

New Theoretical Frameworks for Machine Learning

Maria-Florina Balcan

May 2007

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:

Avrim Blum, Chair

Manuel Blum

Yishay Mansour

Tom Mitchell

Santosh Vempala

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.*

Copyright © 2007 Maria-Florina Balcan

Keywords: Semi-supervised Learning, Active Learning, Co-training, Similarity-based Learning, Kernels, Margins, Low-Dimensional Mappings, Sample Complexity, Mechanism Design, Random Sampling Mechanisms, Profit Maximization, Algorithms for Pricing Problems.

Abstract

This thesis develops and analyzes theoretical frameworks for new emerging paradigms of Machine Learning including Semi-supervised, Active, and Similarity-based Learning. These are areas of significant practical importance and significant activity in Machine Learning, and a number of different algorithmic approaches have been developed for each of them. Standard Learning Theory frameworks such as PAC or Statistical Learning Theory models tend to not capture these learning approaches, hence developing sound and rigorous models that provide a thorough understanding of these new paradigms is desirable. The purpose of this thesis is to propose and to study new theoretical frameworks and algorithms for better understanding and extending some of these learning approaches. In addition, this dissertation also presents new applications of techniques from Machine Learning Theory to new emerging areas of Computer Science at large, such as Auction and Mechanism Design.

In Machine Learning, there has been growing interest in using unlabeled data together with labeled data due to the availability of large amounts of unlabeled data in many applications. As a result, a number of different algorithmic approaches have been developed for this purpose. However, the underlying assumptions of these methods are often quite distinct and not captured by standard theoretical models. This thesis introduces an extension of the PAC model (a standard model for Supervised Learning) to fit Semi-supervised Learning, that can be used to help analyze most of the different approaches taken so far. This includes issues such as "Under what conditions will unlabeled data help and by how much?" and "How much data should I expect to need in order to perform well?". In the context of Kernel methods, this thesis shows how Random Projection techniques can be used to convert a given kernel function into an explicit, distribution dependent, set of features, which can then be fed into more general (not necessarily kernelizable) learning algorithms. In addition, this work shows how such methods can be extended to more general pairwise similarity functions and also gives a formal theory that matches the standard intuition that a good kernel function is one that acts as a good measure of similarity. Another important part of this thesis involves developing algorithms for Active Learning. In particular, this dissertation includes the first Active Learning algorithm which works in the presence of arbitrary forms of noise, as well as a few margin based Active Learning algorithms.

This dissertation also contributes with new connections between Machine Learning and Mechanism Design. Specifically, this thesis presents the first general framework in which machine learning methods can be used for reducing mechanism design problems to standard algorithmic questions for a wide range of revenue maximization problems in an unlimited supply setting. In addition, this dissertation also develops approximations algorithms and online mechanisms for a number of a number of important pricing problems. Some of the online mechanisms presented in this work are based on Online Learning techniques.

Contents

1	Thesis Overview	1
1.1	Incorporating Unlabeled Data in the Learning Process	1
1.2	Similarity Based Learning	2
1.3	Mechanism Design, Machine Learning, and Pricing Problems	3
2	Semi-Supervised and Active Learning	5
2.1	Semi-Supervised Learning	6
2.1.1	A Formal Framework	8
2.1.2	Example of Results in Our Model	10
2.1.3	Related Models	16
2.1.4	Conclusions and Extensions towards Active Learning	17
2.2	Active Learning	18
3	Similarity-based Learning	21
3.1	Preliminaries	22
3.2	Kernels as Features: On Kernels, Margins and Low-Dimensional Mappings	24
3.2.1	Two simple mappings	25
3.2.2	An improved mapping	27
3.3	A General Theory of Learning with Similarity Functions	27
3.3.1	Formal Framework	28
3.3.2	Sufficient Conditions for Learning with Similarity Functions	28
3.3.3	Good Kernels are Good Similarity Functions	33
3.3.4	Perspective	33
4	Mechanism Design, Machine Learning, and Pricing Problems	35
4.1	Generic Random Sampling Based Mechanisms for Revenue Maximization in an Unlim- ited Supply Setting	35
4.1.1	Problem Formulation	35
4.1.2	Definitions	39
4.1.3	Generic Reductions	41
4.1.4	Generic Analyses	42
4.1.5	Structural Risk Minimization	44
4.1.6	Improving the Bounds	45
4.2	Algorithms for Pricing Problems	46

5	Work in Progress, Open Problems, and Future Work	47
5.1	Similarity Based Learning and Clustering	47
5.2	Semi-Supervised Learning and Active Learning	48
5.3	Mechanism Design, Machine Learning, and Pricing Problems	49
	Bibliography	51

Chapter 1

Thesis Overview

This thesis contains two primary thrusts. The first is developing new frameworks for modelling the needs of new machine learning applications and paradigms, with a particular focus on *classification*. The second is developing new connections between Machine Learning Theory and emerging areas of Computer Science, in particular Algorithmic Game Theory.

1.1 Incorporating Unlabeled Data in the Learning Process

Traditionally, machine learning has focused on problems of learning a task from labeled examples only. However, for many contemporary practical problems such as classifying web pages or detecting spam, there is often additional information available; in particular, for many of these settings unlabeled data is often much cheaper and more plentiful than labeled data. As a consequence, there has recently been substantial interest in using unlabeled data together with labeled data for learning [35, 38, 80, 85, 95, 106, 110, 127], since clearly, if useful information can be extracted from it that reduces dependence on labeled examples, this can be a significant benefit [34, 102].

There are currently two settings that have been considered to incorporate unlabeled data in the learning process. The first one is the so-called *Semi-supervised Learning* [51], where, in addition to a set of labeled examples drawn at random from the underlying data distribution, the learning algorithm can also use a (usually larger) set of unlabeled examples from the same distribution. In this setting, unlabeled data becomes informative under *additional* assumptions and beliefs about the learning problem. Examples of such assumptions are the one used by Transductive SVM [85] (namely, that the target function should cut through low density regions of the space), or by Co-training [38] (namely, that the target should be self-consistent in some way). Unlabeled data is then potentially useful in this setting because it allows one to reduce search space from the whole set of hypotheses, down to the set of *a-priori* reasonable with respect to the underlying distribution.

The second setting, an increasingly popular one for the past few years, is *Active Learning* [21, 53, 58]. Here, the learning algorithm has both the capability of drawing random unlabeled examples from the underlying distribution, and that of asking for the labels of *any* of these examples, while the hope is that a good classifier can be learned with significantly fewer labels by *actively* directing the queries to *informative* examples. As opposed to the Semi-supervised learning setting, and similarly to the classical supervised learning settings (PAC and Statistical Learning Theory settings) the only prior belief about the learning problem in the Active Learning setting is that the target function (or a good approximation of it) belongs to a given concept class. Luckily, it turns out that for simple concept classes such as linear separators on the line one can achieve an *exponential* improvement (over the usual supervised

learning setting) in the labeled data sample complexity, under no additional assumptions about the learning problem [21, 53]. In general, however, for more complicated concept classes, the speed-ups achievable in the active learning setting depend on the match between the distribution over example-label pairs and the hypothesis class, and therefore on the target hypothesis in the class. Furthermore, there are simple examples where active learning does not help at all, not even in the realizable case [58].

In our work we study both these settings. For the semi-supervised learning problem, we provide a *unified* PAC-style model that captures many of the ways unlabeled data is typically used, and provides a very general framework for thinking about this issue [15, 18, 51]. This model provides a unified framework for analyzing when and why unlabeled data can help, in which one can discuss both *sample-complexity* and *algorithmic* issues.

Our model can be viewed as an extension of the standard PAC model, where in addition to a concept class C , one also proposes a compatibility function (an abstract prior): a type of compatibility that one believes the target concept should have with the underlying distribution of data. For example, such a belief could be that the target should cut through a low-density region of space, or that it should be self-consistent in some way as in co-training. This belief is then explicitly represented in the model. Unlabeled data is then potentially helpful in this setting because it allows one to estimate compatibility over the space of hypotheses, and to reduce the size of the search space from the whole set of hypotheses C down to those that, according to one's assumptions, are a-priori reasonable with respect to the distribution.

After proposing the model, we analyze sample-complexity issues in this setting: that is, how much of each type of data one should expect to need in order to learn well, and what are the basic quantities that these numbers depend on. We provide examples of sample-complexity bounds both for uniform convergence and ϵ -cover based algorithms, as well as several algorithmic results. We describe in greater detail the high level idea and a few of our results of our extension of the PAC model to fit Semi-Supervised Learning [15, 18, 19, 51] in Chapter 2.1.

For the active learning problem, we propose and analyze a generic version-space based strategy in the purely *agnostic* setting [21]. For details see Chapter 2.2. In recent work we also analyze margin based active learning of linear separators [25].

1.2 Similarity Based Learning

Kernel functions have become an extremely popular tool in contemporary machine learning, with an attractive theory as well [79, 83, 113, 114, 122]. They are used in domains ranging from Computer Vision [78] to Computational Biology [113] to Language and Text Processing [83], with workshops, (e.g. [2, 3, 4, 5]), books [79, 83, 113, 114] [122], and large portions of major conferences (see, e.g., [1]) devoted to kernel methods.

A kernel is a function that takes in two data objects (which could be images, DNA sequences, or points in R^n) and outputs a number, with the property that the function is symmetric and positive-semidefinite. That is, for any kernel K , there must exist an (implicit) mapping ϕ , such that for all inputs x, x' we have $K(x, x') = \phi(x) \cdot \phi(x')$. The kernel is then used inside a “kernelized” learning algorithm such as SVM or kernel-perceptron as the way in which the algorithm interacts with the data. Typical kernel functions for structured data include the polynomial kernel $K(x, x') = (1 + x \cdot x')^d$ and the Gaussian kernel $K(x, x') = e^{-\|x-x'\|^2/2\sigma^2}$, and a number of special-purpose kernels have been developed for sequence data, image data, and other types of data as well [55, 56, 93, 103, 118].

The theory behind kernel functions is based on the fact that many standard algorithms for learning linear separators, such as SVMs and the Perceptron algorithm, can be written so that the only way they interact with their data is via computing dot-products on pairs of examples. Thus, by replacing each

invocation of $x \cdot x'$ with a kernel computation $K(x, x')$, the algorithm behaves exactly as if we had explicitly performed the mapping $\phi(x)$, even though ϕ may be a mapping into a very high-dimensional space. Furthermore, these algorithms have convergence rates that depend only on the *margin* of the best separator, and not on the dimension of the space in which the data resides [12, 116]. Thus, kernel functions are often viewed as providing much of the power of this implicit high-dimensional space, without paying for it computationally (because the ϕ mapping is only implicit) or in terms of sample size (if data is indeed well-separated in that space).

