearly show that our methods incur significantly lesser error than KR. Moreover, RegLand-Sp learns more accurate predictors using the same number of landmarks. For instance, when learning using the Sigmoid kernel on the `CPUData` dataset, at 20 landmarks, RegLand is able to guarantee an MSE of 0.016 whereas RegLand-Sp offers an MSE of less than 0.002; MLKR is only able to guarantee an MSE rate of 0.04 for this dataset. In Table 5.1a, we compare accuracies of the two algorithms when given 50 landmark points with those of KR for the Sigmoid and Manhattan kernels. We find that in all cases, RegLand-Sp gives superior accuracies than KR. Moreover, the Manhattan kernel seems to match or outperform the Sigmoid kernel on all the datasets.

Ordinal Regression: Here, we compare our method with the baseline KR method on benchmark datasets. As mentioned in Section 5.3.3, our method uses the EXC formulation of Chu and Keerthi (2007) along with the landmarking scheme given in Algorithm 5. We implemented a gradient descent-based solver (ORLand) to solve the primal formulation of EXC and used fixed equi-spaced thresholds instead of learning them as suggested by Chu and Keerthi. Of the six datasets considered here, the two `Wine` datasets are ordinal regression datasets where the quality of the wine is to be predicted on a scale from 1 to 10. The remaining four datasets are regression datasets whose labels were subjected to equi-frequency binning to obtain ordinal regression datasets as done by Chu and Keerthi. We measured the average absolute error (AAE) for each method. Figure 5.2b compares ORLand with KR as the number of landmarks increases. Table 5.1b compares accuracies of ORLand for 50 landmark points with those of KR for Sigmoid and Manhattan kernels. In almost all cases, ORLand gives a much better performance than KR. The Sigmoid kernel seems to outperform the Manhattan kernel on a couple of datasets. We refer the reader to Section 5.8 for additional experimental results.



(a) Mean squared error for landmarking (RegLand), sparse landmarking (RegLand-Sp) and kernel regression (KR) on real regression datasets.



(b) Average absolute error for landmarking (ORLand) and kernel regression (KR) on ordinal regression datasets.

Figure 5.2: Performance of landmarking algorithms with increasing number of landmarks on real-valued regression (Figure 5.2a) and ordinal regression (Figure 5.2b) datasets.

Datasets	Sigmoid kernel		Manhattan kernel	
	KR	Land-Sp	KR	Land-Sp
`Abalone` (uci)	2.1e-002	**6.2e-003**	1.7e-002	**6.0e-003**
$N = 4177, d = 8$	(8.3e-004)	(8.4e-004)	(7.1e-004)	(3.7e-004)
`Bodyfat` (sta)	4.6e-004	**9.5e-005**	3.9e-004	**3.5e-005**
$N = 252, d = 14$	(6.5e-005)	(1.3e-004)	(2.2e-005)	(1.3e-005)
`CAHousing` (sta)	5.9e-002	**1.6e-002**	5.8e-002	**1.5e-002**
$N = 20640, d = 8$	(2.3e-004)	(6.2e-004)	(1.9e-004)	(1.4e-004)
`CPUData` (del)	4.1e-002	**1.4e-003**	4.3e-002	**1.2e-003**
$N = 8192, d = 12$	(1.6e-003)	(1.7e-004)	(1.6e-003)	(3.2e-005)
`PumaDyn-8` (del)	2.3e-001	**1.4e-002**	2.3e-001	**1.4e-002**
$N = 8192, d = 8$	(4.6e-003)	(4.5e-004)	(4.5e-003)	(4.8e-004)
`PumaDyn-32` (del)	1.8e-001	**1.4e-002**	1.8e-001	**1.4e-002**
$N = 8192, d = 32$	(3.6e-003)	(3.7e-004)	(3.6e-003)	(3.1e-004)

(a) Mean squared error for real regression problems.

Datasets	Sigmoid kernel		Manhattan kernel	
	KR	ORLand	KR	ORLand
`Wine-Red` (uci)	6.8e-001	**4.2e-001**	6.7e-001	**4.5e-001**
$N = 1599, d = 11$	(2.8e-002)	(3.8e-002)	(3.0e-002)	(3.2e-002)
`Wine-White` (uci)	6.2e-001	**8.9e-001**	6.2e-001	**4.9e-001**
$N = 4898, d = 11$	(2.0e-002)	(8.5e-001)	(2.0e-002)	(1.5e-002)
`Bank-8` (del)	2.9e+000	**6.1e-001**	2.7e+000	**6.3e-001**
$N = 8192, d = 8$	(6.2e-002)	(4.4e-002)	(6.6e-002)	(1.7e-002)
`Bank-32` (del)	2.7e+000	**1.6e+000**	2.6e+000	**1.6e+000**
$N = 8192, d = 32$	(1.2e-001)	(2.3e-002)	(8.1e-002)	(9.4e-002)
`House-8` (del)	2.8e+000	**1.5e+000**	2.7e+000	**1.4e+000**
$N = 22784, d = 8$	(9.3e-003)	(2.0e-002)	(1.0e-002)	(1.2e-002)
`House-16` (del)	2.7e+000	**1.5e+000**	2.8e+000	**1.4e+000**
$N = 22784, d = 16$	(2.0e-002)	(1.0e-002)	(2.0e-002)	(2.3e-002)

(b) Mean absolute error for ordinal regression problems.

Table 5.1: Performance of landmarking-based algorithms (with 50 landmarks) vs. baseline kernel regression (KR). Values in parentheses indicate standard deviation values. Values in the first columns indicate dataset source (in parentheses), size (N) and dimensionality (d). Bold numbers indicate the best performance.

5.5 Discussion

In this work we considered the general problem of supervised learning using non-PSD similarity functions. We provided a goodness criterion for similarity functions w.r.t. various learning tasks. This allowed us to construct efficient learning algorithms with provable generalization error bounds. At the same time, we were able to show, for each learning task, that our criterion is not too restrictive in that it admits all good PSD kernels. We then focused on the problem of identifying influential landmarks with the aim of learning sparse predictors. We presented a model that formalized the intuition that typically only a small fraction of landmarks is influential for a given learning problem. We adapted existing sparse vector recovery algorithms within our model to learn provably sparse predictors with bounded generalization error. Finally, we empirically evaluated our learning algorithms on benchmark regression and ordinal regression tasks. In all cases, our learning methods, especially the sparse recovery algorithm, consistently outperformed the kernel regression baseline.

5.6 Supplementary Theorems

In this section we prove certain generic results that would be used in the utility and admissibility proofs. The first result, given as Lemma 5.22, allows us to analyze the landmarking step (Step 1 of Algorithm 5) and allows us to reduce the learning problem to that of learning a linear predictor over the landmarked space. The second result, given as Lemma 5.23, gives us a succinct re-statement of generalization error bounds proven in (Kakade et al., 2008) that would be used in proving utility bounds. The third result, given as Lemma 5.24, is a technical result that helps us prove admissibility bounds for our goodness definitions.

Lemma 5.22 (Landmarking approximation guarantee (Kar and Jain, 2011))**.** *Given a similarity function K over a domain $\mathcal{X}$ and a bounded function of the form $f(x) = \mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}[\![w(\mathbf{x}')K(\mathbf{x},\mathbf{x}')]\!]$ for some bounded weight function $w : \mathcal{X} \to \{-B, B\}$, for every $\varepsilon, \delta > 0$ there exists a randomized map $\Psi : \mathcal{X} \to \mathbb{R}^d$ for $d = d(\varepsilon, \delta)$ such that with probability at least $1-\delta$, there exists a linear operator $\tilde{f}$ over $\mathbb{R}^d$ such that $\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\!\left[\left|\tilde{f}(\Psi(\mathbf{x})) - f(\mathbf{x})\right|\right]\!\!\right] \leq \varepsilon$.*

Proof. This result essentially allows us to project the learning problem into a Euclidean space where one can show, for the various learning problems considered here, that existing large margin techniques are applicable to solve the original problem. The result appeared in (Kar and Jain, 2011) and is presented here for completeness.

Sample d *landmark points* $\mathcal{L} = \{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$ from $\mathcal{D}$ and construct the map $\Psi_{\mathcal{L}} : \mathbf{x} \mapsto \frac{1}{\sqrt{d}}(K(\mathbf{x}, \mathbf{x}_1), \ldots, K(\mathbf{x}, \mathbf{x}_d))$ and consider the linear operator $\tilde{f}$ over $\mathbb{R}^d$ defined as follows (in the following, we shall always omit the subscript $\mathcal{L}$ for clarity):

$$\tilde{f} : \mathbf{x} \mapsto \frac{1}{d}\sum_{i=1}^{d} w(\mathbf{x}_i)K(\mathbf{x}, \mathbf{x}_i) = \langle \tilde{\mathbf{w}}, \Psi(\mathbf{x}) \rangle$$

for $\mathbf{w} = \frac{1}{\sqrt{d}}(w(\mathbf{x}_1), \ldots, w(\mathbf{x}_d)) \in \mathbb{R}^d$. A standard Hoeffding-style argument shows that for $d = \mathcal{O}\left(\frac{B^2}{\varepsilon^2}\log\frac{1}{\delta^2}\right) = \mathcal{O}\left(\frac{B^2}{\varepsilon^2}\log\frac{1}{\delta}\right)$, $\tilde{f}$ gives a point wise approximation to f, i.e. for all $\mathbf{x} \in \mathcal{X}$, with probability greater than $1-\delta^2$, we have $\left|\tilde{f}(\Psi(\mathbf{x})) - f(\mathbf{x})\right| < \varepsilon$.

Now call the event $\textsf{BAD-APPROX}(\mathbf{x}) := \left|\tilde{f}(\Psi(\mathbf{x})) - f(\mathbf{x})\right| > \varepsilon$. Thus we have for all $\mathbf{x} \in \mathcal{X}$, $\mathop{\mathbb{P}}_{\tilde{f}}[\textsf{BAD-APPROX}(\mathbf{x})] = \mathop{\mathbb{E}}_{\tilde{f}}[\![1_{\textsf{BAD-APPROX}(\mathbf{x})}]\!] < \delta^2$ (here the probabilities are being taken over the construction of $\tilde{f}$ i.e. the choice of the landmark points). Taking expectations over the entire domain, applying Fubini's theorem to switch expectations and applying Markov's inequality we get

$$\mathop{\mathbb{P}}_{\tilde{f}}\left[\mathop{\mathbb{P}}_{\mathbf{x}\sim\mathcal{D}}[\textsf{BAD-APPROX}(\mathbf{x})] > \delta\right] < \delta$$

Thus with confidence $1-\delta$ we have $\mathop{\mathbb{P}}_{\mathbf{x}\sim\mathcal{D}}[\textsf{BAD-APPROX}(\mathbf{x})] < \delta$ and thus

$$\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\!\left[\left|\tilde{f}(\Psi(\mathbf{x})) - f(\mathbf{x})\right|\right]\!\!\right] < (1-\delta)\varepsilon + 2B\delta$$

since $\sup_{\mathbf{x}\in\mathcal{X}}\left|\tilde{f}(\Psi(\mathbf{x}))\right| = \sup_{\mathbf{x}\in\mathcal{X}}|f(\mathbf{x})| = B$. For $\delta < \frac{\varepsilon}{B}$ we get $\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\!\left[\left|\tilde{f}(\Psi(\mathbf{x})) - f(\mathbf{x})\right|\right]\!\!\right] < 2\varepsilon$. □

Lemma 5.23 (Risk bounds for linear predictors (Kakade et al., 2008))**.** *Consider a real-valued prediction problem y over a domain $\mathcal{X} = \{\mathbf{x} : \|\mathbf{x}\|_2 \le C_X\}$ and a linear learning model $\mathcal{F} : \{\mathbf{x} \mapsto \langle\mathbf{w}, \mathbf{x}\rangle : \|\mathbf{w}\|_2 \le C_W\}$ under some fixed loss function $\ell(\cdot,\cdot)$ that is C_L-Lipschitz in its second argument. For any $f \in \mathcal{F}$, let $\mathcal{L}_f = \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}[\![\ell(f(\mathbf{x}), y(\mathbf{x}))]\!]$ and $\hat{\mathcal{L}}_f^n$ be the empirical loss on a set of n i.i.d. chosen points. Then we have, with probability greater than $(1-\delta)$,*

$$\sup_{f\in\mathcal{F}}\left(\mathcal{L}_f - \hat{\mathcal{L}}_f^n\right) \le 3C_L C_X C_W\sqrt{\frac{\log(1/\delta)}{n}}$$

Proof. There exist a few results that provide a unified analysis for the generalization properties of linear predictors (Kakade et al., 2008; Zhang, 2002). However we use the heavy hammer of Rademacher average based analysis since it provides sharper bounds than covering number based analyses.

The result follows from imposing a squared L_2 regularization on the $\mathbf{w}$ vectors. Since the squared L_2 function is 2-strongly convex with respect to the L_2 norm, using (Kakade et al., 2008, Theorem 1), we get a bound on the Rademacher complexity of the function class $\mathcal{F}$ as $\mathcal{R}_n(\mathcal{F}) \le C_X C_W\sqrt{\frac{1}{n}}$. Next, using the Lipschitz properties of the loss function, a result from Bartlett and Mendelson (2002) allows us to bound the excess error by $2C_L\mathcal{R}_n(\mathcal{F}) + C_L C_X C_W\sqrt{\frac{\log(1/\delta)}{2n}}$. The result then follows from simple manipulations. □

Lemma 5.24 (Admissible weight functions for PSD kernels (Srebro, 2007))**.** *Consider a PSD kernel that is (ε_0, γ)-good for a learning problem with respect to some convex loss function ℓ_K. Then there exists a vector $\mathbf{W}' \in \mathcal{H}_K$ and a bounded weight function $w : \mathcal{X} \to$*

$\mathbb{R}$ *such that* $\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}[\![\ell_K(\langle \mathbf{W}', \Phi_K(\mathbf{x})\rangle, y(\mathbf{x}))]\!] \leq \varepsilon_0 + \frac{1}{2C\gamma^2}$ *for some arbitrary positive constant* C *and for all* $\mathbf{x} \in \mathcal{X}$*, we have* $\mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}[\![w(\mathbf{x}')K(\mathbf{x},\mathbf{x}')]\!] = \langle \mathbf{W}', \Phi_K(\mathbf{x})\rangle$.

Proof. Note that the (ε_0, γ)-goodness of K guarantees the existence of a weight vector $\mathbf{W}^* \in \mathcal{H}_K$ with small loss at large margin. Thus $\mathbf{W}'$ acts as a proxy for $\mathbf{W}^*$ providing bounded loss at unit margin but with the additional property of being functionally equivalent to a bounded weighted average of the kernel values as required by the definition of a good similarity function. This will help us prove admissibility results for our similarity learning models.

We start by proving the theorem for a discrete distribution - the generalization to non-discrete distributions will follow by using variational optimization techniques as discussed by Srebro (2007). Consider a discrete learning problem with $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, corresponding distribution $\mathcal{D} = \{p_1, \ldots, p_n\}$ and target $y = \{y_1, \ldots, y_n\}$ such that $\sum p_i = 1$. Set up the following regularized ERM problem (albeit on the entire domain):

$$\min_{\mathbf{W}\in\mathcal{H}_K} \quad \frac{1}{2}\|\mathbf{W}\|^2_{\mathcal{H}_K} + C\sum_{i=1}^{n} p_i \ell_K(\langle \mathbf{W}, \Phi_K(\mathbf{x}_i)\rangle, y_i)$$

Let $\mathbf{W}'$ be the weight vector corresponding to the optima of the above problem. By the Representer Theorem (for example (Schölkopf et al., 2001)), we can choose $\mathbf{W}' = \sum \alpha_i \Phi_K(\mathbf{x}_i)$ for some bounded α_i (the exact bounds on α_i are problem specific). By (ε_0, γ)-goodness of K we have

$$\begin{aligned}
\frac{1}{2}\left\|\mathbf{W}'\right\|^2_{\mathcal{H}_K} + C\sum_{i=1}^{n} p_i \ell_K\left(\left\langle \mathbf{W}', \Phi_K(\mathbf{x}_i)\right\rangle, y_i\right) &\leq \frac{1}{2}\left\|\frac{1}{\gamma}\mathbf{W}^*\right\|^2_{\mathcal{H}_K} + C\sum_{i=1}^{n} p_i \ell_K\left(\frac{\langle \mathbf{W}^*, \Phi_K(\mathbf{x}_i)\rangle}{\gamma}, y_i\right) \\
&= \frac{1}{2\gamma^2} + C\cdot \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\!\left[\ell_K\left(\frac{\langle \mathbf{W}^*, \Phi_K(\mathbf{x})\rangle}{\gamma}, y(\mathbf{x})\right)\right]\!\!\right] \\
&\leq \frac{1}{2\gamma^2} + C\varepsilon_0
\end{aligned}$$

Thus we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\!\left[\ell_K\left(\left\langle \mathbf{W}', \Phi_K(\mathbf{x})\right\rangle, y(\mathbf{x})\right)\right]\!\!\right] &\leq \frac{1}{2C}\left\|\mathbf{W}'\right\|^2_{\mathcal{H}_K} + \sum_{i=1}^{n} p_i \ell_K\left(\left\langle \mathbf{W}', \Phi_K(\mathbf{x}_i)\right\rangle, y_i\right) \\
&\leq \varepsilon_0 + \frac{1}{2C\gamma^2}
\end{aligned}$$

which proves the first part of the claim. For the second part, set up a weight function $w_i = \frac{\alpha_i}{p_i}$. Then, for any $\mathbf{x} \in \mathcal{X}$ we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}[\![w(\mathbf{x}')K(\mathbf{x},\mathbf{x}')]\!] &= \sum_{i=1}^{n} p_i w_i K(\mathbf{x},\mathbf{x}_i) = \sum_{i=1}^{n} p_i \frac{\alpha_i}{p_i} K(\mathbf{x},\mathbf{x}_i) \\
&= \sum_{i=1}^{n} \alpha_i \langle \Phi_K(\mathbf{x}), \Phi_K(\mathbf{x}_i)\rangle = \left\langle \mathbf{W}', \Phi_K(\mathbf{x})\right\rangle
\end{aligned}$$

The weight function is bounded since the α_i are bounded and, this being a discrete learning problem, cannot have vanishing probability masses p_i (actually, in the cases we shall consider, the α_i will itself contain a p_i term that will subsequently get canceled). For non-discrete cases, variational techniques give us similar results. □

5.7 Proofs

We give missing proofs below.

5.7.1 Proof of Theorem 5.5

First of all, we use Lemma 5.22 to project onto a d dimensional space where there exists a linear predictor $\tilde{f} : \mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x} \rangle$ such that $\mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\left| \tilde{f}\left(\Psi\left(\mathbf{x}\right)\right) - f(\mathbf{x}) \right| \right]\!\!\right] \leq 2\varepsilon_1$. Note that $\|\mathbf{w}\|_2 \leq B$ and $\sup_{\mathbf{x} \in \mathcal{X}} \{\|\Psi(\mathbf{x})\|\} \leq 1$ by construction. We will now show that $\tilde{f}$ has bounded ε-insensitive loss.

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\ell_\varepsilon \left(\tilde{f}\left(\Psi\left(\mathbf{x}\right)\right), y(\mathbf{x}) \right) \right]\!\!\right] &= \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\ell_\varepsilon \left(f(\mathbf{x}), y(\mathbf{x}) \right) \right]\!\!\right] + \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\ell_\varepsilon \left(\tilde{f}\left(\Psi\left(\mathbf{x}\right)\right), y(\mathbf{x}) \right) - \ell_\varepsilon \left(f(\mathbf{x}), y(\mathbf{x}) \right) \right]\!\!\right] \\
&\leq \varepsilon_0 + \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\ell_\varepsilon \left(\tilde{f}\left(\Psi\left(\mathbf{x}\right)\right), y(\mathbf{x}) \right) - \ell_\varepsilon \left(f(\mathbf{x}), y(\mathbf{x}) \right) \right]\!\!\right] \\
&\leq \varepsilon_0 + \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\left| \tilde{f}\left(\Psi(\mathbf{x})\right) - f(\mathbf{x}) \right| \right]\!\!\right] \\
&\leq \varepsilon_0 + 2\varepsilon_1
\end{aligned}$$

where in the second step we have used the goodness properties of K, in the third step we used the fact that the ε-insensitive loss function is 1-Lipschitz in its first argument. Note that $\|\mathbf{w}\| \approx \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[w^2(\mathbf{x}) \right]\!\!\right]$ with high probability and if $\mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[w^2(\mathbf{x}) \right]\!\!\right] \ll B$ then we get a much better bound on the norm of $\mathbf{w}$. The excess loss incurred due to this landmarking step is, with probability $1 - \delta$, at most $32B\sqrt{\frac{\log(1/\delta)}{d}}$.

Now consider the following regularized ERM problem on n i.i.d. sample points:

$$\hat{\mathbf{w}} = \mathop{\arg\min}_{\mathbf{w}: \|\mathbf{w}\|_2 \leq B} \frac{1}{n} \sum_{i=1}^{n} \ell_\varepsilon \left(\langle \mathbf{w}, \Psi(\mathbf{x}_i) \rangle, y(\mathbf{x}_i) \right)$$

The final output of our learning algorithm shall be $\mathbf{x} \mapsto \langle \hat{\mathbf{w}}, \Psi(\mathbf{x}) \rangle$. Here we have $C_X = 1$, $C_L = 1$ since $\ell_\varepsilon(\cdot)$ is 1-Lipschitz and $C_W = B$. Thus by Lemma 5.23, we get that the excess loss incurred due to this regularized ERM step is at most $3B\sqrt{\frac{\log 1/\delta}{n}}$.

Since the ε-insensitive loss is related to the absolute error by $|x| \leq \ell_\varepsilon(x) + \varepsilon$ we have the total error (with respect to absolute loss) being incurred by our predictor to be, with

probability at least $1-2\delta$, at most

$$\varepsilon_0 + 32B\sqrt{\frac{\log(1/\delta)}{d}} + 3B\sqrt{\frac{\log 1/\delta}{n}} + \varepsilon$$

Taking $d = \mathcal{O}\left(\frac{B^2}{\varepsilon_1^2}\log\frac{1}{\delta}\right)$ unlabeled landmarks and $n = \mathcal{O}\left(\frac{B^2}{\varepsilon_1^2}\log\frac{1}{\delta}\right)$ labeled training points gives us our desired result.

5.7.2 Proof of Theorem 5.6

We prove the two parts of the result separately.

Part 1: Admissibility: Using Lemma 5.24 it is possible to obtain a vector $\mathbf{W}' = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\Phi_K(\mathbf{x}_i) \in \mathcal{H}_K$ with small loss such that $0 \leq \alpha_i, \alpha_i^* \leq p_i C$ and $\alpha_i\alpha_i^* = 0$ (these inequalities are a consequence of applying the KKT conditions). This allows us to construct a weight function $w_i = \frac{\alpha_i - \alpha_i^*}{p_i}$ such that $|w_i| \leq C$ and $\mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}[\![w(\mathbf{x}')K(\mathbf{x},\mathbf{x}')]\!] = \langle \mathbf{W}', \Phi_K(\mathbf{x})\rangle$ for all $\mathbf{x}\in\mathcal{X}$.

Thus we have

$$\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\!\left[\ell_\varepsilon\left(\mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}[\![w(\mathbf{x}')K(\mathbf{x},\mathbf{x}')]\!], y(\mathbf{x})\right)\right]\!\!\right] = \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}[\![\ell_\varepsilon\left(\langle\mathbf{W}', \Phi_K(\mathbf{x})\rangle, y(\mathbf{x})\right)]\!] \leq \frac{1}{2C\gamma^2} + \varepsilon_0.$$

Setting $C = \frac{1}{2\varepsilon_1\gamma^2}$ gives us our result.

We can use variational techniques to extend this to non-discrete distributions as well.

Part 2: Tightness: The tight example that we provide is an adaptation of the example given for large margin classification in (Srebro, 2007). However, our analysis differs from that of Srebro, partly necessitated by our choice of loss function.

Consider the following regression problem: $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\} \subset \mathbb{R}^3$, $\mathcal{D} = \left\{\frac{1}{2}-\varepsilon, \varepsilon, \varepsilon, \frac{1}{2}-\varepsilon\right\}$, $y = \{+1, +1, -1, -1\}$

$$\begin{aligned}
\mathbf{x}_1 &= \left(\gamma, \gamma, \sqrt{1-2\gamma^2}\right)\\
\mathbf{x}_2 &= \left(\gamma, -\gamma, \sqrt{1-2\gamma^2}\right)\\
\mathbf{x}_3 &= \left(-\gamma, \gamma, \sqrt{1-2\gamma^2}\right)\\
\mathbf{x}_4 &= \left(-\gamma, -\gamma, \sqrt{1-2\gamma^2}\right)
\end{aligned}$$

Clearly the vector $\mathbf{w} = (1,0,0)$ yields a predictor y' with no ε-insensitive loss for $\varepsilon = 0$ (i.e. $\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}[\![\ell_0\left(y(\mathbf{x}) - y'(\mathbf{x})\right)]\!] = 0$) at margin γ. Thus the native inner product $\langle\cdot,\cdot\rangle$ on $\mathbb{R}^3$ is a $(0,\gamma)$-good kernel for this particular regression problem.

Now consider any bounded weighing function on $\mathcal{X}$, $w = \{w_1, w_2, w_3, w_4\}$ and analyze the effectiveness of $\langle\cdot,\cdot\rangle$ as a similarity function. The output $\tilde{y}$ of the resulting predictor on the different points is given by $\tilde{y}_i = \sum_{j=1}^{4} p_j w_j \langle \mathbf{x}_i, \mathbf{x}_j\rangle$.

In particular, consider the output on the *heavy* points $\mathbf{x}_1$ and $\mathbf{x}_4$ (note that the analysis

in (Srebro, 2007) considers the *light* points $\mathbf{x}_2$ and $\mathbf{x}_3$ instead). We have

$$\begin{aligned}
\tilde{y}_1 &= \left(\frac{1}{2}-\varepsilon\right) w_1 + \varepsilon\left(1-2\gamma^2\right)(w_2+w_3) + \left(\frac{1}{2}-\varepsilon\right) w_4\left(1-4\gamma^2\right) \\
&= a + \left(\frac{1}{2}-\varepsilon\right)(w_1 + b w_4) \\
\tilde{y}_4 &= \left(\frac{1}{2}-\varepsilon\right) w_1\left(1-4\gamma^2\right) + \varepsilon\left(1-2\gamma^2\right)(w_2+w_3) + \left(\frac{1}{2}-\varepsilon\right) w_4 \\
&= a + \left(\frac{1}{2}-\varepsilon\right)(b w_1 + w_4)
\end{aligned}$$

for $a = \varepsilon\left(1-2\gamma^2\right)(w_2+w_3), b = \left(1-4\gamma^2\right)$. The main idea behind this choice is that the difference in the value of the predictor on these points is only due to the values of w_1 and w_4. Since the true values at these points are very different, this should force w_1 and w_4 to take large values unless a large error is incurred. To formalize this argument we lower bound the expected $\ell_0(\cdot)$ loss of this predictor by the loss incurred on these heavy points.

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\left[\ell_0\left(y(\mathbf{x})-\tilde{y}(\mathbf{x})\right)\right]\!\right] &\geq \left(\frac{1}{2}-\varepsilon\right)\left(\ell_0\left(y(\mathbf{x}_1)-\tilde{y}(\mathbf{x}_1)\right)+\ell_0\left(y(\mathbf{x}_4)-\tilde{y}(\mathbf{x}_4)\right)\right) \\
&= \left(\frac{1}{2}-\varepsilon\right)\left(|1-\tilde{y}(\mathbf{x}_1)|+|-1-\tilde{y}(\mathbf{x}_4)|\right) \\
&\geq \left(\frac{1}{2}-\varepsilon\right)\left(2-\tilde{y}(\mathbf{x}_1)+\tilde{y}(\mathbf{x}_4)\right) \\
&= \left(\frac{1}{2}-\varepsilon\right)\left(2-\left(\frac{1}{2}-\varepsilon\right)(1-b)(w_4-w_1)\right) \\
&= \left(\frac{1}{2}-\varepsilon\right)\left(2-\left(\frac{1}{2}-\varepsilon\right)\left(4\gamma^2\right)(w_4-w_1)\right)
\end{aligned}$$

where in the second step we use the fact that $\ell_0(x) = |x|$ and in the third step we used the fact that $|a|+|b| \geq a-b$. Thus, in order to have expected error at most ε_1, we require

$$w_4 - w_1 \geq \frac{1}{4\gamma^2}\left(2-\frac{\varepsilon_1}{\frac{1}{2}-\varepsilon}\right)\frac{1}{\frac{1}{2}-\varepsilon} = \frac{1}{4\varepsilon_1\gamma^2}$$

for the setting $\varepsilon = \frac{1}{2}-\varepsilon_1$. Thus we have $|w_1|+|w_4| \geq w_4 - w_1 \geq \frac{1}{4\varepsilon_1\gamma^2}$ which implies $\max\left(|w_1|,|w_4|\right) \geq \frac{1}{8\varepsilon_1\gamma^2}$ which proves the result.

5.7.3 Proof of Theorem 5.8

The proof of this theorem essentially parallels that of (Balcan et al., 2008a, Theorem 8) but diverges later since the aim there is to preserve margin violations whereas we wish to preserve loss under the absolute loss function. Sample d *landmark points* $\mathcal{L} = \{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$ from the distribution $\mathcal{D}$ and construct the map $\Psi_{\mathcal{L}} : \mathbf{x} \mapsto (K(\mathbf{x},\mathbf{x}_1), \ldots, K(\mathbf{x},\mathbf{x}_d))$ and consider the linear operator $\tilde{f} : \mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x}\rangle$ with $\mathbf{w}_i = \frac{w(\mathbf{x}_i)R(\mathbf{x}_i)}{d_{\text{info}}}$ where $d_{\text{info}} = \sum_{i=1}^{d} R(\mathbf{x}_i)$

is the number of informative landmarks. In the following we will refer to $\tilde{f}$ and $\mathbf{w}$ interchangeably. This ensures that $\left\|\tilde{f}\right\|_1 := \|\mathbf{w}\|_1 \leq B$. Note that we have chosen an L_1 normalized weight vector instead of an L_2 normalized one like we had in Lemma 5.22. This is due to a subsequent use of sparsity promoting regularizers whose analysis requires the existence of bounded L_1 norm predictors.

Using the arguments given for Lemma 5.22 and Theorem 5.5, we can show that if $d_{\text{info}} = \Omega\left(\frac{B^2}{\varepsilon_1^2}\log\frac{1}{\delta}\right)$ (i.e. if we have collected enough informative landmarks), then we are done. However, the Chernoff bound (lower tail) tells us that for $d = \Omega\left(\frac{B^2}{\tau\varepsilon_1^2}\log\frac{1}{\delta}\right)$, this will happen with probability $1-\delta$. Moreover, the Chernoff bound (upper tail) tells us that, simultaneously we will also have $d_{\text{inf}} \leq \frac{3d\tau}{2}$. Together these prove the claim.

5.7.4 Proof of Lemma 5.12

The result for non-sparse vectors, that applies here as well, follows in a straightforward manner from (Kakade et al., 2008, Theorem 1, Example 3.1(2)) and (Bartlett and Mendelson, 2002) which we reproduce for completeness. Since the L_1 and L_∞ norms are dual to each other, for any $\mathbf{w} \in (\mathbb{R}^+)^d$ such that $\|\mathbf{w}\|_1 = B$ and any $\mu \in \Delta^d$, where Δ^d is the probability simplex in d dimensions, the Kullback-divergence function $\mathrm{KL}\left(\frac{\mathbf{w}}{B}\middle\|\,\mu\right)$ is $\frac{1}{B^2}$-strongly convex with respect to the L_1 norm. We can remove the positivity constraints on the coordinates of $\mathbf{w}$ by using the standard method of introducing additional dimensions that encode negative components of the (signed) weight vector.

Using (Kakade et al., 2008, Theorem 1), thus, we can bound the Rademacher complexity of the function class $\mathcal{F}$ as $\mathcal{R}_n(\mathcal{F}) \leq C_X C_W\sqrt{\frac{2\log 2d}{n}}$. Next, using the Lipschitz properties of the loss function, a result from (Bartlett and Mendelson, 2002) allows us to bound the excess error by $2C_L\mathcal{R}_n(\mathcal{F}) + C_L C_X C_W\sqrt{\frac{\log(1/\delta)}{2n}}$. The result then follows.

5.7.5 Proof of Theorem 5.9

Using Theorem 5.8, we first bound the excess loss due to landmarking by $32B\sqrt{\frac{\log(1/\tau\delta)}{d}}$. Next we set up the (dummy) Ivanov regularized regression problem (given in Equation 5.2) with the training loss being the objective and regularization parameter $C_W = B$. The training loss incurred by the minimizer of that problem $\mathbf{w}_{\text{inter}}$ is, with probability at least $(1-\delta)$, bounded by $\hat{\mathcal{L}}(\mathbf{w}_{\text{inter}}) \leq \varepsilon_0 + 32B\sqrt{\frac{\log(1/\delta)}{\tau d}} + B\sqrt{\frac{\log(1/\delta)}{n}}$ due to the guarantees of Theorem 5.8. Next, we run the Forward Greedy Selection algorithm of Shalev-Shwartz et al. (specialized to our case in Algorithm 6) and obtain another predictor $\hat{\mathbf{w}}$ with L_1 norm bounded by B that has empirical error at most $\hat{\mathcal{L}}(\hat{\mathbf{w}}) \leq \hat{\mathcal{L}}(\mathbf{w}_{\text{inter}}) + \sqrt{\frac{18B^2}{k}}$. Finally, using Lemma 5.12, we bound the true ε-insensitive loss incurred by $\hat{\mathbf{w}}$ by $\hat{\mathcal{L}}(\hat{\mathbf{w}}) + 2B\sqrt{\frac{2\log(2d)}{n}} + B\sqrt{\frac{\log(1/\delta)}{2n}}$. Adding ε to convert this loss to absolute loss we get that with probability at most $(1-3\delta)$, we will output a k-sparse predictor in a d-dimensional space with absolute

regression loss at most

$$\varepsilon_0 + 32B\sqrt{\frac{\log(1/\delta)}{\tau d}} + \sqrt{\frac{18B^2}{k}} + 2B\sqrt{\frac{2\log(2d)}{n}} + 2B\sqrt{\frac{\log(1/\delta)}{2n}} + \varepsilon$$

5.7.6 Proof of Theorem 5.10

To prove the first part, construct a new weight function $\tilde{w}(\mathbf{x}) = \operatorname{sign}(w(\mathbf{x})) \cdot \bar{w}$. Note that we have $|\tilde{w}(\mathbf{x})| \leq \bar{w} \leq B$. Also construct the choice function as follows: for any $\mathbf{x}$, let $\mathbb{P}[R(\mathbf{x}) = 1|\mathbf{x}] = \frac{|w(\mathbf{x})|}{B}$. This gives us $\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}[\![R(\mathbf{x})]\!] = \frac{\bar{w}}{B}$. Then for any $\mathbf{x}$, we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}[\![\tilde{w}(\mathbf{x}')K(\mathbf{x},\mathbf{x}')|R(\mathbf{x}')]\!] &= \mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}\left[\!\!\left[\operatorname{sign}(w(\mathbf{x}))\,\bar{w}K(\mathbf{x},\mathbf{x}')\frac{|w(\mathbf{x})|}{B}\right]\!\!\right] \Big/ \mathop{\mathbb{P}}_{\mathbf{x}\sim\mathcal{D}}[R(\mathbf{x}) = 1] \\
&= \mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}\left[\!\!\left[w(\mathbf{x})K(\mathbf{x},\mathbf{x}')\frac{\bar{w}}{B}\right]\!\!\right] \Big/ \left(\frac{\bar{w}}{B}\right) \\
&= \mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}[\![w(\mathbf{x}')K(\mathbf{x},\mathbf{x}')]\!]
\end{aligned}$$

Since $f(\mathbf{x}) = \mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}[\![w(\mathbf{x}')K(\mathbf{x},\mathbf{x}')]\!]$ has small ε-insensitive loss by (ε_0, B)-goodness of K, we have our result. To prove the second part, construct a new weight function $\tilde{w}(\mathbf{x}) = \frac{w(\mathbf{x})}{\tau}\mathbb{P}[R(\mathbf{x}) = 1|\mathbf{x}]$. Note that we have $|\tilde{w}(\mathbf{x})| \leq \frac{B}{\tau}$. Then for any $\mathbf{x}$, we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}[\![\tilde{w}(\mathbf{x}')K(\mathbf{x},\mathbf{x}')]\!] &= \mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}\left[\!\!\left[\frac{w(\mathbf{x}')}{\tau}R(\mathbf{x}')K(\mathbf{x},\mathbf{x}')\right]\!\!\right] \\
&= \mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}\left[\!\!\left[\frac{w(\mathbf{x}')}{\tau}K(\mathbf{x},\mathbf{x}')|R(\mathbf{x}')\right]\!\!\right] \mathop{\mathbb{P}}_{\mathbf{x}'\sim\mathcal{D}}\left[R(\mathbf{x}') = 1\right] \\
&= \mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}[\![w(\mathbf{x}')K(\mathbf{x},\mathbf{x}')|R(\mathbf{x}')]\!]
\end{aligned}$$

Since $f(\mathbf{x}) = \mathop{\mathbb{E}}_{\mathbf{x}'\sim\mathcal{D}}[\![w(\mathbf{x}')K(\mathbf{x},\mathbf{x}')|R(\mathbf{x}')]\!]$ has small ε-insensitive loss by (ε_0, B, τ)-goodness of K, we have our result.

Using the above result we get our admissibility guarantee.

Corollary 5.25. *Every PSD kernel that is (ε_0, γ)-good for a regression problem is, for any $\varepsilon_1 > 0$, $\left(\varepsilon_0 + \varepsilon_1, \mathcal{O}\left(\frac{1}{\varepsilon_1\gamma^2}\right), 1\right)$-good as a similarity function as well.*

The above result is rather weak with respect to the sparsity parameter τ since we have made no assumptions on the distribution of the dual variables α_i, α_i^* in the proof of Theorem 5.6 which is why we are forced to use the (weak) inequality $\frac{\bar{w}}{B} \leq 1$. Any stronger assumptions on the kernel goodness shall also strengthen this admissibility result.

5.7.7 Proof of Theorem 5.16

We use Lemma 5.22 to construct a landmarked space with a linear predictor $\tilde{f} : \mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x} \rangle$ such that $\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\!\left[\left|\tilde{f}(\Psi(\mathbf{x})) - f(\mathbf{x})\right|\right]\!\!\right] \leq 2\varepsilon_1$. As before, we have $\|\mathbf{w}\|_2 \leq B$ and $\sup_{\mathbf{x}\in\mathcal{X}}\{\|\Psi(\mathbf{x})\|\} \leq 1$. In the following, we shall first show bounds on the mislabeling error

i.e $\mathop{\mathbb{P}}_{\mathbf{x}\sim\mathcal{D}}[\hat{y}(\mathbf{x}) \neq y(\mathbf{x})]$. Next, we shall convert these bounds into ordinal regression loss by introducing a *spacing* parameter into the model.

Since the γ-margin loss function is 1-Lipschitz, we get

$$\begin{aligned}\left[\tilde{f}(\Psi(\mathbf{x})) - b_{y(\mathbf{x})}\right]_\gamma &\leq \left[f(\mathbf{x}) - b_{y(\mathbf{x})}\right]_\gamma + 2\varepsilon_1 \\ \left[b_{y(\mathbf{x})+1} - \tilde{f}(\Psi(\mathbf{x}))\right]_\gamma &\leq \left[b_{y(\mathbf{x})+1} - f(\mathbf{x})\right]_\gamma + 2\varepsilon_1\end{aligned}$$

Which gives us, upon taking expectations on both sides,

$$\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\left[\tilde{f}(\Psi(\mathbf{x})) - b_{y(\mathbf{x})}\right]_\gamma + \left[b_{y(\mathbf{x})+1} - \tilde{f}(\Psi(\mathbf{x}))\right]_\gamma\right] \leq \varepsilon_0 + 4\varepsilon_1$$

Lemma 5.22 guarantees the excess loss due to landmarking to be at most $64B\sqrt{\frac{\log(1/\delta)}{d}}$. Moreover, since the γ-margin loss is 1-Lipschitz, Lemma 5.23 allows us to bound excess loss due to training by $3B\sqrt{\frac{\log(1/\delta)}{n}}$ so that the learned predictor has γ-margin loss at most $\varepsilon_0 + \varepsilon_1$ for any ε_1 given large enough d and n. Now, from the definition of the γ-margin loss it is clear that if the loss is greater than γ then it indicates a mislabeling. Hence, the mislabeling error is bounded by $\frac{\varepsilon_0+\varepsilon_1}{\gamma}$.

This may be unsatisfactory if $\gamma \ll 1$ - to remedy such situations we show that we can bound the 1-margin loss directly. Starting from $\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\left|\tilde{f}(\Psi(\mathbf{x})) - f(\mathbf{x})\right|\right] < 2\varepsilon_1$, we can also deduce

$$\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\left[1 - \tilde{f}(\Psi(\mathbf{x})) + b_{y(\mathbf{x})}\right]_+ + \left[1 - b_{y(\mathbf{x})+1} + \tilde{f}(\Psi(\mathbf{x}))\right]_+\right] \leq \varepsilon_0 + 4\varepsilon_1$$

We can bound the excess training error for this loss function as well. Since the 1-margin loss directly bounds the mislabeling error, combining the two arguments we get the second part of the claim.

However, the margin losses themselves do not present any bound on the ordinal regression error. This is because, if the thresholds are closely spaced together, then even an instance of gross ordinal regression loss could correspond to very small margin loss. To remedy this, we introduce a *spacing* parameter into the model. We say that a set of thresholds is Δ-spaced if $\min_{i\in[r]}\{|b_i - b_{i+1}|\} \geq \Delta$. Such a condition can easily be incorporated into the model of Chu and Keerthi (2007) as a constraint in the optimization formulation.

Suppose that a given instance has ordinal regression error $\ell_{\text{ord}}(\hat{y}(\mathbf{x}), y(\mathbf{x})) = k$. This can happen if the point was given a label k labels below (or above) its correct label. Also suppose that the γ-margin error in this case is $[\hat{y}(\mathbf{x}) - y(\mathbf{x})]_\gamma = h$. Without loss of generality, assume that the point $\mathbf{x}$ of label $k+1$ was given the label 1 giving an ordinal regression loss of $l_{\text{ord}} = k$ (a similar analysis would hold if the point of label 1 were to be given a label $k+1$ by symmetry of the margin loss formulation with respect to left and right thresholds). In this case the value of the underlying regression function must lie between

b_1 and b_2 and thus, the margin loss h satisfies $h \geq b_{k+1} + \gamma - b_2 = \gamma + \sum_{i=2}^{k} (b_{i+1} - b_i) \geq \gamma + (k-1)\,\Delta$. Thus, if the margin loss is at most h, the ordinal regression error must satisfy $\ell_{\text{ord}}\left(\hat{y}(\mathbf{x}), y(\mathbf{x})\right) \leq \frac{\left[\hat{y}(\mathbf{x}) - b_{y(\mathbf{x})}\right]_\gamma + \left[b_{y(\mathbf{x})+1} - \hat{y}(\mathbf{x})\right]_\gamma - \gamma}{\Delta} + 1$. Let $\psi_\Delta(x) = \frac{x+\Delta-1}{\Delta}$. Using the bounds on the γ-margin and 1-margin losses given above, we get the first part of the claim.

5.7.8 Proof Theorem 5.17

We prove the two parts of the result separately.

Part 1: Admissibility: As before, using Lemma 5.24 it is possible to obtain a vector $\mathbf{W}' = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) \Phi_K(\mathbf{x}_i) \in \mathcal{H}_K$ such that $0 \leq \alpha_i, \alpha_i^* \leq p_i C$ (by applying the KKT conditions) and the following holds:

$$\mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\left[b_{y(\mathbf{x})} + 1 - \left\langle \mathbf{W}', \Phi_K(\mathbf{x}) \right\rangle \right]_+ + \left[\left\langle \mathbf{W}', \Phi_K(\mathbf{x}) \right\rangle - b_{y(\mathbf{x})+1} + 1 \right]_+ \right]\!\!\right] < \frac{1}{2C\gamma^2} + \varepsilon_0 \quad (5.3)$$

This allows us to construct a weight function $w_i = \frac{\alpha_i - \alpha_i^*}{p_i}$ such that $|w_i| \leq 2C$ (since we do not have any guarantee that $\alpha_i \alpha_i^* = 0$) and $\mathop{\mathbb{E}}_{\mathbf{x}' \sim \mathcal{D}} [\![w(\mathbf{x}') K(\mathbf{x}, \mathbf{x}')]\!] = \langle \mathbf{W}', \Phi_K(\mathbf{x}) \rangle$ for all $\mathbf{x} \in \mathcal{X}$. Denoting $f(\mathbf{x}) := \mathop{\mathbb{E}}_{\mathbf{x}' \sim \mathcal{D}} [\![w(\mathbf{x}') K(\mathbf{x}, \mathbf{x}')]\!]$ for convenience gives us

$$\begin{aligned} \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\left[f(\mathbf{x}) - b_{y(\mathbf{x})} \right]_1 + \left[b_{y(\mathbf{x})+1} - f(\mathbf{x}) \right]_1 \right]\!\!\right] &= \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\left[1 - f(\mathbf{x}) + b_{y(\mathbf{x})} \right]_+ + \left[1 - b_{y(\mathbf{x})+1} + f(\mathbf{x}) \right]_+ \right]\!\!\right] \\ &\leq \frac{1}{2C\gamma^2} + \varepsilon_0 \end{aligned}$$

where in the first step we used $[x]_1 = [1 - x]_+$. Now use the fact $[x]_1 = \frac{1}{\gamma} [\gamma x]_\gamma$ to get the following:

$$\mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\left[\gamma_1 f(\mathbf{x}) - \gamma_1 b_{y(\mathbf{x})} \right]_{\gamma_1} + \left[\gamma_1 b_{y(\mathbf{x})+1} - \gamma_1 f(\mathbf{x}) \right]_{\gamma_1} \right]\!\!\right] \leq \frac{\gamma_1}{2C\gamma^2} + \gamma_1 \varepsilon_0$$

Note that it is not possible to perform the analysis on the loss function $[\cdot]_\gamma$ directly since using it requires us to scale the threshold values by a factor of γ_1 that makes the result in Equation 5.3 unusable. Hence we first perform the analysis for $[\cdot]_1$, utilize Equation 5.3 and then interpret the resulting inequality in terms of $[\cdot]_{\gamma_1}$.

Setting $2C = \frac{\gamma_1}{\varepsilon_1 \gamma^2}$, using $w'(\mathbf{x}) = \gamma_1 w(\mathbf{x})$ as weights, using $b'_j = \gamma_1 b_j$ as the thresholds and noting that the new bound on the weights is $|w'_i| \leq 2C\gamma_1$ gives us the result. As before, using variational optimization techniques, this result can be extended to non-discrete distributions as well.

In particular, setting $\gamma_1 = \gamma$ gives us that any PSD kernel that is (ε_0, γ)-good for an ordinal regression problem is also $\left(\gamma\varepsilon_0 + \varepsilon_1, \frac{1}{\varepsilon_1}\right)$-good as a similarity function with respect to the γ-margin loss.

Part 2: Tightness: We adapt our running example (used for proving the lower bound for real regression) for the case of ordinal regression as well. Consider the points with value

-1 as having label 1 and those having value $+1$ as having label 2. Clearly, $w = (1, 0, 0)$ along with the thresholds $b_1 = -\infty$ and $b_2 = 0$ establishes the native inner product as a $(0, \gamma)$-good PSD kernel.

Now consider the heavy points yet again and some weight function and threshold b_2 (b_1 is always fixed at $-\infty$) that is supposed to demonstrate the goodness of the inner product kernel as a similarity function. Clearly we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\!\left[\left[f(\mathbf{x}) - b_{y(\mathbf{x})}\right]_{\gamma_1} + \left[b_{y(\mathbf{x})+1} - f(\mathbf{x})\right]_{\gamma_1}\right]\!\!\right] &\geq \left(\frac{1}{2} - \varepsilon\right)\left(\left[f(\mathbf{x}_1) - b_2\right]_{\gamma_1} + \left[b_2 - f(\mathbf{x}_4)\right]_{\gamma_1}\right) \\
&= \left(\frac{1}{2} - \varepsilon\right)\left(\left[\gamma_1 - f(\mathbf{x}_1) + b_2\right]_{+} + \left[\gamma_1 - b_2 + f(\mathbf{x}_4)\right]_{+}\right) \\
&\geq \left(\frac{1}{2} - \varepsilon\right)\left(2\gamma_1 - f(\mathbf{x}_1) + f(\mathbf{x}_4)\right) \\
&= \left(\frac{1}{2} - \varepsilon\right)\left(2\gamma_1 - \left(\frac{1}{2} - \varepsilon\right)(1 - b)\left(w_4 - w_1\right)\right) \\
&= \left(\frac{1}{2} - \varepsilon\right)\left(2\gamma_1 - \left(\frac{1}{2} - \varepsilon\right)\left(4\gamma^2\right)\left(w_4 - w_1\right)\right)
\end{aligned}$$

where in the third step we have used the fact that $[a]_+ + [b]_+ \geq a + b$. Thus, in order to have expected error at most ε_1, we must have

$$w_4 - w_1 \geq \frac{1}{4\gamma^2}\left(2\gamma_1 - \frac{\varepsilon_1}{\frac{1}{2} - \varepsilon}\right)\frac{1}{\frac{1}{2} - \varepsilon} = \frac{\gamma_1^2}{4\varepsilon_1\gamma^2}$$

by setting $\varepsilon = \frac{1}{2} - \frac{\varepsilon_1}{\gamma_1}$ which then proves the result after applying an averaging argument.

5.7.9 Proof of Theorem 5.20

As before, we use Lemma 5.22 to construct a landmarked space with a linear predictor $\tilde{f} : \mathbf{x} \mapsto \langle \mathbf{w}, \mathbf{x} \rangle$ such that $\mathop{\mathbb{E}}_{\mathbf{z}\sim\mu}\left[\!\!\left[\left|\tilde{f}\left(\Psi\left(\mathbf{z}\right)\right) - f(\mathbf{z})\right|\right]\!\!\right] \leq 2\varepsilon_1$. We have $\|\mathbf{w}\|_2 \leq B$ and $\sup_{\mathbf{x}\in\mathcal{X}}\{\|\Psi(\mathbf{x})\|\} \leq 1$. Now lets overload notation to denote by $\Psi(\mathbf{x})$ the concatenation of the images of the m document-query pairs in $\mathbf{x}$ under $\Psi(\cdot)$ and by $\tilde{f}(\Psi(\mathbf{x}))$, the m-dimensional vector obtained by applying $\tilde{f}$ to each of the m components of $\Psi(\mathbf{x})$.

Since the squared loss function is $2B$-Lipschitz in its first argument in the region of interest, we get

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\!\left[\ell_{\text{sq}}\left(\tilde{f}(\Psi(\mathbf{x})), \eta(r(\mathbf{x}))\right)\right]\!\!\right] &= \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\!\left[\sum_{i=1}^{m}\ell_{\text{sq}}\left(\tilde{f}(\Psi(\mathbf{z}_i)), \eta(r(\mathbf{x}))_i\right)\right]\!\!\right] \\
&= \sum_{i=1}^{m}\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\!\left[\ell_{\text{sq}}\left(\tilde{f}(\Psi(\mathbf{z}_i)), \eta(r(\mathbf{x}))_i\right)\right]\!\!\right] \\
&= \sum_{i=1}^{m}\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}\left[\!\!\left[\ell_{\text{sq}}\left(f(\mathbf{z}_i), \eta(r(\mathbf{x}))_i\right)\right]\!\!\right] +
\end{aligned}$$

$$
\begin{aligned}
& \sum_{i=1}^{m} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}} \left[\!\!\left[\ell_{\mathrm{sq}}\left(\tilde{f}(\Psi(\mathbf{z}_i)), \eta(r(\mathbf{x}))_i\right) - \ell_{\mathrm{sq}}\left(f(\mathbf{z}_i), \eta(r(\mathbf{x}))_i\right) \right]\!\!\right] \\
\leq & \sum_{i=1}^{m} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}} \left[\!\!\left[\ell_{\mathrm{sq}}\left(f(\mathbf{z}_i), \eta(r(\mathbf{x}))_i\right) \right]\!\!\right] + 2B \sum_{i=1}^{m} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}} \left[\!\!\left[\left| \tilde{f}(\Psi(\mathbf{z}_i)) - f(\mathbf{z}_i) \right| \right]\!\!\right] \\
= & \sum_{i=1}^{m} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}} \left[\!\!\left[\ell_{\mathrm{sq}}\left(f(\mathbf{z}_i), \eta(r(\mathbf{x}))_i\right) \right]\!\!\right] + 2B \sum_{i=1}^{m} \mathop{\mathbb{E}}_{\mathbf{z}\sim\mu} \left[\!\!\left[\left| \tilde{f}(\Psi(\mathbf{z})) - f(\mathbf{z}) \right| \right]\!\!\right] \\
\leq & \sum_{i=1}^{m} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}} \left[\!\!\left[\ell_{\mathrm{sq}}\left(f(\mathbf{z}_i), \eta(r(\mathbf{x}))_i\right) \right]\!\!\right] + 4Bm\varepsilon_1 \\
= & \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}} \left[\!\!\left[\sum_{i=1}^{m} \ell_{\mathrm{sq}}\left(f(\mathbf{z}_i), \eta(r(\mathbf{x}))_i\right) \right]\!\!\right] + 4Bm\varepsilon_1 \\
= & \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}} \left[\!\!\left[\ell_{\mathrm{sq}}\left(f(\mathbf{x}), \eta(r(\mathbf{x}))\right) \right]\!\!\right] + 4Bm\varepsilon_1 \\
\leq & \varepsilon_0 + 4Bm\varepsilon_1
\end{aligned}
$$

where $\mathbf{x} = (q, p_1, \ldots, p_m)$ and $\mathbf{z}_i = (p_i, q)$. In the first and the last but one step we have used decomposability of the squared loss, in the fourth step we have used Lipschitz properties of the squared loss, in the fifth step we have used properties of the generative mechanism assumed for ranking instances, in the sixth step we have used the guarantee given by Lemma 5.22. Throughout we have repeatedly used linearity of expectation. This bounds the excess error due to landmarking to d dimensions by $64B^2m^2\sqrt{\frac{\log(1/\delta)}{d}}$ using Lemma 5.22. Similarly, Lemma 5.23 also allows us to bound the excess error due to training by $3B^2\sqrt{\frac{\log(1/\delta)}{n}}$ which puts our total squared loss at $\varepsilon_0 + \varepsilon_1$ for large enough d and n.

We now invoke (Ravikumar et al., 2011, Theorem 10) that states that if the surrogate loss function $\ell(\cdot,\cdot)$ being used is a Bregman divergence generated by a function that is C_S-strongly convex with respect to some norm $\|\cdot\|$ then we can bound $\ell_{\mathrm{NDCG}}(\mathbf{s}, \mathbf{r}) \leq \frac{C_F}{\sqrt{C_S}} \cdot \sqrt{\ell(\mathbf{s}, \mathbf{r})}$ where $C_F = 2\left\| \left(\frac{1}{F(1)}, \ldots, \frac{1}{F(m)}\right)^\top \right\|_*$, F is the decay function used in the definition of NDCG and $\|\cdot\|_*$ is the dual norm of $\|\cdot\|$. Note that we are using the "noiseless" version of the result where $r(\mathbf{x})$ is a deterministic function of $\mathbf{x}$.

In our case the squared loss is 2-strongly convex with respect to the L_2 norm which is its own dual. Hence $C_S = 2$ and $C_F = \mathcal{O}\left(\sqrt{\frac{m}{\log m}}\right)$, if $\hat{f} : \mathbf{x} \mapsto \langle \hat{\mathbf{w}}, \Psi(\mathbf{x}) \rangle$ is our final output, we get, for some constant C,

$$
\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}} \left[\!\!\left[\ell_{\mathrm{NDCG}}\left(\hat{f}(\mathbf{x}), r(\mathbf{x})\right) \right]\!\!\right] \leq C\sqrt{\frac{m}{\log m}} \cdot \sqrt{\varepsilon_0 + 4Bm\varepsilon_1} \leq C\sqrt{\frac{m}{\log m}} \cdot \sqrt{\varepsilon_0} + C\frac{2m}{\sqrt{\log m}} \cdot \sqrt{B\varepsilon_1}
$$

which proves the claim. This affects the bounds given by Lemmata 5.22 and 5.23 since the dependence of the excess error on d and n will now be in terms of the inverse of their fourth roots instead of inverse of the square roots as was the case in regression and ordinal regression.

5.7.10 Proof of Theorem 5.21

For notational convenience, we shall assume that the RKHS $\mathcal{H}_K$ is finite dimensional so that we can talks in terms of finite dimensional matrices and vectors. As before, let $f(\mathbf{z}; \mathbf{W}) = \langle \mathbf{W}, \Phi_K(\mathbf{z}) \rangle$ and let $\mathbf{W}'$ be the minimizer of the following program.

$$\begin{aligned}
& \min_{\mathbf{W} \in \mathcal{H}_K} && \frac{1}{2} \|\mathbf{W}\|_{\mathcal{H}_K}^2 + C \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} [\![\ell_{\text{sq}} (f(\mathbf{x}; \mathbf{W}), \eta(r(\mathbf{x})))]\!] \\
\equiv & \min_{\mathbf{W} \in \mathcal{H}_K} && \frac{1}{2} \|\mathbf{W}\|_{\mathcal{H}_K}^2 + C \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} \left[\!\!\left[\sum_{i=1}^{m} \ell_{\text{sq}} (f(\mathbf{z}_i; \mathbf{W}), \eta(r(\mathbf{x}))_i) \right]\!\!\right] \\
\equiv & \min_{\mathbf{W} \in \mathcal{H}_K} && \frac{1}{2} \|\mathbf{W}\|_{\mathcal{H}_K}^2 + C \sum_{i=1}^{m} \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}} [\![\ell_{\text{sq}} (f(\mathbf{z}_i; \mathbf{W}), \eta(r(\mathbf{x}))_i)]\!] \\
\equiv & \min_{\mathbf{W} \in \mathcal{H}_K} && \frac{1}{2} \|\mathbf{W}\|_{\mathcal{H}_K}^2 + mC \mathop{\mathbb{E}}_{\mathbf{z} \sim \mu} [\![\ell_{\text{sq}} (f(\mathbf{z}; \mathbf{W}), \tilde{r}(\mathbf{z}))]\!] + C_{\mathcal{D}}
\end{aligned}$$

where for any $\mathbf{z} \in \mathcal{Q} \times \mathcal{P}$, $\tilde{r}(\mathbf{z})$ gives us the expected normalized relevance of this document-query pair across ranking instances and $C_{\mathcal{D}}$ is some constant independent of $\mathbf{W}$ and dependent solely on the underlying distributions.

Using the goodness of the kernel K and the argument given in the proof of Lemma 5.24, it is possible to show that the vector $\mathbf{W}'$ has squared loss at most $\frac{1}{2C\gamma^2} + \varepsilon_0$. Hence the only task remaining is to show that their exists a bounded weight function w such that for all $\mathbf{z} \in \mathcal{P} \times \mathcal{Q}$, we have $f(\mathbf{z}; \mathbf{W}) = \langle \mathbf{W}', \Phi_K(\mathbf{z}) \rangle = \mathop{\mathbb{E}}_{\mathbf{z}' \sim \mu} [\![w(\mathbf{z}) K(\mathbf{z}, \mathbf{z}')]\!]$ which will prove the claim.

To do so we assume that the (finite) set of document-query pairs is $(\mathbf{z}_1, \ldots, \mathbf{z}_k)$ with $\mathbf{z}_i$ having probability μ_i and relevance $r_i = \tilde{r}(\mathbf{z}_i)$. Then the above program can equivalently be written as

$$\begin{aligned}
& \min_{\mathbf{W} \in \mathcal{H}_K} && \frac{1}{2} \|\mathbf{W}\|_{\mathcal{H}_K}^2 + mC \sum_{i=1}^{k} \mu_i \ell_{\text{sq}} (\langle \mathbf{W}, \Phi_K(\mathbf{z}_i) \rangle, r_i) \\
\equiv & \min_{\mathbf{W} \in \mathcal{H}_K} && \frac{1}{2} \|\mathbf{W}\|_{\mathcal{H}_K}^2 + mC \left\| \sqrt{P} X^\top \mathbf{W} - \sqrt{P} \mathbf{r} \right\|_2^2 \\
\equiv & \min_{\mathbf{W} \in \mathcal{H}_K} && \frac{1}{2} \|\mathbf{W}\|_{\mathcal{H}_K}^2 + mC \left\| \tilde{X}^\top \mathbf{W} - \tilde{\mathbf{r}} \right\|_2^2 \\
\equiv & \min_{\alpha \in \mathbb{R}^{mn}} && \frac{1}{2} \|X\alpha\|_{\mathcal{H}_K}^2 + mC \left\| \tilde{X}^\top X \alpha - \tilde{\mathbf{r}} \right\|_2^2
\end{aligned}$$

where $X = (\Phi_K(\mathbf{z}_1), \ldots, \Phi_K(\mathbf{z}_k))$, $\mathbf{r} = (r_1, \ldots, r_k)^\top$, P is the $k \times k$ diagonal matrix with $P_{ii} = \mu_i$, $\tilde{X} = X\sqrt{P}$ and $\tilde{\mathbf{r}} = \sqrt{P}\mathbf{r}$. The last step follows by the Representer Theorem which tells us that at the optima, $\mathbf{W}' = X\alpha$ for some $\alpha \in \mathbb{R}^k$.

Some simple linear algebra shows us that the minimizer α has the form

$$\alpha \quad = \quad \left(X^\top \tilde{X} \tilde{X}^\top X + \frac{1}{2mC} X^\top X \right)^{-1} X^\top \tilde{X} \tilde{\mathbf{r}}$$

$$
\begin{aligned}
&= \left(GPG + \frac{G}{2mC}\right)^{-1} GP\mathbf{r} \\
&= \left(PG + \frac{I}{2mC}\right)^{-1} G^{-1}GP\mathbf{r} \\
&= \left(PG + \frac{I}{2mC}\right)^{-1} P\mathbf{r}
\end{aligned}
$$

where $G = X^\top X$ is the Gram matrix given by the kernel K. In the third step we have assumed that G does not have vanishing eigenvalues which can always be ensured by adding a small positive constant to the diagonal. Thus we have

$$\left(PG + \frac{I}{2mC}\right)\alpha = P\mathbf{r}$$

looking at the i^{th} element of both sides we have

$$\mu_i \sum_{j=1}^{k} \alpha_j K(\mathbf{z}_i, \mathbf{z}_j) + \frac{\alpha_i}{m2C} = \mu_i r_i$$

which gives us $\alpha_i = 2mC\mu_i \left(r_i - \langle \mathbf{W}', \Phi_K(\mathbf{z}_i)\rangle\right)$. Now assume, without loss of generality, that the relevance scores are normalized, i.e. $r_i \leq 1$ for all i. Thus we have

$$\frac{1}{2}\left\|\mathbf{W}'\right\|^2_{\mathcal{H}_K} + mC\left\|\tilde{X}^\top \mathbf{W}' - \tilde{\mathbf{r}}\right\|_2^2 \leq \frac{1}{2}\left\|\mathbf{0}\right\|^2_{\mathcal{H}_K} + mC\left\|\tilde{X}^\top \mathbf{0} - \tilde{\mathbf{r}}\right\|_2^2$$

which gives us $\frac{1}{2}\left\|\mathbf{W}'\right\|^2_{\mathcal{H}_K} \leq mC\left\|\tilde{\mathbf{r}}\right\|_2^2 \leq mC\sum_{i=1}^{k}\mu_i = mC$ which gives us $\|\mathbf{W}'\| \leq \sqrt{2mC}$. Since the kernel is already a normalized kernel, $\|\Phi_K(\mathbf{z}_i)\| \leq 1$ which gives us, by an application of Cauchy-Schwartz, $|\alpha_i| \leq 2mC\mu_i(1 + \sqrt{m2C}) \leq 5\mu_i mC\sqrt{mC}$.

If we now establish a weight function over the domain $w_i = \frac{\alpha_i}{\mu_i}$, then $|w_i| \leq 5mC\sqrt{mC}$ and we can show that for all $\mathbf{z}$, we have $\langle \mathbf{W}', \Phi_K(\mathbf{z})\rangle = \mathop{\mathbb{E}}_{\mathbf{z}'\sim\mu}[\![w(\mathbf{z})K(\mathbf{z}, \mathbf{z}')]\!]$. Setting $C = \frac{1}{2\varepsilon_1\gamma^2}$ finishes the proof.

5.8 Supplementary Experimental Results

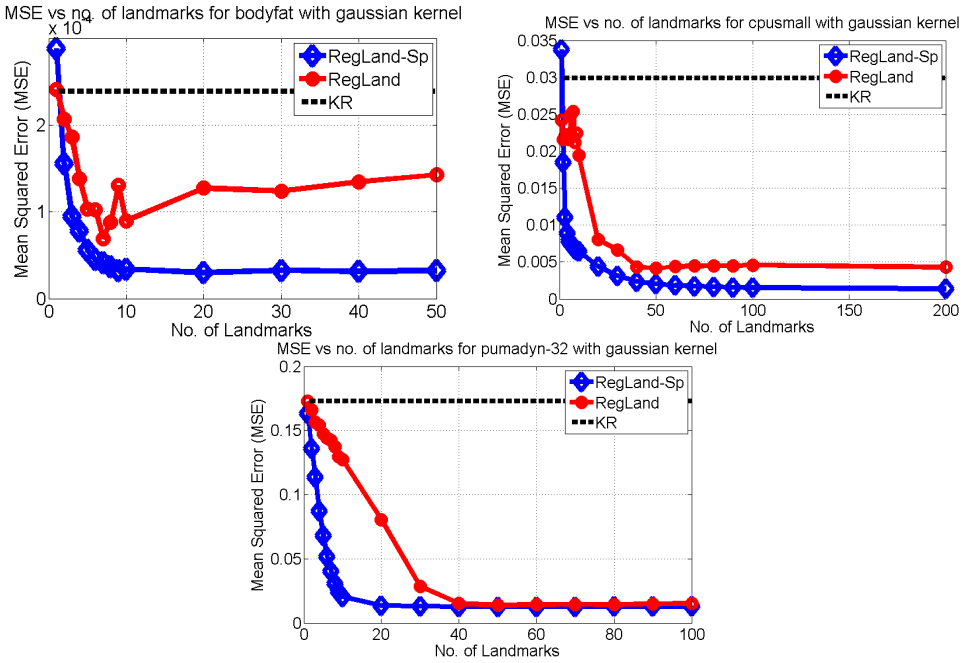
Below we present additional experimental results for regression and ordinal regression problems.

Regression Experiments

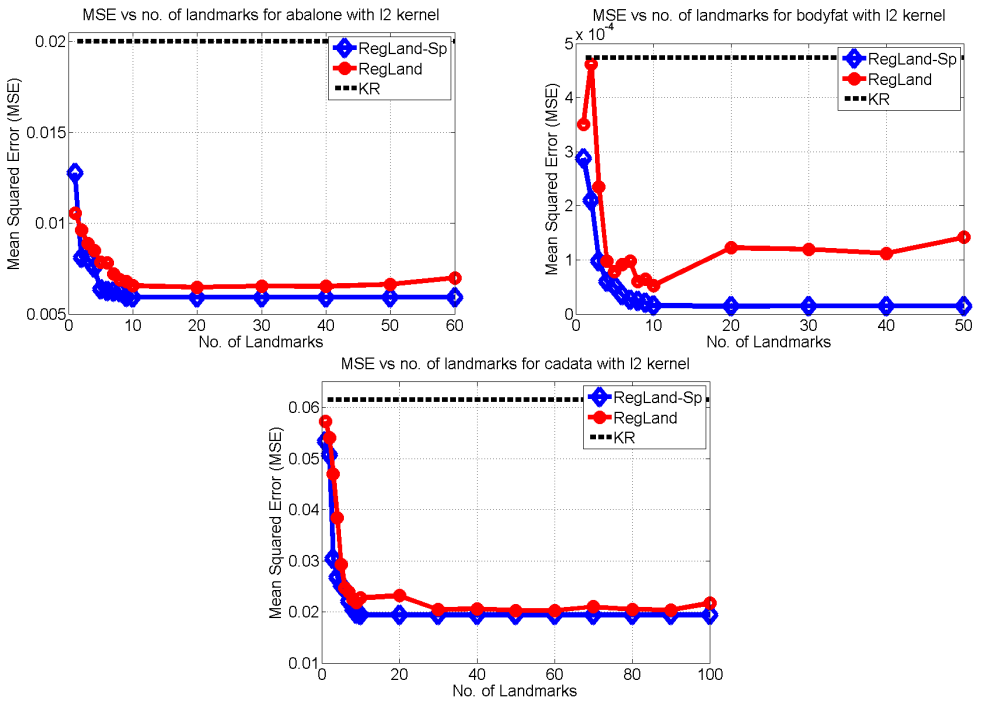
We present results on various benchmark datasets considered in Section 5.4 for Gaussian $K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{y}\|_2^2}{2\sigma^2}\right)$ and Euclidean: $K(\mathbf{x}, \mathbf{y}) = -\left\|\mathbf{x} - \mathbf{y}\right\|_2^2$ kernels. Following standard practice, we fixed σ to be the average pairwise distance between data points in the training set.

Ordinal Regression Experiments

We present results on various benchmark datasets considered in Section 5.4 for Gaussian $K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{y}\|_2^2}{2\sigma^2}\right)$ and Manhattan: $K(\mathbf{x}, \mathbf{y}) = -\|\mathbf{x} - \mathbf{y}\|_1$ kernels. Following standard practice, we fixed σ to be the average pairwise distance between data points in the training set.

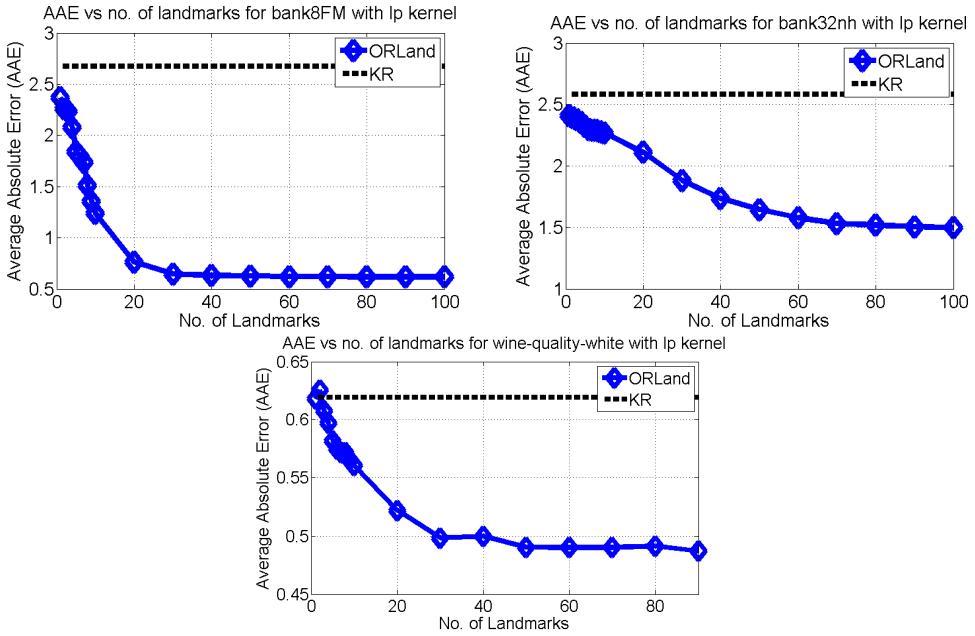


(a) Mean squared error for landmarking (RegLand), sparse landmarking (RegLand-Sp) and kernel regression (KR) for the Gaussian kernel.