While the above theory is quite elegant, it has a few limitations. First, when designing a kernel function for some learning problem, the intuition typically employed is that a good kernel would be one that serves as a good similarity function for the given problem [113]. On the other hand, the above theory talks about margins in an implicit and possibly very high-dimensional space. So, in this sense the theory is not that helpful for providing intuition when selecting or designing a kernel function. Second, it may be that the most natural similarity function for a given problem is not positive-semidefinite, and it could require substantial work, possibly reducing the quality of the function, to coerce it into a legal form. Finally, from a complexity-theoretic perspective, it is a bit unsatisfying for the explanation of the effectiveness of some algorithm to depend on properties of an implicit high-dimensional mapping that one may not even be able to calculate. In particular, the standard theory at first blush has a “something for nothing” feel to it (all the power of the implicit high-dimensional space without having to pay for it) and perhaps there is a more prosaic explanation of what it is that makes a kernel useful for a given learning problem. For these reasons, it would be helpful to have a theory that was in terms of more tangible quantities.

In our work [22], we show how Random Projection techniques can be used to convert a given kernel function into an explicit, distribution dependent, set of features, which can then be fed into more general (not necessarily kernelizable) learning algorithms. For details see Chapter 3.2. In addition, we also show how such methods can be extended to more *general* pairwise similarity functions and also gives a formal theory that matches the standard intuition that a good kernel function is one that acts as a good measure of similarity [16].

In particular, we define a notion of what it means for a pairwise function $K(x, x')$ to be a “good similarity function” for a given learning problem that (a) does not require the notion of an implicit space and allows for functions that are not positive semi-definite, (b) is provably sufficient for learning, and (c) is broad, in sense that a good kernel in the standard sense (large margin in the implicit ϕ -space) will also satisfy our definition of a good similarity function, though with some loss in the parameters. In this way, we provide the first theory that describes the effectiveness of a given kernel (or more general similarity function) in terms of natural similarity-based properties. For details see Chapter 3.3.

1.3 Mechanism Design, Machine Learning, and Pricing Problems

In this thesis we also present explicit connections between Machine Learning Theory and certain contemporary problems in Economics.

With the Internet developing as the single most important arena for resource sharing among parties with diverse and selfish interests, traditional algorithmic and distributed systems approaches are insufficient. Instead, in protocols for the Internet, game-theoretic and economic issues must be considered. A fundamental research endeavor in this new field is the design and analysis of auction mechanisms and pricing algorithm. In our work [20] we show how Machine Learning methods can be used in the design of auctions and other pricing mechanisms with guarantees on their performance.

In particular, we show how Sample Complexity techniques in Statistical Learning Theory can be used to reduce problems of incentive-compatible mechanism design to standard algorithmic questions, for a

wide range of revenue-maximizing problems in an unlimited supply setting. In doing so, we obtain a unified approach for considering a variety of profit maximizing mechanism design problems, including many that have been previously considered in the literature. In particular, we use techniques from Sample Complexity in machine learning theory, *both for analyzing and designing our mechanisms*. We apply our reductions to a diverse set of revenue maximizing pricing problems, such as the problem of auctioning a digital good, the attribute auction problem, and the problem of item pricing in unlimited supply combinatorial auctions.

Our results *substantially generalize* the previous work on Random Sampling mechanisms (see e.g., [74, 75]) – by both broadening the applicability of such mechanisms and by simplifying the analysis. Also, from a learning perspective, these settings present several unique challenges and particularities: the loss function is discontinuous and asymmetric, and the range of bidders' valuations may be large. We describe in greater detail the high level idea and a few of these reductions in Chapter 4.

Chapter 2

Semi-Supervised and Active Learning

As mentioned in Chapter 1, there are currently two main settings that have been considered to incorporate unlabeled data in the learning process. The first one is the so-called *Semi-supervised Learning* [51], where, in addition to a set of labeled examples drawn at random from the underlying data distribution, the learning algorithm can also use a (usually larger) set of unlabeled examples from the same distribution. In this setting, unlabeled data becomes informative under *additional* assumptions and beliefs about the learning problem.

The second setting, an increasingly popular one for the past few years in the Machine Learning (Theory) community is *Active Learning* [21, 53, 58]. Here, the learning algorithm has both the capability of drawing random unlabeled examples from the underlying distribution and that of asking for the labels of *any* of these examples, and the hope is that a good classifier can be learned with significantly fewer labels by *actively* directing the queries to *informative* examples. As opposed to the Semi-supervised learning setting, and similarly to the classical supervised learning settings (PAC and Statistical Learning Theory settings) the only prior belief about the learning problem in the Active Learning setting is that the target function (or a good approximation of it) belongs to a given concept class. Luckily, it turns out that for simple concept classes such as linear separators on the line one can achieve an *exponential* improvement (over the usual supervised learning setting) in the labeled data sample complexity, under no additional assumptions about the learning problem [21, 53].¹ In general, however, for more complicated concept classes, the speed-ups achievable in the active learning setting depend on the match between the distribution over example-label pairs and the hypothesis class, and therefore on the target hypothesis in the class. Furthermore, there are simple examples where active learning does not help at all, even if there in the realizable case (see, for example, [58]). Recent work of Dasgupta [58] gives a generic characterization of the sample complexity aspect of active learning in the *realizable* case.

In our work we study both these settings. For the semi-supervised learning problem, we provide a *unified* PAC-style model that captures many of the ways unlabeled data is typically used, and provides a very general framework for thinking about this issue [15, 18, 51].

For the active learning problem, we propose and analyze a generic version-space based strategy in the purely *agnostic* setting. We also analyze margin based active learning of linear separators. Specifically, we present an Agnostic Active learning algorithm, A^2 , in [21]. The only assumption we rely upon is that samples are drawn *i.i.d.* from some underlying distribution. In particular, we make no assumptions about the mechanism producing noise (e.g., class/target misfit, fundamental randomization, adversarial situations). This is the first result of this form. In recent work, we also analyze a generic margin based

¹For this simple concept class one can achieve a pure exponential improvement [53] in the realizable case, while in the agnostic case the improvement depends upon the noise rate [21].

active learning algorithm for learning linear separators [25].

2.1 Semi-Supervised Learning

As mentioned in Chapter 1 and in the previous paragraph, given the easy availability of unlabeled data in many settings, there has been growing interest in methods that try to use such data together with the (more expensive) labeled data for learning. A number of *Semi-supervised Learning* techniques have been developed for doing this, along with experimental results on a variety of different learning problems. These include label propagation for word-sense disambiguation [125]; co-training for classifying web pages [38], parsing [80], improving visual detectors [95], and document classification [110]; transductive SVM [85] and EM [106] for text classification; graph-based methods [35, 127] and others. In fact, the problem of learning from labeled and unlabeled data was the topic of a few workshops at the recent ICML conferences (ICML 2003 and ICML 2005).

One difficulty from a theoretical point of view is that standard discriminative learning models do not really capture how and why unlabeled data can be of help. In particular, both in the PAC model [43, 88, 120] and the Statistical Learning Theory framework [122] there is purposefully a complete disconnect between the data distribution D and the target function f being learned. The only prior belief is that f belongs to some class C : even if D is known fully, any function $f \in C$ is still possible. For instance, in the PAC model, it is perfectly natural (and common) to talk about the problem of learning a concept class such as DNF formulas [96, 124] or an intersection of halfspaces [29, 37, 91, 123] over the uniform distribution; but clearly in this case unlabeled data is useless — you can just generate it yourself. For learning over an unknown distribution (as in the the standard PAC and Statistical Learning Theory settings), unlabeled data can help somewhat, by allowing one to use distribution-specific sample-complexity bounds, but this does not seem to fully capture the power of unlabeled data in practice.

In *generative*-model settings, one *can* easily talk theoretically about the use of unlabeled data, e.g., [49, 50]. However, these results typically make strong assumptions that essentially imply that there is only one natural distinction to be made for a given (unlabeled) data distribution. For instance, a typical generative-model setting would be that we assume positive examples are generated by one Gaussian, and negative examples are generated by another Gaussian. In this case, given enough unlabeled data, we could in principle recover the Gaussians and would need labeled data only to tell us which Gaussian is the positive one and which is the negative one.² This is too strong an assumption for most real-world settings. Instead, we would like our model to allow for a distribution over data (e.g., documents we want to classify) where there are a number of plausible distinctions we might want to make.³ In addition, we would like a general framework that can be used to model many different uses of unlabeled data.

In our work [15, 18, 51], we present a PAC-style framework that bridges between these positions and can be used to help think about many of the ways unlabeled data is typically used. We extend the PAC model in a way that allows one to express not only the form of target function one is considering, but also relationships that one hopes the target function and underlying distribution will possess. We then analyze sample-complexity issues in this setting: that is, how much of each type of data one should expect to need in order to learn well, and also give examples of algorithmic results in this model. We emphasize that we derive all the results in an extension of the PAC framework so that we can formally talk about *algorithmic* results and not only about sample complexity or generalization bounds, but the same high level idea can

²[49, 50] do not assume Gaussians in particular, but they do assume the distributions are distinguishable, which from our perspective has the same issue.

³In fact, there has been recent work in the generative model setting on the practical side that goes in this direction [104, 106]. We discuss connections to generative models further in [51].

be used to extend the Statistical Learning Framework too.

Specifically, the idea of the proposed model is to augment the PAC notion of a *concept class*, which is a set of functions (like linear separators or decision trees), with a notion of *compatibility* between a function and the data distribution that we hope the target function will satisfy. Then, rather than talking of “learning a concept class C ,” we will talk of “learning a concept class C under compatibility notion χ .” For example, suppose we believe there should exist a good linear separator, and that furthermore, if the data happens to cluster, then this separator probably does not slice through the middle of any such clusters. Then we would want a compatibility notion that penalizes functions that do, in fact, slice through clusters. In this framework, the extent to which unlabeled data helps depends on two quantities: first, the extent to which the true target function satisfies the given assumption, and second, the extent to which the distribution allows this assumption to rule out alternative hypotheses. For instance, if the data does not cluster at all, then all functions equally satisfy this compatibility notion and the assumption ends up not helping. From a Bayesian perspective, one can think of this as a PAC model for a setting in which one’s prior is not just over functions, but also over how the function and underlying distribution relate to each other.

To make our model formal, we will need to ensure that the degree of compatibility be something that can be estimated from a finite sample. To do this, we will require that the compatibility notion χ actually be a function from $C \times X$ to $[0, 1]$, where the compatibility of a function f with the data distribution D is $\mathbf{E}_{x \sim D}[\chi(f, x)]$. The degree of *incompatibility* is then something we can think of as a kind of “unlabeled error rate” that measures how a-priori unreasonable we believe some proposed hypothesis to be. For instance, in the example above of a “margin-style” compatibility, we could define $\chi(f, x)$ to be an increasing function of the distance of x to the separator f . In this case, the unlabeled error rate, $1 - \chi(f, D)$, is a measure of the probability mass close to the proposed separator. In co-training, where each example x has two “views” ($x = \langle x_1, x_2 \rangle$), the underlying belief is that the true target c^* can be decomposed into functions $\langle c_1^*, c_2^* \rangle$ over each view such that for most examples, $c_1^*(x_1) = c_2^*(x_2)$. In this case, we can define $\chi(\langle f_1, f_2 \rangle, \langle x_1, x_2 \rangle) = 1$ if $f_1(x_1) = f_2(x_2)$, and 0 if $f_1(x_1) \neq f_2(x_2)$. Then the compatibility of a hypothesis $\langle f_1, f_2 \rangle$ with an underlying distribution D is $\Pr_{\langle x_1, x_2 \rangle \sim D}[f_1(x_1) = f_2(x_2)]$.