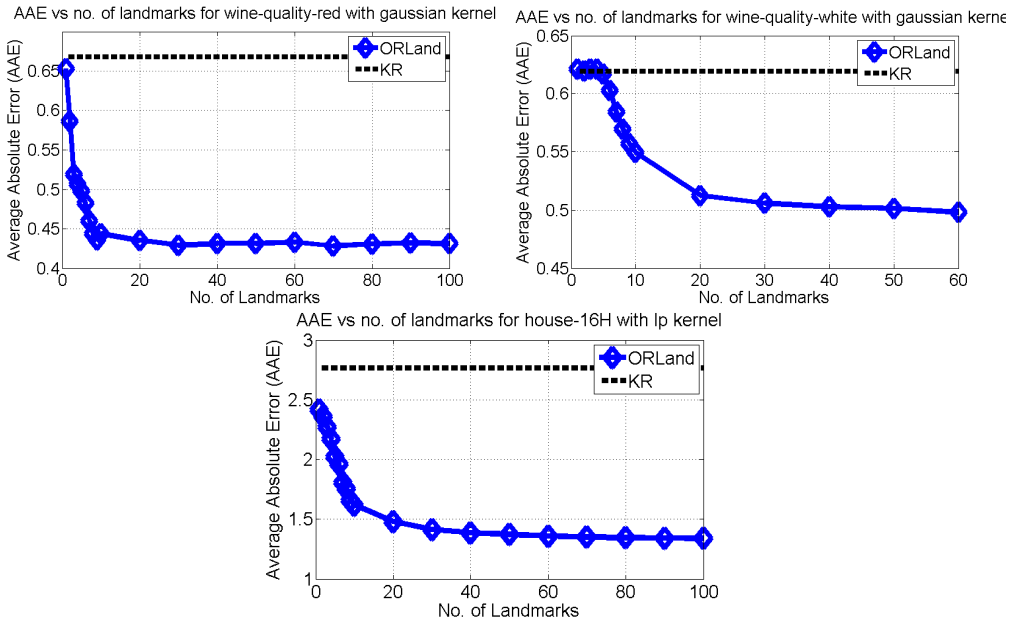


(b) Mean squared error for landmarking (RegLand), sparse landmarking (RegLand-Sp) and kernel regression (KR) for the Euclidean kernel.

Figure 5.3: Performance of landmarking algorithms with increasing number of landmarks on real-valued regression datasets.



(a) Average absolute error for landmarking (ORLand) and kernel regression (KR) on ordinal regression datasets for the Manhattan kernel.



(b) Average absolute error for landmarking (ORLand) and kernel regression (KR) on ordinal regression datasets for the Gaussian kernel.

Figure 5.4: Performance of landmarking algorithms with increasing number of landmarks on ordinal regression datasets.

Chapter 6

Online Learning with Pairwise Loss Functions

Contents

Abstract *In this chapter, we study the generalization properties of online learning based stochastic methods for supervised learning problems where the loss function is dependent on more than one training sample (e.g., metric learning, ranking). We present a generic decoupling technique that enables us to provide Rademacher complexity-based generalization error bounds. Our bounds are in general tighter than those obtained by Wang et al. (2012) for the same problem. Using our decoupling technique, we are further able to obtain fast convergence rates for strongly convex pairwise loss functions. We are also able to analyze a class of memory efficient online learning algorithms for pairwise learning problems that use only a bounded subset of past training samples to update the hypothesis at each step. Finally, in order to complement our generalization bounds, we propose a novel memory efficient online learning algorithm for higher order learning problems with bounded regret guarantees.*

6.1 Introduction

Several supervised learning problems involve working with pairwise or higher order loss functions, i.e., loss functions that depend on more than one training sample. Take for example the *metric learning* problem (Jin et al., 2009), where the goal is to learn a metric M that brings points of a similar label together while keeping differently labeled points apart. In this case the loss function used is a pairwise loss function $\ell(M, (\mathbf{x}, y), (\mathbf{x}', y')) = \phi\left(yy'\left(1 - M(\mathbf{x}, \mathbf{x}')\right)\right)$ where ϕ is the hinge loss function. In general, a pairwise loss function is of the form $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$ where $\mathcal{H}$ is the hypothesis space and $\mathcal{X}$ is the input domain. Other examples include preference learning (Xing et al., 2002), ranking (Agarwal and Niyogi, 2009), AUC maximization (Zhao et al., 2011) and multiple kernel learning (Kumar et al., 2012).

In practice, algorithms for such problems use intersecting pairs of training samples to learn. Hence the training data pairs are not i.i.d. and consequently, standard generalization error analysis techniques do not apply to these algorithms. Recently, the analysis of *batch* algorithms learning from such coupled samples has received much attention (Cao et al., 2012; Clémençon et al., 2008; Brefeld and Scheffer, 2005) where a dominant idea has been to use an alternate representation of the U-statistic and provide uniform convergence bounds. Another popular approach has been to use algorithmic stability (Agarwal and Niyogi, 2009; Jin et al., 2009) to obtain algorithm-specific results.

While batch algorithms for pairwise (and higher-order) learning problems have been studied well theoretically, online learning based stochastic algorithms are more popular in practice due to their scalability. However, their generalization properties were not studied until recently. Wang et al. (2012) provided the first generalization error analysis of online learning methods applied to pairwise loss functions. In particular, they showed that such higher-order online learning methods also admit online to batch conversion bounds (similar

to those for first-order problems obtained by Cesa-Bianchi et al. (2001)) which can be combined with regret bounds to obtain generalization error bounds. However, due to their proof technique and dependence on L_∞ covering numbers of function classes, their bounds are not tight and have a strong dependence on the dimensionality of the input space.

In literature, there are several instances where Rademacher complexity based techniques achieve sharper bounds than those based on covering numbers (Kakade et al., 2008). However, the coupling of different input pairs in our problem does not allow us to use such techniques directly.

In this chapter we introduce a generic technique for analyzing online learning algorithms for higher order learning problems. Our technique, that uses an extension of Rademacher complexities to higher order function classes (instead of covering numbers), allows us to give bounds that are tighter than those of Wang et al. and that, for several learning scenarios, have no dependence on input dimensionality at all.

The key to our proof is a technique we call *Symmetrization of Expectations* which acts as a decoupling step and allows us to reduce excess risk estimates to Rademacher complexities of function classes. Wang et al., on the other hand, perform a symmetrization with probabilities which, apart from being more involved, yields suboptimal bounds. Another advantage of our technique is that it allows us to obtain *fast* convergence rates for learning algorithms that use *strongly convex* loss functions. Our result, which uses a novel two stage proof technique, extends a similar result in the first order setting by Kakade and Tewari (2008) to the pairwise setting.

Wang et al. (and our results mentioned above) assume an online learning setup in which a stream of points $\mathbf{z}_1, \ldots, \mathbf{z}_n$ is observed and the penalty function used at the t^{th} step is $\hat{\mathcal{L}}_t(h) = \frac{1}{t-1}\sum_{\tau=1}^{t-1} \ell(h, \mathbf{z}_t, \mathbf{z}_\tau)$. Consequently, the results of Wang et al. expect regret bounds with respect to these *all-pairs* penalties $\hat{\mathcal{L}}_t$. This requires one to use/store all previously seen points which is computationally/storage wise expensive and hence in practice, learning algorithms update their hypotheses using only a bounded subset of the past samples (Zhao et al., 2011).

In the above mentioned setting, we are able to give generalization bounds that only require algorithms to give regret bounds with respect to *finite-buffer* penalty functions such as $\hat{\mathcal{L}}_t^{\text{buf}}(h) = \frac{1}{|B|}\sum_{\mathbf{z}\in B} \ell(h, \mathbf{z}_t, \mathbf{z})$ where B is a *buffer* that is updated at each step. Our proofs hold for any *stream oblivious* buffer update policy including FIFO and the widely used reservoir sampling policy (Vitter, 1985; Zhao et al., 2011)[1].

To complement our online to batch conversion bounds, we also provide a memory efficient online learning algorithm that works with bounded buffers. Although our algorithm is constrained to observe and learn using the *finite-buffer* penalties $\hat{\mathcal{L}}_t^{\text{buf}}$ alone, we are still able to provide high confidence regret bounds with respect to the *all-pairs* penalty

[1] ↑ Independently, Wang et al. (2013) also extended their proof to give similar guarantees. However, their bounds hold only for the FIFO update policy and have worse dependence on dimensionality in several cases (see Section 6.5).

functions $\hat{\mathcal{L}}_t$. We note that Zhao et al. (2011) also propose an algorithm that uses finite buffers and claim an *all-pairs* regret bound for the same. However, their regret bound does not hold due to a subtle mistake in their proof.

We also provide empirical validation of our proposed online learning algorithm on AUC maximization tasks and show that our algorithm performs competitively with that of Zhao et al., in addition to being able to offer theoretical regret bounds.

Our Contributions:

(a) We provide a generic online-to-batch conversion technique for higher-order supervised learning problems offering bounds that are sharper than those of Wang et al..

(b) We obtain fast convergence rates when loss functions are *strongly convex*.

(c) We analyze online learning algorithms that are constrained to learn using a finite buffer.

(d) We propose a novel online learning algorithm that works with finite buffers but is able to provide a high confidence regret bound with respect to the *all-pairs* penalty functions.

6.2 Problem Setup

For ease of exposition, we introduce an online learning model for higher order supervised learning problems in this section; concrete learning instances such as AUC maximization and metric learning are given in Section 6.6. For sake of simplicity, we restrict ourselves to pairwise problems in this chapter; our techniques can be readily extended to higher order problems as well.

For pairwise learning problems, our goal is to learn a real valued *bivariate* function $h^* : \mathcal{X} \times \mathcal{X} \to \mathcal{Y}$, where $h^* \in \mathcal{H}$, under some loss function $\ell : \mathcal{H} \times \mathbb{Z} \times \mathbb{Z} \to \mathbb{R}^+$ where $\mathbb{Z} = \mathcal{X} \times \mathcal{Y}$.

The online learning algorithm is given sequential access to a stream of elements $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n$ chosen i.i.d. from the domain $\mathbb{Z}$. Let $Z^t := \{\mathbf{z}_1, \ldots, \mathbf{z}_t\}$. At each time step $t = 2 \ldots n$, the algorithm posits a hypothesis $h_{t-1} \in \mathcal{H}$ upon which the element $\mathbf{z}_t$ is revealed and the algorithm incurs the following penalty:

$$\hat{\mathcal{L}}_t(h_{t-1}) = \frac{1}{t-1} \sum_{\tau=1}^{t-1} \ell(h_{t-1}, \mathbf{z}_t, \mathbf{z}_\tau). \tag{6.1}$$

For any $h \in \mathcal{H}$, we define its expected risk as:

$$\mathcal{L}(h) := \mathop{\mathbb{E}}_{\mathbf{z},\mathbf{z}'} \left[\!\!\left[\ell(h, \mathbf{z}, \mathbf{z}') \right]\!\!\right]. \tag{6.2}$$

Our aim is to present an ensemble $h_1, \ldots, h_{n-1}$ such that the expected risk of the ensemble is small. More specifically, we desire that, for some small $\varepsilon > 0$,

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \varepsilon,$$

where $h^* = \arg\min_{h\in\mathcal{H}} \mathcal{L}(h)$ is the population risk minimizer. Note that this allows us to do hypothesis selection in a way that ensures small expected risk. Specifically, if one chooses a hypothesis as $\hat{h} := \frac{1}{(n-1)}\sum_{t=2}^{n} h_{t-1}$ (for convex ℓ) or $\hat{h} := \arg\min_{t=2,\ldots,n} \mathcal{L}(h_t)$ then we have $\mathcal{L}(\hat{h}) \leq \mathcal{L}(h^*) + \varepsilon$.

Since the model presented above requires storing all previously seen points, it becomes unusable in large scale learning scenarios. Instead, in practice, a *sketch* of the stream is maintained in a buffer B of capacity s. At each step, the penalty is now incurred only on the pairs $\{(\mathbf{z}_t, \mathbf{z}) : \mathbf{z} \in B_t\}$ where B_t is the state of the buffer at time t. That is,

$$\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) = \frac{1}{|B_t|}\sum_{\mathbf{z}\in B_t}\ell(h_{t-1}, \mathbf{z}_t, \mathbf{z}). \tag{6.3}$$

We shall assume that the buffer is updated at each step using some *stream oblivious* policy such as FIFO or Reservoir sampling (Vitter, 1985) (see Section 6.5).

In Section 6.3, we present online-to-batch conversion bounds for online learning algorithms that give regret bounds w.r.t. penalty functions given by (6.1). In Section 6.4, we extend our analysis to algorithms using strongly convex loss functions. In Section 6.5 we provide generalization error bounds for algorithms that give regret bounds w.r.t. *finite-buffer* penalty functions given by (6.3). Finally in section 6.7 we present a novel memory efficient online learning algorithm with regret bounds.

6.3 Online to Batch Conversion Bounds for Bounded Loss Functions

We now present our generalization bounds for algorithms that provide regret bounds with respect to the *all-pairs* loss functions (see Eq. (6.1)). Our results give tighter bounds and have a much better dependence on input dimensionality than the bounds given by Wang et al.. See Section 6.3.1 for a detailed comparison.

As was noted by Wang et al., the generalization error analysis of online learning algorithms in this setting does not follow from existing techniques for first-order problems (such as (Cesa-Bianchi et al., 2001; Kakade and Tewari, 2008)). The reason is that the terms $V_t = \hat{\mathcal{L}}_t(h_{t-1})$ do not form a martingale due to the intersection of training samples in V_t and $V_\tau, \tau < t$.

Our technique, that aims to utilize the Rademacher complexities of function classes in order to get tighter bounds, faces yet another challenge at the *symmetrization* step,

a precursor to the introduction of Rademacher complexities. As the reader will recall[2], there are two main aspects involved in Rademacher-average based proofs. For a learning problem, say binary classification, given training points $\mathbf{x}_1, \ldots, \mathbf{x}_n$, we first rewrite the population risk as the expected empirical risk on a fresh randomly sampled set of points $\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_n$ - since these points are never actually seen and are simply introduced for the sake of analysis, they are often referred to as "ghost variables".

However, since these ghost points are chosen from the same distribution as the training points, they are indistinguishable from them. More specifically, the following two estimators (A) and (B) are indistinguishable in the sense that $\mathbb{E}[\![(A)]\!] = \mathbb{E}[\![(B)]\!]$

$$\begin{aligned}(A) &: \sup_{h \in \mathcal{H}} \{\ell(h, \mathbf{x}_1) - \ell(h, \tilde{\mathbf{x}}_1)\} \\ (B) &: \sup_{h \in \mathcal{H}} \{\ell(h, \tilde{\mathbf{x}}_1) - \ell(h, \mathbf{x}_1)\}\end{aligned}$$

This, combined with the fact that $(A) = -(B)$ gives us $\mathbb{E}[\![(A)]\!] = \mathbb{E}[\![-(A)]\!] = \mathbb{E}[\![\varepsilon \cdot (A)]\!]$ where ε is a Rademacher variable which establishes the so-called symmetrization step. However, in the case of pairwise loss functions it turns out that due to the coupling between the "head" variable $\mathbf{z}_t$ and the "tail" variables $\mathbf{z}_\tau$ in the loss function $\hat{\mathcal{L}}_t$, a standard symmetrization between true $\mathbf{z}_\tau$ and ghost $\tilde{\mathbf{z}}_\tau$ samples does not succeed in generating Rademacher averages and instead yields complex looking terms.

More specifically, suppose we have *true* variables $\mathbf{z}_t$ and *ghost* variables $\tilde{\mathbf{z}}_t$ and are in the process of bounding the expected excess risk by analyzing expressions of the form

$$E_{\text{orig}} = \ell(h_{t-1}, \mathbf{z}_t, \mathbf{z}_\tau) - \ell(h_{t-1}, \tilde{\mathbf{z}}_t, \tilde{\mathbf{z}}_\tau).$$

Performing a traditional symmetrization of the variables $\mathbf{z}_\tau$ with $\tilde{\mathbf{z}}_\tau$ would give us expressions of the form

$$E_{\text{symm}} = \ell(h_{t-1}, \mathbf{z}_t, \tilde{\mathbf{z}}_\tau) - \ell(h_{t-1}, \tilde{\mathbf{z}}_t, \mathbf{z}_\tau).$$

At this point the analysis hits a barrier since unlike first order situations, we cannot relate E_{symm} to E_{orig} by means of introducing Rademacher variables.

We circumvent this problem by using a technique that we call *Symmetrization of Expectations*. The technique allows us to use standard symmetrization to obtain Rademacher complexities. More specifically, we analyze expressions of the form

$$E'_{\text{orig}} = \mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h_{t-1}, \mathbf{z}, \mathbf{z}_\tau)]\!] - \mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h_{t-1}, \mathbf{z}, \tilde{\mathbf{z}}_\tau)]\!]$$

which upon symmetrization yield expressions such as

$$E'_{\text{symm}} = \mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h_{t-1}, \mathbf{z}, \tilde{\mathbf{z}}_\tau)]\!] - \mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h_{t-1}, \mathbf{z}, \mathbf{z}_\tau)]\!]$$

[2] ↑ We refer the reader to Section 2.3.2 for an introduction to the basic technique of proving Rademacher-average based generalization bounds.

which allow us to introduce Rademacher variables since $E'_{\text{symm}} = -E'_{\text{orig}}$. This idea is exploited by the lemma given below that relates the expected risk of the ensemble to the penalties incurred during the online learning process. In the following we use the following extension of Rademacher averages (Kakade et al., 2008) to bivariate function classes:

$$\mathcal{R}_n(\mathcal{H}) = \mathbb{E}\left[\!\!\left[\sup_{h\in\mathcal{H}} \frac{1}{n}\sum_{\tau=1}^{n} \varepsilon_\tau h(\mathbf{z}, \mathbf{z}_\tau)\right]\!\!\right]$$

where the expectation is over ε_τ, $\mathbf{z}$ and $\mathbf{z}_\tau$. We shall denote composite function classes as follows : $\ell \circ \mathcal{H} := \{(\mathbf{z}, \mathbf{z}') \mapsto \ell(h, \mathbf{z}, \mathbf{z}'), h \in \mathcal{H}\}$.

Lemma 6.1. *Let $h_1, \ldots, h_{n-1}$ be an ensemble of hypotheses generated by an online learning algorithm working with a bounded loss function $\ell : \mathcal{H} \times \mathbb{Z} \times \mathbb{Z} \to [0, B]$. Then for any $\delta > 0$, we have with probability at least $1 - \delta$,*

$$\frac{1}{n-1}\sum_{t=2}^{n} \mathcal{L}(h_{t-1}) \leq \frac{1}{n-1}\sum_{t=2}^{n} \hat{\mathcal{L}}_t(h_{t-1}) + \frac{2}{n-1}\sum_{t=2}^{n} \mathcal{R}_{t-1}(\ell \circ \mathcal{H}) + 3B\sqrt{\frac{\log\frac{n}{\delta}}{n-1}}.$$

The proof of the lemma involves decomposing the excess risk term into a martingale difference sequence and a residual term in a manner similar to Wang et al.. The martingale sequence, being a bounded one, is shown to converge using the Azuma-Hoeffding inequality. The residual term is handled using uniform convergence techniques involving Rademacher averages. The complete proof of the lemma is given in the Section 6.12.1.

Similar to Lemma 6.1, the following converse relation between the population and empirical risk of the population risk minimizer h^* can also be shown.

Lemma 6.2. *For any $\delta > 0$, we have with probability at least $1 - \delta$,*

$$\frac{1}{n-1}\sum_{t=2}^{n} \hat{\mathcal{L}}_t(h^*) \leq \mathcal{L}(h^*) + \frac{2}{n-1}\sum_{t=2}^{n} \mathcal{R}_{t-1}(\ell \circ \mathcal{H}) + 3B\sqrt{\frac{\log\frac{1}{\delta}}{n-1}}.$$

An online learning algorithm will be said to have an *all-pairs* regret bound $\mathfrak{R}_n$ if it presents an ensemble $h_1, \ldots, h_{n-1}$ such that

$$\sum_{t=2}^{n} \hat{\mathcal{L}}_t(h_{t-1}) \leq \inf_{h\in\mathcal{H}} \sum_{t=2}^{n} \hat{\mathcal{L}}_t(h) + \mathfrak{R}_n.$$

Suppose we have an online learning algorithm with a regret bound $\mathfrak{R}_n$. Then combining Lemmata 6.1 and 6.2 gives us the following online to batch conversion bound:

Theorem 6.3. *Let $h_1, \ldots, h_{n-1}$ be an ensemble of hypotheses generated by an online learning algorithm working with a B-bounded loss function ℓ that guarantees a regret bound of $\mathfrak{R}_n$. Then for any $\delta > 0$, we have with probability at least $1 - \delta$,*

$$\frac{1}{n-1}\sum_{t=2}^{n} \mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{4}{n-1}\sum_{t=2}^{n} \mathcal{R}_{t-1}(\ell \circ \mathcal{H}) + \frac{1}{n-1}\mathfrak{R}_n + 6B\sqrt{\frac{\log\frac{n}{\delta}}{n-1}}.$$

As we shall see in Section 6.6, for several learning problems, the Rademacher complexities behave as $\mathcal{R}_{t-1}(\ell \circ \mathcal{H}) \leq C_d \cdot \mathcal{O}\left(\frac{1}{\sqrt{t-1}}\right)$ where C_d is a constant dependent only on the dimension d of the input space and the $\mathcal{O}(\cdot)$ notation hides constants dependent on the domain size and the loss function. This allows us to bound the excess risk as follows:

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{1}{n-1}\mathfrak{R}_n + \mathcal{O}\left(\frac{C_d + \sqrt{\log(n/\delta)}}{\sqrt{n-1}}\right).$$

Here, the error decreases with n at a standard $1/\sqrt{n}$ rate (up to a $\sqrt{\log n}$ factor), similar to that obtained by Wang et al.. However, for several problems the above bound can be significantly tighter than those offered by covering number based arguments. We provide below a detailed comparison of our results with those of Wang et al..

6.3.1 Discussion on the nature of our bounds

As mentioned above, our proof enables us to use Rademacher complexities which are typically easier to analyze and provide tighter bounds Kakade et al. (2008). In particular, as shown in Section 6.6, for L_2 regularized learning formulations, the Rademacher complexities are dimension independent i.e. $C_d = 1$. Consequently, unlike the bounds of Wang et al. that have a linear dependence on d, our bound becomes independent of the input space dimension. For sparse learning formulations with L_1 or trace norm regularization, we have $C_d = \sqrt{\log d}$ giving us a mild dependence on the input dimensionality.

Our bounds are also tighter that those of Wang et al. in general. Whereas we provide a confidence bound of $\delta < \exp\left(-n\varepsilon^2 + \log n\right)$, Wang et al. offer a weaker bound $\delta < (1/\varepsilon)^d \exp\left(-n\varepsilon^2 + \log n\right)$.

An artifact of the proof technique of Wang et al. (2012) is that their proof is required to exclude a constant fraction of the ensemble $(h_1, \ldots, h_{cn})$ from the analysis, failing which their bounds turn vacuous. Our proof on the other hand is able to give guarantees for the *entire* ensemble.

In addition to this, as the following sections show, our proof technique enjoys the flexibility of being extendable to give fast convergence guarantees for strongly convex loss functions as well as being able to accommodate learning algorithms that use finite buffers.

6.4 Fast Convergence Rates for Strongly Convex Loss Functions

In this section we extend results of the previous section to give *fast* convergence guarantees for online learning algorithms that use strongly convex loss functions of the following form:

$$\ell(h, \mathbf{z}, \mathbf{z}') = g(\langle h, \phi(\mathbf{z}, \mathbf{z}')\rangle) + r(h),$$

where g is a convex function and $r(h)$ is a σ-strongly convex regularizer (see Section 6.6 for examples) i.e. $\forall h_1, h_2 \in \mathcal{H}$ and $\alpha \in [0,1]$, we have

$$r(\alpha h_1 + (1-\alpha)h_2) \leq \alpha r(h_1) + (1-\alpha) r(h_2) - \frac{\sigma}{2}\alpha(1-\alpha)\left\|h_1 - h_2\right\|^2 .$$

For any norm $\|\cdot\|$, let $\|\cdot\|_*$ denote its dual norm.

Our analysis reduces the pairwise problem to a first order problem and a martingale convergence problem. We require the following *fast* convergence bound in the standard first order *batch* learning setting:

Theorem 6.4. *Let $\mathcal{F}$ be a closed and convex set of functions over $\mathcal{X}$. Let $\wp(f, \mathbf{x}) = p(\langle f, \phi(\mathbf{x})\rangle) + r(f)$, for a σ-strongly convex function r, be a loss function with $\mathcal{P}$ and $\hat{\mathcal{P}}$ as the associated population and empirical risk functionals and f^* as the population risk minimizer. Suppose $\wp$ is L-Lipschitz and $\|\phi(\mathbf{x})\|_* \leq R, \forall \mathbf{x} \in \mathcal{X}$. Then w.p. $1-\delta$, for any $\varepsilon > 0$, we have for all $f \in \mathcal{F}$,*

$$\mathcal{P}(f) - \mathcal{P}(f^*) \leq (1+\varepsilon)\left(\hat{\mathcal{P}}(f) - \hat{\mathcal{P}}(f^*)\right) + \frac{C_\delta}{\varepsilon\sigma n}$$

where $C_\delta = C_d^2 \cdot (4(1+\varepsilon)LR)^2\,(32 + \log(1/\delta))$ and C_d is the dependence of the Rademacher complexity of the class $\mathcal{F}$ on the input dimensionality d.

The above theorem is a minor modification of a similar result by Sridharan et al. (2008) and the proof (given in Section 6.12.2) closely follows their proof as well. We can now state our online to batch conversion result for strongly convex loss functions.

Theorem 6.5. *Let $h_1, \ldots, h_{n-1}$ be an ensemble of hypotheses generated by an online learning algorithm working with a B-bounded, L-Lipschitz and σ-strongly convex loss function ℓ. Further suppose the learning algorithm guarantees a regret bound of $\mathfrak{R}_n$. Let $\mathfrak{V}_n = \max\left\{\mathfrak{R}_n, 2C_d^2 \log n \log(n/\delta)\right\}$ Then for any $\delta > 0$, we have with probability at least $1-\delta$,*

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{1}{n-1}\mathfrak{R}_n + C_d \cdot \mathcal{O}\left(\frac{\sqrt{\mathfrak{V}_n \log n \log(n/\delta)}}{n-1}\right),$$

where the $\mathcal{O}(\cdot)$ notation hides constants dependent on domain size and the loss function such as L, B and σ.

The decomposition of the excess risk in this case is not made explicitly but rather emerges as a side-effect of the proof progression. The proof starts off by applying Theorem 6.4 to the hypothesis in each round with the following loss function $\wp(h, \mathbf{z}') := \mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h, \mathbf{z}, \mathbf{z}')]\!]$. Applying the regret bound to the resulting expression gives us a martingale difference sequence which we then bound using Bernstein-style inequalities and a proof technique from (Kakade and Tewari, 2008). The complete proof is given in Section 6.12.3.

We now note some properties of this result. The effective dependence of the above bound on the input dimensionality is C_d^2 since the expression $\sqrt{\mathfrak{V}_n}$ hides a C_d term. We

have $C_d^2 = 1$ for non sparse learning formulations and $C_d^2 = \log d$ for sparse learning formulations. We note that our bound matches that of Kakade and Tewari (for *first*-order learning problems) up to a logarithmic factor.

6.5 Analyzing Online Learning Algorithms that use Finite Buffers

In this section, we present our online to batch conversion bounds for algorithms that work with *finite-buffer* loss functions $\hat{\mathcal{L}}_t^{\text{buf}}$. In our finite buffer online learning model, one observes a stream of elements $\mathbf{z}_1, \ldots, \mathbf{z}_n$. A *sketch* of these elements is maintained in a buffer B of size s, i.e., at each step $t = 2, \ldots, n$, the buffer contains a subset of the elements Z^{t-1} of size at most s. At each step $t = 2 \ldots n$, the online learning algorithm posits a hypothesis $h_{t-1} \in \mathcal{H}$, upon which the element $\mathbf{z}_t$ is revealed and the algorithm incurs the loss

$$\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) = \frac{1}{|B_t|} \sum_{\mathbf{z} \in B_t} \ell(h_{t-1}, \mathbf{z}_t, \mathbf{z}),$$

where B_t is the state of the buffer at time t. Note that $|B_t| \leq s$. We would be interested in algorithms that are able to give a *finite-buffer* regret bound, i.e., for which, the proposed ensemble $h_1, \ldots, h_{n-1}$ satisfies

$$\sum_{t=2}^{n} \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) - \inf_{h \in \mathcal{H}} \sum_{t=2}^{n} \hat{\mathcal{L}}_t^{\text{buf}}(h) \leq \mathfrak{R}_n^{\text{buf}}.$$

We assume that the buffer is updated after each step in a *stream-oblivious* manner (see discussion below). For randomized buffer update policies (such as reservoir sampling Vitter (1985)), this corresponds to the assumption that we are supplied at each step with some fresh randomness r_t (see examples below) along with the data point $\mathbf{z}_t$. Thus the data received at time t is a tuple $\mathbf{w}_t = (\mathbf{z}_t, r_t)$. We shall refer to the random variables r_t as *auxiliary* variables. It is important to note that stream obliviousness dictates that r_t as a random variable is independent of z_t. Let $W^{t-1} := \{\mathbf{w}_1, \ldots, \mathbf{w}_{t-1}\}$ and $R^{t-1} := \{r_1, \ldots, r_{t-1}\}$. Note that R^{t-1} completely decides the indices present in the buffer B_t at step t independent of Z^{t-1}. For any $h \in \mathcal{H}$, define

$$\tilde{\mathcal{L}}_t^{\text{buf}} := \mathop{\mathbb{E}}_{\mathbf{z}_t} \left[\left[\hat{\mathcal{L}}_t^{\text{buf}} \middle| W^{t-1} \right] \right].$$

For our guarantees to hold, we require the buffer update policy used by the learning algorithm to be *stream oblivious*. More specifically, we require the buffer update rule to decide upon the inclusion of a particular point $\mathbf{z}_i$ in the buffer based only on its stream index $i \in [n]$. Some examples of stream oblivious policies are given below. Stream oblivious policies allow us to decouple buffer construction randomness from training sample randomness which makes analysis easier; we leave the analysis of *stream aware* buffer update policies as a topic of future research.

6.5.1 Examples of Stream Oblivious Policies

Below we give some examples of stream oblivious policies for updating the buffer:

1. **FIFO**: in this policy, the data point $\mathbf{z}_t$ arriving at time $t > s$ is inducted into the buffer by evicting the data point $\mathbf{z}_{(t-s)}$ from the buffer. Since this is a non-randomized policy, there is no need for auxiliary randomness and we can assume that r_t follows the trivial law
$$r_t \sim \mathbb{1}_{\{r=1\}}.$$

2. **RS** : the Reservoir Sampling policy was introduced by Vitter (1985). In this policy, at time $t > s$, the incoming data point $\mathbf{z}_t$ is inducted into the buffer with probability s/t. If chosen to be included, it results in the eviction of a random element of the buffer. In this case the auxiliary random variable is a 2-tuple that follows the law
$$r_t = (r_t^1, r_t^2) \sim \left(\text{Bernoulli}\left(\frac{s}{t}\right), \frac{1}{s}\sum_{i=1}^{s} \mathbb{1}_{\{r_2=i\}} \right).$$

3. **RS-x** (see Algorithm 7): in this policy, the incoming data point $\mathbf{z}_t$ at time $t > s$, replaces each data point in the buffer independently with probability $1/t$. Thus the incoming point has the potential to evict multiple buffer points while establishing multiple copies of itself in the buffer. In this case, the auxiliary random variable is defined by a Bernoulli process:
$$r_t = (r_t^1, r_t^2 \ldots, r_t^s) \sim \left(\text{Bernoulli}\left(\frac{1}{t}\right), \text{Bernoulli}\left(\frac{1}{t}\right), \ldots, \text{Bernoulli}\left(\frac{1}{t}\right) \right).$$

4. **RS-x^2** (see Algorithm 9): this is a variant of **RS-x** in which the number of evictions is first decided by a Binomial trial and then those many random points in the buffer are replaced by the incoming data point. This can be implemented as follows:
$$r_t = (r_t^1, r_t^2) \sim \left(\text{Binomial}\left(s, \frac{1}{t}\right), \text{Perm}(s) \right)$$
where $\text{Perm}(s)$ gives a random permutation of s elements.

In the above mentioned setting, we can prove the following online to batch conversion bound for bounded loss functions:

Theorem 6.6. *Let $h_1, \ldots, h_{n-1}$ be an ensemble of hypotheses generated by an online learning algorithm working with a finite buffer of capacity s and a B-bounded loss function ℓ. Moreover, suppose that the algorithm guarantees a regret bound of $\mathfrak{R}_n^{buf}$. Then for any $\delta > 0$, we have with probability at least $1 - \delta$,*
$$\frac{1}{n-1}\sum_{t=2}^{n} \mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{1}{n-1}\mathfrak{R}_n^{buf} + \mathcal{O}\left(\frac{C_d}{\sqrt{s}} + B\sqrt{\frac{\log \frac{n}{\delta}}{s}} \right)$$

where C_d is the dependence of $\mathcal{R}_n(\mathcal{H})$ on the input dimensionality d.

Proof Sketch. We shall prove the result in two steps. In the first step we shall prove the following uniform convergence style result

Lemma 6.7. *Let $h_1, \ldots, h_{n-1}$ be an ensemble of hypotheses generated by an online learning algorithm working with a B-bounded loss function ℓ and a finite buffer of capacity s. Then for any $\delta > 0$, we have with probability at least $1 - \delta$,*

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \frac{1}{n-1}\sum_{t=2}^{n}\hat{\mathcal{L}}_t^{buf}(h_{t-1}) + B\sqrt{\frac{2\log\frac{n}{\delta}}{s}} + \frac{2}{n-1}\sum_{t=2}^{n}\mathcal{R}_{\min\{t-1,s\}}(\ell\circ\mathcal{H}).$$

At a high level, our proof progression shall follow that of Lemma 6.1. However, the execution of the proof will have to be different in order to accommodate the finiteness of the buffer and randomness used to construct it. Similarly, we shall also be able to show the following result.

Lemma 6.8. *For any $\delta > 0$, we have with probability at least $1 - \delta$,*

$$\frac{1}{n-1}\sum_{t=2}^{n}\hat{\mathcal{L}}_t^{buf}(h^*) \leq \mathcal{L}(h^*) + 3B\sqrt{\frac{\log\frac{n}{\delta}}{s}} + \frac{2}{n-1}\sum_{t=2}^{n}\mathcal{R}_{\min\{t-1,s\}}(\ell\circ\mathcal{H}).$$

Note that for classes whose Rademacher averages behave as $\mathcal{R}_n(\mathcal{H}) \leq C_d \cdot \mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$, applying Lemma 6.10 gives us $\mathcal{R}_n(\ell\circ\mathcal{H}) \leq C_d \cdot \mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$ as well which allows us to show

$$\frac{2}{n-1}\sum_{t=2}^{n}\mathcal{R}_{\min\{t-1,s\}}(\ell\circ\mathcal{H}) = C_d \cdot \mathcal{O}\left(\frac{1}{\sqrt{s}}\right).$$

Combining Lemmata 6.7 and 6.8 along with the definition of bounded buffer regret $\mathfrak{R}_n^{\text{buf}}$ gives us the first part of Theorem 6.6. □

We can strengthen the above result for strongly convex loss functions as well:

Theorem 6.9. *Let $h_1, \ldots, h_{n-1}$ be an ensemble of hypotheses generated by an online learning algorithm working with a finite buffer of capacity s and a B-bounded loss function that is Lipschitz and strongly convex as well, then with confidence at least $1 - \delta$, we have*

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{1}{n-1}\mathfrak{R}_n^{buf} + C_d \cdot \mathcal{O}\left(\sqrt{\frac{\mathfrak{W}_n\log\frac{n}{\delta}}{sn}}\right)$$

where $\mathfrak{W}_n = \max\left\{\mathfrak{R}_n^{buf}, \frac{2C_d^2 n\log(n/\delta)}{s}\right\}$ and C_d is the dependence of $\mathcal{R}_n(\mathcal{H})$ on the input dimensionality d.

The above bound guarantees an excess error of $\tilde{\mathcal{O}}\left(1/s\right)$ for algorithms (such as Follow-the-leader (Hazan et al., 2006)) that offer logarithmic regret $\mathfrak{R}_n^{\text{buf}} = \mathcal{O}\left(\log n\right)$. We stress

that this theorem is not a direct corollary of our results for the *infinite buffer* case (Theorems 6.3 and 6.5). Instead, our proof requires a more careful analysis of the excess risk in order to accommodate the finiteness of the buffer and the randomness (possibly) used in constructing it.

More specifically, care needs to be taken to handle randomized buffer update policies such as **RS** which introduce additional randomness into the analysis. A naive application of techniques used to prove results for the unbounded buffer case would result in bounds that give non trivial generalization guarantees only for large buffer sizes such as $s = \omega(\sqrt{n})$. Our bounds, on the other hand, only require $s = \tilde{\omega}(1)$.

Key to our proofs is a conditioning step where we first analyze the conditional excess risk by conditioning upon randomness used by the buffer update policy. Such conditioning is made possible by the stream-oblivious nature of the update policy and thus, stream-obliviousness is required by our analysis. Subsequently, we analyze the excess risk by taking expectations over randomness used by the buffer update policy.

Note that the above results only require an online learning algorithm to provide regret bounds w.r.t. the *finite-buffer* penalties $\hat{\mathcal{L}}_t^{\text{buf}}$ and do not require any regret bounds w.r.t the *all-pairs* penalties $\hat{\mathcal{L}}_t$.

For instance, the finite buffer based online learning algorithms OAM_{seq} and OAM_{gra} proposed in Zhao et al. are able to provide a regret bound w.r.t. $\hat{\mathcal{L}}_t^{\text{buf}}$ (Zhao et al., 2011, Lemma 2) but are not able to do so w.r.t the *all-pairs* loss function (see Section 6.7 for a discussion). Using Theorem 6.6, we are able to give a generalization bound for OAM_{seq} and OAM_{gra} and hence explain the good empirical performance of these algorithms as reported in Zhao et al.. Note that Wang et al. are not able to analyze OAM_{seq} and OAM_{gra} since their analysis is restricted to algorithms that use the (deterministic) FIFO update policy whereas OAM_{seq} and OAM_{gra} use the (randomized) **RS** policy of Vitter.

6.6 Applications

In this section we make explicit our online to batch conversion bounds for several learning scenarios and also demonstrate their dependence on input dimensionality by calculating their respective Rademacher complexities. Recall that our definition of Rademacher complexity for a pairwise function class is given by,

$$\mathcal{R}_n(\mathcal{H}) = \mathbb{E}\left[\!\!\left[\sup_{h \in \mathcal{H}} \frac{1}{n} \sum_{\tau=1}^{n} \varepsilon_\tau h(\mathbf{z}, \mathbf{z}_\tau) \right]\!\!\right].$$

For our purposes, we would be interested in the Rademacher complexities of *composition classes* of the form $\ell \circ \mathcal{H} := \{(\mathbf{z}, \mathbf{z}') \mapsto \ell(h, \mathbf{z}, \mathbf{z}'), h \in \mathcal{H}\}$ where ℓ is some Lipschitz loss function. Frequently we have $\ell(h, \mathbf{z}, \mathbf{z}') = \phi\left(h(\mathbf{x}, \mathbf{x}')Y(y, y')\right)$ where $Y(y, y') = y - y'$ or $Y(y, y') = yy'$ and $\phi : \mathbb{R} \to \mathbb{R}$ is some margin loss function (Steinwart and Christmann, 2008b). Suppose ϕ is L-Lipschitz and $Y = \sup_{y, y' \in \mathcal{Y}} |Y(y, y')|$. Then we have

Theorem 6.10. $\mathcal{R}_n(\ell \circ \mathcal{H}) \leq LY\mathcal{R}_n(\mathcal{H})$.

The proof uses standard contraction inequalities and is given in Section 6.12.6. This reduces our task to computing the values of $\mathcal{R}_n(\mathcal{H})$ which we do using a two stage proof technique (see discussion below). For any subset X of a Banach space and any norm $\|\cdot\|_p$, we define $\|X\|_p := \sup_{\mathbf{x} \in X} \|\mathbf{x}\|_p$. Let the domain $\mathcal{X} \subset \mathbb{R}^d$. We also define norm bounded balls in the Banach space as $\mathcal{B}_p(r) := \left\{\mathbf{x} : \|\mathbf{x}\|_p \leq r\right\}$ for any $r > 0$.

6.6.1 Rademacher Complexity Derivations

In this section we shall derive Rademacher complexity bounds for hypothesis classes used in various learning problems. Crucial to our derivations shall be the following result by Kakade et al. (2008). Recall the usual definition of Rademacher complexity of a *univariate* function class $\mathcal{F} = \{f : \mathcal{X} \to \mathbb{R}\}$

$$\mathcal{R}_n(\mathcal{F}) = \mathbb{E}\left[\!\!\left[\sup_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \varepsilon_i f(\mathbf{x}_i)\right]\!\!\right].$$

Theorem 6.11 (Kakade et al. (2008), Theorem 1)**.** *Let $\mathcal{W}$ be a closed and convex subset of some Banach space equipped with a norm $\|\cdot\|$ and dual norm $\|\cdot\|_*$. Let $F : \mathcal{W} \to \mathbb{R}$ be σ-strongly convex with respect to $\|\cdot\|_*$. Assume $\mathcal{W} \subseteq \left\{\mathbf{w} : F(\mathbf{w}) \leq W_*^2\right\}$. Furthermore, let $\mathcal{X} = \{\mathbf{x} : \|\mathbf{x}\| \leq X\}$ and $\mathcal{F}_{\mathcal{W}} := \{\mathbf{w} \mapsto \langle \mathbf{w}, \mathbf{x}\rangle : \mathbf{w} \in \mathcal{W}, \mathbf{x} \in \mathcal{X}\}$. Then, we have*

$$\mathcal{R}_n(\mathcal{F}_{\mathcal{W}}) \leq XW_* \sqrt{\frac{2}{\sigma n}}.$$

We note that Theorem 6.11 is applicable only to first order learning problems since it gives bounds for univariate function classes. However, our hypothesis classes consist of bivariate functions which makes a direct application difficult. Recall our extension of Rademacher averages to bivariate function classes:

$$\mathcal{R}_n(\mathcal{H}) = \mathbb{E}\left[\!\!\left[\sup_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \varepsilon_i h(\mathbf{z}, \mathbf{z}_i)\right]\!\!\right]$$

where the expectation is over ε_i, $\mathbf{z}$ and $\mathbf{z}_i$. To overcome the above problem we will use the following two step proof technique:

1. **Order reduction**: We shall cast our learning problems in a modified input domain where predictors behave linearly as univariate functions. More specifically, given a hypothesis class $\mathcal{H}$ and domain $\mathcal{X}$, we shall construct a modified domain $\tilde{\mathcal{X}}$ and a map $\psi : \mathcal{X} \times \mathcal{X} \to \tilde{\mathcal{X}}$ such that for any $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ and $h \in \mathcal{H}$, we have $h(\mathbf{x}, \mathbf{x}') = \langle h, \psi(\mathbf{x}, \mathbf{x}')\rangle$.

Hypothesis class	Rademacher Complexity
$\mathcal{B}_q(\|\mathcal{W}\|_q)$	$2\|\mathcal{X}\|_p \|\mathcal{W}\|_q \sqrt{\frac{p-1}{n}}$
$\mathcal{B}_1(\|\mathcal{W}\|_1)$	$2\|\mathcal{X}\|_\infty \|\mathcal{W}\|_1 \sqrt{\frac{e \log d}{n}}$

Table 6.1: Rademacher complexity bounds for AUC maximization. We have $1/p+1/q = 1$ and $q > 1$.

2. **Conditioning**: For every $\mathbf{x} \in \mathcal{X}$, we will create a function class

$$\mathcal{F}_{\mathbf{x}} = \left\{ \mathbf{x}' \mapsto \left\langle h, \psi(\mathbf{x}, \mathbf{x}') \right\rangle : h \in \mathcal{H} \right\}.$$

Since $\mathcal{F}_{\mathbf{x}}$ is a univariate function class, we will use Theorem 6.11 to bound $\mathcal{R}_n(\mathcal{F}_{\mathbf{x}})$. Since $\mathcal{R}_n(\mathcal{H}) = \mathop{\mathbb{E}}_{\mathbf{x}} [\![\mathcal{R}_n(\mathcal{F}_{\mathbf{x}})]\!]$, we shall obtain Rademacher complexity bounds for $\mathcal{H}$.

We give below some examples of learning situations where these results may be applied. For sake of convenience we present the examples using loss functions for classification tasks but the same can be extended to other learning problems such as regression, multi-class classification and ordinal regression.

6.6.2 AUC maximization for Linear Prediction

In this case the goal is to maximize the area under the ROC curve for a linear classification problem at hand. This translates itself to a learning situation where $\mathcal{W}, \mathcal{X} \subseteq \mathbb{R}^d$. We have $h_{\mathbf{w}}(\mathbf{x}, \mathbf{x}') = \mathbf{w}^\top \mathbf{x} - \mathbf{w}^\top \mathbf{x}'$ and $\ell(h_{\mathbf{w}}, z_1, z_2) = \phi\left((y - y')h_{\mathbf{w}}(\mathbf{x}, \mathbf{x}')\right)$ where ϕ is the hinge loss or the exponential loss Zhao et al. (2011).

In order to apply Theorem 6.11, we rewrite the hypothesis as $h_{\mathbf{w}}(\mathbf{x}, \mathbf{x}') = \mathbf{w}^\top(\mathbf{x} - \mathbf{x}')$ and consider the input domain $\tilde{\mathcal{X}} = \{\mathbf{x} - \mathbf{x}' : \mathbf{x}, \mathbf{x}' \in \mathcal{X}\}$ and the map $\psi : (\mathbf{x}, \mathbf{x}') \mapsto \mathbf{x} - \mathbf{x}'$. Clearly if $\mathcal{X} \subseteq \{\mathbf{x} : \|\mathbf{x}\| \leq X\}$ then $\tilde{\mathcal{X}} \subseteq \{\mathbf{x} : \|\mathbf{x}\| \leq 2X\}$ and thus we have $\left\|\tilde{\mathcal{X}}\right\| \leq 2\|\mathcal{X}\|$ for any norm $\|\cdot\|$. It is now possible to regularize the hypothesis class $\mathcal{W}$ using a variety of norms.

If we wish to define our hypothesis class as $\mathcal{B}_q(\cdot), q > 1$, then in order to apply Theorem 6.11, we can use the regularizer $F(\mathbf{w}) = \|\mathbf{w}\|_q^2$. If we wish the sparse hypotheses class, $\mathcal{B}_1(W_1)$, we can use the regularizer $F(\mathbf{w}) = \|\mathbf{w}\|_q^2$ with $q = \frac{\log d}{\log d - 1}$ as this regularizer is strongly convex with respect to the L_1 norm Kakade et al. (2012). Table 6.1 gives a succinct summary of such possible regularizations and corresponding Rademacher complexity bounds.

Note that we obtain dimension independence, for example when the classifiers are L_2 regularized which allows us to bound the Rademacher complexities of kernelized function classes for bounded kernels as well.

Kernelized AUC maximization: Since the L_2 regularized hypothesis class has a dimension independent Rademacher complexity, it is possible to give guarantees for algorithms performing AUC maximization using kernel classifiers as well. In this case we have a Mercer kernel K with associated reproducing kernel Hilbert space $\mathcal{H}_K$ and

Hypothesis Class	Rademacher Complexity
$\mathcal{B}_{2,2}(\|\mathcal{W}\|_{2,2})$	$\|\mathcal{X}\|_2^2 \|\mathcal{W}\|_{2,2} \sqrt{\frac{1}{n}}$
$\mathcal{B}_{2,1}(\|\mathcal{W}\|_{2,1})$	$\|\mathcal{X}\|_2 \|\mathcal{X}\|_\infty \|\mathcal{W}\|_{2,1} \sqrt{\frac{e \log d}{n}}$
$\mathcal{B}_{1,1}(\|\mathcal{W}\|_{1,1})$	$\|\mathcal{X}\|_\infty^2 \|\mathcal{W}\|_{1,1} \sqrt{\frac{2e \log d}{n}}$
$\mathcal{B}_{S(1)}(\|\mathcal{W}\|_{S(1)})$	$\|\mathcal{X}\|_2^2 \|\mathcal{W}\|_{S(1)} \sqrt{\frac{e \log d}{n}}$

Table 6.2: Rademacher complexity bounds for Similarity and Metric learning

feature map $\Phi_K : \mathcal{X} \to \mathcal{H}_K$. Our predictors lie in the RKHS, i.e., $\mathbf{w} \in \mathcal{H}_K$ and we have $h_{\mathbf{w}}(\mathbf{x}, \mathbf{x}') = \mathbf{w}^\top (\Phi_K(\mathbf{x}) - \Phi_K(\mathbf{x}'))$. In this case we will have to use the map $\psi : (\mathbf{x}, \mathbf{x}') \mapsto \Phi_K(\mathbf{x}) - \Phi_K(\mathbf{x}') \in \mathcal{H}_K$. If the kernel is bounded, i.e., for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, we have $|K(\mathbf{x}, \mathbf{x}')| \leq \kappa^2$, then we can get a Rademacher average bound of $2\kappa \|\mathcal{W}\|_2 \sqrt{\frac{1}{n}}$.

6.6.3 Linear Similarity and Mahalanobis Metric learning

A variety of applications, such as in vision, require one to fine tune one's notion of proximity by learning a similarity or metric function over the input space. We consider some such examples below. In the following, we have $\mathbf{W} \in \mathbb{R}^{d \times d}$.

1. *Mahalanobis metric learning*: in this case we wish to learn a Mahalanobis metric $M_{\mathbf{W}}(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^\top \mathbf{W} (\mathbf{x} - \mathbf{x}')$ using the loss function

$$\ell(M_{\mathbf{W}}, \mathbf{z}, \mathbf{z}') = \phi\left(yy'\left(1 - M_{\mathbf{W}}^2(\mathbf{x}, \mathbf{x}')\right)\right)$$

   Jin et al. (2009) for a hypothesis class $\mathcal{W} \subset \mathbb{R}^{d \times d}$.

2. *Linear kernel learning*: in this case we wish to learn a linear kernel function $K_{\mathbf{W}}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{W} \mathbf{x}', \mathbf{W} \succeq 0$. A variety of loss functions have been proposed to aid the learning process

   (a) *Kernel-target Alignment*: the loss function used is $\ell(K_{\mathbf{W}}, \mathbf{z}, \mathbf{z}') = \phi\left(yy' K_{\mathbf{W}}(\mathbf{x}, \mathbf{x}')\right)$ where ϕ is used to encode some notion of alignment Cristianini et al. (2001); Cortes et al. (2010b).

   (b) *S-Goodness*: this is used in case one wishes to learn a *good* similarity function that need not be positive semi definite Bellet et al. (2012); Balcan and Blum (2006) by defining $\ell(K_{\mathbf{W}}, \mathbf{z}) = \phi\left(y \mathop{\mathbb{E}}_{(\mathbf{x}', y')} [\![y' K_{\mathbf{W}}(\mathbf{x}, \mathbf{x}')]\!]\right)$.

In order to apply Theorem 6.11, we will again rewrite the hypothesis and consider a different input domain. For the similarity learning problem, write the similarity function as $K_{\mathbf{W}}(\mathbf{x}, \mathbf{x}') = \left\langle \mathbf{W}, \mathbf{x}\mathbf{x}'^\top \right\rangle$ and consider the input space $\tilde{\mathcal{X}} = \left\{\mathbf{x}\mathbf{x}'^\top : \mathbf{x}, \mathbf{x}' \in \mathcal{X}\right\} \subseteq \mathbb{R}^{d \times d}$ along with the map $\psi : (\mathbf{x}, \mathbf{x}') \mapsto \mathbf{x}\mathbf{x}'^\top$. For the metric learning problem, rewrite the metric as $M_{\mathbf{W}}(\mathbf{x}, \mathbf{x}') = \left\langle \mathbf{W}, (\mathbf{x} - \mathbf{x}')(\mathbf{x} - \mathbf{x}')^\top \right\rangle$ and consider the input space $\tilde{\mathcal{X}} = \left\{(\mathbf{x} - \mathbf{x}')(\mathbf{x} - \mathbf{x}')^\top : \mathbf{x}, \mathbf{x}' \in \mathcal{X}\right\} \subseteq \mathbb{R}^{d \times d}$ along with the map $\psi : (\mathbf{x}, \mathbf{x}') \mapsto (\mathbf{x} - \mathbf{x}')(\mathbf{x} - \mathbf{x}')^\top$.

Hypothesis Class	Rademacher Avg. Bound
$\mathcal{S}_2(1)$	$\kappa^2\sqrt{\frac{p}{n}}$
$\Delta(1)$	$\kappa^2\sqrt{\frac{e\log p}{n}}$

Table 6.3: Rademacher complexity bounds for Multiple kernel learning

In this case it is possible to apply a variety of matrix norms to regularize the hypothesis class. We consider the following (mixed) matrix norms : $\|\cdot\|_{1,1}$, $\|\cdot\|_{2,1}$ and $\|\cdot\|_{2,2}$. Note that the $(2,2)$-norm regularization offers a dimension independent bound.

We also consider the Schatten norm $\|\mathbf{X}\|_{S(p)} := \|\boldsymbol{\sigma}(\mathbf{X})\|_p$ that includes the widely used *trace norm* $\|\boldsymbol{\sigma}(\mathbf{X})\|_1$. As before, we define norm bounded balls in the Banach space as follows: $\mathcal{B}_{p,q}(r) := \left\{\mathbf{x} : \|\mathbf{x}\|_{p,q} \leq r\right\}$.

Using results on construction of strongly convex functions with respect to theses norms from Kakade et al. (2012), it is possible to get bounds on the Rademacher averages of the various hypothesis classes. However these bounds involve norm bounds for the modified domain $\tilde{\mathcal{X}}$. We make these bounds explicit by expressing norm bounds for $\tilde{\mathcal{X}}$ in terms of those for $\mathcal{X}$. From the definition of $\tilde{\mathcal{X}}$ for the similarity learning problems, we get, for any $p, q \geq 1$, $\left\|\tilde{\mathcal{X}}\right\|_{p,q} \leq \|\mathcal{X}\|_p \|\mathcal{X}\|_q$. Also, since every element of $\tilde{\mathcal{X}}$ is of the form $\mathbf{x}\mathbf{x}'^{\top}$, it has only one non zero singular value $\|\mathbf{x}\|_2 \|\mathbf{x}'\|_2$ which gives us $\left\|\tilde{X}\right\|_{S(p)} \leq \|\mathcal{X}\|_2^2$ for any $p \geq 1$.

For the metric learning problem, we can similarly get $\left\|\tilde{X}\right\|_{p,q} \leq 4\|\mathcal{X}\|_p \|\mathcal{X}\|_q$ and $\left\|\tilde{X}\right\|_{S(p)} \leq 4\|\mathcal{X}\|_2^2$ for any $p \geq 1$ which allows us to get similar bounds as those for similarity learning but for an extra constant factor. We summarize our bounds in Table 6.2. We note that Cao et al. (2012) devote a substantial amount of effort to calculate these values for the mixed norms on a case-by-case basis (and do not consider Schatten norms either) whereas, using results exploiting strong convexity and strong smoothness from (Kakade et al., 2012), we are able to get the same as simple corollaries.

6.6.4 Two-stage Multiple kernel learning

The analysis of the previous example can be replicated for learning non-linear Mercer kernels as well. Additionally, since all Mercer kernels yield Hilbertian metrics, these methods can be extended to learning Hilbertian metrics as well. However, since Hilbertian metric learning has not been very popular in literature, we restrict our analysis to kernel learning alone. We present this example using the framework proposed by Kumar et al. (2012) due to its simplicity and generality.

The goal here is to improve the SVM classification algorithm by learning a *good* kernel K that is a positive combination of *base* kernels. We are given p Mercer kernels $K_1, \ldots, K_p$ that are bounded, i.e., for all i, $|K_i(\mathbf{x}, \mathbf{x}')| \leq \kappa^2$ for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ and our task is to find a combination of these kernels given by a vector $\boldsymbol{\mu} \in \mathbb{R}^p, \boldsymbol{\mu} \geq 0$ such that the kernel $K_{\boldsymbol{\mu}}(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^p \boldsymbol{\mu}_i K_i(\mathbf{x}, \mathbf{x}')$ is a *good* kernel Balcan and Blum (2006). In this case the

loss function used is $\ell(\boldsymbol{\mu}, \mathbf{z}, \mathbf{z}') = \phi\left(yy' K_{\boldsymbol{\mu}}(\mathbf{x}, \mathbf{x}')\right)$ where $\phi(\cdot)$ is meant to encode some notion of alignment. Kumar et al. (2012) take $\phi(\cdot)$ to be the hinge loss.

To apply Theorem 6.11, we simply use the "K-space" construction proposed in Kumar et al. (2012). We write $K_{\boldsymbol{\mu}}(\mathbf{x}, \mathbf{x}') = \langle \boldsymbol{\mu}, z(\mathbf{x}, \mathbf{x}') \rangle$ where $z(\mathbf{x}, \mathbf{x}') = (K_1(\mathbf{x}, \mathbf{x}'), \ldots, K_p(\mathbf{x}, \mathbf{x}'))$. Consequently our modified input space looks like $\tilde{\mathcal{X}} = \{z(\mathbf{x}, \mathbf{x}') : \mathbf{x}, \mathbf{x}' \in \mathcal{X}\} \subseteq \mathbb{R}^p$ with the map $\psi : (\mathbf{x}, \mathbf{x}') \mapsto z(\mathbf{x}, \mathbf{x}')$. Popular regularizations on the kernel combination vector $\boldsymbol{\mu}$ include the sparsity inducing L_1 regularization that constrains $\boldsymbol{\mu}$ to lie on the unit simplex $\Delta(1) = \{\boldsymbol{\mu} : \|\boldsymbol{\mu}\|_1 = 1, \boldsymbol{\mu} \geq 0\}$ and L_2 regularization that restricts $\boldsymbol{\mu}$ to lie on the unit sphere $\mathcal{S}_2(1) = \{\boldsymbol{\mu} : \|\boldsymbol{\mu}\|_2 = 1, \boldsymbol{\mu} \geq 0\}$. Arguments similar to the one used to discuss the case of AUC maximization for linear predictors give us bounds on the Rademacher averages for these two hypothesis classes in terms of $\left\|\tilde{X}\right\|_2$ and $\left\|\tilde{X}\right\|_\infty$. Since $\left\|\tilde{X}\right\|_2 \leq \kappa^2\sqrt{p}$ and $\left\|\tilde{X}\right\|_\infty \leq \kappa^2$, we obtain explicit bounds on the Rademacher averages that are given in Table 6.3.

Several previous works (eg. Cortes et al. (2010b)) have provided Rademacher average bounds for the hypothesis class that consists of *predictors* that come from these kernel spaces. More specifically, the hypothesis class in these works is

$$\mathcal{F} = \left\{ f \in \mathcal{H}_K : \|f\|_{\mathcal{H}_K} \leq 1, K = \sum_{i=1}^{p} \boldsymbol{\mu}_i K_i, \boldsymbol{\mu} \geq 0, \|\boldsymbol{\mu}\|_q = 1 \right\}$$

where $q = 1, 2$. The bounds presented above, however, consider the hypothesis class of the kernel functions themselves, true to the spirit of two-stage multiple kernel learning. In other words, our hypothesis class is

$$\mathcal{K} = \left\{ K : K = \sum_{i=1}^{p} \boldsymbol{\mu}_i K_i, \boldsymbol{\mu} \geq 0, \|\boldsymbol{\mu}\|_q = 1 \right\}$$

where $q = 1, 2$. Our objective here is to simply learn a good kernel and leave the predictor learning task to the second stage.

We note that for the L_1 regularized case, our bound has a similar dependence on the number of kernels, i.e., $\sqrt{\log p}$ as the bounds presented in Cortes et al. (2010a). For the L_2 case however, we have a worse dependence of $\sqrt{p}$ than Cortes et al. (2010a) who get a $\sqrt[4]{p}$ dependence. This may be attributed to the fact that we analyze the problem in a black box manner and use fairly generic bounds by looking at the class $\mathcal{K}$ and are thus not able to exploit the internal structure and additional constraints placed on the hypothesis class $\mathcal{F}$. However, it is a bit unfair to compare the two bounds since Cortes et al. (2010a) consider single stage kernel learning algorithms that try to learn the kernel combination as well as the classifier in a single step whereas we are dealing with a two-stage process where classifier learning is disjoint from the kernel learning step.

In particular, we have a worse dependence on the κ parameter - whereas our dependence is quadratic, Cortes et al. (2010b) have only a linear dependence. This is due to the extra constraint $\|f\|_{\mathcal{H}_K} \leq 1$ placed on the elements of the hypothesis class $\mathcal{F}$. This tightly

Algorithm 7 RS-x : Stream Subsampling with Replacement

Input: Buffer B, new point $\mathbf{z}_t$, buffer size s, timestep t.

if $|B| < s$ **then** {There is space}
  $B \leftarrow B \cup \{\mathbf{z}_t\}$
else {Overflow situation}
  if $t = s + 1$ **then** {Repopulation step}
    TMP $\leftarrow B \cup \{\mathbf{z}_t\}$
    Repopulate B with s points sampled uniformly with replacement from TMP.
  else {Normal update step}
    Independently, replace each point of B with $\mathbf{z}_t$ with probability $1/t$.
  end if
end if
return B

Algorithm 8 OLP : Online Learning with Pairwise Loss Functions

Input: Step length scale η, Buffer size s
Output: An ensemble $\mathbf{w}_2, \ldots, \mathbf{w}_n \in \mathcal{W}$ with low regret

$\mathbf{w}_0 \leftarrow \mathbf{0}$, $B \leftarrow \phi$
for $t = 1$ **to** n **do**
  Obtain a training point $\mathbf{z}_t$
  Set step length $\eta_t \leftarrow \frac{\eta}{\sqrt{t}}$
  $\mathbf{w}_t \leftarrow \Pi_{\mathcal{W}} \left[\mathbf{w}_{t-1} + \frac{\eta_t}{|B|} \sum_{\mathbf{z} \in B} \nabla_{\mathbf{w}} \ell(\mathbf{w}_{t-1}, \mathbf{z}_t, \mathbf{z}) \right]$ {$\Pi_{\mathcal{W}}$ projects onto the set $\mathcal{W}$}
  $B \leftarrow$ Update-buffer$(B, \mathbf{z}_t, s, t)$ {using **RS-x**}
end for
return $\mathbf{w}_2, \ldots, \mathbf{w}_n$

bounds the components of the function f in the various RKHSes $\mathcal{H}_{K_1}, \ldots, \mathcal{H}_{K_p}$ whereas our analysis is completely oblivious to functions in the RKHSes and hence considers an enlarged hypothesis class. It should be possible, as well as would be interesting, to get better bounds by doing a fine case-by-case analysis for the multiple kernel learning case.

6.7 OLP : Online Learning with Pairwise Loss Functions

In this section, we present an online learning algorithm for learning with pairwise loss functions in a finite buffer setting. The key contribution in this section is a buffer update policy that when combined with a variant of the GIGA algorithm (Zinkevich, 2003) allows us to give high probability regret bounds.

In previous work, Zhao et al. presented an online learning algorithm that uses finite buffers with the **RS** policy and proposed an *all-pairs* regret bound. The **RS** policy ensures, over the randomness used in buffer updates, that at any given time, the buffer contains a uniform sample from the preceding stream. Using this property, (Zhao et al., 2011, Lemma 2) claimed that $\mathbb{E}\left[\!\left[\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1})\right]\!\right] = \hat{\mathcal{L}}_t(h_{t-1})$ where the expectation is taken over the randomness used in buffer construction. However, a property such as $\mathbb{E}\left[\!\left[\hat{\mathcal{L}}_t^{\text{buf}}(h)\right]\!\right] = \hat{\mathcal{L}}_t(h)$ holds only for functions h that are either fixed or obtained independently of the random

variables used in buffer updates (over which the expectation is taken). Since h_{t-1} is learned from points in the buffer itself, the above property, and consequently the regret bound, does not hold.

We remedy this issue by showing a relatively weaker claim; we show that with high probability we have $\hat{\mathcal{L}}_t(h_{t-1}) \leq \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) + \varepsilon$. At a high level, this claim is similar to showing uniform convergence bounds for $\hat{\mathcal{L}}_t^{\text{buf}}$. However, the reservoir sampling algorithm is not particularly well suited to prove such uniform convergence bounds as it essentially performs sampling without replacement (see Section 6.10 for a discussion). We overcome this hurdle by proposing a new buffer update policy **RS-x** (see Algorithm 7) that, at each time step, guarantees s i.i.d. samples from the preceding stream. The following theorem formalizes the properties of the sampling algorithm.

Theorem 6.12. *Suppose we have a stream of elements $\mathbf{z}_1, \ldots, \mathbf{z}_n$ being sampled into a buffer B of size s using the **RS-x** algorithm. Then at any time $t \geq s+2$, each element of B is an i.i.d. sample from the set Z^{t-1}.*

However, it turns out that are certain issues in implementing the **RS-x** buffer update policy. To remedy this, we propose an efficient alternate implementation of the same policy, details of which can be found in Section 6.11. Our algorithm uses this buffer update policy in conjunction with an online learning algorithm **OLP** (see Algorithm 8) that is a variant of the well-known GIGA algorithm of Zinkevich. We provide the following *all-pairs* regret guarantee for our algorithm:

Theorem 6.13. *Suppose the **OLP** algorithm working with an s-sized buffer generates an ensemble $\mathbf{w}_1, \ldots, \mathbf{w}_{n-1}$. Then with probability at least $1-\delta$,*

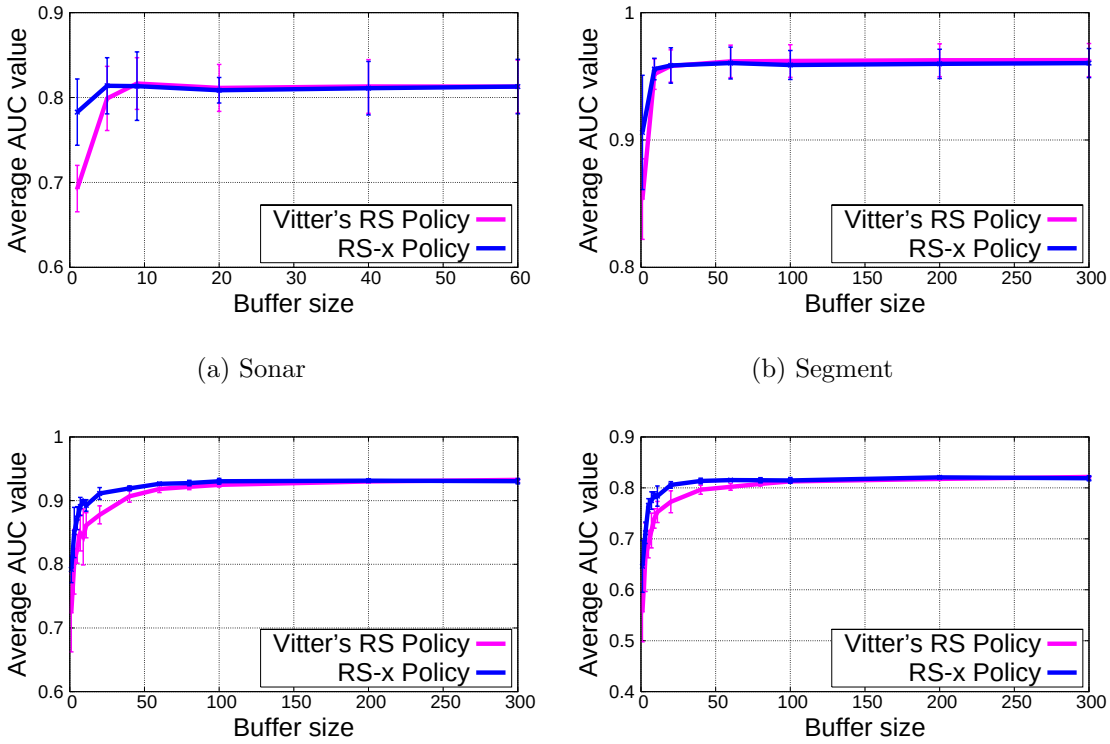
$$\frac{\mathfrak{R}_n}{n-1} \leq \mathcal{O}\left(C_d\sqrt{\frac{\log\frac{n}{\delta}}{s}} + \sqrt{\frac{1}{n-1}}\right)$$

See Section 6.12.8 for the proof. A drawback of our bound is that it offers sublinear regret only for buffer sizes $s = \omega(\log n)$. A better regret bound for constant s or a lower-bound on the regret is an open problem.

6.8 Experimental Evaluation

In this section we present experimental evaluation of our proposed **OLP** algorithm. We stress that the aim of this evaluation is to show that our algorithm, that enjoys high confidence regret bounds, also performs competitively in practice with respect to the OAM_{gra} algorithm proposed by Zhao et al. since our results in Section 6.5 show that OAM_{gra} does enjoy good generalization guarantees despite the lack of an *all-pairs* regret bound.

In our experiments, we adapted the **OLP** algorithm to the AUC maximization problem and compared it with OAM_{gra} on 18 different benchmark datasets. We used 60% of



(a) Sonar

(b) Segment

(c) IJCNN

(d) Covertype

Figure 6.1: Performance of **OLP** (using **RS-x**) and OAM_{gra} (using **RS**) by Zhao et al. (2011) on AUC maximization tasks with varying buffer sizes.

the available data points up to a maximum of 20000 points to train both algorithms. We refer the reader to Section 6.11 for a discussion on the implementation of the **RS-x** algorithm. Figure 6.1 presents the results of our experiments on 4 datasets across 5 random training/test splits. Results on other datasets can be found in Section 6.13. The results demonstrate that **OLP** performs competitively to OAM_{gra} while in some cases having slightly better performance for small buffer sizes.