This setup allows us to analyze the ability of a finite unlabeled sample to reduce our dependence on labeled examples, as a function of the compatibility of the target function (i.e., how correct we were in our assumption) and various measures of the “helpfulness” of the distribution. In particular, in our model, we find that unlabeled data can help in several distinct ways.

- If the target function is highly compatible with D , then if we have enough unlabeled data to estimate compatibility over all $f \in C$, we can in principle reduce the size of the search space from C down to just those $f \in C$ whose estimated compatibility is high. For instance, if D is “helpful”, then the set of such functions will be much smaller than the entire set C .
- By providing an estimate of D , unlabeled data can allow us to use a more refined distribution-specific notion of “hypothesis space size” such as Annealed VC-entropy [63], Rademacher complexities [26, 46, 92] or the size of the smallest ϵ -cover [31], rather than VC-dimension [43, 88]. In fact, for natural cases (such as those above) we find that the sense in which unlabeled data reduces the “size” of the search space is best described in these distribution-specific measures.
- Finally, if the distribution is especially nice, we may find that not only does the set of “compatible” $f \in C$ have a small ϵ -cover, but also the elements of the cover are far apart. In that case, if we assume the target function is fully compatible, we may be able to learn from even fewer labeled examples than the $1/\epsilon$ needed just to *verify* a good hypothesis! (Though here D is effectively committing to the target as in generative models.)

Our framework also allows us to address the issue of how much *unlabeled* data we should expect to

need. Roughly, the “VCdim/ ϵ^2 ” form of standard PAC sample complexity bounds now becomes a bound on the number of *unlabeled* examples we need. However, technically, the set whose VC-dimension we now care about is not C but rather a set defined by both C and χ : that is, the overall complexity depends both on the complexity of C and the complexity of the notion of compatibility (see Section 2.1.2). One consequence of our model is that if the target function and data distribution are both well behaved with respect to the compatibility notion, then the sample-size bounds we get for labeled data can *substantially beat* what one could hope to achieve through pure labeled-data bounds, and we illustrate this with a few different examples in [15, 18].

We present in the following the formal description of our model, a few examples, a few sample complexity results and describe our main algorithmic result. For details, see [15, 18, 51].

2.1.1 A Formal Framework

In this section we formally introduce what we mean by a *notion of compatibility*, and illustrate it through a number of relevant examples including *margins* and *co-training*.

We assume that examples (both labeled and unlabeled) come according to a fixed unknown distribution D over an instance space X , and they are labeled by some unknown target function c^* . As in the standard PAC model, a *concept class* or *hypothesis space* is a set of functions over the instance space X , and we will often make the assumption (the “realizable case”) that the target function belongs to a given class C . For a given hypothesis f , the (true) error rate of f is defined as $err(f) = err_D(f) = \Pr_{x \sim D}[f(x) \neq c^*(x)]$. For any two hypotheses $f_1, f_2 \in C$, the distance with respect to D between f_1 and f_2 is defined as $d(f_1, f_2) = d_D(f_1, f_2) = \Pr_{x \sim D}[f_1(x) \neq f_2(x)]$. We will use $\widehat{err}(f)$ to denote the empirical error rate of f on a given labeled sample and $\widehat{d}(f_1, f_2)$ to denote the empirical distance between f_1 and f_2 on a given unlabeled sample.

We define a *notion of compatibility* to be a mapping from a hypothesis f and a distribution D to $[0, 1]$ indicating how “compatible” f is with D . In order for this to be estimable from a finite sample, we require that compatibility be an expectation over individual examples. (Though one could imagine more general notions with this property as well.) Specifically, we define:

Definition 2.1.1 A legal notion of compatibility is a function $\chi : C \times X \rightarrow [0, 1]$ where we (overloading notation) define $\chi(f, D) = \mathbf{E}_{x \sim D}[\chi(f, x)]$. Given a sample S , we define $\chi(f, S)$ to be the empirical average over the sample.

Note 1 One could also allow compatibility functions over k -tuples of examples, in which case our (unlabeled) sample-complexity bounds would simply increase by a factor of k . For settings in which D is actually known in advance (e.g., transductive learning, see Section 2.1.3) we can drop this requirement entirely and allow any notion of compatibility $\chi(f, D)$ to be legal.

Definition 2.1.2 Given compatibility notion χ , the incompatibility of f with D is $1 - \chi(f, D)$. We will also call this its unlabeled error rate, $err_{unl}(f)$, when χ and D are clear from context. For a given sample S , we use $\widehat{err}_{unl}(f)$ to denote the empirical average over S .

Finally, we also need a notation for the set of functions whose incompatibility is at most some given value τ .

Definition 2.1.3 Given threshold τ , we define $C_{D,\chi}(\tau) = \{f \in C : err_{unl}(f) \leq \tau\}$. So, e.g., $C_{D,\chi}(1) = C$. Similarly, for a sample S , we define $C_{S,\chi}(\tau) = \{f \in C : \widehat{err}_{unl}(f) \leq \tau\}$

We now give several examples to illustrate this framework:

Example 1. Suppose examples are points in R^d and C is the class of linear separators. A natural belief in this setting is that data should be “well-separated”: not only should the target function separate the

positive and negative examples, but it should do so by some reasonable *margin* γ . This is the assumption used by Transductive SVM [85]. In this case, if we are given γ up front, we could define $\chi(f, x) = 1$ if x is farther than distance γ from the hyperplane defined by f , and $\chi(f, x) = 0$ otherwise. So, the incompatibility of f with D is probability mass within distance γ of $f \cdot x = 0$. Or we could define $\chi(f, x)$ to be a smooth function of the distance of x to the separator, if we do not want to commit to a specific γ in advance. (In contrast, defining compatibility of a hypothesis based on the largest γ such that D has probability mass *exactly zero* within distance γ of the separator would *not* fit our model: it cannot be written as an expectation over individual examples and indeed would not be a good definition since one cannot distinguish “zero” from “exponentially close to zero” from a small sample of unlabeled data).

Example 2. In co-training [38], we assume examples come as pairs $\langle x_1, x_2 \rangle$, and our goal is to learn a pair of functions $\langle f_1, f_2 \rangle$. For instance, if our goal is to classify web pages, x_1 might represent the words on the page itself and x_2 the words attached to links pointing *to* this page from other pages. The hope that underlies co-training is that the two parts of the example are consistent, which then allows the co-training algorithm to bootstrap from unlabeled data. For example, *iterative co-training* uses a small amount of labeled data to get some initial information (e.g., if a link with the words “my advisor” points to a page then that page is probably a faculty member’s home page) and then when it finds an unlabeled example where one half is confident (e.g., the link says “my advisor”), it uses that to label the example for training its hypothesis over the other half. This approach and several variants have been used for a variety of learning problems, including named entity classification [54], text classification [70, 105], natural language processing [111], large scale document classification [109], and visual detectors [95]. As mentioned the assumptions underlying co-training fit naturally into our framework. In particular, we can define the incompatibility of some hypothesis $\langle f_1, f_2 \rangle$ with distribution D as $\Pr_{\langle x_1, x_2 \rangle \sim D} [f_1(x_1) \neq f_2(x_2)]$.

Example 3. In transductive graph-based methods, we are given a set of unlabeled examples connected in a graph g , where the interpretation of an edge is that we believe the two endpoints of the edge should have the *same* label. Given a few labeled vertices, various graph-based methods then attempt to use them to infer labels for the remaining points. If we are willing to view D as a distribution over *edges* (a uniform distribution if g is unweighted), then as in co-training we can define the incompatibility of some hypothesis f as the probability mass of edges that are cut by f , which then motivates various cut-based algorithms. For instance, if we require f to be boolean, then the mincut method of [35] finds the most-compatible hypothesis consistent with the labeled data; if we allow f to be fractional and define $1 - \chi(f, \langle x_1, x_2 \rangle) = (f(x_1) - f(x_2))^2$, then the algorithm of [127] finds the most-compatible consistent hypothesis. If we do not wish to view D as a distribution over edges, we could have D be a distribution over *vertices* and broaden Definition 2.1.1 to allow for χ to be a function over *pairs* of examples. In fact, as mentioned in Note 1, since we have perfect knowledge of D in this setting we can allow any compatibility function $\chi(f, D)$ to be legal. We discuss more connections with graph-based methods in Section 2.1.3.

Example 4. As a special case of co-training, suppose examples are pairs of points in R^d , C is the class of linear separators, and we believe the two points in each pair should both be on the *same* side of the target function. (So, this is a version of co-training where we require $f_1 = f_2$.) The motivation is that we want to use pairwise information as in Example 3, but we also want to use the features of each data point. For instance, in the word-sense disambiguation problem studied by [125], the goal is to determine which of several dictionary definitions is intended for some target word in a piece of text (e.g., is “plant” being used to indicate a tree or a factory?). The local context around each word can be viewed as placing

it into R^d , but the edges correspond to a completely different type of information: the belief that if a word appears twice in the same document, it is probably being used in the *same* sense both times. In this setting, we could use the same compatibility function as in Example 3, but rather than having the concept class C be all possible functions, we reduce C to just linear separators.

Example 5. In a related setting to co-training, considered by [94], examples are single points in X but we have a pair of hypothesis spaces $\langle C_1, C_2 \rangle$ (or more generally a k -tuple $\langle C_1, \dots, C_k \rangle$), and the goal is to find a pair of hypotheses $\langle f_1, f_2 \rangle \in C_1 \times C_2$ with low error over labeled data and that agree over the distribution. For instance, if data is sufficiently “well-separated”, one might expect there to exist both a good linear separator and a good decision tree, and one would like to use this assumption to reduce the need for labeled data. In this case one could define compatibility of $\langle f_1, f_2 \rangle$ with D as $\Pr_{x \sim D}[f_1(x) = f_2(x)]$, or the similar notion given in [94, 112, 117].

2.1.2 Example of Results in Our Model

We now present several examples of results in our model. We start by describing examples of sample-complexity bounds that fall out of this framework, showing how unlabeled data, together with a suitable compatibility notion, can reduce the need for labeled examples. We do not focus on giving the tightest possible bounds, but we instead focus more on the types of bounds and the type of quantities on which they depend, in order to better understand what it is about the learning *problem* one can hope to leverage from via unlabeled data. We finally end this section by describing our main *algorithmic* result in this model [15, 18, 51].

A few Sample Complexity Results

The basic structure of all of these results is as follows. First, given enough unlabeled data (where “enough” will be a function of some measure of the complexity of C and possibly of χ as well), we can uniformly estimate the true compatibilities of all functions in C by their empirical compatibilities over the sample. Then, by using this quantity to give a preference ordering over the functions in C , we can reduce “ C ” down to “the set of functions in C whose compatibility is not much larger than the true target function” in bounds for the number of *labeled* examples needed for learning. The specific bounds differ in terms of the exact complexity measures used (and a few other issues such as stratification and realizability) and we provide examples illustrating when and how certain complexity measures can be significantly more powerful than others.

Uniform Convergence Bounds

We begin with uniform convergence bounds (later in Section 2.1.2 we give tighter ϵ -cover bounds that apply to algorithms of a particular form). For clarity, we begin with the case of finite hypothesis spaces where we measure the “size” of a set of functions by just the number of functions in the set. We then discuss several issues that arise when considering infinite hypothesis spaces, such as what is an appropriate measure for the “size” of the set of compatible functions, and the need to account for the complexity of the compatibility notion itself. Note that in the standard PAC model, one typically talks of either the realizable case, where we assume that $c^* \in C$, or the agnostic case where we do not [88]. In our setting, we have the additional issue of *unlabeled* error rate, and can either make an a-priori assumption that the target function’s unlabeled error is low, or else aim for a more “Occam-style” bound in which we have a stream of labeled examples and halt once they are sufficient to justify the hypothesis produced.

Finite hypothesis spaces We first give a bound for the “doubly realizable” case.

Theorem 2.1.1 *If we see m_u unlabeled examples and m_l labeled examples, where*

$$m_u \geq \frac{1}{\epsilon} \left[\ln |C| + \ln \frac{2}{\delta} \right] \quad \text{and} \quad m_l \geq \frac{1}{\epsilon} \left[\ln |C_{D,\chi}(\epsilon)| + \ln \frac{2}{\delta} \right],$$

then with probability at least $1 - \delta$, all $f \in C$ with $\widehat{err}(f) = 0$ and $\widehat{err}_{unl}(f) = 0$ have $err(f) \leq \epsilon$.

Proof: The probability that a given hypothesis f with $err_{unl}(f) > \epsilon$ has $\widehat{err}_{unl}(f) = 0$ is at most $(1 - \epsilon)^{m_u} < \frac{\delta}{2|C|}$ for the given value of m_u . Therefore, by the union bound, the number of unlabeled examples is sufficient to ensure that with probability $1 - \frac{\delta}{2}$, only hypotheses in $C_{D,\chi}(\epsilon)$ have $\widehat{err}_{unl}(f) = 0$. The number of labeled examples then similarly ensures that with probability $1 - \frac{\delta}{2}$, none of those whose true error is at least ϵ have an empirical error of 0, yielding the theorem. ■

Interpretation: So, if the target function indeed is perfectly correct and compatible, then Theorem 2.1.1 gives sufficient conditions on the number of examples needed to ensure that an algorithm that optimizes both quantities over the observed data will, in fact, achieve a PAC guarantee. To emphasize this, we will say that an algorithm efficiently PAC_{unl} -learns the pair (C, χ) if it is able to achieve a PAC guarantee using time and sample sizes polynomial in the bounds of Theorem 2.1.1.

We can think of Theorem 2.1.1 as bounding the number of labeled examples we need as a function of the “helpfulness” of the distribution D with respect to our notion of compatibility. That is, in our context, a helpful distribution is one in which $C_{D,\chi}(\epsilon)$ is small, and so we do not need much labeled data to identify a good function among them. We can get a similar bound in the situation when the target function is not fully compatible:

Theorem 2.1.2 *Given $t \in [0, 1]$, if we see m_u unlabeled examples and m_l labeled examples, where*

$$m_u \geq \frac{2}{\epsilon^2} \left[\ln |C| + \ln \frac{4}{\delta} \right] \quad \text{and} \quad m_l \geq \frac{1}{\epsilon} \left[\ln |C_{D,\chi}(t + 2\epsilon)| + \ln \frac{2}{\delta} \right],$$

then with probability at least $1 - \delta$, all $f \in C$ with $\widehat{err}(f) = 0$ and $\widehat{err}_{unl}(f) \leq t + \epsilon$ have $err(f) \leq \epsilon$, and furthermore all $f \in C$ with $err_{unl}(f) \leq t$ have $\widehat{err}_{unl}(f) \leq t + \epsilon$.

In particular, this implies that if $err_{unl}(c^*) \leq t$ and $err(c^*) = 0$ then with high probability the $f \in C$ that optimizes $\widehat{err}(f)$ and $\widehat{err}_{unl}(f)$ has $err(f) \leq \epsilon$.

Proof: Same as Theorem 2.1.1 except apply Hoeffding bounds [63] to the unlabeled error rates. ■

Finally, we give a simple Occam/luckiness type of bound for this setting. Given a sample S , let us define $\text{desc}_S(f) = \ln |C_{S,\chi}(\widehat{err}_{unl}(f))|$. That is, $\text{desc}_S(f)$ is the description length of f (in “nats”) if we sort hypotheses by their empirical compatibility and output the index of f in this ordering. Similarly, define $\epsilon\text{-desc}_D(f) = \ln |C_{D,\chi}(err_{unl}(f) + \epsilon)|$. This is an upper-bound on the description length of f if we sort hypotheses by an ϵ -approximation to their true compatibility. Then we can get a bound as follows:

Theorem 2.1.3 *For any set S of unlabeled data, given m_l labeled examples, with probability at least $1 - \delta$, all $f \in C$ satisfying $\widehat{err}(f) = 0$ and $\text{desc}_S(f) \leq \epsilon m_l - \ln(1/\delta)$ have $err(f) \leq \epsilon$. Furthermore, if $|S| \geq \frac{2}{\epsilon^2} [\ln |C| + \ln \frac{2}{\delta}]$, then with probability at least $1 - \delta$, all $f \in C$ satisfy $\text{desc}_S(f) \leq \epsilon\text{-desc}_D(f)$.*

Interpretation: The point of this theorem is that an algorithm can use observable quantities to determine if it can be confident. Furthermore, if we have enough unlabeled data, the observable quantities will be no worse than if we were learning a slightly less compatible function using an infinite-size unlabeled sample.

Note that if we begin with a non-distribution-dependent ordering of hypotheses, inducing some description length $\text{desc}(f)$, and our compatibility assumptions turn out to be wrong, then it could well be that $\text{desc}_D(c^*) > \text{desc}(c^*)$. In this case our use of unlabeled data would end up hurting rather than helping.

Infinite hypothesis spaces To reduce notation, we will assume in the rest of this chapter that $\chi(f, x) \in \{0, 1\}$ so that $\chi(f, D) = \Pr_{x \sim D}[\chi(f, x) = 1]$. However, all our sample complexity results can be easily extended to the general case.

For infinite hypothesis spaces, the first issue that arises is that in order to achieve uniform convergence of *unlabeled* error rates, the set whose complexity we care about is not C but rather $\chi(C) = \{\chi_f : f \in C\}$ where we define $\chi_f(x) = \chi(f, x)$. For instance, suppose examples are just points on the line, and $C = \{f_a(x) : f_a(x) = 1 \text{ iff } x \leq a\}$. In this case, $\text{VCdim}(C) = 1$. However, we could imagine a compatibility function such that $\chi(f_a, x)$ depends on some complicated relationship between the real numbers a and x . In this case, $\text{VCdim}(\chi(C))$ is much larger, and indeed we would need many more unlabeled examples to estimate compatibility over all of C .

A second issue is that we need an appropriate measure for the “size” of the set of surviving functions. VC-dimension tends not to be a good choice: for instance, if we consider the case of Example 1 (margins), then even if data is concentrated in two well-separated “blobs”, the set of compatible separators still has as large a VC-dimension as the entire class even though they are all very similar with respect to D . Instead, it is better to consider distribution dependent complexity measures such as annealed VC-entropy or Rademacher averages. For this we introduce some notation. Specifically, for any C , we denote by $C[m, D]$ the expected number of splits of m points (drawn i.i.d.) from D with concepts in C . Also, for a given (fixed) $S \subseteq X$, we will denote by \bar{S} the uniform distribution over S , and by $C[m, \bar{S}]$ the expected number of splits of m points (drawn i.i.d.) from \bar{S} with concepts in C . Then we can get bounds as follows:

Theorem 2.1.4 *An unlabeled sample of size*

$$m_u = O\left(\frac{\text{VCdim}(\chi(C))}{\epsilon^2} \log \frac{1}{\epsilon} + \frac{1}{\epsilon^2} \log \frac{2}{\delta}\right)$$

and a labeled sample of size

$$m_l > \frac{2}{\epsilon} \left[\log(2s) + \log \frac{2}{\delta} \right], \text{ where } s = C_{D, \chi}(t + 2\epsilon)[2m_l, D]$$

(i.e., s is the expected number of splits of $2m_l$ points drawn from D using concepts in C of unlabeled error rate $\leq t + 2\epsilon$) is sufficient so that with probability $1 - \delta$, all $f \in C$ with $\widehat{err}(f) = 0$ and $\widehat{err}_{unl}(f) \leq t + \epsilon$ have $err(f) \leq \epsilon$, and furthermore all $f \in C$ have $|err_{unl}(f) - \widehat{err}_{unl}(f)| \leq \epsilon$.

This is the analog of Theorem 2.1.2 for the infinite case. In particular, this implies that if $err(c^*) = 0$ and $err_{unl}(c^*) \leq t$, then with high probability the $f \in C$ that optimizes $\widehat{err}(f)$ and $\widehat{err}_{unl}(f)$ has $err(f) \leq \epsilon$.

Proof: By standard VC-bounds [63, 122], the number of unlabeled examples is sufficient to ensure that with probability $1 - \frac{\delta}{2}$ we can estimate, within ϵ , $\Pr_{x \in D}[\chi_f(x) = 1]$ for all $\chi_f \in \chi(C)$. Since $\chi_f(x) = \chi(f, x)$, this implies we can estimate, within ϵ , the unlabeled error rate $err_{unl}(f)$ for all $f \in C$, and so the set of hypotheses with $\widehat{err}_{unl}(f) \leq t + \epsilon$ is contained in $C_{D, \chi}(t + 2\epsilon)$.

The bound on the number of labeled examples follows from [63] (where it is shown that the expected number of partitions can be used instead of the maximum in the standard VC proof). This bound ensures that with probability $1 - \frac{\delta}{2}$, none of the functions in $C_{D, \chi}(t + 2\epsilon)$ whose true (labeled) error is at least ϵ have an empirical (labeled) error of 0. ■

We can also give a bound where we specify the number of labeled examples as a function of the *unlabeled sample*; this is useful because we can imagine our learning algorithm performing some calculations over the unlabeled data and then deciding how many labeled examples to purchase.

Theorem 2.1.5 *Given $t \geq 0$, an unlabeled sample S of size*

$$O\left(\frac{\max[\text{VCdim}(C), \text{VCdim}(\chi(C))]}{\epsilon^2} \log \frac{1}{\epsilon} + \frac{1}{\epsilon^2} \log \frac{2}{\delta}\right)$$

is sufficient so that if we label m_l examples drawn uniformly at random from S , where

$$m_l > \frac{4}{\epsilon} \left[\log(2s) + \log \frac{2}{\delta} \right] \quad \text{and} \quad s = C_{S,\chi}(t + \epsilon) [2m_l, \bar{S}]$$

then with probability $\geq 1 - \delta$, all $f \in C$ with $\widehat{err}(f) = 0$ and $\widehat{err}_{unl}(f) \leq t + \epsilon$ have $err(f) \leq \epsilon$. Furthermore all $f \in C$ have $|err_{unl}(f) - \widehat{err}_{unl}(f)| \leq \epsilon$.