6.9 Discussion

In this work we studied the generalization capabilities of online learning algorithms for pairwise loss functions from several different perspectives. Using the method of *Symmetrization of Expectations*, we first provided sharp online to batch conversion bounds for algorithms that offer *all-pairs* regret bounds. Our results for bounded and strongly convex loss functions closely match their first order counterparts. We also extended our analysis to algorithms that are only able to provide *finite-buffer* regret bounds using which we were able to explain the good empirical performance of some existing algorithms. Finally we presented a new memory-efficient online learning algorithm that is able to provide *all-pairs* regret bounds in addition to performing well empirically.

Several interesting directions can be pursued for future work, foremost being the development of online learning algorithms that can guarantee sub-linear regret at constant buffer sizes or else a regret lower bound for finite buffer algorithms. Secondly, the idea of a *stream-aware* buffer update policy is especially interesting both from an empirical as well as theoretical point of view and would possibly require novel proof techniques for its analysis. Lastly, scalability issues that arise when working with higher order loss functions also pose an interesting challenge.

6.10 Regret Bounds for Reservoir Sampling Algorithms

The Reservoir Sampling algorithm (Vitter, 1985) essentially performs sampling without replacement which means that the samples present in the buffer are not i.i.d. samples from the preceding stream. Due to this, proving regret bounds by way of uniform convergence arguments becomes a bit more difficult. However, there has been a lot of work on analyzing learning algorithms that learn from non-i.i.d. data such as data generated by ergodic processes. Of particular interest is a result by Serfling [3] that gives Hoeffding style bounds for data generated from a finite population without replacement.

Although Serfling's result does provide a way to analyze the **RS** algorithm, doing so directly would require using arguments that involve covering numbers that offer bounds that are dimension dependent and that are not tight. It would be interesting to see if equivalents of the McDiarmid's inequality and Rademacher averages can be formulated for samples obtained without replacement to get tighter results. For our purposes, we remedy the situation by proposing a new sampling algorithm that gives us i.i.d. samples in the buffer allowing existing techniques to be used to obtain regret bounds.

6.11 Implementing the RS-x Algorithm

Although the **RS-x** algorithm presented in the chapter allows us to give clean regret bounds, it suffers from a few drawbacks. From a theoretical point of view, the algorithm is inferior to Vitter's **RS** algorithm in terms of randomness usage. The **RS** algorithm (see Zhao et al. (2011) for example) uses a Bernoulli random variable and a discrete uniform random variable at each time step. The discrete random variable takes values in $[s]$ as a result of which the algorithm uses a total of $\mathcal{O}(\log s)$ random bits at each step.

The **RS-x** algorithm as proposed, on the other hand, uses s Bernoulli random variables at each step (to decide which buffer elements to replace with the incoming point) taking its randomness usage to $\mathcal{O}(s)$ bits. From a practical point of view this has a few negative consequences:

1. Due to increased randomness usage, the variance of the resulting algorithm increases.

[3] ↑ R. J. Serfling, Probability Inequalities for the Sum in Sampling without Replacement, *The Annals of Statistics*, 2(1):39-48, 1974.

Algorithm 9 RS-x^2 : An Alternate Implementation of the **RS-x** Algorithm

Input: Buffer B, new point $\mathbf{z}_t$, buffer size s, timestep t
Output: Updated buffer B_{new}

if $|B| < s$ **then** {There is space}
  $B_{\text{new}} \leftarrow B \cup \{\mathbf{z}_t\}$
else {Overflow situation}
  if $t = s + 1$ **then** {Repopulation step}
    TMP $= B \cup \{\mathbf{z}_t\}$
    $B_{\text{new}} = \phi$
    for $i = 1$ **to** s **do**
      Select random $\mathbf{r} \in$ TMP with replacement
      $B_{\text{new}} \leftarrow B_{\text{new}} \cup \{\mathbf{r}\}$
    end for
  else {Normal update step}
    $B_{\text{new}} \leftarrow B$
    Sample $k \sim \text{Binomial}(s, 1/t)$
    Remove k random elements from B_{new}
    $B_{\text{new}} \leftarrow B_{\text{new}} \cup \left(\coprod_{i=1}^{k} \{\mathbf{z}_t\}\right)$
  end if
end if
return B_{new}

2. At step t, the Bernoulli random variables required all have success probability $1/t$. This quantity drops down to negligible values for even moderate values of t. Note that Vitter's **RS** on the other hand requires a Bernoulli random variable with success probability s/t which dies down much more slowly.

3. Due to the requirement of such high precision random variables, the imprecisions of any pseudo random generator used to simulate this algorithm become apparent resulting in poor performance.

In order to ameliorate the situation, we propose an alternate implementation of the *normal* update step of the **RS-x** algorithm in Algorithm 9. We call this new sampling policy **RS-x^2** . We shall formally demonstrate the equivalence of the **RS-x** and the **RS-x^2** policies by showing that both policies result in a buffer whose each element is a uniform sample from the preceding stream with replacement. This shall be done by proving that the joint distribution of the buffer elements remains the same whether the **RS-x** *normal* update is applied or the **RS-x^2** *normal* step is applied (note that **RS-x** and **RS-x^2** have identical *repopulation steps*). This will ensure that any learning algorithm will be unable to distinguish between the two update mechanisms and consequently, our regret guarantees shall continue to hold.

First we analyze the randomness usage of the **RS-x^2** update step. The update step first samples a number $K_t \sim B(s, 1/t)$ from the binomial distribution and then replaces K_t random locations with the incoming point. Choosing k locations without replacement from a pool of s locations requires at most $k \log s$ bits of randomness. Since K_t is sampled from

the binomial distribution $B(s, 1/t)$, we have $K_t = \mathcal{O}(1)$ in expectation (as well as with high probability) since $t > s$ whenever this step is applied. Hence our randomness usage per update is at most $\mathcal{O}(\log s)$ random bits which is much better than the randomness usage of **RS-x** and that actually matches that of Vitter's **RS** upto a constant.

To analyze the statistical properties of the **RS-x²** update step, let us analyze the state of the buffer after the update step. In the **RS-x** algorithm, the state of the buffer after an update is completely specified once we enumerate the locations that were replaced by the incoming point. Let the indicator variable R_i indicate whether the i^{th} location was replaced or not. Let $r \in \{0,1\}^s$ denote a *fixed* pattern of replacements. Then the original implementation of the update step of **RS-x** guarantees that

$$\mathop{\mathbb{P}}_{\textbf{RS-x}}\left[\bigwedge_{i=1}^{s}(R_i = r_i)\right] = \left(\frac{1}{t}\right)^{\|r\|_1}\left(1-\frac{1}{t}\right)^{s-\|r\|_1}$$

To analyze the same for the alternate implementation of the **RS-x²** update step, we first notice that choosing k items from a pool of s without replacement is identical to choosing the first k locations from a random permutation of the s items. Let us denote $\|r\|_1 = k$. Then we have,

$$\begin{aligned}\mathop{\mathbb{P}}_{\textbf{RS-x}^\textbf{2}}\left[\bigwedge_{i=1}^{s}(R_i = r_i)\right] &= \sum_{j=1}^{s}\mathbb{P}\left[\bigwedge_{i=1}^{s}(R_i = r_i)\wedge K_t = j\right]\\ &= \mathbb{P}\left[\bigwedge_{i=1}^{s}(R_i = r_i)\wedge K_t = k\right]\\ &= \mathbb{P}\left[\bigwedge_{i=1}^{s}(R_i = r_i)\,\middle|\, K_t = k\right]\mathbb{P}\left[K_t = k\right]\end{aligned}$$

We have

$$\mathbb{P}\left[K_t = k\right] = \binom{s}{k}\left(\frac{1}{t}\right)^k\left(1-\frac{1}{t}\right)^{s-k}$$

The number of arrangements of s items such that some specific k items fall in the first k positions is $k!(s-k)!$. Thus we have

$$\begin{aligned}\mathop{\mathbb{P}}_{\textbf{RS-x}^\textbf{2}}\left[\bigwedge_{i=1}^{s}(R_i = r_i)\right] &= \binom{s}{k}\left(\frac{1}{t}\right)^k\left(1-\frac{1}{t}\right)^{s-k}\frac{k!(s-k)!}{s!}\\ &= \left(\frac{1}{t}\right)^k\left(1-\frac{1}{t}\right)^{s-k}\\ &= \mathop{\mathbb{P}}_{\textbf{RS-x}}\left[\bigwedge_{i=1}^{s}(R_i = r_i)\right]\end{aligned}$$

which completes the argument.

6.12 Proofs

6.12.1 Proof of Lemma 6.1

As a first step, we decompose the excess risk in a manner similar to Wang et al.. For any $h \in \mathcal{H}$ let

$$\tilde{\mathcal{L}}_t(h) := \mathop{\mathbb{E}}_{\mathbf{z}_t}\left[\!\!\left[\hat{\mathcal{L}}_t(h) \middle| Z^{t-1} \right]\!\!\right].$$

This allows us to decompose the excess risk as follows:

$$\frac{1}{n-1}\sum_{t=2}^{n} \mathcal{L}(h_{t-1}) - \hat{\mathcal{L}}_t(h_{t-1}) = \frac{1}{n-1}\left(\sum_{t=2}^{n} \underbrace{\mathcal{L}(h_{t-1}) - \tilde{\mathcal{L}}_t(h_{t-1})}_{P_t} + \underbrace{\tilde{\mathcal{L}}(h_{t-1}) - \hat{\mathcal{L}}_t(h_{t-1})}_{Q_t}\right).$$

By construction, we have $\mathop{\mathbb{E}}_{\mathbf{z}_t}\left[\!\!\left[Q_t \middle| Z^{t-1} \right]\!\!\right] = 0$ and hence the sequence $Q_2, \ldots, Q_n$ forms a martingale difference sequence. Since $|Q_t| \leq B$ as the loss function is bounded, an application of the Azuma-Hoeffding inequality shows that with probability at least $1-\delta$

$$\frac{1}{n-1}\sum_{t=2}^{n} Q_t \leq B\sqrt{\frac{2\log\frac{1}{\delta}}{n-1}}. \tag{6.4}$$

We now analyze each term P_t individually. By linearity of expectation, we have for a ghost sample $\tilde{Z}^{t-1} = \{\tilde{\mathbf{z}}_1, \ldots, \tilde{\mathbf{z}}_{t-1}\}$,

$$\mathcal{L}(h_{t-1}) = \mathop{\mathbb{E}}_{\tilde{Z}^{t-1}}\left[\!\!\left[\frac{1}{t-1}\sum_{\tau=1}^{t-1} \mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1}, \mathbf{z}, \tilde{\mathbf{z}}_\tau) \right]\!\right] \right]\!\!\right]. \tag{6.5}$$

The expression of $\mathcal{L}(h_{t-1})$ as a nested expectation is the precursor to performing symmetrization with expectations and plays a crucial role in overcoming coupling problems. This allows us to write P_t as

$$\begin{aligned} P_t &= \mathop{\mathbb{E}}_{\tilde{Z}^{t-1}}\left[\!\!\left[\frac{1}{t-1}\sum_{\tau=1}^{t-1} \mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1}, \mathbf{z}, \tilde{\mathbf{z}}_\tau) \right]\!\right] \right]\!\!\right] - \tilde{\mathcal{L}}_t(h_{t-1}) \\ &\leq \underbrace{\sup_{h\in\mathcal{H}}\left[\mathop{\mathbb{E}}_{\tilde{Z}^{t-1}}\left[\!\!\left[\frac{1}{t-1}\sum_{\tau=1}^{t-1} \mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h, \mathbf{z}, \tilde{\mathbf{z}}_\tau) \right]\!\right] \right]\!\!\right] - \tilde{\mathcal{L}}_t(h) \right]}_{g_t(\mathbf{z}_1, \ldots, \mathbf{z}_{t-1})}. \end{aligned}$$

Since $\tilde{\mathcal{L}}_t(h) = \mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\!\left[\frac{1}{t-1}\sum_{\tau=1}^{t-1} \ell(h, \mathbf{z}, \mathbf{z}_\tau) \middle| Z^{t-1} \right]\!\!\right]$ and ℓ is bounded, the expression $g_t(\mathbf{z}_1, \ldots, \mathbf{z}_{t-1})$ can have a variation of at most $B/(t-1)$ when changing any of its $(t-1)$ variables. Hence an application of McDiarmid's inequality gives us, with probability at least $1-\delta$,

$$g_t(\mathbf{z}_1, \ldots, \mathbf{z}_{t-1}) \leq \mathop{\mathbb{E}}_{Z^{t-1}}\left[\!\left[g_t(\mathbf{z}_1, \ldots, \mathbf{z}_{t-1}) \right]\!\right] + B\sqrt{\frac{\log\frac{1}{\delta}}{2(t-1)}}.$$

For any $h \in \mathcal{H}, \mathbf{z}' \in \mathcal{Z}$, let $\wp(h, \mathbf{z}') := \frac{1}{t-1}\mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h, \mathbf{z}, \mathbf{z}')]\!]$. Then we can write

$$\begin{aligned}
\mathop{\mathbb{E}}_{Z^{t-1}}[\![g(\mathbf{z}_1, \ldots, \mathbf{z}_{t-1})]\!] &= \mathop{\mathbb{E}}_{Z^{t-1}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left[\mathop{\mathbb{E}}_{\tilde{Z}^{t-1}}\left[\!\!\left[\sum_{\tau=1}^{t-1}\wp(h, \tilde{\mathbf{z}}_\tau)\right]\!\!\right] - \sum_{\tau=1}^{t-1}\wp(h, \mathbf{z}_\tau)\right]\right]\!\!\right] \\
&\leq \mathop{\mathbb{E}}_{Z^{t-1}, \tilde{Z}^{t-1}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left[\sum_{\tau=1}^{t-1}\wp(h, \tilde{\mathbf{z}}_\tau) - \sum_{\tau=1}^{t-1}\wp(h, \mathbf{z}_\tau)\right]\right]\!\!\right] \\
&= \mathop{\mathbb{E}}_{Z^{t-1}, \tilde{Z}^{t-1}, \{\varepsilon_\tau\}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left[\sum_{\tau=1}^{t-1}\varepsilon_\tau\left(\wp(h, \tilde{\mathbf{z}}_\tau) - \wp(h, \mathbf{z}_\tau)\right)\right]\right]\!\!\right] \\
&\leq \frac{2}{t-1}\mathop{\mathbb{E}}_{Z^{t-1}, \{\varepsilon_\tau\}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left[\sum_{\tau=1}^{t-1}\varepsilon_\tau\mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h, \mathbf{z}, \mathbf{z}_\tau)]\!]\right]\right]\!\!\right] \\
&\leq \frac{2}{t-1}\mathop{\mathbb{E}}_{\mathbf{z}, Z^{t-1}, \{\varepsilon_\tau\}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left[\sum_{\tau=1}^{t-1}\varepsilon_\tau\ell(h, \mathbf{z}, \mathbf{z}_\tau)\right]\right]\!\!\right] \\
&= 2\mathcal{R}_{t-1}(\ell \circ \mathcal{H}).
\end{aligned}$$

Note that in the third step, the symmetrization was made possible by the decoupling step in Eq. (6.5) where we decoupled the "head" variable $\mathbf{z}_t$ from the "tail" variables by absorbing it inside an expectation. This allowed us to symmetrize the true and ghost samples $\mathbf{z}_\tau$ and $\tilde{\mathbf{z}}_\tau$ in a standard manner. Thus we have, with probability at least $1 - \delta$,

$$P_t \leq 2\mathcal{R}_{t-1}(\ell \circ \mathcal{H}) + B\sqrt{\frac{\log\frac{1}{\delta}}{2(t-1)}}.$$

Applying a union bound on the bounds for $P_t, t = 2, \ldots, n$ gives us with probability at least $1 - \delta$,

$$\frac{1}{n-1}\sum_{t=2}^{n} P_t \leq \frac{2}{n-1}\sum_{t=2}^{n}\mathcal{R}_{t-1}(\ell \circ \mathcal{H}) + B\sqrt{\frac{2\log\frac{n}{\delta}}{n-1}}. \tag{6.6}$$

Adding Equations (6.4) and (6.6) gives us the result.

6.12.2 Proof of Theorem 6.4

We begin with a lemma implicit in the proof of Theorem 1 in (Sridharan et al., 2008). For the function class $\mathcal{F}$ and loss function $\wp$ as above, define a new loss function $\mu : (f, \mathbf{x}) \mapsto \wp(f, \mathbf{x}) - \wp(f^*, \mathbf{x})$ with $\mathcal{M}$ and $\hat{\mathcal{M}}$ as the associated population and empirical risk functionals. Let $r = \frac{4L^2R^2C_d^2(32+\log(1/\delta))}{\sigma n}$. Then we have the following

Lemma 6.14. *For any $\varepsilon > 0$, with probability at least $1 - \delta$, the following happens*

1. *For all $f \in \mathcal{F}$ such that $\mathcal{M}(f) \leq 16\left(1 + \frac{1}{\varepsilon}\right)^2 r$, we have $\mathcal{M}(f) \leq \hat{\mathcal{M}}(f) + 4\left(1 + \frac{1}{\varepsilon}\right) r$.*
2. *For all $f \in \mathcal{F}$ such that $\mathcal{M}(f) > 16\left(1 + \frac{1}{\varepsilon}\right)^2 r$, we have $\mathcal{M}(f) \leq (1 + \varepsilon)\hat{\mathcal{M}}(f)$.*

The difference in our proof technique lies in the way we combine these two cases. We do so by proving the following two simple results.

Lemma 6.15. *For all f s.t. $\mathcal{M}(f) \leq 16\left(1+\frac{1}{\varepsilon}\right)^2 r$, we have*

$$\mathcal{M}(f) \leq (1+\varepsilon)\left(\hat{\mathcal{M}}(f) + 4\left(1+\frac{1}{\varepsilon}\right)r\right).$$

Proof. We notice that for all $f \in \mathcal{F}$, we have $\mathcal{M}(f) = \mathcal{P}(f) - \mathcal{P}(f^*) \geq 0$. Thus, using Lemma 6.14, Part 1, we have $\hat{\mathcal{M}}(f) + 4\left(1+\frac{1}{\varepsilon}\right)r \geq \mathcal{M}(f) \geq 0$. Since for any $a, \varepsilon > 0$, we have $a \leq (1+\varepsilon)a$, the result follows. □

Lemma 6.16. *For all f s.t. $\mathcal{M}(f) > 16\left(1+\frac{1}{\varepsilon}\right)^2 r$, we have*

$$\mathcal{M}(f) \leq (1+\varepsilon)\left(\hat{\mathcal{M}}(f) + 4\left(1+\frac{1}{\varepsilon}\right)r\right).$$

Proof. We use the fact that $r > 0$ and thus $4(1+\varepsilon)\left(1+\frac{1}{\varepsilon}\right)r > 0$ as well. The result then follows from an application of Part 2 of Lemma 6.14. □

From the definition of the loss function μ, we have for any $f \in \mathcal{F}$, $\mathcal{M}(f) = \mathcal{P}(f) - \mathcal{P}(f^*)$ and $\hat{\mathcal{M}}(f) = \hat{\mathcal{P}}(f) - \hat{\mathcal{P}}(f^*)$. Combining the above lemmata with this observation completes the proof.

6.12.3 Proof of Theorem 6.5

The decomposition of the excess risk shall not be made explicitly in this case but shall emerge as a side-effect of the proof progression. Consider the loss function $\wp(h, \mathbf{z}') := \mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h, \mathbf{z}, \mathbf{z}')]\!]$ with $\mathcal{P}$ and $\hat{\mathcal{P}}$ as the associated population and empirical risk functionals. Clearly, if ℓ is L-Lipschitz and σ-strongly convex then so is $\wp$. As Equation (6.5) shows, for any $h \in \mathcal{H}$, $\mathcal{P}(h) = \mathcal{L}(h)$. Also it is easy to see that for any Z^{t-1}, $\hat{\mathcal{P}}(h) = \tilde{\mathcal{L}}_t(h)$. Applying Theorem 6.4 on h_{t-1} with the loss function $\wp$ gives us w.p. $1-\delta$,

$$\mathcal{L}(h_{t-1}) - \mathcal{L}(h^*) \leq (1+\varepsilon)\left(\tilde{\mathcal{L}}_t(h_{t-1}) - \tilde{\mathcal{L}}_t(h^*)\right) + \frac{C_\delta}{\varepsilon\sigma(t-1)}$$

which, upon summing across time steps and taking a union bound, gives us with probability at least $1-\delta$,

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{C_{(\delta/n)}\log n}{\varepsilon\sigma(n-1)} + \frac{1+\varepsilon}{n-1}\sum_{t=2}^{n}\left(\tilde{\mathcal{L}}_t(h_{t-1}) - \tilde{\mathcal{L}}_t(h^*)\right).$$

Let $\xi_t := \left(\tilde{\mathcal{L}}_t(h_{t-1}) - \tilde{\mathcal{L}}_t(h^*)\right) - \left(\hat{\mathcal{L}}_t(h_{t-1}) - \hat{\mathcal{L}}_t(h^*)\right)$. Then using the regret bound $\mathfrak{R}_n$ we can write,

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{1+\varepsilon}{n-1}\left(\mathfrak{R}_n + \sum_{t=2}^{n}\xi_t\right) + \frac{C_{(\delta/n)}\log n}{\varepsilon\sigma(n-1)}.$$

We now use Bernstein type inequalities to bound the sum $\sum_{t=2}^{n} \xi_t$ using a proof technique used in (Kakade and Tewari, 2008; Cesa-Bianchi and Gentile, 2008). We first note some properties of the sequence below.

Lemma 6.17. *The sequence $\xi_2, \ldots, \xi_n$ is a bounded martingale difference sequence with bounded conditional variance.*

Proof. That ξ_t is a martingale difference sequence follows by construction: we can decompose the term $\xi_t = \phi_t - \psi_t$ where $\phi_t = \tilde{\mathcal{L}}_t(h_{t-1}) - \hat{\mathcal{L}}_t(h_{t-1})$ and $\psi_t = \tilde{\mathcal{L}}_t(h^*) - \hat{\mathcal{L}}_t(h^*)$, both of which are martingale difference sequences with respect to the common filtration $\mathcal{F} = \{\mathcal{F}_n : n = 0, 1, \ldots\}$ where $\mathcal{F}_n = \sigma(\mathbf{z}_i : i = 1, \ldots, n)$.

Since the loss function takes values in $[0, B]$, we have $|\xi_t| \leq 2B$ which proves that our sequence is bounded.

To prove variance bounds for the sequence, we first use the Lipschitz properties of the loss function to get

$$\xi_t = \left(\tilde{\mathcal{L}}_t(h_{t-1}) - \tilde{\mathcal{L}}_t(h^*)\right) - \left(\hat{\mathcal{L}}_t(h_{t-1}) - \hat{\mathcal{L}}_t(h^*)\right) \leq 2L \left\|h_{t-1} - h^*\right\|.$$

Recall that the hypothesis space is embedded in a Banach space equipped with the norm $\|\cdot\|$. Thus we have $\mathbb{E}\left[\!\left[\xi_t^2 \middle| Z^{t-1}\right]\!\right] \leq 4L^2 \left\|h_{t-1} - h^*\right\|^2$. Now using σ-strong convexity of the loss function we have

$$\frac{\mathcal{L}(h_{t-1}) + \mathcal{L}(h^*)}{2} \geq \mathcal{L}\left(\frac{h_{t-1} + h^*}{2}\right) + \frac{\sigma}{8}\left\|h_{t-1} - h^*\right\|^2 \geq \mathcal{L}(h^*) + \frac{\sigma}{8}\left\|h_{t-1} - h^*\right\|^2.$$

Let $\sigma_t^2 := \frac{16L^2}{\sigma}\left(\mathcal{L}(h_{t-1}) - \mathcal{L}(h^*)\right)$. Combining the two inequalities we get $\mathbb{E}\left[\!\left[\xi_t^2 \middle| Z^{t-1}\right]\!\right] \leq \sigma_t^2$. □

We note that although Kakade and Tewari state their result with a requirement that the loss function be strongly convex in a point wise manner, i.e., for all $\mathbf{z}, \mathbf{z}' \in \mathbb{Z}$, the function $\ell(h, \mathbf{z}, \mathbf{z}')$ be strongly convex in h, they only require the result in expectation. More specifically, our notion of strong convexity where we require the population risk functional $\mathcal{L}(h)$ to be strongly convex actually suits the proof of Kakade and Tewari as well.

We now use a Bernstein type inequality for martingales proved in Kakade and Tewari. The proof is based on a fundamental result on martingale convergence due to Freedman (1975).

Theorem 6.18. *Given a martingale difference sequence $X_t, t = 1 \ldots n$ that is uniformly B-bounded and has conditional variance $\mathbb{E}\left[\!\left[X_t^2 | X_1, \ldots, X_{t-1}\right]\!\right] \leq \sigma_t^2$, we have for any $\delta < 1/e$ and $n \geq 3$, with probability at least $1 - \delta$,*

$$\sum_{t=1}^{n} X_t \leq \max\left\{2\sigma^*, 3B\sqrt{\log\frac{4\log n}{\delta}}\right\}\sqrt{\log\frac{4\log n}{\delta}},$$

where $\sigma^* = \sqrt{\sum_{t=1}^{n} \sigma_t^2}$.

Let $\mathfrak{D}_n = \sum_{t=2}^{n} (\mathcal{L}(h_{t-1}) - \mathcal{L}(h^*))$. Then we can write the variance bound as

$$\sigma^* = \sqrt{\sum_{t=1}^{n} \sigma_t^2} = \sqrt{\sum_{t=1}^{n} \frac{16L^2}{\sigma} (\mathcal{L}(h_{t-1}) - \mathcal{L}(h^*))} = 4L\sqrt{\frac{\mathfrak{D}_n}{\sigma}}.$$

Thus, with probability at least $1 - \delta$, we have

$$\sum_{t=1}^{n} \xi_t \leq \max\left\{8L\sqrt{\frac{\mathfrak{D}_n}{\sigma}}, 6B\sqrt{\log\frac{4\log n}{\delta}}\right\}\sqrt{\log\frac{4\log n}{\delta}}.$$

Denoting $\Delta = \sqrt{\log\frac{4\log n}{\delta}}$ for notational simplicity and using the above bound in the online to batch conversion bound gives us

$$\frac{\mathfrak{D}_n}{n-1} \leq \frac{1+\varepsilon}{n-1}\left(\mathfrak{R}_n + \max\left\{8L\sqrt{\frac{\mathfrak{D}_n}{\sigma}}, 6B\Delta\right\}\Delta\right) + \frac{C_{(\delta/n)}\log n}{\varepsilon\sigma(n-1)}.$$

Solving this quadratic inequality is simplified by a useful result given in (Kakade and Tewari, 2008, Lemma 4)

Lemma 6.19. *For any* $s, r, d, b, \Delta > 0$ *such that*

$$s \leq r + \max\left\{4\sqrt{ds}, 6b\Delta\right\}\Delta,$$

we also have

$$s \leq r + 4\sqrt{dr}\Delta + \max\{16d, 6b\}\Delta^2.$$

Using this result gives us a rather nasty looking expression which we simplify by absorbing constants inside the $\mathcal{O}(\cdot)$ notation. We also make a simplifying ad-hoc assumption that we shall only set $\varepsilon \in (0, 1]$. The resulting expression is given below:

$$\mathfrak{D}_n \leq (1+\varepsilon)\,\mathfrak{R}_n + \mathcal{O}\left(\frac{C_d^2\log n\log(n/\delta)}{\varepsilon} + \log\frac{\log n}{\delta} + \sqrt{\left(\mathfrak{R}_n + \frac{C_d^2\log n\log(n/\delta)}{\varepsilon}\right)\log\frac{\log n}{\delta}}\right).$$

Let $\mathfrak{V}_n = \max\left\{\mathfrak{R}_n, 2C_d^2\log n\log(n/\delta)\right\}$. Concentrating only on the portion of the expression involving ε and ignoring the constants, we get

$$\begin{aligned}
&\varepsilon\mathfrak{R}_n + \frac{C_d^2\log n\log(n/\delta)}{\varepsilon} + \sqrt{\frac{C_d^2\log n\log(n/\delta)}{\varepsilon}\log\frac{\log n}{\delta}} \\
&\leq \varepsilon\mathfrak{R}_n + \frac{2C_d^2\log n\log(n/\delta)}{\varepsilon} \leq \varepsilon\mathfrak{V}_n + \frac{2C_d^2\log n\log(n/\delta)}{\varepsilon} \\
&\leq 2C_d\sqrt{2\mathfrak{V}_n\log n\log(n/\delta)},
\end{aligned}$$

where the second step follows since $\varepsilon \leq 1$ and the fourth step follows by using $\varepsilon =$

$\sqrt{\frac{2C_d^2 \log n \log(n/\delta)}{\mathfrak{V}_n}} \leq 1$. Putting this into the excess risk expression gives us

$$\frac{1}{n-1}\sum_{t=2}^{n} \mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{1}{n-1}\mathfrak{R}_n + C_d \cdot \mathcal{O}\left(\frac{\sqrt{\mathfrak{V}_n \log n \log(n/\delta)}}{n-1}\right)$$

which finishes the proof.

6.12.4 Proof of Lemma 6.7

We first decompose the excess risk term as before

$$\sum_{t=2}^{n} \mathcal{L}(h_{t-1}) - \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) = \sum_{t=2}^{n} \underbrace{\mathcal{L}(h_{t-1}) - \tilde{\mathcal{L}}_t^{\text{buf}}(h_{t-1})}_{P_t} + \underbrace{\tilde{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) - \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1})}_{Q_t}.$$

By construction, the sequence Q_t forms a martingale difference sequence, i.e., $\mathop{\mathbb{E}}_{\mathbf{z}_t}\left[\!\left[Q_t \middle| Z^{t-1} \right]\!\right] = 0$ and hence by an application of Azuma Hoeffding inequality we have

$$\frac{1}{n-1}\sum_{t=2}^{n} Q_t \leq B\sqrt{\frac{2\log\frac{1}{\delta}}{n-1}}. \tag{6.7}$$

We now analyze each term P_t individually. To simplify the analysis a bit we assume that the buffer update policy keeps admitting points into the buffer as long as there is space so that for $t \leq s+1$, the buffer contains an exact copy of the preceding stream. This is a very natural assumption satisfied by FIFO as well as reservoir sampling. We stress that our analysis works even without this assumption but requires a bit more work. In case we do make this assumption, the analysis of Lemma 6.1 applies directly and we have, for any $t \leq s+1$, with probability at least $1-\delta$,

$$P_t \leq \mathcal{R}_{t-1}(\ell \circ \mathcal{H}) + B\sqrt{\frac{\log\frac{1}{\delta}}{2(t-1)}}$$

For $t > s+1$, for an independent ghost sample $\{\tilde{\mathbf{w}}_1, \ldots, \tilde{\mathbf{w}}_{t-1}\}$ we have,

$$\begin{aligned}
\mathop{\mathbb{E}}_{\tilde{W}^{t-1}}\left[\!\left[\tilde{\mathcal{L}}_t^{\text{buf}} \right]\!\right] &= \mathop{\mathbb{E}}_{\tilde{W}^{t-1}}\left[\!\left[\frac{1}{s}\sum_{\tilde{\mathbf{z}}\in\tilde{B}_t} \mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1}, \mathbf{z}, \tilde{\mathbf{z}}) \right]\!\right] \right]\!\right] \\
&= \mathop{\mathbb{E}}_{\tilde{R}^{t-1}}\left[\!\left[\mathop{\mathbb{E}}_{\tilde{Z}^{t-1}}\left[\!\left[\frac{1}{s}\sum_{\tilde{\mathbf{z}}\in\tilde{B}_t} \mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1}, \mathbf{z}, \tilde{\mathbf{z}}) \right]\!\right] \middle| \tilde{R}^{t-1} \right]\!\right] \right]\!\right].
\end{aligned}$$

The conditioning performed above is made possible by stream obliviousness. Now suppose that given $\tilde{R}^{t-1}$ the indices $\tilde{\tau}_1, \ldots, \tilde{\tau}_s$ are present in the buffer $\tilde{B}_t$ at time t. Recall that this choice of indices is independent of $\tilde{Z}^{t-1}$ because of stream obliviousness. Then we can

write the above as

$$\begin{aligned}\mathop{\mathbb{E}}_{\tilde{R}^{t-1}}\left[\!\!\left[\mathop{\mathbb{E}}_{\tilde{Z}^{t-1}}\left[\!\!\left[\frac{1}{s}\sum_{\tilde{\mathbf{z}}\in\tilde{B}_t}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1},\mathbf{z},\tilde{\mathbf{z}})\right]\!\right]\,\middle|\,\tilde{R}^{t-1}\right]\!\!\right]\right] &= \mathop{\mathbb{E}}_{\tilde{R}^{t-1}}\left[\!\!\left[\mathop{\mathbb{E}}_{\tilde{Z}^{t-1}}\left[\!\!\left[\frac{1}{s}\sum_{j=1}^{s}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1},\mathbf{z},\tilde{\mathbf{z}}_{\tilde{\tau}_j})\right]\!\right]\right]\!\!\right]\right] \\ &= \mathop{\mathbb{E}}_{\tilde{R}^{t-1}}\left[\!\!\left[\mathop{\mathbb{E}}_{\tilde{\mathbf{z}}_1,\ldots,\tilde{\mathbf{z}}_s}\left[\!\!\left[\frac{1}{s}\sum_{j=1}^{s}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1},\mathbf{z},\tilde{\mathbf{z}}_j)\right]\!\right]\right]\!\!\right]\right] \\ &= \mathop{\mathbb{E}}_{\tilde{R}^{t-1}}\left[\!\left[\mathcal{L}(h_{t-1})\right]\!\right] = \mathcal{L}(h_{t-1}).\end{aligned}$$

We thus have

$$\mathop{\mathbb{E}}_{\tilde{W}^{t-1}}\left[\!\!\left[\frac{1}{s}\sum_{\tilde{\mathbf{z}}\in\tilde{B}_t}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1},\mathbf{z},\tilde{\mathbf{z}})\right]\!\right]\right]\!\!\right] = \mathcal{L}(h_{t-1}). \tag{6.8}$$

We now upper bound P_t as

$$\begin{aligned}P_t &= \mathcal{L}(h_{t-1}) - \tilde{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) \\ &= \mathop{\mathbb{E}}_{\tilde{W}^{t-1}}\left[\!\!\left[\frac{1}{s}\sum_{\tilde{\mathbf{z}}\in\tilde{B}_t}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1},\mathbf{z},\tilde{\mathbf{z}})\right]\!\right]\right]\!\!\right] - \tilde{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) \\ &\leq \underbrace{\sup_{h\in\mathcal{H}}\left[\mathop{\mathbb{E}}_{\tilde{W}^{t-1}}\left[\!\!\left[\frac{1}{s}\sum_{\tilde{\mathbf{z}}\in\tilde{B}_t}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h,\mathbf{z},\tilde{\mathbf{z}})\right]\!\right]\right]\!\!\right] - \tilde{\mathcal{L}}_t^{\text{buf}}(h)\right]}_{g_t(\mathbf{w}_1,\ldots,\mathbf{w}_{t-1})}.\end{aligned}$$

Now it turns out that applying McDiarmid's inequality to $g_t(\mathbf{w}_1,\ldots,\mathbf{w}_{t-1})$ directly would yield a very loose bound. This is because of the following reason: since $\hat{\mathcal{L}}_t^{\text{buf}}(h) = \frac{1}{|B_t|}\sum_{\mathbf{z}\in B_t}\ell(h,\mathbf{z}_t,\mathbf{z})$ depends only on s data points, changing any one of the $(t-1)$ variables $\mathbf{w}_i$ brings about a perturbation in g_t of magnitude at most $\mathcal{O}(1/s)$. The problem is that g_t is a function of $(t-1)\gg s$ variables and hence a direct application of McDiarmid's inequality would yield an excess error term of $\sqrt{\frac{t\log(1/\delta)}{s^2}}$ which would in the end require $s=\omega(\sqrt{n})$ to give any non trivial generalization bounds. In contrast, we wish to give results that would give non trivial bounds for $s=\tilde{\omega}(1)$.