Proof: Standard VC-bounds (in the same form as for Theorem 2.1.4) imply that the number of *labeled* examples m_l is sufficient to guarantee the conclusion of the theorem with “ $err(f)$ ” replaced by “ $err_{\bar{S}}(f)$ ” (the error with respect to \bar{S}) and “ ϵ ” replaced with “ $\epsilon/2$ ”. The number of *unlabeled* examples is enough to ensure that, with probability $\geq 1 - \frac{\delta}{2}$, for all $f \in C$, $|err(f) - err_{\bar{S}}(f)| \leq \epsilon/2$. Combining these two statements yields the theorem. ■

So, if $err(c^*) = 0$ and $err_{unl}(c^*) \leq t$, then with high probability the $f \in C$ that optimizes $\widehat{err}(f)$ and $\widehat{err}_{unl}(f)$ has $err(f) \leq \epsilon$. If we assume $err_{unl}(c^*) = 0$ then we can use $C_{S,\chi}(0)$ instead of $C_{S,\chi}(t + \epsilon)$.

Remark: Notice that for the case of Example 1, in the worst case (over distributions D) this will essentially recover the standard margin sample-complexity bounds. In particular, $C_{S,\chi}(0)$ contains only those separators that split S with margin $\geq \gamma$, and therefore, s is no greater than the maximum number of ways of splitting $2m_l$ points with margin γ . However, if the distribution is nice, then the bounds can be much better because there may be many fewer ways of splitting S with margin γ . For instance, in the case of two well-separated “blobs” discussed above, if S is large enough, we would have just $s = 4$.

We finally give a stratified version of Theorem 2.1.5 as follows:

Theorem 2.1.6 *An unlabeled sample S of size*

$$O \left(\frac{\max[VCdim(C), VCdim(\chi(C))]}{\epsilon^2} \log \frac{1}{\epsilon} + \frac{1}{\epsilon^2} \log \frac{2}{\delta} \right)$$

is sufficient so that with probability $\geq 1 - \delta$ we have that simultaneously for every $k \geq 0$ the following is true: if we label m_k examples drawn uniformly at random from S , where

$$m_k > \frac{4}{\epsilon} \left[\log(2s) + \log \frac{2(k+1)(k+2)}{\delta} \right] \quad \text{and} \quad s = C_{S,\chi}((k+1)\epsilon) [2m_k, \bar{S}]$$

then all $f \in C$ with $\widehat{err}(f) = 0$ and $\widehat{err}_{unl}(f) \leq (k+1)\epsilon$ have $err(f) \leq \epsilon$.

Interpretation: This theorem is an analog of Theorem 2.1.3 and it essentially justifies a stratification based on the estimated unlabeled error rates. Since $k \leq \frac{1}{\epsilon}$, we clearly have a fallback property: the number of labeled examples required is never much worse than the number of labeled examples required by a standard supervised learning algorithm.

We can also imagine having data dependent bounds for both labeled and unlabeled data, and also doing a double stratification, with respect to both labeled and unlabeled error rates. In particular, we can derive a bound as follows:

Theorem 2.1.7 *An unlabeled sample S of size*

$$O \left(\frac{\max[VCdim(C), VCdim(\chi(C))]}{\epsilon^2} \log \frac{1}{\epsilon} + \frac{1}{\epsilon^2} \log \frac{2}{\delta} \right)$$

is sufficient so that with probability $\geq 1 - \delta$ we have that simultaneously for every $i \geq 0$, $k \geq 0$ the following is true: if we label $m_{k,i}$ examples drawn uniformly at random from S , where

$$m_{k,i} > \frac{8}{\epsilon^2} \left[\log(2s) + \log \frac{4(k+1)(k+2)(i+1)(i+2)}{\delta} \right] \quad \text{and} \quad s = C_{S,\chi}((k+1)\epsilon) [2m_{k,i}, \bar{S}]$$

then all $f \in C$ with $\widehat{err}(f) \leq (i+1)\epsilon$ and $\widehat{err}_{unl}(f) \leq (k+1)\epsilon$ have $err(f) \leq (i+2) \cdot \epsilon$.

One could further derive tighter bounds, both in terms of labeled and unlabeled examples, by using more recent stronger concentration results [45, 46]. As mentioned already, our focus here is not on giving the tightest possible bounds, but we instead focus more on the types of bounds and the type of quantities on which they depend, in order to better understand what it is about the learning *problem* one can hope to leverage from via unlabeled data.

ϵ -Cover-based Bounds

The bounds in the previous section are for uniform convergence: they provide guarantees for *any* algorithm that optimizes well on the observed data. In this section, we consider stronger bounds based on ϵ -covers that can be obtained for algorithms that behave in a specific way: they first use the unlabeled examples to choose a “representative” set of compatible hypotheses, and then use the labeled sample to choose among these. Bounds based on ϵ -covers exist in the classical PAC setting, but in our framework these bounds and algorithms of this type are *especially natural and convenient*.

Recall that a set $C_\epsilon \subseteq 2^X$ is an ϵ -cover for C with respect to D if for every $f \in C$ there is a $f' \in C_\epsilon$ which is ϵ -close to f . That is, $\Pr_{x \sim D}(f(x) \neq f'(x)) \leq \epsilon$.

To illustrate how this can produce stronger bounds, consider the setting of Example 3 (graph-based algorithms) where the graph g consists of two cliques of $n/2$ vertices, connected together by $o(n^2)$ edges (in particular, the number of edges connecting the cliques is small compared to ϵn^2). Suppose the target function labels one of the cliques as positive and one as negative, and we define compatibility of a hypothesis to be the fraction of edges in g that are cut by it (so the target function indeed has unlabeled error rate less than ϵ). Now, given any set S_L of $m_l \ll \epsilon n$ labeled examples, there is always a highly-compatible hypothesis consistent with S_L that just separates the positive points in S_L from the entire rest of the graph: the number of edges cut will be at most $nm_l \ll \epsilon n^2$. However, such a hypothesis clearly has high true error since it is so unbalanced. So, we do not have uniform convergence. On the other hand, the set of functions of unlabeled error rate less than $\epsilon/4$ has a small ϵ -cover: in particular, *any* partition of g that cuts less than $\epsilon n^2/4$ edges must be ϵ -close to (a) the all-positive function, (b) the all-negative function, (c) the target function c^* , or (d) the complement of the target function $1 - c^*$. So, ϵ -cover bounds act as if the concept class had only 4 functions, and so require only a constant number of labeled examples.⁴

For another case where ϵ -cover bounds can beat uniform-convergence bounds, imagine examples are *pairs* of points in $\{0, 1\}^d$, C is the class of linear separators, and compatibility is determined by whether both points are on the same side of the separator (i.e., the case of Example 4). Now suppose for simplicity that the target function just splits the hypercube on the first coordinate, and the distribution is uniform over pairs having the same first coordinate (so the target is fully compatible). It is not hard to show that given polynomially many unlabeled examples S_U and $\frac{1}{4} \log d$ labeled examples S_L , with high probability there will exist high-error functions consistent with S_L and compatible with S_U .⁵ So, we do not yet have uniform convergence. In contrast, the cover-size of the set of functions compatible with S_U is constant, so ϵ -cover based bounds again allow learning from just a constant number of labeled examples.

In particular, we can give an ϵ -cover based bound as follows.

⁴Effectively, ϵ -cover bounds allow one to rule out a hypothesis that, say, just separates the positive points in S_L from the rest of the graph by noting that this hypothesis is very close (with respect to D) to the all-negative hypothesis, and *that* hypothesis has a high labeled-error rate.

⁵Proof: Let V be the set of all variables that (a) appear in *every* positive example of S_L and (b) appear in *no* negative example of S_L . Over the draw of S_L , each variable has a $(1/2)^{|S_L|} = 1/\sqrt{d}$ chance of belonging to V , so with high probability V has size at least $\frac{1}{2}\sqrt{d}$. Now, consider the hypothesis corresponding to the conjunction of all variables in V . This correctly classifies the examples in S_L , and w.h.p. it classifies *every* other example in S_U negative because each example in S_U has only a $1/2^{|V|}$ chance of satisfying every variable in V , and the size of S_U is much less than $2^{|V|}$. So, this means it is compatible with S_U and consistent with S_L , even though its true error is high.

Theorem 2.1.8 *If t is an upper bound for $err_{unl}(c^*)$ and p is the size of a minimum ϵ -cover for $C_{D,\chi}(t + 4\epsilon)$, then using m_u unlabeled examples and m_l labeled examples for*

$$m_u = O\left(\frac{VCdim(\chi(C))}{\epsilon^2} \log \frac{1}{\epsilon} + \frac{1}{\epsilon^2} \log \frac{2}{\delta}\right) \quad \text{and} \quad m_l = O\left(\frac{1}{\epsilon} \ln \frac{p}{\delta}\right),$$

we can with probability $1 - \delta$ identify a hypothesis which is 10ϵ close to c^ .*

Proof: See [15, 18]. ■

As an interesting case where unlabeled data helps substantially, consider a co-training setting where the target c^* is fully compatible and D satisfies the conditional independence given the label property. As shown by [38], one can boost any weak hypothesis from unlabeled data in this setting (assuming one has enough labeled data to produce a weak hypothesis). Related sample complexity results are given in [60]. We can actually show that given enough unlabeled data, in fact we can learn from just a *single* labeled example. Specifically, it is possible to show that for any concept classes C_1 and C_2 , we have:

Theorem 2.1.9 *Assume that $err(c^*) = err_{unl}(c^*) = 0$ and D satisfies independence given the label. Then using m_u unlabeled examples and m_l labeled examples we can find a hypothesis that with probability $1 - \delta$ has error at most ϵ , provided that*

$$m_u = O\left(\frac{1}{\epsilon} \cdot \left[VCdim(C_1) + VCdim(C_2)\right] \cdot \ln\left(\frac{1}{\epsilon}\right) + \ln\left(\frac{1}{\delta}\right)\right)$$

and

$$m_l = O\left(\log_{\left(\frac{1}{\epsilon}\right)}\left(\frac{1}{\delta}\right)\right)$$

Proof: See [15, 18]. ■

It is worth noting that, his result can be extended to the case considered in [19] that D^+ and D^- merely satisfy constant expansion.

This example illustrates that if data is especially well behaved with respect to the compatibility notion, then our bounds on labeled data can be *extremely good*.

For the case of linear separators and independence given the label, we can give *efficient* algorithms, achieving the bounds in Theorem 2.1.9 in terms of labeled examples by a polynomial time algorithm (for details see the “Algorithmic Results” Section). Note, however, that both these bounds rely heavily on the assumption that the target is fully compatible. If the assumption is more of a “hope” than a belief, then one would need additional labeled examples just to validate the hypothesis produced.

Algorithmic Results

In our work, we also give several examples of *efficient* algorithms in our model.

Our main algorithmic result, which we briefly describe here, is for a case of co-training where the hypothesis class is the class of linear separators. For simplicity we focus first on the case of Example 4: the target function is a linear separator in R^d and each example is a *pair* of points, both of which are assumed to be on the same side of the separator (i.e., an example is a line-segment that does not cross the target hyperplane).⁶

Clearly, a natural approach for attacking the algorithmic problem is to try to solve the “consistency” problem: given a set of labeled and unlabeled data, our goal is to find a separator that is consistent with the labeled examples and compatible with the unlabeled ones (i.e., it gets the labeled data correct and doesn’t

⁶ We show in [51] how our results can be extended to the more general setting.

cut too many edges). Unfortunately, this consistency problem is NP-hard: given a graph g embedded in R^d with two distinguished points s and t , it is NP-hard to find the linear separator that cuts the minimum number of edges, *even if the minimum is 0* [65]. For this reason, we will make an additional assumption, that the two points in an example are each drawn *independently given the label*. That is, there is a single distribution \overline{D} over R^d , and with some probability p_+ , two points are drawn i.i.d. from \overline{D}_+ (\overline{D} restricted to the positive side of the target function) and with probability $1 - p_+$, the two are drawn i.i.d. from \overline{D}_- (\overline{D} restricted to the negative side of the target function). Note that our sample complexity results in Section 2.1.2 extend to weaker assumptions such as distributional expansion introduced by [19], but we need true independence for our algorithmic results. [38] have also given positive algorithmic results for co-training when (a) the two halves of an example are drawn independently given the label (which we are assuming now), (b) the underlying function is learnable via Statistical Query algorithms⁷ (which is true for linear separators [39]), and (c) we have enough labeled data to produce a weakly-useful hypothesis (defined below) on one of the halves to begin with. We present [51] an improvement over that result by showing how we can run the algorithm in [38] with only *a single* labeled example, thus obtaining an efficient algorithm in our model. It is worth noticing that in the process, we also simplify the results of [39] somewhat.

2.1.3 Related Models

We briefly describe here a transductive analog of our model, as well as the relationship between our model and the *Luckiness Framework*.

A Transductive Analog of our Model

We can also talk about a transductive analog of our (inductive) model, that incorporates many of the existing transductive methods for learning with labeled and unlabeled data. In a transductive setting one assumes that the unlabeled sample S is given, a random small subset is labeled, and the goal is to predict well on the rest of S . In order to make use of unlabeled examples, we will again express the relationship we hope the target function has with the distribution through a compatibility notion χ . However, since in this case the compatibility between a given hypothesis and D is completely determined by S (which is known), we will not need to require that compatibility be an expectation over unlabeled examples. Given this setup, from the sample complexity point of view we only care about how much labeled data we need, and algorithmically we need to find a highly compatible hypothesis with low error on the labeled data.

Rather than presenting general theorems, we instead focus on the modeling aspect and give here several examples in the context of graph-based semi-supervised algorithms for binary classification. In these methods one usually assumes that there is weighted graph g defined over S , which is given a-priori and encodes the prior knowledge. In the following we denote by W the weighted adjacency matrix of g and by C_S the set of all binary functions over S .

Minimum cut: Suppose for $f \in C_S$ we define the incompatibility of f to be the weight of the cut in g determined by f . This is the implicit notion of compatibility considered in [35], and algorithmically the goal is to find the most compatible hypothesis that gets the labeled data correct, which can be solved efficiently using network flow. From a sample-complexity point of view, the number of labeled examples we need is proportional to the VC-dimension of the class of hypotheses that are at least as compatible as the target function, which is known to be $O\left(\frac{k}{\lambda}\right)$ [89], [90], where k is the number of edges cut by c^* and λ is the size of the global minimum cut in the graph. Also note that

⁷For a detailed description of the Statistical Query model see [87] and [88].

the Randomized Mincut algorithm (considered by [41]), which is an extension of the basic mincut approach can be viewed as motivated by a PAC-Bayes sample complexity analysis of the problem.

Normalized Cut: Consider the normalized cut setting of [84] and for $f \in C_S$ define $size(f)$ to be the weight of the cut in g determined by f , and let f_{neg} and f_{pos} be the number of points in S on which h predicts negative and positive, respectively. For $f \in C_S$, define the incompatibility of f to be $\frac{size(f)}{f_{neg} \cdot f_{pos}}$. Note that this is the implicit compatibility function used in [84], and again, algorithmically the goal would be to find a highly compatible hypothesis that gets the labeled data correct. Unfortunately, the corresponding optimization problem in this case is NP-hard. Still, several approximate solutions have been considered, leading to different semi-supervised learning algorithms. For instance, [84] considers a spectral relaxation that leads to the ‘‘SGT algorithm’’; another relaxation based on Semi-Definite programming is considered by [32].

Harmonic Function: We can also model the algorithms introduced in [127] as follows. If we consider f to be a probabilistic prediction function defined over S , then the incompatibility of f is given by $\sum_{i,j} w_{i,j} (f(i) - f(j))^2 = f^T L f$, where L is the un-normalized Laplacian of g . Similarly we can model the algorithm introduced by [126] by noticing that the incompatibility of f is given by $f^T \mathcal{L} f$ where \mathcal{L} is the normalized Laplacian of g . More generally, all the Graph Kernel methods can be viewed in our framework if we consider that the incompatibility of f is given by $\|f\|_K = f^T K f$ where K is a kernel derived from the graph (see for instance [128]).

Connections to the Luckiness Framework

It is worth noticing that there is a strong connection between our approach and the luckiness framework [99, 115]. In both cases, the idea is to define an ordering of hypotheses that depends on the data, in the hope that we will be ‘‘lucky’’ and find that not too many other functions are as compatible as the target. There are two main differences, however. The first is that the luckiness framework (being designed for supervised learning only) uses labeled data both for estimating compatibility and for learning: this is a more difficult task, and as a result our bounds on labeled data can be significantly better. For instance, in Example 4 described in Section 2.1.1, for any non-degenerate distribution, a dataset of $\frac{d}{2}$ pairs can with probability 1 be completely shattered by fully-compatible hypotheses, so the luckiness framework does not help. In contrast, with a larger (unlabeled) sample, one can potentially reduce the space of compatible functions quite significantly, and learn from $o(d)$ or even $O(1)$ labeled examples depending on the distribution – see Section 2.1.2 and 2.1.2. Secondly, the luckiness framework talks about compatibility between a hypothesis and a *sample*, whereas we define compatibility with respect to a distribution. This allows us to talk about the amount of unlabeled data needed to estimate true compatibility. There are also a number of differences at the technical level of the definitions.

2.1.4 Conclusions and Extensions towards Active Learning

Given the easy availability of unlabeled data in many settings, there has been growing interest in methods that try to use such data together with the (more expensive) labeled data for learning. Nonetheless, there is still substantial disagreement and no clear consensus about when unlabeled data helps and by how much. In our work, we have provided a PAC-style model for semi-supervised learning that captures many of the ways unlabeled data is typically used, and provides a very general framework for thinking about this issue. The high level main implication of our analysis is that unlabeled data is useful if (a) we have a good notion of compatibility so that the target function indeed has a low unlabeled error rate, (b) the distribution D is *helpful* in the sense that not too many other hypotheses also have a low unlabeled error rate, and (c) we

have enough *unlabeled* data to estimate unlabeled error rates well. One consequence of our model is that if the target function and data distribution are both well behaved with respect to the compatibility notion, then the sample-size bounds we get for labeled data can substantially beat what one could hope to achieve through pure labeled-data bounds. Note that in this model we have assumed that the *labeled* examples are drawn uniformly at random from the set of unlabeled examples.

2.2 Active Learning

As mentioned at the beginning of this chapter, what distinguishes active learning from the Semi-Supervised learning setting is that the algorithm can *choose* the examples he wants to label from all the examples in the unlabeled pool. The hope is that a good classifier can be learned with significantly fewer labels by actively directing the queries to informative examples. There is yet no satisfactory theoretical framework that fully explains when and why active learning helps, and most of the existing positive results so far rely on *very strong additional* assumptions.

For instance the Query by Committee analysis [69] assumes realizability (i.e., there exists a perfect classifier in a known set) and a correct Bayesian prior on the set of hypotheses [69]; furthermore, while the algorithm and the analysis in [69] are general, the only specific setting for which a significant improvement in the labeled data sample complexity is shown is for learning homogeneous linear separators with respect to an input distribution which is *uniform* over the unit sphere. The Perceptron active learning algorithm for learning linear separators described in [61] is analyzed under the same *strong* assumption: uniform distribution over the surface of the unit sphere.

In the general active learning setting, for the realizable case, Cohen, Atlas and Ladner have introduced in [53] a *generic* active learning algorithm. This algorithm is a sequential algorithm that keeps track of two spaces — the current *version space* H_i , defined as the set of hypotheses in H consistent with all labels revealed so far, and the current *region of uncertainty* R_i , defined as the set of all $x \in X$, for which there exists a pair of hypotheses in H_i that disagrees on x . In round i , the algorithm picks a random unlabeled example from R_i and queries it, eliminating all hypotheses in H_i inconsistent with the received label. The algorithm then eliminates those $x \in R_i$ on which all surviving hypotheses agree, and recurses.⁸ This process fundamentally relies on the assumption that there exists a consistent hypothesis in H . In the agnostic case, we cannot eliminate a hypothesis based on its disagreement with a single example. We need to be more conservative, or we risk eliminating best hypotheses in the class. In [21] we present an extension of this algorithm to the *agnostic* case and we show that in certain cases our algorithm, which we call A^2 does provide a significant improvement in the sample complexity.

For clarity, we describe here the precise agnostic setting we analyze in [21]. We denote by X the instance space, by $Y = \{-1, 1\}$ the set of possible label, and by H the hypothesis class – a set of functions mapping from X to Y . We assume there is a distribution P over instances in X , and that the instances are labeled by a possibly randomized oracle O . The *error rate* of a hypothesis h with respect to a distribution P over $X \times Y$ is defined as $err_P(h) = \Pr_{x,y \sim P}[h(x) \neq y]$. Let $\eta = \min_{h \in H} (err_{D,O}(h))$ denote the minimum error rate of any hypothesis in H with respect to the distribution (D, O) induced by D and the labeling oracle O . The goal is to find a hypothesis $h \in H$ with $err_{D,O}(h)$ within ϵ of η , where ϵ is some target error.

Our algorithm A^2 relies on a subroutine, which computes a lower bound $LB(S, h, \delta)$ and an upper bound $UB(S, h, \delta)$ on the true error rate $err_P(h)$ of h by using a sample S of examples drawn *i.i.d.* from

⁸Note that while in general this algorithm might be computationally inefficient, it can be made efficient for certain concept classes in the realizable case.

P . Each of these bounds must hold for all h simultaneously with probability at least $1 - \delta$. The subroutine is formally defined below.

Definition 2.2.1 A subroutine for computing $LB(S, h, \delta)$ and $UB(S, h, \delta)$ is said to be legal if for all distributions P over $X \times Y$, and for all $m \in \mathbb{N}$,

$$LB(S, h, \delta) \leq \text{err}_P(h) \leq UB(S, h, \delta)$$

holds for all $h \in H$ simultaneously, with probability $1 - \delta$ over the draw of S according to P^m .