In order to get around this problem, we need to reduce the number of variables in the statistic while applying McDiarmid's inequality. Fortunately, we observe that g_t *effectively* depends only on s variables, the data points that end up in the buffer at time t. This allows us to do the following. For any R^{t-1}, define

$$\delta(R^{t-1}) := \mathop{\mathbb{P}}_{Z^{t-1}}\left[\,g_t(\mathbf{w}_1,\ldots,\mathbf{w}_{t-1}) > \varepsilon \,|\, R^{t-1}\right].$$

We will first bound $\delta(R^{t-1})$. This will allow us to show

$$\mathop{\mathbb{P}}_{W^{t-1}}\left[g_t(\mathbf{w}_1,\ldots,\mathbf{w}_{t-1}) > \varepsilon\right] \leq \mathop{\mathbb{E}}_{R^{t-1}}\left[\!\left[\delta(R^{t-1})\right]\!\right],$$

where we take expectation over the distribution on R^{t-1} induced by the buffer update policy. Note that since we are oblivious to the nature of the distribution over R^{t-1}, our proof works for any stream oblivious buffer update policy. Suppose that given R^{t-1} the indices $\tau_1, \ldots, \tau_s$ are present in the buffer B_t at time t. Then we have

$$\begin{aligned} g_t(\mathbf{w}_1, \ldots, \mathbf{w}_{t-1}; R^{t-1}) &= \sup_{h \in \mathcal{H}} \left[\mathop{\mathbb{E}}_{\tilde{W}^{t-1}} \left[\left[\frac{1}{s} \sum_{\tilde{\mathbf{z}} \in \tilde{B}_t} \mathop{\mathbb{E}}_{\mathbf{z}} \left[\ell(h, \mathbf{z}, \tilde{\mathbf{z}}) \right] \right] - \frac{1}{s} \sum_{j=1}^{s} \mathop{\mathbb{E}}_{\mathbf{z}} \left[\ell(h, \mathbf{z}, \mathbf{z}_{\tau_j}) \right] \right] \right] \\ &=: \tilde{g}_t(\mathbf{z}_{\tau_1}, \ldots, \mathbf{z}_{\tau_s}). \end{aligned}$$

The function $\tilde{g}_t$ can be perturbed at most B/s due to a change in one of $\mathbf{z}_{\tau_j}$. Applying McDiarmid's inequality to the function $\tilde{g}_t$ we get with probability at least $1 - \delta$,

$$\tilde{g}_t(\mathbf{z}_{\tau_1}, \ldots, \mathbf{z}_{\tau_s}) \leq \mathop{\mathbb{E}}_{Z^{t-1}} \left[\tilde{g}_t(\mathbf{z}_{\tau_1}, \ldots, \mathbf{z}_{\tau_s}) \right] + B \sqrt{\frac{\log \frac{1}{\delta}}{2s}}$$

We analyze $\mathop{\mathbb{E}}_{Z^{t-1}} \left[\tilde{g}_t(\mathbf{z}_{\tau_1}, \ldots, \mathbf{z}_{\tau_s}) \right]$ below. In the third step in the calculations we symmetrize the true random variable $\mathbf{z}_{\tau_j}$ with the ghost random variable $\tilde{\mathbf{z}}_{\tilde{\tau}_j}$. This is contrasted with traditional symmetrization where we would symmetrize $\mathbf{z}_i$ with $\tilde{\mathbf{z}}_i$. In our case, we let the buffer construction dictate the matching at the symmetrization step.

$$\begin{aligned} & \mathop{\mathbb{E}}_{Z^{t-1}} \left[\tilde{g}_t(\mathbf{z}_{\tau_1}, \ldots, \mathbf{z}_{\tau_s}) \right] \\ = & \mathop{\mathbb{E}}_{Z^{t-1}} \left[\left[\sup_{h \in \mathcal{H}} \left[\mathop{\mathbb{E}}_{\tilde{W}^{t-1}} \left[\left[\frac{1}{s} \sum_{\tilde{\mathbf{z}} \in \tilde{B}_t} \mathop{\mathbb{E}}_{\mathbf{z}} \left[\ell(h, \mathbf{z}, \tilde{\mathbf{z}}) \right] \right] - \frac{1}{s} \sum_{j=1}^{s} \mathop{\mathbb{E}}_{\mathbf{z}} \left[\ell(h, \mathbf{z}, \mathbf{z}_{\tau_j}) \right] \right] \right] \right] \\ \leq & \mathop{\mathbb{E}}_{\tilde{R}^{t-1}} \left[\left[\mathop{\mathbb{E}}_{Z^{t-1}, \tilde{Z}^{t-1}} \left[\left[\sup_{h \in \mathcal{H}} \left[\frac{1}{s} \sum_{j=1}^{s} \mathop{\mathbb{E}}_{\mathbf{z}} \left[\ell(h, \mathbf{z}, \tilde{\mathbf{z}}_{\tilde{\tau}_j}) \right] - \frac{1}{s} \sum_{j=1}^{s} \mathop{\mathbb{E}}_{\mathbf{z}} \left[\ell(h, \mathbf{z}, \mathbf{z}_{\tau_j}) \right] \right] \right] \right] \middle| \tilde{R}^{t-1} \right] \right] \\ = & \mathop{\mathbb{E}}_{\tilde{R}^{t-1}} \left[\left[\mathop{\mathbb{E}}_{Z^{t-1}, \tilde{Z}^{t-1}, \varepsilon_j} \left[\left[\sup_{h \in \mathcal{H}} \left[\frac{1}{s} \sum_{j=1}^{s} \varepsilon_j \left(\mathop{\mathbb{E}}_{\mathbf{z}} \left[\ell(h, \mathbf{z}, \tilde{\mathbf{z}}_{\tilde{\tau}_j}) \right] - \mathop{\mathbb{E}}_{\mathbf{z}} \left[\ell(h, \mathbf{z}, \mathbf{z}_{\tau_j}) \right] \right) \right] \right] \right] \middle| \tilde{R}^{t-1} \right] \right] \\ \leq & 2 \mathop{\mathbb{E}}_{\tilde{R}^{t-1}} \left[\left[\mathop{\mathbb{E}}_{Z^{t-1}, \varepsilon_j} \left[\left[\sup_{h \in \mathcal{H}} \left[\frac{1}{s} \sum_{j=1}^{s} \varepsilon_j \mathop{\mathbb{E}}_{\mathbf{z}} \left[\ell(h, \mathbf{z}, \mathbf{z}_{\tau_j}) \right] \right] \right] \right] \middle| \tilde{R}^{t-1} \right] \right] \\ \leq & 2 \mathop{\mathbb{E}}_{\tilde{R}^{t-1}} \left[\mathcal{R}_s(\ell \circ \mathcal{H}) \right] \leq 2 \mathcal{R}_s(\ell \circ \mathcal{H}). \end{aligned}$$

Thus we get, with probability at least $1 - \delta$ over $\mathbf{z}_1, \ldots, \mathbf{z}_{t-1}$,

$$g_t(\mathbf{w}_1, \ldots, \mathbf{w}_{t-1}; R^{t-1}) \leq 2 \mathcal{R}_s(\ell \circ \mathcal{H}) + B \sqrt{\frac{\log \frac{1}{\delta}}{2s}}$$

which in turn, upon taking expectations with respect to R^{t-1}, gives us with probability

at least $1-\delta$ over $\mathbf{w}_1, \ldots, \mathbf{w}_{t-1}$,

$$P_t = g_t(\mathbf{w}_1, \ldots, \mathbf{w}_{t-1}) \leq 2\mathcal{R}_s(\ell \circ \mathcal{H}) + B\sqrt{\frac{\log \frac{1}{\delta}}{2s}}.$$

Applying a union bound on the bounds for $P_t, t = 2, \ldots, n$ gives us with probability at least $1-\delta$,

$$\frac{1}{n-1}\sum_{t=2}^{n} P_t \leq \frac{2}{n-1}\sum_{t=2}^{n} \mathcal{R}_{\min\{t-1,s\}}(\ell \circ \mathcal{H}) + B\sqrt{\frac{\log \frac{n}{\delta}}{2s}}. \tag{6.9}$$

Adding Equations (6.7) and (6.9) gives us the result.

6.12.5 Proof of Theorem 6.9

Our task here is to prove bounds on the following quantity

$$\frac{1}{n-1}\sum_{t=2}^{n} \mathcal{L}(h_{t-1}) - \mathcal{L}(h^*).$$

Proceeding as before, we will first prove the following result

$$\mathop{\mathbb{P}}_{Z^n}\left[\frac{1}{n-1}\sum_{t=2}^{n} \mathcal{L}(h_{t-1}) - \mathcal{L}(h^*) > \varepsilon \,\middle|\, R^n\right] \leq \delta. \tag{6.10}$$

This will allow us, upon taking expectations over R^n, show the following

$$\mathop{\mathbb{P}}_{W^n}\left[\frac{1}{n-1}\sum_{t=2}^{n} \mathcal{L}(h_{t-1}) - \mathcal{L}(h^*) > \varepsilon\right] \leq \delta,$$

which shall complete the proof.

In order to prove the statement given in Equation (6.10), we will use Theorem 6.4. As we did in the case of all-pairs loss functions, consider the loss function $\wp(h, \mathbf{z}') := \mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h, \mathbf{z}, \mathbf{z}')]\!]$ with $\mathcal{P}$ and $\hat{\mathcal{P}}$ as the associated population and empirical risk functionals. Clearly, if ℓ is L-Lipschitz and σ-strongly convex then so is $\wp$. By linearity of expectation, for any $h \in \mathcal{H}$, $\mathcal{P}(h) = \mathcal{L}(h)$. Suppose that given R^{t-1} the indices $\tau_1, \ldots, \tau_s$ are present in the buffer B_t at time t. Applying Theorem 6.4 on h_{t-1} at the t^{th} step with the loss function $\wp$ gives us that given R^{t-1}, with probability at least $1-\delta$ over the choice of Z^{t-1},

$$\mathcal{L}(h_{t-1}) - \mathcal{L}(h^*) \leq (1+\varepsilon)\left(\tilde{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) - \tilde{\mathcal{L}}_t^{\text{buf}}(h^*)\right) + \frac{C_\delta}{\varepsilon\sigma(\min\{s, t-1\})},$$

where we have again made the simplifying (yet optional) assumption that prior to time $t = s+1$, the buffer contains an exact copy of the stream. Summing across time steps and taking a union bound, gives us that given R^n, with probability at least $1-\delta$ over the

choice of Z^n,

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{C_{(\delta/n)}}{\varepsilon\sigma}\left(\frac{\log 2s}{n-1} + \frac{1}{s}\right) + \frac{1+\varepsilon}{n-1}\sum_{t=2}^{n}\tilde{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) - \tilde{\mathcal{L}}_t^{\text{buf}}(h^*).$$

Let us define as before

$$\xi_t := \left(\tilde{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) - \tilde{\mathcal{L}}_t^{\text{buf}}(h^*)\right) - \left(\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) - \hat{\mathcal{L}}_t^{\text{buf}}(h^*)\right).$$

Then using the regret bound $\mathfrak{R}_n^{\text{buf}}$ we can write,

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{1+\varepsilon}{n-1}\left(\mathfrak{R}_n^{\text{buf}} + \sum_{t=2}^{n}\xi_t\right) + \frac{C_{(\delta/n)}}{\varepsilon\sigma}\left(\frac{\log 2s}{n-1} + \frac{1}{s}\right).$$

Assuming $s < n/\log n$ simplifies the above expression to the following:

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{1+\varepsilon}{n-1}\left(\mathfrak{R}_n^{\text{buf}} + \sum_{t=2}^{n}\xi_t\right) + \frac{2C_{(\delta/n)}}{\varepsilon\sigma s}.$$

Note that this assumption is neither crucial to our proof nor very harsh since for $s = \Omega(n)$, we can always apply the results from the *infinite-buffer* setting using Theorem 6.5. Moving forward, by using the Bernstein-style inequality from Kakade and Tewari (2008), one can show with that probability at least $1-\delta$, we have

$$\sum_{t=1}^{n}\xi_t \leq \max\left\{8L\sqrt{\frac{\mathfrak{D}_n}{\sigma}}, 6B\sqrt{\log\frac{4\log n}{\delta}}\right\}\sqrt{\log\frac{4\log n}{\delta}},$$

where $\mathfrak{D}_n = \sum_{t=2}^{n}(\mathcal{L}(h_{t-1}) - \mathcal{L}(h^*))$. This gives us

$$\frac{\mathfrak{D}_n}{n-1} \leq \frac{1+\varepsilon}{n-1}\left(\mathfrak{R}_n^{\text{buf}} + \max\left\{8L\sqrt{\frac{\mathfrak{D}_n}{\sigma}}, 6B\Delta\right\}\Delta\right) + \frac{2C_{(\delta/n)}}{\varepsilon\sigma s}.$$

Using (Kakade and Tewari, 2008, Lemma 4) and absorbing constants inside the $\mathcal{O}(\cdot)$ notation we get:

$$\mathfrak{D}_n \leq (1+\varepsilon)\,\mathfrak{R}_n^{\text{buf}} + \mathcal{O}\left(\frac{C_d^2 n\log(n/\delta)}{\varepsilon s} + \log\frac{\log n}{\delta}\right) + \mathcal{O}\left(\sqrt{\left(\mathfrak{R}_n^{\text{buf}} + \frac{C_d^2 n\log(n/\delta)}{\varepsilon s}\right)\log\frac{\log n}{\delta}}\right).$$

Let $\mathfrak{W}_n = \max\left\{\mathfrak{R}_n^{\text{buf}}, \frac{2C_d^2 n\log(n/\delta)}{s}\right\}$. Concentrating only on the portion of the expression involving ε and ignoring the constants, we get

$$\begin{aligned}&\varepsilon\mathfrak{R}_n^{\text{buf}} + \frac{C_d^2 n\log(n/\delta)}{\varepsilon s} + \sqrt{\frac{C_d^2 n\log(n/\delta)}{\varepsilon s}\log\frac{\log n}{\delta}}\\ &\leq \varepsilon\mathfrak{R}_n^{\text{buf}} + \frac{2C_d^2 n\log(n/\delta)}{\varepsilon s} \leq \varepsilon\mathfrak{W}_n + \frac{2C_d^2 n\log(n/\delta)}{\varepsilon s}\end{aligned}$$

$$\leq 2C_d\sqrt{\frac{2\mathfrak{W}_n n\log(n/\delta)}{s}},$$

where the second step follows since $\varepsilon \leq 1$ and $s \leq n$ and the fourth step follows by using $\varepsilon = \sqrt{\frac{2C_d^2 n\log(n/\delta)}{\mathfrak{W}_n s}} \leq 1$ Putting this into the excess risk expression gives us

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{1}{n-1}\mathfrak{R}_n^{\text{buf}} + C_d\cdot\mathcal{O}\left(\sqrt{\frac{\mathfrak{W}_n\log(n/\delta)}{sn}}\right),$$

which finishes the proof. Note that in case $\mathfrak{W}_n = \mathfrak{R}_n^{\text{buf}}$, we get

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{1}{n-1}\mathfrak{R}_n^{\text{buf}} + C_d\cdot\mathcal{O}\left(\sqrt{\frac{\mathfrak{R}_n^{\text{buf}}\log(n/\delta)}{sn}}\right).$$

On the other hand if $\mathfrak{R}_n^{\text{buf}} \leq \frac{2C_d^2 n\log(n/\delta)}{s}$, we get

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{1}{n-1}\mathfrak{R}_n^{\text{buf}} + C_d^2\cdot\mathcal{O}\left(\frac{\log(n/\delta)}{s}\right).$$

6.12.6 Proof of Theorem 6.10

Recall that we are considering a composition classes of the form $\ell\circ\mathcal{H} := \{(\mathbf{z},\mathbf{z}') \mapsto \ell(h,\mathbf{z},\mathbf{z}'), h\in\mathcal{H}\}$ where ℓ is some Lipschitz loss function. We have $\ell(h,z_1,z_2) = \phi\left(h(x_1,x_2)Y(y_1,y_2)\right)$ where $Y(y_1,y_2) = y_1 - y_2$ or $Y(y_1,y_2) = y_1y_2$ and $\phi:\mathbb{R}\to\mathbb{R}$ involves some margin loss function. We also assume that ϕ is point wise L-Lipschitz. Let $Y = \sup_{y_1,y_2\in\mathcal{Y}}|Y(y_1,y_2)|$.

Let $\tilde{\phi}(x) = \phi(x) - \phi(0)$. Note that $\tilde{\phi}(\cdot)$ is point wise L-Lipschitz as well as satisfies $\tilde{\phi}(0) = 0$. Let $Y = \sup_{y,y'\in\mathcal{Y}}|Y(y,y')|$.

We will require the following contraction lemma that we state below.

Theorem 6.20 (Implicit in proof of Ledoux and Talagrand (2002), Theorem 4.12)**.** *Let $\mathcal{H}$ be a set of bounded real valued functions from some domain $\mathcal{X}$ and let $\mathbf{x}_1,\ldots,\mathbf{x}_n$ be arbitrary elements from $\mathcal{X}$. Furthermore, let $\phi_i:\mathbb{R}\to\mathbb{R}$, $i=1,\ldots,n$ be L-Lipschitz functions such that $\phi_i(0)=0$ for all i. Then we have*

$$\mathbb{E}\left[\!\!\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i\phi_i(h(\mathbf{x}_i))\right]\!\!\right] \leq L\mathbb{E}\left[\!\!\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i h(\mathbf{x}_i)\right]\!\!\right].$$

Using the above inequality we can state the following chain of (in)equalities:

$$\begin{aligned}\mathcal{R}_n(\ell\circ\mathcal{H}) &= \mathbb{E}\left[\!\!\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i\ell(h,\mathbf{z},\mathbf{z}_i)\right]\!\!\right]\\ &= \mathbb{E}\left[\!\!\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i\phi\left(h(\mathbf{x},\mathbf{x}_i)Y(y,y_i)\right)\right]\!\!\right]\end{aligned}$$

$$
\begin{aligned}
&= \mathbb{E}\left[\!\!\left[\sup_{h\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} \varepsilon_i \tilde{\phi}\left(h(\mathbf{x},\mathbf{x}_i)Y(y,y_i)\right)\right]\!\!\right] + \phi(0)\mathbb{E}\left[\!\!\left[\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i\right]\!\!\right] \\
&= \mathbb{E}\left[\!\!\left[\sup_{h\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} \varepsilon_i \tilde{\phi}\left(h(\mathbf{x},\mathbf{x}_i)Y(y,y_i)\right)\right]\!\!\right] \\
&\leq LY\mathbb{E}\left[\!\!\left[\sup_{h\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} \varepsilon_i h(\mathbf{x},\mathbf{x}_i)\right]\!\!\right] \\
&= LY\mathcal{R}_n(\mathcal{H}),
\end{aligned}
$$

where the fourth step follows from linearity of expectation. The fifth step is obtained by applying the contraction inequality to the functions $\psi_i : x \mapsto \tilde{\phi}(a_i x)$ where $a_i = Y(y, y_i)$. We exploit the fact that the contraction inequality is actually proven for the empirical Rademacher averages due to which we can take $a_i = Y(y, y_i)$ to be a constant dependent only on i, use the inequality, and subsequently take expectations. We also have, for any i and any $x, y \in \mathbb{R}$,

$$
\begin{aligned}
|\psi_i(x) - \psi_i(y)| &= \left|\tilde{\phi}(a_i x) - \tilde{\phi}(a_i y)\right| \\
&\leq L\left|a_i x - a_i y\right| \\
&\leq L\left|a_i\right|\left|x - y\right| \\
&\leq LY\left|x - y\right|,
\end{aligned}
$$

which shows that every function $\psi_i(\cdot)$ is LY-Lipschitz and satisfies $\psi_i(0) = 0$. This makes an application of the contraction inequality possible on the empirical Rademacher averages which upon taking expectations give us the result.

6.12.7 Proof of Theorem 6.12

To prove the results, let us assume that the buffer contents are addressed using the variables $\zeta_1, \ldots, \zeta_s$. We shall first concentrate on a fixed element, say ζ_1 (which we shall call simply ζ for notational convenience) of the buffer and inductively analyze the probability law $\mathcal{P}_t$ obeyed by ζ at each time step $t \geq s + 2$.

We will prove that the probability law obeyed by ζ at time t is $\mathcal{P}_t(\zeta) = \frac{1}{t-1}\sum_{\tau=1}^{t-1} \mathbb{1}_{\{\zeta=\mathbf{z}_\tau\}}$. The law is interpreted as saying the following: for any $\tau \leq t-1$, $\mathbb{P}\left[\zeta = \mathbf{z}_\tau\right] = \frac{1}{t-1}$ and shows that the element ζ is indeed a uniform sample from the set Z^{t-1}. We would similarly be able to show this for all locations $\zeta_2, \ldots, \zeta_s$ which would prove that the elements in the buffer are indeed identical samples from the preceding stream. Since at each step, the **RS-x** algorithm updates all buffer locations independently, the random variables $\zeta_1, \ldots, \zeta_s$ are independent as well which would allow us to conclude that at each step we have s i.i.d. samples in the buffer as claimed.

We now prove the probability law for ζ. We note that the repopulation step done at time $t = s + 1$ explicitly ensures that at step $t = s + 2$, the buffer contains s i.i.d samples from Z^{s+1} i.e. $\mathcal{P}_{s+2}(\zeta) = \frac{1}{s+1}\sum_{\tau=1}^{s+1} \mathbb{1}_{\{\zeta=\mathbf{z}_\tau\}}$. This forms the initialization of our

inductive argument. Now suppose that at the t^{th} time step, the claim is true and ζ obeys the law $\mathcal{P}_t(\zeta) = \frac{1}{t-1}\sum_{\tau=1}^{t-1} \mathbb{1}_{\{\zeta=\mathbf{z}_\tau\}}$. At the t^{th} step, we would update the buffer by making the incoming element $\mathbf{z}_t$ replace the element present at the location indexed by ζ with probability $1/(t+1)$. Hence ζ would obey the following law after the update

$$\left(1 - \frac{1}{t}\right)\mathcal{P}_t(\zeta) + \frac{1}{t}\mathbb{1}_{\{\zeta=\mathbf{z}_t\}} = \frac{1}{t}\sum_{\tau=1}^{t} \mathbb{1}_{\{\zeta=\mathbf{z}_\tau\}}$$

which shows that at the $(t+1)^{\text{th}}$ step, ζ would follow the law $\mathcal{P}_{t+1}(\zeta) = \frac{1}{t}\sum_{\tau=1}^{t} \mathbb{1}_{\{\zeta=\mathbf{z}_\tau\}}$ which completes the inductive argument and the proof.

6.12.8 Proof of Theorem 6.13

We now prove Theorem 6.13 that gives a high confidence regret bound for the **OLP** learning algorithm when used along with the **RS-x** buffer update policy. Our proof proceeds in two steps: in the first step we prove a uniform convergence type guarantee that would allow us to convert regret bounds with respect to the *finite-buffer* penalties $\hat{\mathcal{L}}_t^{\text{buf}}$ into regret bounds in in terms of the *all-pairs* loss functions $\hat{\mathcal{L}}_t$. In the second step we then prove a regret bound for **OLP** with respect to the *finite-buffer* penalties.

We proceed with the first step of the proof by proving the lemma given below. Recall that for any sequence of training examples $\mathbf{z}_1, \ldots, \mathbf{z}_n$, we define, for any $h \in \mathcal{H}$, the all-pairs loss function as $\hat{\mathcal{L}}_t(h) = \frac{1}{t-1}\sum_{\tau=1}^{t-1} \ell(h, \mathbf{z}_t, \mathbf{z}_\tau)$. Moreover, if the online learning process uses a buffer, the we also define the *finite-buffer* loss function as $\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) = \frac{1}{|B_t|}\sum_{\mathbf{z}\in B_t} \ell(h_{t-1}, \mathbf{z}_t, \mathbf{z})$.

Lemma 6.21. *Suppose we have an online learning algorithm that incurs buffer penalties based on a buffer B of size s that is updated using the **RS-x** algorithm. Suppose further that the learning algorithm generates an ensemble $h_1, \ldots, h_{n-1}$. Then for any $t \in [1, n-1]$, with probability at least $1-\delta$ over the choice of the random variables used to update the buffer B until time t, we have*

$$\hat{\mathcal{L}}_t(h_{t-1}) \leq \hat{\mathcal{L}}_t^{buf}(h_{t-1}) + C_d \cdot \mathcal{O}\left(\sqrt{\frac{\log\frac{1}{\delta}}{s}}\right)$$

Proof. Suppose $t \leq s+1$, then since at that point the buffer stores the stream exactly, we have

$$\hat{\mathcal{L}}_t(h_{t-1}) = \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1})$$

which proves the result. Note that, as Algorithm 8 indicates, at step $t = s+1$ the buffer is updated (using the repopulation step) only after the losses have been calculated and hence step $t = s+1$ still works with a buffer that stores the stream exactly.

We now analyze the case $t > s+1$. At each step $\tau > s$, the **RS-x** algorithm uses s independent Bernoulli random variables (which we call *auxiliary random variables*) to

update the buffer, call them $r_1^\tau, \ldots, r_s^\tau$ where r_j^τ is used to update the j^{th} item ζ_j in the buffer. Let $\mathfrak{r}_j^t := \{r_j^{s+1}, r_j^2, \ldots, r_j^t\} \in \{0,1\}^t$ denote an ensemble random variable composed of $t-s$ independent Bernoulli variables. It is easy to see that the element ζ_j is completely determined at the t^{th} step given $\mathfrak{r}_j^{t-1}$.

Theorem 6.12 shows, for any $t > s+1$, that the buffer contains s i.i.d. samples from the set Z^{t-1}. Thus, for any *fixed* function $h \in \mathcal{H}$, we have for any $j \in [s]$,

$$\mathop{\mathbb{E}}_{\mathfrak{r}_j^{t-1}} [\![\ell(h, \mathbf{z}_t, \zeta_j)]\!] = \frac{1}{t-1} \sum_{\tau=1}^{t-1} \ell(h, \mathbf{z}_t, \mathbf{z}_\tau)$$

which in turn shows us that

$$\mathop{\mathbb{E}}_{\mathfrak{r}_1^{t-1}, \ldots, \mathfrak{r}_s^{t-1}} \left[\!\!\left[\hat{\mathcal{L}}_t^{\text{buf}}(h) \right]\!\!\right] = \frac{1}{t-1} \sum_{\tau=1}^{t-1} \ell(h, \mathbf{z}_t, \mathbf{z}_\tau) = \hat{\mathcal{L}}_t(h)$$

Now consider a ghost sample of auxiliary random variables $\tilde{\mathfrak{r}}_1^{t-1}, \ldots, \tilde{\mathfrak{r}}_s^{t-1}$. Since our hypothesis h_{t-1} is independent of these ghost variables, we can write

$$\mathop{\mathbb{E}}_{\tilde{\mathfrak{r}}_1^{t-1}, \ldots, \tilde{\mathfrak{r}}_s^{t-1}} \left[\!\!\left[\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) \right]\!\!\right] = \hat{\mathcal{L}}_t(h_{t-1})$$

We recall that error in the proof presented in Zhao et al. (2011) was to apply such a result on the *true* auxiliary variables upon which h_{t-1} is indeed dependent. Thus we have

$$\begin{aligned} \hat{\mathcal{L}}_t(h_{t-1}) - \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) &= \mathop{\mathbb{E}}_{\tilde{\mathfrak{r}}_1^{t-1}, \ldots, \tilde{\mathfrak{r}}_s^{t-1}} \left[\!\!\left[\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) \right]\!\!\right] - \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) \\ &\leq \underbrace{\sup_{h \in \mathcal{H}} \left[\mathop{\mathbb{E}}_{\tilde{\mathfrak{r}}_1^{t-1}, \ldots, \tilde{\mathfrak{r}}_s^{t-1}} \left[\!\!\left[\hat{\mathcal{L}}_t^{\text{buf}}(h) \right]\!\!\right] - \hat{\mathcal{L}}_t^{\text{buf}}(h) \right]}_{g_t(\mathfrak{r}_1^{t-1}, \ldots, \mathfrak{r}_s^{t-1})} \end{aligned}$$

Now, the perturbation to any of the ensemble variables $\mathfrak{r}_j$ (a perturbation to an ensemble variable implies a perturbation to one or more variables forming that ensemble) can only perturb only the element ζ_j in the buffer. Since $\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) = \frac{1}{s} \sum_{\mathbf{z} \in B_t} \ell(h_{t-1}, \mathbf{z}_t, \mathbf{z})$ and the loss function is B-bounded, this implies that a perturbation to any of the ensemble variables can only perturb $g(\mathfrak{r}_1^{t-1}, \ldots, \mathfrak{r}_s^{t-1})$ by at most B/s. Hence an application of McDiarmid's inequality gives us, with probability at least $1-\delta$,

$$g_t(\mathfrak{r}_1^{t-1}, \ldots, \mathfrak{r}_s^{t-1}) \leq \mathop{\mathbb{E}}_{\mathfrak{r}_j^{t-1}} [\![g_t(\mathfrak{r}_1^{t-1}, \ldots, \mathfrak{r}_s^{t-1})]\!] + B\sqrt{\frac{\log \frac{1}{\delta}}{2s}}$$

Analyzing the expectation term we get

$$\mathop{\mathbb{E}}_{\mathfrak{r}_j^{t-1}} [\![g_t(\mathfrak{r}_1^{t-1}, \ldots, \mathfrak{r}_s^{t-1})]\!] = \mathop{\mathbb{E}}_{\mathfrak{r}_j^{t-1}} \left[\!\!\left[\sup_{h \in \mathcal{H}} \left[\mathop{\mathbb{E}}_{\tilde{\mathfrak{r}}_1^{t-1}, \ldots, \tilde{\mathfrak{r}}_s^{t-1}} \left[\!\!\left[\hat{\mathcal{L}}_t^{\text{buf}}(h) \right]\!\!\right] - \hat{\mathcal{L}}_t^{\text{buf}}(h) \right] \right]\!\!\right]$$

$$
\begin{aligned}
&\leq \mathop{\mathbb{E}}_{\mathfrak{r}_j^{t-1},\tilde{\mathfrak{r}}_j^{t-1}} \left[\!\!\left[\sup_{h\in\mathcal{H}} \left[\frac{1}{s}\sum_{j=1}^{s} \ell(h,\mathbf{z}_t,\tilde{\zeta}_j) - \ell(h,\mathbf{z}_t,\zeta_j) \right] \right]\!\!\right] \\
&= \mathop{\mathbb{E}}_{\mathfrak{r}_j^{t-1},\tilde{\mathfrak{r}}_j^{t-1},\varepsilon_j} \left[\!\!\left[\sup_{h\in\mathcal{H}} \left[\frac{1}{s}\sum_{j=1}^{s} \varepsilon_j \left(\ell(h,\mathbf{z}_t,\tilde{\zeta}_j) - \ell(h,\mathbf{z}_t,\zeta_j) \right) \right] \right]\!\!\right] \\
&\leq 2 \mathop{\mathbb{E}}_{\mathfrak{r}_j^{t-1},\tilde{\mathfrak{r}}_j^{t-1},\varepsilon_j} \left[\!\!\left[\sup_{h\in\mathcal{H}} \left[\frac{1}{s}\sum_{j=1}^{s} \varepsilon_j \ell(h,\mathbf{z}_t,\zeta_j) \right] \right]\!\!\right] \leq 2\mathcal{R}_s(\ell\circ\mathcal{H})
\end{aligned}
$$

where in the third step we have used the fact that symmetrizing a pair of true and ghost ensemble variables is equivalent to symmetrizing the buffer elements they determine. In the last step we have exploited the definition of Rademacher averages with the (empirical) measure $\frac{1}{t-1}\sum_{\tau=1}^{t-1}\delta_{\mathbf{z}_\tau}$ imposed over the domain $\mathbb{Z}$.

For hypothesis classes for which $\hat{\mathcal{R}}_s(\ell\circ\mathcal{H}) = C_d\cdot\mathcal{O}\left(\sqrt{\frac{1}{s}}\right)$, this proves the claim. □

Using a similar proof progression we can also show the following:

Lemma 6.22. *For any fixed $h\in\mathcal{H}$ and any $t\in[1,n-1]$, with probability at least $1-\delta$ over the choice of the random variables used to update the buffer B until time t, we have*

$$
\hat{\mathcal{L}}_t^{buf}(h) \leq \hat{\mathcal{L}}_t(h) + C_d\cdot\mathcal{O}\left(\sqrt{\frac{\log\frac{1}{\delta}}{s}}\right)
$$

Combining Lemmata 6.21 and 6.22 and taking a union bound over all time steps, the following corollary gives us a *buffer to all-pairs* conversion bound.

Lemma 6.23. *Suppose we have an online learning algorithm that incurs buffer penalties based on a buffer B of size s that is updated using the **RS-x** algorithm. Suppose further that the learning algorithm generates an ensemble $h_1,\ldots,h_{n-1}$. Then with probability at least $1-\delta$ over the choice of the random variables used to update the buffer B, we have*

$$
\mathfrak{R}_n \leq \mathfrak{R}_n^{buf} + C_d\,(n-1)\cdot\mathcal{O}\left(\sqrt{\frac{\log\frac{n}{\delta}}{s}}\right),
$$

where we recall the definition of the all-pairs *regret as*

$$
\mathfrak{R}_n := \sum_{t=2}^{n}\hat{\mathcal{L}}_t(h_{t-1}) - \inf_{h\in\mathcal{H}}\sum_{t=2}^{n}\hat{\mathcal{L}}_t(h)
$$

and the finite-buffer *regret as*

$$
\mathfrak{R}_n^{buf} := \sum_{t=2}^{n}\hat{\mathcal{L}}_t^{buf}(h_{t-1}) - \inf_{h\in\mathcal{H}}\sum_{t=2}^{n}\hat{\mathcal{L}}_t^{buf}(h).
$$

Proof. Let $\hat{h} := \arg\inf_{h\in\mathcal{H}} \sum_{t=2}^{n}\hat{\mathcal{L}}_t(h)$. Then Lemma 6.22 gives us, upon summing over t

and taking a union bound,

$$\sum_{t=2}^{n} \hat{\mathcal{L}}_t^{\text{buf}}(\hat{h}) \leq \sum_{t=2}^{n} \hat{\mathcal{L}}_t(\hat{h}) + C_d(n-1) \cdot \mathcal{O}\left(\sqrt{\frac{\log \frac{n}{\delta}}{s}}\right), \tag{6.11}$$

whereas Lemma 6.21 similarly guarantees

$$\sum_{t=2}^{n} \hat{\mathcal{L}}_t(h_{t-1}) \leq \sum_{t=2}^{n} \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) + C_d(n-1) \cdot \mathcal{O}\left(\sqrt{\frac{\log \frac{n}{\delta}}{s}}\right), \tag{6.12}$$

where both results hold with high confidence. Adding the Equations (6.11) and (6.12) and using $\sum_{t=2}^{n} \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) \leq \inf_{h \in \mathcal{H}} \sum_{t=2}^{n} \hat{\mathcal{L}}_t^{\text{buf}}(\hat{h}) + \mathfrak{R}_n^{\text{buf}}$ completes the proof. □

To finish the proof, we give a *finite-buffer* regret bound for the **OLP** algorithm.