We briefly describe here the A^2 algorithm – a formal specification of our algorithm A^2 is given in Algorithm 1. Let H_i be the set of hypotheses still under consideration by A^2 in round i . If all hypotheses in H_i agree on some region of the instance space, this region can be safely eliminated. To help us keep track of progress in decreasing the region of uncertainty, define $\text{DISAGREE}_D(H_i)$ as the probability that there exists a pair of hypotheses in H_i that disagrees on a random example drawn from D :

$$\text{DISAGREE}_D(H_i) = \Pr_{x \sim D}[\exists h_1, h_2 \in H_i : h_1(x) \neq h_2(x)].$$

Hence $\text{DISAGREE}_D(H_i)$ is the volume of the current region of uncertainty with respect to D .

Let D_i be the distribution D restricted to the current region of uncertainty. Formally, $D_i = D(x \mid \exists h_1, h_2 \in H_i : h_1(x) \neq h_2(x))$. In round i , A^2 samples a set of examples S_i from D_i , O , and uses it to compute upper and lower bounds for all hypotheses in H_i . It then eliminates all hypotheses whose lower bound is greater than the minimum upper bound. Round i completes when S_i is large enough to eliminate at least half of the current region of uncertainty. Since we eliminate only those examples on which the surviving hypotheses agree, an optimal hypothesis in H_i with respect to D_i remains an optimal hypothesis in H_{i+1} with respect to D_{i+1} . Since each round i cuts $\text{DISAGREE}_D(H_i)$ down by half, the number of rounds is bounded by $\log \frac{1}{\epsilon}$.

In [21] we give examples of distributions and hypothesis classes for which A^2 requires only a small number of labeled examples to transition between rounds, yielding an *exponential improvement in sample complexity*.

It is worth mentioning that the *membership-query* setting [9, 10, 48, 82] is similar to active learning considered here except that no unlabeled data is given. Instead, the learning algorithm is allowed to query examples of its own choice. This is problematic in several applications because natural oracles, such as hired humans, have difficulty labeling synthetic examples [28].

Algorithm 1 A^2 (allowed error rate ϵ , sampling oracle for D , labeling oracle O , hypothesis class H)

Set $i = 1$, $P_i = D$, $H_i = H$, $S_i = \emptyset$, and $k = 1$.

while $\text{DISAGREE}_D(H_i)(\min_{h \in H_i} \text{UB}(S_i, h, \mu_k) - \min_{h \in H_i} \text{LB}(S_i, h, \mu_k)) > \epsilon$

1. set $S_i = \emptyset$, $H'_i = H_i$, $k \leftarrow k + 1$.

2. **while** $\text{DISAGREE}_D(H'_i) \geq \frac{1}{2} \text{DISAGREE}_D(H_i)$

(a) **if** $\text{DISAGREE}_D(H'_i)(\min_{h \in H'_i} \text{UB}(S_i, h, \mu_k) - \min_{h \in H'_i} \text{LB}(S_i, h, \mu_k)) \leq \epsilon$

(b) return $h = \underset{h \in H'_i}{\text{argmin}} \text{UB}(S_i, h, \mu_k)$.

(c) **else**

i. $S'_i =$ Rejection sample $2|S_i| + 1$ samples x from D satisfying:

$$\exists h_1, h_2 \in H_i : h_1(x) \neq h_2(x)$$

ii. $S_i \leftarrow S_i \cup \{(x, O(x)) : x \in S'_i\}$; $k \leftarrow k + 1$;

iii. $H'_i = \{h \in H_i : \text{LB}(S_i, h, \mu_k) \leq \min_{h' \in H_i} \text{UB}(S_i, h', \mu_k)\}$, $k \leftarrow k + 1$.

end if

end while

3. $H_{i+1} \leftarrow H'_i$;

$D_{i+1} \leftarrow D_i$ conditioned on the disagreement $\exists h_1, h_2 \in H_i : h_1(x) \neq h_2(x)$;

$i \leftarrow i + 1$.

end while

Return $h = \underset{h \in H_i}{\text{argmin}} \text{UB}(S_i, h, \mu_k)$.

Chapter 3

Similarity-based Learning

As mentioned in Chapter 1, kernel functions have become an extremely popular tool in Machine Learning, with an attractive theory as well [79, 83, 113, 114, 122].

A kernel is a function that takes in two data objects (which could be images, DNA sequences, or points in R^n) and outputs a number, with the property that the function is symmetric and positive-semidefinite. That is, for any kernel K , there must exist an (implicit) mapping ϕ , such that for all inputs x, x' we have $K(x, x') = \phi(x) \cdot \phi(x')$. The kernel is then used inside a “kernelized” learning algorithm such as SVM or kernel-perceptron as the way in which the algorithm interacts with the data.

The theory behind kernel functions is based on the fact that many standard algorithms for learning linear separators, such as SVMs and the Perceptron algorithm, can be written so that the only way they interact with their data is via computing dot-products on pairs of examples. Thus, by replacing each invocation of $x \cdot x'$ with a kernel computation $K(x, x')$, the algorithm behaves exactly as if we had explicitly performed the mapping $\phi(x)$, even though ϕ may be a mapping into a very high-dimensional space. Furthermore, these algorithms have convergence rates that depend only on the *margin* γ of the best separator, and not on the dimension of the space in which the data resides [12, 116]. Thus, kernel functions are often viewed as providing much of the power of this implicit high-dimensional space, without paying for it computationally (because the ϕ mapping is only implicit) or in terms of sample size (if data is indeed well-separated in that space).

In our work [22], we point out that the Johnson-Lindenstrauss [59] lemma suggests that in the presence of a large margin, a kernel function can also be viewed as a mapping to a *low*-dimensional space, one of dimension only $\tilde{O}(1/\gamma^2)$. We then explore the question of whether one can efficiently produce such low-dimensional mappings, using only black-box access to a kernel function. That is, given just a program that computes $K(x, y)$ on inputs x, y of our choosing, can we efficiently construct an explicit (small) set of features that effectively capture the power of the implicit high-dimensional space? We answer this question in the affirmative if our method is also allowed black-box access to the underlying data distribution (i.e., unlabeled examples).

Our positive result can be viewed as saying that designing a good kernel function is much like designing a good feature space. Given a kernel, by running it in a black-box manner on random *unlabeled* examples, we can *efficiently* generate an explicit set of $\tilde{O}(1/\gamma^2)$ features, such that if the data was linearly separable with margin γ under the kernel, then it is approximately separable in this new feature space. We describe these results more thoroughly in Section 3.2 below.

One interesting aspect our simplest method considered in [22]¹ is that it can be applied to any generic

¹As explained in Section 3.2 below, this mapping is very simple: we only need to choose x_1, \dots, x_d from the underlying distribution over the instance space D and then to use the mapping $x \mapsto (K(x, x_1), \dots, K(x, x_d))$.

“similarity” function $K(x, y)$, even those that are not necessarily legal kernels and do not necessarily have the same interpretation as computing a dot-product in some implicit ϕ -space. In subsequent work [16] we extend some of these guarantees to this more general setting.

Specifically, in [16] we develop an alternative, more general theory of learning with similarity functions (i.e., sufficient conditions for a similarity function to allow one to learn well) that does not require reference to implicit spaces, and does not require the function to be positive semi-definite (or even symmetric). These results also *generalize* the standard theory in the sense that any good kernel function under the usual definition can be shown to also be a good similarity function under our definition (though with some loss in the parameters). Moreover, our theory provides the first steps towards a theory of kernels that describes the effectiveness of a given kernel function in terms of natural *similarity-based* properties. We describe the main motivation and the implications of these results more thoroughly in Section 3.3 below.

3.1 Preliminaries

We present here the learning model and a few definitions we use throughout this chapter.

We assume that examples are given to us according to some probability distribution D over an instance space X and labeled by some unknown target function $c^* : X \rightarrow \{-1, +1\}$. We use $P = (D, c^*)$ to denote the combined distribution over labeled examples. Given some sample S of labeled training examples (each drawn independently from D and labeled by c^*), our objective is to come up with a hypothesis h with low true error: that is, we want $\Pr_{x \sim D}(h(x) \neq c^*(x))$ to be low. In the discussion below, by a “learning problem” we mean a distribution $P = (D, c^*)$ over labeled examples.

We will be considering learning algorithms whose only access to their data is via a pairwise similarity function $K(x, x')$ that given two examples outputs a number in the range $[-1, 1]$. Specifically,

Definition 3.1.1 *A similarity function over X is any pairwise function $K : X \times X \rightarrow [-1, 1]$. We say that K is a symmetric similarity function if $K(x, x') = K(x', x)$ for all x, x' .*

Definition 3.1.2 *A similarity function K is a kernel if there exists a function ϕ from the instance space X into a (possibly implicit) “ ϕ -space” such that $K(x, x') = \phi(x) \cdot \phi(x')$.*

Definition 3.1.3 *We say that a set S of labeled examples is **linearly separable by margin γ** if there exists a unit-length vector w such that:*

$$\min_{(x, \ell) \in S} [\ell(w \cdot x) / \|x\|] \geq \gamma.$$

That is, the separator $w \cdot x \geq 0$ has margin γ if every labeled example in S is correctly classified and furthermore the cosine of the angle between w and x has magnitude at least γ .² For simplicity, we are only considering separators that pass through the origin, though results we discuss can be adapted to the general case as well.

We can similarly talk in terms of the distribution P rather than a sample S .

Definition 3.1.4 *We say that P is **linearly separable by margin γ** if there exists a unit-length vector w such that:*

$$\Pr_{(x, \ell) \sim P} [\ell(w \cdot x) / \|x\| < \gamma] = 0,$$

*and we say that P is **separable with error α at margin γ** if there exists a unit-length vector w such that:*

$$\Pr_{(x, \ell) \sim P} [\ell(w \cdot x) / \|x\| < \gamma] \leq \alpha.$$

²Often margin is defined without normalizing by the length of the examples, though in that case the “ γ^2 ” term in sample complexity bounds becomes “ γ^2/R^2 ”, where R is the maximum $\|x\|$ over $x \in S$. Technically, normalizing produces a stronger bound because we are taking the minimum of a ratio, rather than the ratio of a minimum to a maximum.

A powerful theoretical result in machine learning is that if a learning problem is linearly separable by a large margin γ , then that makes the problem “easy” in the sense that to achieve good generalization one needs only a number of examples that depends (polynomially) on $1/\gamma$, with no dependence on the dimension of the ambient space X that examples lie in. In fact, two results of this form are:

1. The classic Perceptron Convergence Theorem that the Perceptron Algorithm makes at most $1/\gamma^2$ mistakes on any sequence of examples separable by margin γ [33, 101, 108]. Thus, if the Perceptron algorithm is run on a sample of size $1/(\epsilon\gamma^2)$, the expected error rate of its hypothesis at a random point in time is at most ϵ . (For further results of this form, see [68, 98]).
2. The more recent margin bounds of [27, 116] that state that $|S| = O(\frac{1}{\epsilon}[\frac{1}{\gamma^2} \log^2(\frac{1}{\gamma\epsilon}) + \log \frac{1}{\delta}])$ is sufficient so that with high probability, *any* linear separator of S with margin γ has true error at most ϵ . Thus, this provides a sample complexity bound that applies to any algorithm that finds large-margin separators.