Lemma 6.24. *Suppose the **OLP** algorithm working with an s-sized buffer generates an ensemble $\mathbf{w}_1, \ldots, \mathbf{w}_{n-1}$. Further suppose that the loss function ℓ being used is L-Lipschitz and the space of hypotheses $\mathcal{W}$ is a compact subset of a Banach space with a finite diameter D with respect to the Banach space norm. Then we have*

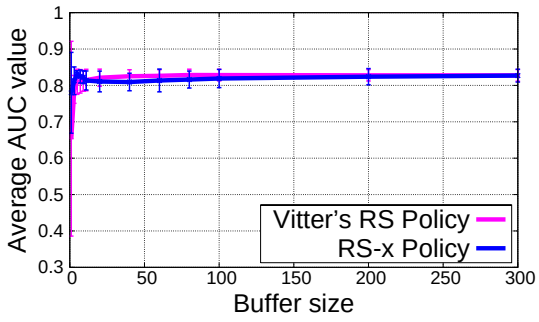
$$\mathfrak{R}_n^{buf} \leq LD\sqrt{n-1}$$

Proof. We observe that the algorithm **OLP** is simply a variant of the GIGA algorithm Zinkevich (2003) being applied with the loss functions $\ell_t^{\text{GIGA}} : \mathbf{w} \mapsto \hat{\mathcal{L}}_t^{\text{buf}}(\mathbf{w})$. Since ℓ_t^{GIGA} inherits the Lipschitz constant of $\hat{\mathcal{L}}_t^{\text{buf}}$ which in turn inherits it from ℓ, we can use the analysis given by Zinkevich (2003) to conclude the proof. □

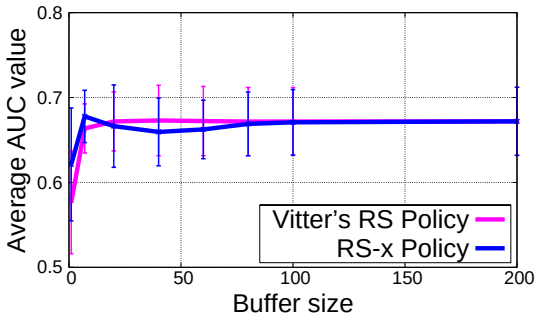
Combining Lemmata 6.23 and 6.24 completes the proof of Theorem 6.13

6.13 Additional Experimental Results

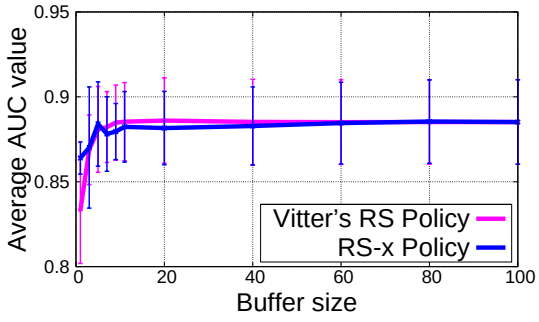
Here we present experimental results on 14 different benchmark datasets (refer to Figure 6.2) comparing the **OLP** algorithm using the **RS-x^2** buffer policy with the OAM$_{\text{gra}}$ algorithm using the **RS** buffer policy. We continue to observe the trend that **OLP** performs competitively to OAM$_{\text{gra}}$ while enjoying a slight advantage in small buffer situations in most cases.



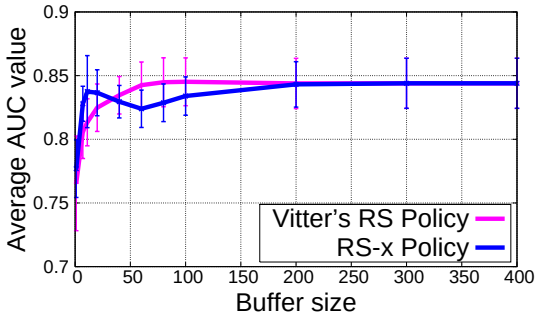
(a) Fourclass



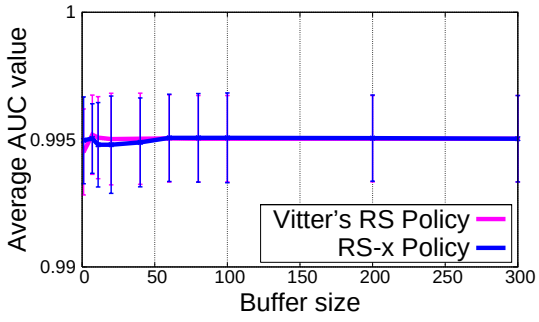
(b) Liver Disorders



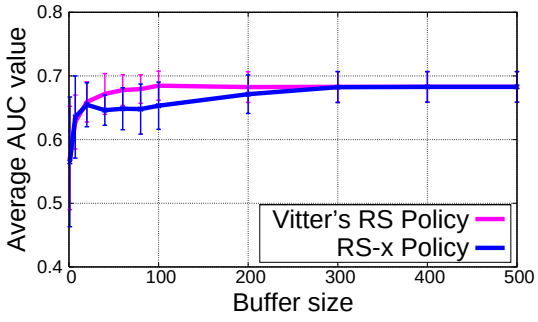
(c) Heart



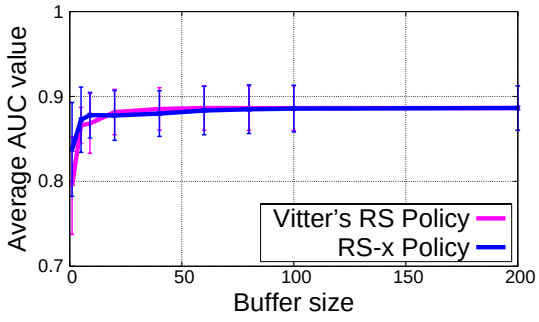
(d) Diabetes



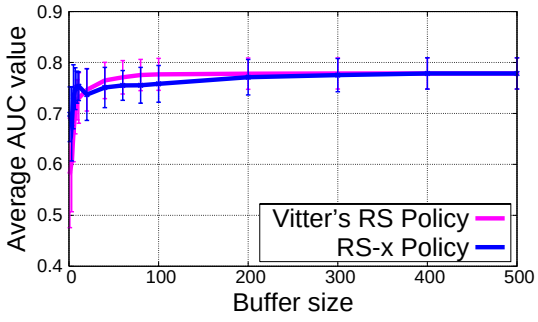
(e) Breast Cancer



(f) Vowel

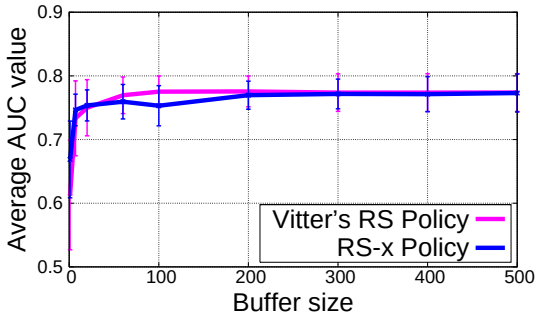


(g) Ionosphere

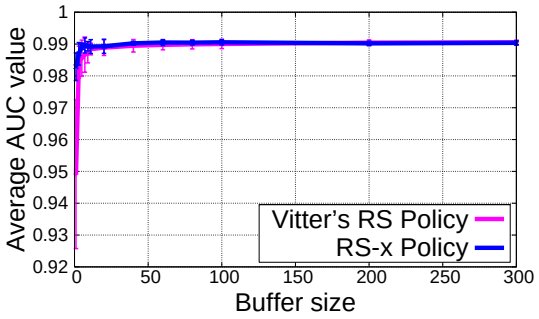


(h) German

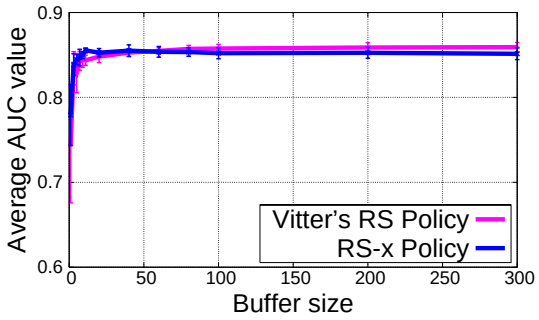
Figure 6.2: Comparison between OAM_{gra} (using **RS** policy) and **OLP** (using **RS-x** policy) on AUC maximization tasks - Part I.



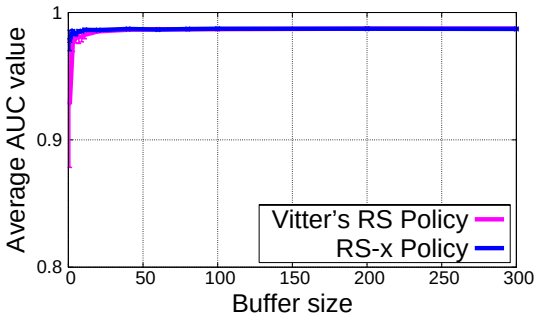
(a) SVMguide3



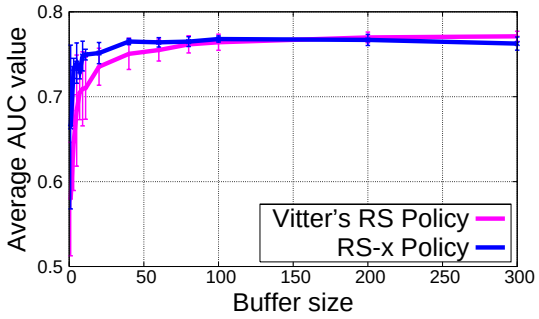
(b) SVMguide1



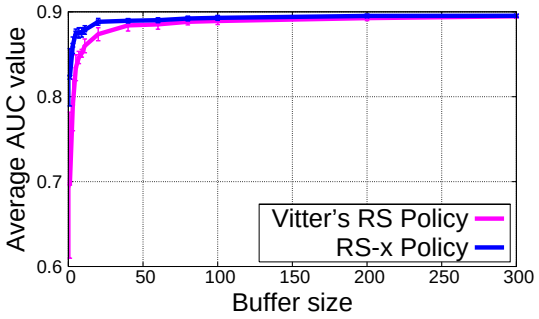
(c) Statlog



(d) Cod RNA



(e) Letter



(f) Adult

Figure 6.3: Comparison between OAM_{gra} (using **RS** policy) and **OLP** (using **RS-x** policy) on AUC maximization tasks - Part II.

Chapter 7

Conclusion and Future Work

Contents

We addressed several problems as a part of this thesis, details of which were presented in the preceding chapters. The purpose of this chapter is to assess the contributions and look for avenues of future research. This we will do in three parts corresponding to the three main problem areas addresses by the thesis (refer to Chapter 2).

7.1 Accelerated Kernel Learning

Chapter 3 addressed the problem of speeding up the training and prediction routines of kernel classifiers by way of constructing random features for dot product kernels. The chapter presented crisp theoretical guarantees for the approach as well as empirically demonstrated its utility on some benchmark data sets.

There has been continuing work in this area of random features; of particular interest is the work of Pham and Pagh (2013) which extends the work to give faster random feature constructions for the special case of polynomial kernels $K(\mathbf{x}, \mathbf{y}) = (\langle \mathbf{x}, \mathbf{y} \rangle + c)^p$. Their work makes innovative use of sketching techniques prevalent in the data streaming community to get a near quadratic reduction in feature construction time for these kernels. This seems to be very encouraging although it remains to be seen if other kernels in the dot product family can also admit such *fast* feature constructions. Moreover, our method seems to be preferable for non-homogeneous kernels ($c > 0$) which also opens up possibilities for improvements both ways.

Of particular interest are methods that bypass the kernel approximation step and directly aim to approximate the decision boundary offered by a particular kernel (for example the work of Jose et al. (2013)). These methods have an advantage over random feature based techniques in that they are data and label aware whereas random feature methods are completely oblivious to data as well as labels (which albeit offers them an indirect advantage of being reusable for different tasks and data sets). It would be very interesting to develop data-aware methods that offer theoretical guarantees such as random features. Nyström methods come close as in they take into account the distribution, but they are still task (i.e. label) oblivious.

The issue of speeding up training and prediction routines can also be raised for indefinite kernel learning but we shall address that in the next section.

7.2 Indefinite Kernel Learning

In Chapters 4 and 5, we addressed the problem of learning with indefinite kernels by presenting a model of learning, building up on earlier work of Balcan and Blum (2006), that is capable of utilizing indefinite kernels in the learning process. We showed that our models offered fast training and prediction routines, as well as crisp generalization bounds. For the case of real valued regression, we were able to extend the model to one that admitted a provable support vector effect, something we later confirmed experimentally as well.

These models present several interesting questions - first of all, is it possible to extend the models, and develop corresponding algorithms for other learning problems such as classification and ranking, so as to have the support vector effect ? This would be interesting from both a theoretical, as well as a practical point of view. This might involve novel applications of sparse learning techniques in the learning process and would, in some sense, unify kernel based learning as a paradigm without prejudice to a certain class of kernels.

The second most important problem is that of the choice of the kernel, something that the Mercer kernel learning community has also been interested for some time now (see (Varma and Babu, 2009) for references). There have been very few explorations into this problem for indefinite kernels (for example (Bellet et al., 2012)) and given our learning models that hinge on "good" similarity functions, the problem becomes equally interesting for indefinite kernels as well. We shall look more into this question in the next section.

Thirdly, there are other learning problems such as clustering which have not received much attention from the indefinite kernel learning community (except perhaps the work of Balcan et al. (2008b) which is very theoretical). It would be interesting to take these problems up for further investigation.

7.3 The Kernel Choice Problem

In Chapter 6, we addressed the problem of learning pairwise kernels in an online fashion. Our model supported learning with arbitrary pairwise loss functions that covered kernel learning without prejudice to the type of the kernel and addressed problems such as Mercer kernel learning, indefinite kernel learning, metric learning, bipartite ranking etc. We provided crisp online-to-batch conversion bounds that offered optimal convergence rates for strongly convex loss functions. We also proposed an online learning based algorithm based on the classical GIGA algorithm and a novel stream sampling algorithm (**RS-x** and **RS-x^2**) that offered sublinear regret with (poly-)logarithmic space usage.

Online techniques are particularly attractive due to their resource efficiency and have even been used to build solvers for batch problems (for example (Shalev-Shwartz et al., 2011)). The same would be very desirable to solve kernel learning problems as well. However, for that to happen, some more work is required in the online learning setup. For instance, the current regret bounds given by the **OLP** algorithm scale as $\mathcal{O}\left(\sqrt{\frac{\log n}{s}}\right)$. It is an open problem to see if this can be improved to $\frac{1}{n^{\Omega(1)}}$ while still using $s = \log^{\mathcal{O}(1)} n$ sized buffer or else to show that there exists a regret lower bound preventing so.

Doing so might involve, among other things, coming up with better buffer management strategies. In particular, stream-aware buffer policies are expected to yield better regret as well as generalization bounds. It would be an interesting problem to devise and analyze such policies.

Appendices

Appendix A

Extending the models of Balcan and Blum, and Wang *et al*

Contents

Abstract *This appendix is with reference to Chapter 4. Here we formally show that our proposed model for classification with indefinite kernels is a generalization of previous models presented in Balcan and Blum (2006) and Wang et al. (2007). We split the discussion into two sections each of which discusses the model presented in one of the previous works.*

A.1 Extending the distance-based model of Wang et al.

Wang et al. (2007) consider a model of learning with distance functions. Their model is similar to ours but for the difference that they restrict themselves to the use of a single transfer function namely the sign function $f = \text{sgn}()$. More formally, they have the following notion of a *good distance function.*

Definition A.1 ((Wang et al., 2007) Definition 4)**.** *A distance function $\mathcal{X}$, $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is said to be an (ε, γ, B)-good distance for a classification problem where $\varepsilon, \gamma, B > 0$ if there exist two class conditional probability distributions $\tilde{\mathcal{D}}(x|\ell(x) = 1)$ and $\tilde{\mathcal{D}}(x|\ell(x) = -1)$ such that for all $x \in \mathcal{X}$, $\frac{\tilde{\mathcal{D}}(x|\ell(x)=1)}{\mathcal{D}(x|\ell(x)=1)} < \sqrt{B}$ and $\frac{\tilde{\mathcal{D}}(x|\ell(x)=-1)}{\mathcal{D}(x|\ell(x)=-1)} < \sqrt{B}$ where $\mathcal{D}(x|\ell(x) = 1)$ and $\mathcal{D}(x|\ell(x) = -1)$ are the class conditional probability distributions of the problem, such that at least a $1 - \varepsilon$ probability mass of examples $x \sim \mathcal{D}$ satisfies*

$$\underset{x',x''\sim\tilde{\mathcal{D}}\times\tilde{\mathcal{D}}}{\tilde{\mathcal{D}}} \left[d(x, x') < d(x, x'')|\ell(x') = \ell(x), \ell(x'') \neq \ell(x)\right] \geq \frac{1}{2} + \gamma \tag{A.1}$$

It can be shown (and is implicit in the proof of Theorem 5 in Wang et al. (2007)) that the above condition is equivalent to

$$\underset{x',x''\sim\mathcal{D}\times\mathcal{D}}{\mathbb{E}} \left[\!\left[w_{\ell(x)}(x')w_{-\ell(x)}(x'') \,\text{sgn}\left(d(x, x'') - d(x, x')\right) |\ell(x') = \ell(x), \ell(x'') \neq \ell(x)\right]\!\right] \geq 2\gamma$$

where $w_1(x) := \frac{\tilde{\mathcal{D}}(x|\ell(x)=1)}{\mathcal{D}(x|\ell(x)=1)}$ and $w_{-1}(x) := \frac{\tilde{\mathcal{D}}(x|\ell(x)=-1)}{\mathcal{D}(x|\ell(x)=-1)}$. Now define $\varpi(x', x'') := w_{\ell(x')}(x')w_{\ell(x'')}(x'')$ and take $f = \text{sgn}()$ as the transfer function in our model. We have, for a $1 - \varepsilon$ fraction of

points,

$$\mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}}\left[\!\left[\varpi\left(x',x''\right)f\left(K(x,x')-K(x,x'')\right)|\ell(x')=\ell(x),\ell(x'')\neq\ell(x)\right]\!\right]\geq C_f\gamma$$

which is clearly equivalent to

$$\mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}}\left[\!\left[w_{\ell(x)}(x')w_{-\ell(x)}(x'')\operatorname{sgn}\left(K(x,x')-K(x,x'')\right)|\ell(x')=\ell(x),\ell(x'')\neq\ell(x)\right]\!\right]\geq\gamma$$

since $C_f=1$ for the sgn() function. Thus the model of learning presented in (Wang et al., 2007) is an instantiation of our proposed model.

A.2 Extending the similarity-based model of Balcan and Blum

Balcan and Blum (2006) present a model of learning with similarity functions. Their model does not consider landmark pairs, just singletons. Accordingly, instead of assigning a weight to each landmark pair, they simply assign a weight to each element of the domain. Consequently one arrives at the following notion of a *good similarity*.

Definition A.2 ((Balcan and Blum, 2006), Definition 3)**.** *A similarity measure $K:\mathcal{X}\times\mathcal{X}\to\mathbb{R}$ is said to be an (ε,γ)-good similarity for a classification problem where $\varepsilon,\gamma>0$ if for some weighing function $w:\mathcal{X}\to[-1,1]$, at least a $1-\varepsilon$ probability mass of examples $x\sim\mathcal{D}$ satisfies*

$$\mathop{\mathbb{E}}_{x'\sim\mathcal{D}}\left[\!\left[w\left(x'\right)K(x,x')|\ell(x')=\ell(x)\right]\!\right]\geq\mathop{\mathbb{E}}_{x'\sim\mathcal{D}}\left[\!\left[w\left(x'\right)K(x,x')|\ell(x')\neq\ell(x)\right]\!\right]+\gamma \qquad \text{(A.2)}$$

Now define $w_+:=\mathop{\mathbb{E}}_{x'\sim\mathcal{D}}\left[\!\left[w\left(x\right)|\ell(x)=1\right]\!\right]$ and $w_-:=\mathop{\mathbb{E}}_{x'\sim\mathcal{D}}\left[\!\left[w\left(x\right)|\ell(x)=-1\right]\!\right]$. Furthermore, take $\varpi(x',x'')=w(x')w(x'')$ as the weight function and $f=\mathrm{id}()$ as the transfer function in our model. Then we have, for a $1-\varepsilon$ fraction of the points,

$$\begin{aligned}
&\mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}}\left[\!\left[\varpi\left(x',x''\right)f\left(K(x,x')-K(x,x'')\right)|\ell(x')=\ell(x),\ell(x'')\neq\ell(x)\right]\!\right]\geq C_f\gamma\\
\equiv\ &\mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}}\left[\!\left[\varpi\left(x',x''\right)\left(K(x,x')-K(x,x'')\right)|\ell(x')=\ell(x),\ell(x'')\neq\ell(x)\right]\!\right]\geq\gamma\\
\equiv\ &\mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}}\left[\!\left[\varpi\left(x',x''\right)K(x,x')|\ell(x')=\ell(x),\ell(x'')\neq\ell(x)\right]\!\right]\geq\\
&\mathop{\mathbb{E}}_{x',x''\sim\mathcal{D}\times\mathcal{D}}\left[\!\left[\varpi\left(x',x''\right)K(x,x'')|\ell(x')=\ell(x),\ell(x'')\neq\ell(x)\right]\!\right]+\gamma\\
\equiv\ &w_{-\ell(x)}\mathop{\mathbb{E}}_{x'\sim\mathcal{D}}\left[\!\left[w(x')K(x,x')|\ell(x')=\ell(x)\right]\!\right]\geq w_{\ell(x)}\mathop{\mathbb{E}}_{x'\sim\mathcal{D}}\left[\!\left[w(x')K(x,x')|\ell(x')\neq\ell(x)\right]\!\right]+\gamma\\
\equiv\ &\mathop{\mathbb{E}}_{x'\sim\mathcal{D}}\left[\!\left[w'(x')K(x,x')|\ell(x')=\ell(x)\right]\!\right]\geq\mathop{\mathbb{E}}_{x'\sim\mathcal{D}}\left[\!\left[w'(x')K(x,x')|\ell(x')\neq\ell(x)\right]\!\right]+\gamma
\end{aligned}$$

where $C_f=1$ for the id() function and $w'(x)=w(x)w_{-\ell(x)}$. Note that this again guarantees a classifier with margin γ in the landmarked space. Thus the model of (Balcan and Blum, 2006) can also be derived in our model.

Appendix B

Generalization Bounds in Presence of Double-dipping

Contents

Abstract *This appendix is with reference to Chapters 4 and 5. Here we present generalization bounds for learning algorithms that use landmarking techniques (such as those proposed in the aforementioned chapters) and re-use landmark points as training points.*

B.1 Introduction

Our analyses presented in Chapters 4 and 5 (as well as the analyses presented in Balcan and Blum (2006); Wang et al. (2007)) use some data as landmark points and then require a fresh batch of training points to learn a classifier in the landmarked space. In practice, however, it might be useful to reuse training data to act as landmark points as well. This is especially true of the approaches outlined in Wang et al. (2007) and Chapter 4 which require labeled landmarks. We give below, generalization bounds for similarity-based learning algorithms that indulge in such "double dipping". The argument uses a technique outlined in Ben-David et al. (2008) and utilizes Rademacher-average based proof techniques used elsewhere in the thesis. We present a generic argument that, in a manner similar to Lemma 5.23, can be specialized to the various learning problems considered in Chapters 4 and 5.

To make the presentation easier we set up some notation. For any predictor f, let $\mathcal{L}_f = \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}} [\![\ell(f(\mathbf{x}), y(\mathbf{x}))]\!]$ denote the population risk and for any training set S of size n, let $\hat{\mathcal{L}}_f^S = \frac{1}{n}\sum_{\mathbf{x}_i\in S} \ell(f(\mathbf{x}_i), y(\mathbf{x}_i))$ denote the empirical risk. For any landmark set $S = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, we let $\Psi_S : \mathbf{x} \mapsto (K(\mathbf{x}, \mathbf{x}_1), \ldots, K(\mathbf{x}, \mathbf{x}_n))$. For any weight vector $\mathbf{w} \in \mathbb{R}^n, \|\mathbf{w}\|_\infty \leq B$ in the landmarked space, denote the predictor $f_{(S,\mathbf{w})} := \frac{1}{n}\langle \mathbf{w}, \Psi_S(\mathbf{x})\rangle = \mathbf{x} \mapsto \frac{1}{n}\sum_{i=1}^{n} \mathbf{w}_i K(\mathbf{x}, \mathbf{x}_i)$. Also let $\mathcal{F}_S := \left\{\mathbf{x} \mapsto \frac{1}{n}\langle \mathbf{w}, \Psi_S(\mathbf{x})\rangle\right\} = \left\{f_{(S,\mathbf{w})} : \mathbf{w} \in \mathbb{R}^n, \|\mathbf{w}\|_\infty \leq B\right\}$.

B.2 *Stable* Embeddings and Sample-dependent Analyses

We note that the embedding defined above is "stable" in the sense that changing a single landmark does not change the embedding too much with respect to bounded predictors. More formally, for any set of n points $S = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, define $g(S) := \sup_{f \in \mathcal{F}_S} \left\{ \mathcal{L}_f - \hat{\mathcal{L}}_f^S \right\}$.

Let S^i be another set of n points that (arbitrarily) differs from S just at the i^{th} point and coincides with S on the rest. Then we have, for any fixed $\mathbf{w}$ of bounded L_∞ norm (i.e. $\|\mathbf{w}\|_\infty \leq B$) and bounded similarity function (i.e. $K(\mathbf{x}, \mathbf{y}) \leq 1$),

$$\begin{aligned}
\sup_{\mathbf{x}} \left\{ \left| f_{(S,\mathbf{w})}(\mathbf{x}) - f_{(S^i,\mathbf{w})}(\mathbf{x}) \right| \right\} &= \sup_{\mathbf{x}} \left\{ \left| \frac{1}{n} \sum_{j=1}^{n} \mathbf{w}_j K(\mathbf{x}, \mathbf{x}_j) - \frac{1}{n} \sum_{i=1}^{n} \mathbf{w}_j K(\mathbf{x}, \mathbf{x}'_j) \right| \right\} \\
&= \sup_{\mathbf{x}} \left\{ \left| \frac{1}{n} \mathbf{w}_i \left(K(\mathbf{x}, \mathbf{x}_i) - K(\mathbf{x}, \mathbf{x}'_i) \right) \right| \right\} \\
&\leq \frac{2B}{n}
\end{aligned}$$

Note that, although Chapter 4 uses pairs of labeled points to define the embedding, the following argument can easily be extended to incorporate this since the embedding is identical to the embedding Ψ_S described above with respect to being "stable". In fact this analysis holds for any stable embedding defined using training points.

Our argument proceeds by showing that with high probability (over choice of the set S) we have

$$\sup_{\mathbf{w}} \left\{ \left| \mathcal{L}_{f_{(S,\mathbf{w})}} - \hat{\mathcal{L}}^S_{f_{(S,\mathbf{w})}} \right| \right\} \leq \varepsilon$$

By the definition of $\mathcal{F}_S$, the above requirement translates to showing that with high probability,

$$\sup_{f \in \mathcal{F}_S} \left\{ \left| \mathcal{L}_f - \hat{\mathcal{L}}^S_f \right| \right\} \leq \varepsilon$$

which highlights the fact that we are dealing with a problem of sample dependent hypothesis spaces[1]. Note that this exactly captures the double dipping procedure of reusing training points as landmark points.

Such a result can be used to give a crisp generalization bound as follows: using Lemma 5.22 and task specific guarantees, we have, with high probability, the existence of a good predictor in the landmarked space of a randomly chosen landmark set S i.e. with very high probability over choice of S, we have $\inf_{f \in \mathcal{F}_S} \{\mathcal{L}_f\} \leq \varepsilon_0$. Let this be achieved by the predictor f^*. Using the uniform convergence guarantee above we get $\hat{\mathcal{L}}^S_{f^*} \leq \varepsilon_0 + \varepsilon$ (with some loss of confidence due to application of a union bound).

Now consider the predictor $\hat{f} := \inf_{f \in \mathcal{F}_S} \left\{ \hat{\mathcal{L}}^S_f \right\}$. Clearly $\hat{\mathcal{L}}^S_{\hat{f}} \leq \hat{\mathcal{L}}^S_{f^*} \leq \varepsilon_0 + \varepsilon$. Invoking the

[1] ↑ We were not able to find any written manuscript detailing the argument of Ben-David et al. (2008). However the argument itself is fairly generic in allowing one to prove generalization bounds for sample dependent hypothesis spaces.

uniform convergence bound yet again shows us that

$$\mathcal{L}_{\hat{f}} \leq \hat{\mathcal{L}}^S_{\hat{f}} + \sup_{f \in \mathcal{F}_S} \left\{ \mathcal{L}_f - \hat{\mathcal{L}}^S_f \right\} \leq \varepsilon_0 + 2\varepsilon$$

Note that we incur some more loss of confidence due to another application of the union bound. This tells us that with high probability, a predictor learned by choosing a random landmark set and training on the landmark set itself would yield a good predictor.

B.3 Double-dipping via Sample-dependent Hypothesis Spaces

We will proceed via a vanilla uniform convergence argument involving symmetrization and an application of McDiarmid's inequality (restated below for convenience). However, proving the stability prerequisite for the application of McDiarmid's inequality shall require use of stability of both the predictor $f_{(S,\mathbf{w})}$ as well as the embedding Ψ_S. Let the loss function ℓ be C_L-Lipschitz in its first argument.

Theorem B.1 (McDiarmid's inequality McDiarmid (1989))**.** *Let $X_1, \ldots, X_n$ be independent random variables taking values in some set $\mathcal{X}$. Furthermore, let $f : \mathcal{X}^n \to \mathbb{R}$ be a function of n variables that satisfies, for all $i \in [n]$ and all $x_1, \ldots, x_n, x_i' \in \mathcal{X}$,*

$$\left| f\left(x_1, \ldots, x_i, \ldots, x_n\right) - f\left(x_1, \ldots, x_i', \ldots, x_n\right) \right| \leq c_i$$

then for all $\varepsilon > 0$, we have

$$\mathbb{P}\left[f - \mathbb{E}\left[\!\left[f \right]\!\right] > \varepsilon \right] \leq \exp\left(\frac{-2\varepsilon^2}{\sum_{i=1}^n c_i^2} \right)$$

Note that the inequality can be made two sided simply by taking $f = -f$. For this reason we shall not worry about the sidedness of the application of this inequality. We shall invoke McDiarmid's inequality on the function $g(S) := \sup_{f \in \mathcal{F}_S} \left\{ \mathcal{L}_f - \hat{\mathcal{L}}^S_f \right\}$ with $S = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ being the random variables. To do so we first prove the stability of the function $g(S)$ with respect to its variables and then bound the value of $\mathop{\mathbb{E}}_S \left[\!\left[g(S) \right]\!\right]$.

Theorem B.2. *For any S, S^i, we have $\left| g(S) - g(S^i) \right| \leq \frac{6BC_L}{n}$.*

Proof. We have

$$\begin{aligned}
g(S) &= \sup_{f \in \mathcal{F}_S} \left\{ \mathcal{L}_f - \hat{\mathcal{L}}^S_f \right\} = \sup_{f \in \mathcal{F}_S} \left\{ \mathcal{L}_f - \hat{\mathcal{L}}^S_f - \hat{\mathcal{L}}^{S^i}_f + \hat{\mathcal{L}}^{S^i}_f \right\} \\
&\leq \sup_{f \in \mathcal{F}_S} \left\{ \mathcal{L}_f - \hat{\mathcal{L}}^{S^i}_f \right\} + \sup_{f \in \mathcal{F}_S} \left\{ \hat{\mathcal{L}}^S_f - \hat{\mathcal{L}}^{S^i}_f \right\} \\
&\leq \sup_{f \in \mathcal{F}_S} \left\{ \mathcal{L}_f - \hat{\mathcal{L}}^{S^i}_f \right\} + \frac{2BC_L}{n}
\end{aligned}$$

where in the fourth step we have used the fact that the loss function is Lipschitz and the

embedding function Ψ_S is bounded. We also have

$$\begin{aligned}
\sup_{f\in\mathcal{F}_S}\left\{\mathcal{L}_f-\hat{\mathcal{L}}_f^{S^i}\right\} &= \sup_{\mathbf{w}}\left\{\mathcal{L}_{f_{(S,\mathbf{w})}}-\hat{\mathcal{L}}^{S^i}_{f_{(S,\mathbf{w})}}\right\}\\
&= \sup_{\mathbf{w}}\left\{\mathcal{L}_{f_{(S,\mathbf{w})}}-\mathcal{L}_{f_{(S^i,\mathbf{w})}}+\mathcal{L}_{f_{(S^i,\mathbf{w})}}-\hat{\mathcal{L}}^{S^i}_{f_{(S^i,\mathbf{w})}}+\hat{\mathcal{L}}^{S^i}_{f_{(S^i,\mathbf{w})}}-\hat{\mathcal{L}}^{S^i}_{f_{(S,\mathbf{w})}}\right\}\\
&\leq \sup_{\mathbf{w}}\left\{\mathcal{L}_{f_{(S^i,\mathbf{w})}}-\hat{\mathcal{L}}^{S^i}_{f_{(S^i,\mathbf{w})}}\right\}+\sup_{\mathbf{w}}\left\{\mathcal{L}_{f_{(S,\mathbf{w})}}-\mathcal{L}_{f_{(S^i,\mathbf{w})}}\right\}\\
&\quad +\sup_{\mathbf{w}}\left\{\hat{\mathcal{L}}^{S^i}_{f_{(S^i,\mathbf{w})}}-\hat{\mathcal{L}}^{S^i}_{f_{(S,\mathbf{w})}}\right\}\\
&\leq \sup_{\mathbf{w}}\left\{\mathcal{L}_{f_{(S^i,\mathbf{w})}}-\hat{\mathcal{L}}^{S^i}_{f_{(S^i,\mathbf{w})}}\right\}+\frac{2BC_L}{n}+\frac{2BC_L}{n}\\
&= \sup_{f\in\mathcal{F}_{S^i}}\left\{\mathcal{L}_f-\hat{\mathcal{L}}_f^{S^i}\right\}+\frac{4BC_L}{n}\\
&= g(S^i)+\frac{4BC_L}{n}
\end{aligned}$$

where in the fourth step we have used the stability of the embedding function and that the loss function is C_L-Lipschitz in its first argument. This ensures that for all $\mathbf{x}$ we have $\left|\ell\left(f_{(S,\mathbf{w})}(\mathbf{x}),y(\mathbf{x})\right)-\ell\left(f_{(S^i,\mathbf{w})}(\mathbf{x}),y(\mathbf{x})\right)\right|\leq\frac{2BC_L}{n}$. We note that since this holds for all $\mathbf{x}$, it also holds in expectation over any (empirical) distribution as well. Putting the two inequalities together gives us $g(S)\leq g(S^i)+\frac{6BC_L}{n}$. Similarly we also have $g(S^i)\leq g(S)+\frac{6BC_L}{n}$ which gives us the result. where in the fourth step we have used the stability of the embedding function and that the loss function is C_L-Lipschitz in its first argument. This ensures that for all $\mathbf{x}$ we have $\left|\ell\left(f_{(S,\mathbf{w})}(\mathbf{x}),y(\mathbf{x})\right)-\ell\left(f_{(S^i,\mathbf{w})}(\mathbf{x}),y(\mathbf{x})\right)\right|\leq\frac{2BC_L}{n}$. We note that since this holds for all $\mathbf{x}$, it also holds in expectation over any (empirical) distribution as well. Putting the two inequalities together gives us $g(S)\leq g(S^i)+\frac{6BC_L}{n}$. Similarly we also have $g(S^i)\leq g(S)+\frac{6BC_L}{n}$ which gives us the result. □

We now have that the function $g(S)$ is $\mathcal{O}\left(\frac{1}{n}\right)$-stable with respect to each of its inputs. We now move on to bound its expectation. For any function class $\mathcal{F}$ we define its *empirical Rademacher average* as follows

$$\hat{\mathcal{R}}_n(\mathcal{F}):=\mathop{\mathbb{E}}_{\sigma}\left[\!\left[\sup_{f\in\mathcal{F}}\left\{\frac{1}{n}\sum_{\mathbf{x}_i\in S}\sigma_i f(\mathbf{x}_i)\right\}\middle|S\right]\!\right]$$

Also let $\mathcal{F}:=\{\mathbf{x}\mapsto\langle\mathbf{w},\mathbf{x}\rangle:\|\mathbf{w}\|_2\leq B\}$ and $\mathcal{X}:=\{\mathbf{x}:\|\mathbf{x}\|_2\leq 1\}$.

Theorem B.3. $\mathop{\mathbb{E}}_{S}\left[\!\left[\sup_{f\in\mathcal{F}_S}\left\{\mathcal{L}_f-\hat{\mathcal{L}}_f^S\right\}\right]\!\right]\leq 2BC_L\sqrt{\frac{1}{n}}$

Proof. We have

$$\begin{aligned}
\mathop{\mathbb{E}}_{S}\left[\!\!\left[\sup_{f\in\mathcal{F}_S}\left\{\mathcal{L}_f-\hat{\mathcal{L}}_f^S\right\}\right]\!\!\right] &= \mathop{\mathbb{E}}_{S}\left[\!\!\left[\sup_{f\in\mathcal{F}_S}\left\{\mathop{\mathbb{E}}_{S'}\left[\!\!\left[\hat{\mathcal{L}}_f^{S'}\right]\!\!\right]-\hat{\mathcal{L}}_f^S\right\}\right]\!\!\right] \leq \mathop{\mathbb{E}}_{S,S'}\left[\!\!\left[\sup_{f\in\mathcal{F}_S}\left\{\hat{\mathcal{L}}_f^{S'}-\hat{\mathcal{L}}_f^S\right\}\right]\!\!\right] \\
&\leq \mathop{\mathbb{E}}_{S,S'}\left[\!\!\left[\sup_{f\in\mathcal{F}_{S\cup S'}}\left\{\hat{\mathcal{L}}_f^{S'}-\hat{\mathcal{L}}_f^S\right\}\right]\!\!\right] \\
&= \mathop{\mathbb{E}}_{S,S',\sigma}\left[\!\!\left[\sup_{f\in\mathcal{F}_{S\cup S'}}\left\{\frac{1}{n}\sum_{\mathbf{x}_i\in S,\mathbf{x}_i'\in S'}\sigma_i\left(\ell(f(\mathbf{x}_i'),y(\mathbf{x}_i'))-\ell(f(\mathbf{x}_i),y(\mathbf{x}_i))\right)\right\}\right]\!\!\right] \\
&\leq 2\mathop{\mathbb{E}}_{S,S',\sigma}\left[\!\!\left[\sup_{f\in\mathcal{F}_{S\cup S'}}\left\{\frac{1}{n}\sum_{\mathbf{x}_i\in S}\sigma_i\ell(f(\mathbf{x}_i),y(\mathbf{x}_i))\right\}\right]\!\!\right] \\
&= 2\mathop{\mathbb{E}}_{S,S',\sigma}\left[\!\!\left[\sup_{\mathbf{w}}\left\{\frac{1}{n}\sum_{\mathbf{x}_i\in S}\sigma_i\ell(f_{(S\cup S',\mathbf{w})}(\mathbf{x}_i),y(\mathbf{x}_i))\right\}\right]\!\!\right] \\
&\leq 2\mathop{\mathbb{E}}_{S,S',\sigma}\left[\!\!\left[\sup_{f\in\mathcal{F}}\left\{\frac{1}{n}\sum_{\mathbf{x}_i\in S}\sigma_i\ell(f(\mathbf{x}_i),y(\mathbf{x}_i))\right\}\right]\!\!\right] \\
&= 2\mathop{\mathbb{E}}_{S}\left[\!\!\left[\hat{\mathcal{R}}_n(\ell\circ\mathcal{F})\right]\!\!\right] \leq 2C_L\mathop{\mathbb{E}}_{S}\left[\!\!\left[\hat{\mathcal{R}}_n(\mathcal{F})\right]\!\!\right] \leq 2BC_L\sqrt{\frac{1}{n}}
\end{aligned}$$

where in the third step we have used the fact that $\mathcal{F}_S \supseteq \mathcal{F}_{S'}$ if $S \supseteq S'$ (this is the monotonicity requirement in Ben-David et al. (2008)). Note that this is essential to establish symmetry so that Rademacher variables can be introduced in the next (symmetrization) step. In the seventh step, we have used the fact that for every S such that $|S| = n$ and $\mathbf{w} \in \mathbb{R}^n$ such that $\|\mathbf{w}\|_\infty \leq B$, there exists a function $f \in \mathcal{F}$ such that for all $\mathbf{x}$, there exists a $\mathbf{x}' \in \mathcal{X}$ such that $f_{(S,\mathbf{w})}(\mathbf{x}) = f(\mathbf{x}')$. In the last step we have used a result from Ambroladze and Shawe-Taylor (2004) which allows calculation of Rademacher averages for composition classes and an intermediate result from the proof of Lemma 5.23 which gives us Rademacher averages for the function class $\mathcal{F}$. □

Thus, by an application of McDiarmid's inequality we have, with probability $(1-\delta)$ over choice of the landmark (training) set,

$$\sup_{f\in\mathcal{F}_S}\left\{\mathcal{L}_f-\hat{\mathcal{L}}_f^S\right\} \leq \mathbb{E}\left[\!\!\left[\sup_{f\in\mathcal{F}_S}\left\{\mathcal{L}_f-\hat{\mathcal{L}}_f^S\right\}\right]\!\!\right] + 6BC_L\sqrt{\frac{\log 1/\delta}{2n}} \leq 4BC_L\sqrt{\frac{\log 1/\delta}{n}}.$$

Similarly, we can also prove

$$\sup_{f\in\mathcal{F}_S}\left\{\hat{\mathcal{L}}_f^S-\mathcal{L}_f\right\} \leq 4BC_L\sqrt{\frac{\log 1/\delta}{n}},$$

which concludes our argument justifying double dipping.

Bibliography

[del()] Delve Dataset Repository. http://www.cs.toronto.edu/~delve/data/datasets.html. University of Toronto. (Cited on pages 94 and 97.)

[sta()] StatLib Dataset Repository. http://lib.stat.cmu.edu/datasets/. Carnegie Mellon University. (Cited on pages 94 and 97.)

[uci()] UCI Machine Learning Repository. http://archive.ics.uci.edu/ml. University of California, Irvine, School of Information and Computer Sciences. (Cited on pages 50, 68, 94 and 97.)

[Achlioptas et al.(2001)] Dimitris Achlioptas, Frank McSherry, and Bernhard Schölkopf. Sampling Techniques for Kernel Methods. In *14th Annual Conference on Neural Information Processing Systems (NIPS)*, 2001. (Cited on page 35.)

[Agarwal(2008)] Shivani Agarwal. Generalization Bounds for Some Ordinal Regression Algorithms. In *19th International Conference on Algorithmic Learning Theory (ALT)*, pages 7–21, 2008. (Cited on pages 89, 90 and 91.)

[Agarwal and Niyogi(2009)] Shivani Agarwal and Partha Niyogi. Generalization Bounds for Ranking Algorithms via Algorithmic Stability. *Journal of Machine Learning Research*, 10:441–474, 2009. (Cited on pages 93 and 116.)

[Ambroladze and Shawe-Taylor(2004)] Amiran Ambroladze and John Shawe-Taylor. Complexity of Pattern Classes and Lipschitz Property. In *15th International Conference on Algorithmic Learning Theory (ALT)*, pages 181–193, 2004. (Cited on page 169.)

[Angluin(1987)] Dana Angluin. Learning Regular Sets from Queries and Counterexamples. *Information and Computation*, 75(2):87–106, 1987. (Cited on page 9.)

[Argyriou et al.(2009)] Andreas Argyriou, Charles A. Micchelli, and Massimiliano Pontil. When Is There a Representer Theorem? Vector Versus Matrix Regularizers. *Journal of Machine Learning Research*, 10:2507–2529, November 2009. (Cited on page 34.)

[Arora et al.(1997)] Sanjeev Arora, László Babai, Jacques Stern, and Z. Sweedyk. The Hardness of Approximate Optima in Lattices, Codes, and Systems of Linear Equations. *Journal of Computer and System Sciences*, 54(2):317–331, April 1997. (Cited on page 64.)

[Bach and Jordan(2005)] Francis R. Bach and Michael I. Jordan. Predictive Low-rank Decomposition for Kernel Methods. In *22nd International Conference on Machine Learning (ICML)*, pages 33–40, 2005. (Cited on page 37.)

[Balcan and Blum(2006)] Maria-Florina Balcan and Avrim Blum. On a Theory of Learning with Similarity Functions. In *23rd Annual International Conference on Machine Learning (ICML)*, pages 73–80, 2006. (Cited on pages x, xi, xvi, 25, 32, 57, 58, 59, 60, 61, 64, 66, 78, 79, 80, 81, 130, 131, 158, 163, 164 and 165.)

[Balcan et al.(2006)] Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. Kernels as Features: On Kernels, Margins, and Low-dimensional Mappings. *Machine Learning*, 65(1):79–94, 2006. (Cited on page 60.)

[Balcan et al.(2008a)] Maria-Florina Balcan, Avrim Blum, and Nathan Srebro. Improved Guarantees for Learning via Similarity Functions. In *21st Annual Conference on Computational Learning Theory (COLT)*, pages 287–298, 2008a. (Cited on pages 58, 85 and 103.)

[Balcan et al.(2008b)] Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. A Discriminative Framework for Clustering via Similarity Functions. In *ACM Annual Symposium on Theory of Computing (STOC)*, pages 671–680, 2008b. (Cited on page 158.)

[Bartlett and Mendelson(2002)] Peter Bartlett and Shahar Mendelson. Rademacher and Gaussian complexities : Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002. (Cited on pages 28, 99 and 104.)

[Bellet et al.(2012)] Aurélien Bellet, Amaury Habrard, and Marc Sebban. Similarity Learning for Provably Accurate Sparse Linear Classification. In *29th International Conference on Machine Learning (ICML)*, 2012. (Cited on pages 32, 130 and 158.)

[Ben-David et al.(2008)] Shai Ben-David, Ali Rahimi, and Nathan Srebro. Generalization Bounds for Indefinite Kernel Machines. In *NIPS 2008 Workshop: New Challenges in Theoretical Machine Learning*, 2008. (Cited on pages 79, 165, 166 and 169.)

[Bengio et al.(2005)] Yoshua Bengio, Olivier Delalleau, and Nicolas Le Roux. The Curse of Highly Variable Functions for Local Kernel Machines. In *19th Annual Conference on Neural Information Processing Systems (NIPS)*, 2005. (Cited on page 34.)

[Bi et al.(2003)] Jinbo Bi, Kristin P. Bennett, Mark J. Embrechts, Curt M. Breneman, and Minghu Song. Dimensionality Reduction via Sparse Support Vector Machines. *Journal of Machine Learning Research*, 3:1229–1243, March 2003. (Cited on page 35.)

[Brefeld and Scheffer(2005)] Ulf Brefeld and Tobias Scheffer. AUC Maximizing Support Vector Learning. In *ICML workshop on ROC Analysis in Machine Learning*, 2005. (Cited on page 116.)

[Burges and Bernhard Scholkopf(1996)] Christopher J. C. Burges and Bernhard Bernhard Scholkopf. Improving the Accuracy and Speed of Support Vector Machines. In *9th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 375–381, 1996. (Cited on page 35.)

[Cao et al.(2012)] Qiong Cao, Zheng-Chu Guo, and Yiming Ying. Generalization Bounds for Metric and Similarity Learning, 2012. arXiv:1207.5437. (Cited on pages 116 and 131.)

[Cesa-Bianchi and Gentile(2008)] Nicoló Cesa-Bianchi and Claudio Gentile. Improved Risk Tail Bounds for On-Line Algorithms. *IEEE Transactions on Information Theory*, 54(1):286–390, 2008. (Cited on page 142.)

[Cesa-Bianchi et al.(2001)] Nicoló Cesa-Bianchi, Alex Conconi, and Claudio Gentile. On the Generalization Ability of On-Line Learning Algorithms. In *14th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 359–366, 2001. (Cited on pages 117 and 119.)

[Chang and Lin(2011)] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2 (3):27:1–27:27, 2011. (Cited on pages 50 and 64.)

[Chen et al.(2009a)] Yihua Chen, Eric K. Garcia, Maya R. Gupta, Ali Rahimi, and Luca Cazzanti. Similarity-based Classification: Concepts and Algorithms. *Journal of Machine Learning Research*, 10:747–776, 2009a. (Cited on pages 24, 58, 65, 67, 80 and 84.)

[Chen et al.(2009b)] Yihua Chen, Maya R. Gupta, and Benjamin Recht. Learning Kernels from Indefinite Similarities. In *26th Annual International Conference on Machine Learning (ICML)*, pages 145–152, 2009b. (Cited on pages 78 and 80.)

[Chitta et al.(2012)] Radha Chitta, Rong Jin, and Anil K. Jain. Efficient Kernel Clustering Using Random Fourier Features. In *12th IEEE International Conference on Data Mining (ICDM)*, pages 161–170, 2012. (Cited on page 35.)

[Chu and Keerthi(2007)] Wei Chu and S. Sathiya Keerthi. Support Vector Ordinal Regression. *Neural Computation*, 19(3):792–815, 2007. (Cited on pages 89, 91, 92, 95 and 106.)

[Clémençon et al.(2008)] Stéphan Clémençon, Gábor Lugosi, and Nicolas Vayatis. Ranking and empirical minimization of U-statistics. *Annals of Statistics*, 36:844–874, 2008. (Cited on page 116.)

[Cortes et al.(2010a)] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Generalization Bounds for Learning Kernels. In *27th International Conference on Machine Learning (ICML)*, pages 247–254, 2010a. (Cited on pages 32 and 132.)

[Cortes et al.(2010b)] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Two-Stage Learning Kernel Algorithms. In *27th International Conference on Machine Learning (ICML)*, pages 239–246, 2010b. (Cited on pages 32, 130 and 132.)

[Cortes et al.(2010c)] Corinna Cortes, Mehryar Mohri, and Ameet Talwalkar. On the Impact of Kernel Approximation on Learning Accuracy. In *13th International Conference*

on Artificial Intelligence and Statistics (AISTATS), pages 113–120, 2010c. (Cited on page 36.)

[Cossalter et al.(2011)] Michele Cossalter, Rong Yan, and Lu Zheng. Adaptive Kernel Approximation for Large-Scale Non-Linear SVM Prediction. In *28th International Conference on Machine Learning (ICML)*, 2011. (Cited on page 35.)

[Cristianini et al.(2001)] Nello Cristianini, John Shawe-Taylor, André Elisseeff, and Jaz S. Kandola. On Kernel-Target Alignment. In *14th International Conference on Neural Information Processing Systems (NIPS)*, pages 367–373, 2001. (Cited on page 130.)

[Cucker and Smale(2001)] Felipe Cucker and Steve Smale. On the Mathematical Foundations of Learning. *Bulletin of the American Mathematical Society*, 39(1):1–49, 2001. (Cited on pages 26 and 42.)

[Duda et al.(2000)] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Wiley-Interscience, second edition, 2000. (Cited on pages 4, 20 and 23.)

[Dudley(2002)] Richard Mansfield Dudley. *Real Analysis and Probability*. Cambridge University Press, Cambridge, 2002. (Cited on page 12.)

[Fan et al.(2008)] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008. (Cited on pages 50, 58, 59 and 67.)

[FitzGerald et al.(1995)] Carl H. FitzGerald, Charles A. Micchelli, and Allan Pinkus. Functions That Preserve Families of Positive Semidefinite Matrices. *Linear Algebra and its Applications*, 221:83–102, 1995. (Cited on page 44.)

[Freedman(1975)] David A. Freedman. On Tail Probabilities for Martingales. *Annals of Probability*, 3(1):100–118, 1975. (Cited on page 142.)

[Garey and Johnson(1979)] M. R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the theory of NP-Completeness*. Freeman, San Francisco, 1979. (Cited on page 64.)

[Gilad-Bachrach et al.(2006)] Ran Gilad-Bachrach, Amir Navot, and Naftali Tishby. Query By Committee Made Real. In *20th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 443–450, 2006. (Cited on page 9.)

[Goldfarb(1984)] Lev Goldfarb. A Unified Approach to Pattern Recognition. *Pattern Recognition*, 17(5):575–582, 1984. (Cited on page 24.)

[Gottlieb et al.(2010)] Lee-Ad Gottlieb, Aryeh (Leonid) Kontorovich, and Robert Krauthgamer. Efficient Classification for Metric Data. In *Annual Conference on Computational Learning Theory (COLT)*, 2010. (Cited on page 24.)

[Graepel et al.(1998)] Thore Graepel, Ralf Herbrich, Peter Bollmann-Sdorra, and Klaus Obermayer. Classification on Pairwise Proximity Data. In *11th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 438–444, 1998. (Cited on page 58.)

[Haasdonk(2005)] Bernard Haasdonk. Feature Space Interpretation of SVMs with Indefinite Kernels. *IEEE Transactions on Pattern Analysis and Machince Intelligence*, 27(4): 482–492, 2005. (Cited on pages 24, 58, 78 and 80.)

[Hazan et al.(2006)] Elad Hazan, Adam Kalai, Satyen Kale, and Amit Agarwal. Logarithmic Regret Algorithms for Online Convex Optimization. In *Annual Conference on Learning Theory*, pages 499–513, 2006. (Cited on page 126.)

[Hilbert and López(2011)] Martin Hilbert and Priscila López. The World's Technological Capacity to Store, Communicate, and Compute Information. *Science*, 332(6025):60–65, 2011. (Cited on page 2.)

[Ho and Lin(2012)] Chia-Hua Ho and Chih-Jen Lin. Large-scale Linear Support Vector Regression. `http://www.csie.ntu.edu.tw/~cjlin/papers/linear-svr.pdf`, retrieved on May 18, 2012, 2012. (Cited on pages 83 and 95.)

[Hoeffding(1963)] Wassily Hoeffding. Probability Inequalities for Sums of Bounded Random Variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963. (Cited on page 15.)

[Indyk and Motwani(1998)] Piotr Indyk and Rajeev Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *30th Annual ACM Symposium on Theory of Computing (STOC)*, pages 604–613, 1998. (Cited on pages 36 and 39.)

[Indyk and Thaper(2003)] Piotr Indyk and Nitin Thaper. Fast Image Retrieval via Embeddings. In *International Workshop Statistical and Computational Theories of Vision*, 2003. (Cited on page 58.)

[Jacobs et al.(2000)] David W. Jacobs, Daphna Weinshall, and Yoram Gdalyahu. Classification with Nonmetric Distances: Image Retrieval and Class Representation. *IEEE Transactions on Pattern Analysis and Machince Intelligence*, 22(6):583–600, 2000. (Cited on page 24.)

[Jain et al.(2012)] Prateek Jain, Brian Kulis, Jason V. Davis, and Inderjit S. Dhillon. Metric and Kernel Learning using a Linear Transformation. *Journal of Machine Learning Research*, 13:519–547, 2012. (Cited on page 59.)

[Järvelin and Kekäläinen(2000)] Kalervo Järvelin and Jaana Kekäläinen. IR evaluation methods for retrieving highly relevant documents. In *23rd Annual ACM SIGIR Confer-*

ence on Research and Development in Information Retrieval, pages 41–48, 2000. (Cited on page 93.)

[Jin et al.(2009)] Rong Jin, Shijun Wang, and Yang Zhou. Regularized Distance Metric Learning: Theory and Algorithm. In *23rd Annual Conference on Neural Information Processing Systems (NIPS)*, pages 862–870, 2009. (Cited on pages xi, 116 and 130.)

[Joachims and Yu(2009)] Thorsten Joachims and Chun-Nam Yu. Sparse Kernel SVMs via Cutting-Plane Training. *Machine Learning*, 76(2):179–193, 2009. (Cited on page 35.)

[Jose et al.(2013)] Cijo Jose, Prasoon Goyal, Parv Aggrwal, and Manik Varma. Local Deep Kernel Learning for Efficient Non-linear SVM Prediction. In *30th International Conference on Machine Learning (ICML)*, 2013. (Cited on pages 35 and 157.)

[Kakade and Tewari(2008)] Sham M. Kakade and Ambuj Tewari. On the Generalization Ability of Online Strongly Convex Programming Algorithms. In *22nd Annual Conference on Neural Information Processing Systems (NIPS)*, pages 801–808, 2008. (Cited on pages 117, 119, 123, 124, 142, 143 and 148.)

[Kakade et al.(2008)] Sham M. Kakade, Karthik Sridharan, and Ambuj Tewari. On the Complexity of Linear Prediction: Risk Bounds, Margin Bounds, and Regularization. In *22nd Annual Conference on Neural Information Processing Systems (NIPS)*, 2008. (Cited on pages 28, 82, 88, 98, 99, 104, 117, 121, 122 and 128.)

[Kakade et al.(2012)] Sham M. Kakade, Shai Shalev-Shwartz, and Ambuj Tewari. Regularization Techniques for Learning with Matrices. *Journal of Machine Learning Research*, 13:1865–1890, 2012. (Cited on pages 129 and 131.)

[Kar and Jain(2011)] Purushottam Kar and Prateek Jain. Similarity-based Learning via Data Driven Embeddings. In *25th Annual Conference on Neural Information Processing Systems (NIPS)*, 2011. (Cited on pages viii, x, xi, 78, 79, 80, 84 and 98.)

[Kar and Jain(2012)] Purushottam Kar and Prateek Jain. Supervised Learning with Similarity Functions. In *26th Annual Conference on Neural Information Processing Systems (NIPS)*, 2012. (Cited on pages viii and xi.)

[Kar and Karnick(2012)] Purushottam Kar and Harish C. Karnick. Random Feature Maps for Dot Product Kernels. In *15th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012. (Cited on pages vii and viii.)

[Kar et al.(2013)] Purushottam Kar, Bharath Sriperumbudur, Prateek Jain, and Harish C. Karnick. On the Generalization Ability of Online Learning Algorithms for Pairwise Loss Functions. In *30th International Conference on Machine Learning (ICML)*, 2013. (Cited on pages viii and xii.)

[Kumar et al.(2012)] Abhishek Kumar, Alexandru Niculescu-Mizil, Koray Kavukcuoglu, and Hal Daumé III. A Binary Classification Framework for Two-Stage Multiple Kernel Learning. In *29th International Conference on Machine Learning (ICML)*, 2012. (Cited on pages 32, 116, 131 and 132.)

[Kumar and Kannan(2010)] Amit Kumar and Ravindran Kannan. Clustering with Spectral Norm and the k-Means Algorithm. In *51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 299–308, 2010. (Cited on page 8.)

[Ledoux and Talagrand(2002)] Michel Ledoux and Michel Talagrand. *Probability in Banach Spaces: Isoperimetry and Processes*. Springer, 2002. (Cited on page 149.)

[Li et al.(2010)] Fuxin Li, Catalin Ionescu, and Cristian Sminchisescu. Random Fourier Approximations for Skewed Multiplicative Histogram Kernels. In *32nd DAGM Conference on Pattern Recognition*, 2010. (Cited on page 36.)

[Luss and d'Aspremont(2007)] Ronny Luss and Alexandre d'Aspremont. Support Vector Machine Classification with Indefinite Kernels. In *21st Annual Conference on Neural Information Processing Systems (NIPS)*, 2007. (Cited on pages 24, 78 and 80.)

[Mahajan et al.(2012)] Meena Mahajan, Prajakta Nimbhorkar, and Kasturi R. Varadarajan. The Planar k-means Problem is NP-hard. *Theoretical Computer Science*, 442: 13–21, 2012. (Cited on page 8.)

[Maji and Berg(2009)] Subhransu Maji and Alexander C. Berg. Max-margin Additive Classifiers for Detection. In *12th IEEE International Conference on Computer Vision (ICCV)*, pages 40–47, 2009. (Cited on page 36.)

[Mashable()] Mashable. A Day in the Life of the Internet. `http://mashable.com/2012/03/06/one-day-internet-data-traffic/`. accessed January 13th, 2013. (Cited on page 2.)

[McDiarmid(1989)] Colin McDiarmid. On the Method of Bounded Differences. *Surveys in Combinatorics*, 141:148–188, 1989. (Cited on pages 13, 15 and 167.)

[Mercer(1909)] James Mercer. Functions of Positive and Negative Type, and their Connection with the Theory of Integral Equations. *Philosophical Transactions of the Royal Society A*, 209:415–446, 1909. (Cited on page 21.)

[Mitchell(1997)] Tom Mitchell. *Machine Learning*. McGraw Hill, 1997. (Cited on page 4.)

[Ong et al.(2004)] Cheng Soon Ong, Xavier Mary, Stéphane Canu, and Alexander J. Smola. Learning with non-positive Kernels. In *21st Annual International Conference on Machine Learning (ICML)*, 2004. (Cited on pages 25, 58, 78 and 80.)

[Pękalska and Duin(2000)] Elżbieta Pękalska and Robert P. W. Duin. Classifiers for Dissimilarity-Based Pattern Recognition. In *International Conference on Pattern Recognition (ICPR)*, pages 2012–2016, 2000. (Cited on page 24.)

[Pękalska and Duin(2001)] Elżbieta Pękalska and Robert P. W. Duin. On Combining Dissimilarity Representations. In *Multiple Classifier Systems*, pages 359–368, 2001. (Cited on page 58.)

[Pękalska and Duin(2002)] Elżbieta Pękalska and Robert P. W. Duin. Dissimilarity representations allow for building good classifiers. *Pattern Recognition Letters*, 23(8):943–956, 2002. (Cited on page 24.)

[Pękalska et al.(2001)] Elżbieta Pękalska, Pavel Paclík, and Robert P. W. Duin. A Generalized Kernel Approach to Dissimilarity-based Classification. *Journal of Machine Learning Research*, 2:175–211, 2001. (Cited on page 24.)

[Perronnin et al.(2010)] Florent Perronnin, Jorge Sánchez, and Yan Liu. Large-scale Image Categorization with Explicit Data embedding. In *23rd IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2297–2304, 2010. (Cited on page 37.)

[Pham and Pagh(2013)] Ninh Pham and Rasmus Pagh. Fast and Scalable Polynomial Kernels via Explicit Feature Maps. In *19th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2013. (Cited on page 157.)

[Rahimi and Recht(2007)] Ali Rahimi and Benjamin Recht. Random Features for Large-Scale Kernel Machines. In *20th Neural Information Processing Systems (NIPS)*, 2007. (Cited on pages 36, 37, 39, 42 and 43.)

[Ravikumar et al.(2011)] Pradeep Ravikumar, Ambuj Tewari, and Eunho Yang. On NDCG Consistency of Listwise Ranking Methods. In *14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 15 of *JMLR Conference and Workshop Proceedings*, pages 618–626, 2011. (Cited on pages 92, 93, 94 and 109.)

[Rudin(1962)] Walter Rudin. *Fourier Analysis on Groups*. lnterscience Publishers, New. York, 1962. (Cited on pages 36 and 38.)

[Schoenberg(1942)] Isaac Jacob Schoenberg. Positive Definite Functions on Spheres. *Duke Mathematical Journal*, 9(1):96–108, 1942. (Cited on pages ix, 37 and 53.)

[Schölkopf(2000)] Bernhard Schölkopf. The Kernel Trick for Distances. In *13th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 301–307, 2000. (Cited on page 24.)

[Schölkopf and Smola(2002)] Bernhard Schölkopf and Alex J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002. (Cited on pages 21, 29, 34, 38, 39, 53, 78 and 83.)

[Schölkopf et al.(1998)] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Mller. Nonlinear Component Analysis as a Kernel Eigenvalue Problem. *Neural Computation*, 10(5):1299–1319, 1998. (Cited on page 29.)

[Schölkopf et al.(2001)] Bernhard Schölkopf, Ralf Herbrich, and Alex J. Smola. A Generalized Representer Theorem. In *14th Annual Conference on Computational Learning Theory (COLT)*, pages 416–426, 2001. (Cited on page 100.)

[Settles(2012)] Burr Settles. *Active Learning*. Morgan & Claypool, 2012. (Cited on page 9.)

[Shalev-Shwartz et al.(2010a)] Shai Shalev-Shwartz, Ohad Shamir, Nathan Srebro, and Karthik Sridharan. Learnability, Stability and Uniform Convergence. *Journal of Machine Learning Research*, 11:2635–2670, 2010a. (Cited on pages 17 and 25.)

[Shalev-Shwartz et al.(2010b)] Shai Shalev-Shwartz, Nathan Srebro, and Tong Zhang. Trading Accuracy for Sparsity in Optimization Problems with Sparsity Constraints. *SIAM Journal on Optimization*, 20(6):2807–2832, 2010b. (Cited on pages xi, xxi, 79, 85, 86, 87, 88, 95 and 104.)

[Shalev-Shwartz et al.(2011)] Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. *Mathematical Programming*, 127(1):3–30, 2011. (Cited on page 159.)

[Shawe-Taylor et al.(1998)] John Shawe-Taylor, Peter L. Bartlett, Robert C. Williamson, and Martin Anthony. Structural Risk Minimization Over Data-Dependent Hierarchies. *IEEE Transactions on Information Theory*, 44(5):1926–1940, 1998. (Cited on page 16.)

[Srebro(2007)] Nathan Srebro. How Good Is a Kernel When Used as a Similarity Measure? In *Annual Conference on Computational Learning Theory (COLT)*, pages 323–335, 2007. (Cited on pages x, 60, 79, 82, 99, 100, 102 and 103.)

[Sridharan et al.(2008)] Karthik Sridharan, Shai Shalev-Shwartz, and Nathan Srebro. Fast Rates for Regularized Objectives. In *21st Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1545–1552, 2008. (Cited on pages 123 and 140.)

[Steinwart(2001)] Ingo Steinwart. On the Influence of the Kernel on the Consistency of Support Vector Machines. *Journal of Machine Learning Research*, 2:67–93, November 2001. (Cited on page 39.)

[Steinwart(2003)] Ingo Steinwart. Sparseness of Support Vector Machines—Some Asymptotically Sharp Bounds. In *16th Annual Conference on Neural Information Processing Systems (NIPS)*, 2003. (Cited on page 34.)

[Steinwart and Christmann(2008a)] Ingo Steinwart and Andreas Christmann. Sparsity of SVMs that use the epsilon-insensitive loss. In *14th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1569–1576, 2008a. (Cited on page 34.)

[Steinwart and Christmann(2008b)] Ingo Steinwart and Andreas Christmann. *Support Vector Machines*. Information Science and Statistics. Springer-Verlag, 2008b. (Cited on pages 6 and 127.)

[Tsang et al.(2005)] Ivor W. Tsang, T. James, and Pak-Ming Cheung. Very Large SVM training using Core Vector Machine. In *8th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2005. (Cited on page 35.)

[Vapnik(2000)] Vladimir Vapnik. *The Nature of Statistical Learning Theory*. Information Science and Statistics. Springer-Verlag, 2000. (Cited on pages 10, 16, 17 and 25.)

[Varma and Babu(2009)] Manik Varma and Bodla Rakesh Babu. More Generality in Efficient Multiple Kernel Learning. In *26th International Conference on Machine Learning (ICML)*, pages 1065–1072, 2009. (Cited on pages 32, 59 and 158.)

[Vedaldi and Zisserman(2010)] Andrea Vedaldi and Andrew Zisserman. Efficient Additive Kernels via Explicit Feature Maps. In *23rd IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3539–3546, 2010. (Cited on pages 36, 37, 39 and 44.)

[Vempati et al.(2010)] Sreekanth Vempati, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Generalized RBF feature maps for Efficient Detection. In *21st British Machine Vision Conference (BMVC)*, 2010. (Cited on page 36.)

[Venkataramani and Kumar(2009)] Krithika Venkataramani and B. V. K. Vijaya Kumar. Designing classifiers for fusion-based biometric verification. In Plataniotis Boulgouris and Micheli-Tzankou, editors, *Biometrics: Theory, Methods and Applications*. Springer, 2009. (Cited on page 65.)

[Vitter(1985)] Jeffrey Scott Vitter. Random Sampling with a Reservoir. *ACM Transactions on Mathematical Software*, 11(1):37–57, 1985. (Cited on pages 117, 119, 124, 125, 127 and 136.)

[von Luxburg and Bousquet(2004)] Ulrike von Luxburg and Olivier Bousquet. Distance-Based Classification with Lipschitz Functions. *Journal of Machine Learning Research*, 5:669–695, 2004. (Cited on page 24.)

[Wang et al.(2007)] Liwei Wang, Cheng Yang, and Jufu Feng. On Learning with Dissimilarity Functions. In *24th International Conference on Machine Learning (ICML)*, pages 991–998, 2007. (Cited on pages xvi, 32, 57, 58, 59, 60, 61, 66, 67, 68, 78, 79, 80, 163, 164 and 165.)

[Wang et al.(2012)] Yuyang Wang, Roni Khardon, Dmitry Pechyony, and Rosie Jones. Generalization Bounds for Online Learning Algorithms with Pairwise Loss Functions. *Proceedings of the 25th Annual Conference on Learning Theory (COLT 2012), JMLR - Proceedings Track*, 23:13.1–13.22, 2012. (Cited on pages xii, xiii, 116, 117, 118, 119, 121, 122 and 139.)

[Wang et al.(2013)] Yuyang Wang, Roni Khardon, Dmitry Pechyony, and Rosie Jones. Online Learning with Pairwise Loss Functions, 2013. arXiv:1301.5332. (Cited on pages 117 and 127.)

[Weinberger and Saul(2009)] Kilian Q. Weinberger and Lawrence K. Saul. Distance Metric Learning for Large Margin Nearest Neighbor Classification. *Journal of Machine Learning Research*, 10:207–244, 2009. (Cited on pages 24 and 32.)

[Weinberger and Tesauro(2007)] Kilian Q. Weinberger and Gerald Tesauro. Metric Learning for Kernel Regression. In *11th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 612–619, 2007. (Cited on page 94.)

[Weinshall et al.(1998)] Daphna Weinshall, David W. Jacobs, and Yoram Gdalyahu. Classification in Non-Metric Spaces. In *11th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 838–846, 1998. (Cited on page 24.)

[Wikipedia()] Wikipedia. Information Age. `http://en.wikipedia.org/wiki/Information_Age`. accessed January 13th, 2013. (Cited on page 2.)

[Williams and Seeger(2000)] Christopher K. I. Williams and Matthias Seeger. Using the Nyström Method to Speed Up Kernel Machines. In *13th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 682–688, 2000. (Cited on page 35.)

[Xing et al.(2002)] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart J. Russell. Distance Metric Learning with Application to Clustering with Side-Information. In *15th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 505–512, 2002. (Cited on page 116.)

[Zhang(2002)] Tong Zhang. Covering Number Bounds of Certain Regularized Linear Function Classes. *Journal of Machine Learning Research*, 2:527–550, 2002. (Cited on page 99.)

[Zhao et al.(2011)] Peilin Zhao, Steven C. H. Hoi, Rong Jin, and Tianbao Yang. Online AUC Maximization. In *28th International Conference on Machine Learning (ICML)*, pages 233–240, 2011. (Cited on pages xvii, 116, 117, 118, 127, 129, 133, 134, 135, 136 and 152.)

[Zinkevich(2003)] Martin Zinkevich. Online Convex Programming and Generalized Infinitesimal Gradient Ascent. In *20th International Conference on Machine Learning (ICML)*, pages 928–936, 2003. (Cited on pages 133, 134 and 154.)