We present below a few other definitions we use throughout this chapter, as well as briefly describe the classical Johnson-Lindenstrauss Lemma. In analogy to Definition 3.1.4, we will say that:

Definition 3.1.5 *P is separable by margin γ in the ϕ -space if there exists a unit-length vector w in the ϕ -space such that $\Pr_{(x,\ell)\sim P}[\ell(w \cdot \phi(x))/\|\phi(x)\| < \gamma] = 0$. Similarly that P is separable with error α at margin γ in the ϕ -space if the above holds with “= 0” replaced by “ $\leq \alpha$ ”.*

For a set of vectors v_1, v_2, \dots, v_k in Euclidean space, let $\text{span}(v_1, \dots, v_k)$ denote the span of these vectors: that is, the set of vectors u that can be written as a linear combination $a_1v_1 + \dots + a_kv_k$. Also, for a vector u and a subspace Y , let $\text{proj}(u, Y)$ be the orthogonal projection of u down to Y . So, for instance, $\text{proj}(u, \text{span}(v_1, \dots, v_k))$ is the orthogonal projection of u down to the space spanned by v_1, \dots, v_k .³

The Johnson-Lindenstrauss Lemma The Johnson-Lindenstrauss Lemma [59, 81, 86] states that given a set S of points in R^n , if we perform an orthogonal projection of those points onto a random d -dimensional subspace, then $d = O(\frac{1}{\gamma^2} \log |S|)$ is sufficient so that with high probability all pairwise distances are preserved up to $1 \pm \gamma$ (up to scaling). Conceptually, one can think of a random projection as first applying a random rotation to R^n and then reading off the first d coordinates. In fact, a number of different forms of “random projection” are known to work (including some that are especially efficient to perform computationally, considered in [6, 13]). In particular, if we think of performing the projection via multiplying all points, viewed as row-vectors of length n , by an $n \times d$ matrix A , then several methods for selecting A that provide the desired result are:

1. Choosing its columns to be d random orthogonal unit-length vectors (a true random orthogonal projection).
2. Choosing each entry in A independently from a standard Gaussian (so the projection can be viewed as selecting d vectors u_1, u_2, \dots, u_d from a spherical gaussian and mapping a point p to $(p \cdot u_1, \dots, p \cdot u_d)$).
3. Choosing each entry in A to be 1 or -1 independently at random.

Formally, here is a convenient form of the Johnson-Lindenstrauss Lemma given in [13]. Let $N(0, 1)$ denote the standard Normal distribution with mean 0 and variance 1, and $U(-1, 1)$ denote the distribution that has probability $1/2$ on -1 and probability $1/2$ on 1 .

Theorem 3.1.1 (Neuronal RP [13]) *Let $u, v \in R^n$. Let $u' = \frac{1}{\sqrt{d}}uA$ and $v' = \frac{1}{\sqrt{d}}vA$ where A is a $n \times d$*

³We note that given a set of vectors v_1, \dots, v_k and the ability to compute dot-products, this projection can be computed efficiently by solving a set of linear equalities.

random matrix whose entries are chosen independently from either $N(0, 1)$ or $U(-1, 1)$. Then,

$$\Pr_A [(1 - \gamma)\|u - v\|^2 \leq \|u' - v'\|^2 \leq (1 + \gamma)\|u - v\|^2] \geq 1 - 2e^{-(\gamma^2 - \gamma^3)\frac{d}{4}}.$$

Theorem 3.1.1 suggests a natural learning algorithm: first randomly project data into a lower dimensional space, and then run some other algorithm in that space, taking advantage of the speedup produced by working over fewer dimensions. Theoretical results for some algorithms of this form are given in [13], and experimental results are given in [57, 66, 71].

3.2 Kernels as Features: On Kernels, Margins and Low-Dimensional Mappings

The Johnson-Lindenstrauss lemma provides a particularly intuitive way to see why one should be able to generalize well from only a small amount of training data when a learning problem is separable by a large margin. In particular, imagine a set S of data in some high-dimensional space, and suppose that we randomly project the data down to R^d . By the Johnson-Lindenstrauss Lemma, $d = O(\gamma^{-2} \log |S|)$ is sufficient so that with high probability, all *angles* between points (viewed as vectors) change by at most $\pm\gamma/2$.⁴ In particular, consider projecting all points in S and the target vector c^* ; if initially data was separable by margin γ , then after projection, since angles with c^* have changed by at most $\gamma/2$, the data is still separable (and in fact separable by margin $\gamma/2$). Thus, this means our problem was in some sense really only a d -dimensional problem after all. Moreover, if we replace the “ $\log |S|$ ” term in the bound for d with “ $\log \frac{1}{\epsilon}$ ”, then we can use Theorem 3.1.1 to get that with high probability at least a $1 - \epsilon$ fraction of S will be separable. Formally, talking in terms of the true distribution P , one can state the following theorem. (Proofs appear in, e.g., [13, 22].)

Theorem 3.2.1 *If P is linearly separable by margin γ , then $d = O\left(\frac{1}{\gamma^2} \log\left(\frac{1}{\epsilon\delta}\right)\right)$ is sufficient so that with probability at least $1 - \delta$, a random projection down to R^d will be linearly separable with error at most ϵ at margin $\gamma/2$.*

So, Theorem 3.2.1 can be viewed as stating that a learning problem separable by margin γ is really only an “ $O(1/\gamma^2)$ -dimensional problem” after all.

Combining kernel functions with the Johnson-Lindenstrauss Lemma (in particular, Theorem 3.2.1 above), we have that if a learning problem indeed has the large margin property under kernel $K(x, y) = \phi(x) \cdot \phi(y)$, then a *random* linear projection of the “ ϕ -space” down to a *low* dimensional space approximately preserves linear separability. This means that for any kernel K under which the learning problem is linearly separable by margin γ in the ϕ -space, we can, in principle, think of K as mapping the input space X into an $\tilde{O}(1/\gamma^2)$ -dimensional space, in essence serving as a method for representing the data in a new (and not too large) feature space.

The question we now consider is whether, given kernel K as a black-box function, we can in fact produce such a mapping efficiently. The problem with the above observation is that it requires explicitly computing the function $\phi(x)$. Since for a given kernel K , the dimensionality of the ϕ -space might be quite large, this is not efficient.⁵ Instead, what we would like is an efficient procedure that given $K(\cdot, \cdot)$ as a black-box program, produces a mapping with the desired properties using running time that depends (polynomially) only on $1/\gamma$ and the time to compute the kernel function K , with no dependence on the

⁴The Johnson-Lindenstrauss Lemma talks about relative *distances* being approximately preserved, but it is a straightforward calculation to show that this implies angles must be approximately preserved as well.

⁵In addition, it is not totally clear how to apply Theorem 3.1.1 if the dimension of the ϕ -space is infinite.

dimensionality of the ϕ -space. This would mean we can effectively convert a kernel K that is good for some learning problem into an explicit set of features, without a need for “kernelizing” our learning algorithm. In this section, we describe several methods for doing so; this work appears in [22].

Specifically, we will show the following. Given black-box access to a kernel function $K(x, y)$, access to unlabeled examples from distribution D , and parameters γ , ε , and δ , we can in polynomial time construct a mapping $F : X \rightarrow R^d$ (i.e., to a set of d real-valued features) where $d = O\left(\frac{1}{\gamma^2} \log \frac{1}{\varepsilon\delta}\right)$, such that if the target concept indeed has margin γ in the ϕ -space, then with probability $1 - \delta$ (over randomization in our choice of mapping function), the induced distribution in R^d is separable with error $\leq \varepsilon$. In fact, not only will the data in R^d be separable, but it will be separable with a margin $\gamma' = \Omega(\gamma)$. (If the original learning problem was separable with error α at margin γ then the induced distribution is separable with error $\alpha + \varepsilon$ at margin γ' .)

To give a feel of what such a mapping might look like, suppose we are willing to use dimension $d = O\left(\frac{1}{\varepsilon}\left[\frac{1}{\gamma^2} + \ln \frac{1}{\delta}\right]\right)$ (so this is linear in $1/\varepsilon$ rather than logarithmic) and we are not concerned with preserving margins and only want approximate separability. Then we show the following especially simple procedure suffices. Just draw a random sample of d unlabeled points x_1, \dots, x_d from D and define $F(x) = (K(x, x_1), \dots, K(x, x_d))$.⁶ That is, if we think of K not so much as an implicit mapping into a high-dimensional space but just as a similarity function over examples, what we are doing is drawing d “reference” points and then defining the i th feature of x to be its similarity with reference point i . Corollary 3.2.3 (in Section 3.2.1 below) shows that under the assumption that the target function has margin γ in the ϕ space, with high probability the data will be approximately separable under this mapping. Thus, this gives a particularly simple way of using the kernel and unlabeled data for feature generation.

Given these results, a natural question is whether it might be possible to perform mappings of this type without access to the underlying distribution. In [22] we show that this is in general *not* possible, given only black-box access (and polynomially-many queries) to an *arbitrary* kernel K . However, it may well be possible for specific standard kernels such as the polynomial kernel or the gaussian kernel.

3.2.1 Two simple mappings

Our goal is a procedure that given black-box access to a kernel function $K(\cdot, \cdot)$, unlabeled examples from distribution D , and a margin value γ , produces a mapping $F : X \rightarrow R^d$ with the following property: if the target function indeed has margin γ in the ϕ -space, then with high probability F approximately preserves linear separability. In this section, we analyze two methods that both produce a space of dimension $O\left(\frac{1}{\varepsilon}\left[\frac{1}{\gamma^2} + \ln \frac{1}{\delta}\right]\right)$, such that with probability $1 - \delta$ the result is separable with error at most ε . The second of these mappings in fact satisfies a stronger condition that its output will be approximately separable at margin $\gamma/2$ (rather than just approximately separable). This property will allow us to use this mapping as a first step in a better mapping in Section 3.3.2.

The following lemma is key to our analysis.

Lemma 3.2.2 *Consider any distribution over labeled examples in Euclidean space such that there exists a linear separator $w \cdot x = 0$ with margin γ . If we draw*

$$d \geq \frac{8}{\varepsilon} \left[\frac{1}{\gamma^2} + \ln \frac{1}{\delta} \right]$$

examples z_1, \dots, z_d iid from this distribution, with probability $\geq 1 - \delta$, there exists a vector w' in $\text{span}(z_1, \dots, z_d)$ that has error at most ε at margin $\gamma/2$.

⁶In contrast, the Johnson-Lindenstrauss Lemma as presented in Theorem 3.1.1 would draw d Gaussian (or uniform $\{-1, +1\}$) random points r_1, \dots, r_d in the ϕ -space and define $F(x) = (\phi(x) \cdot r_1, \dots, \phi(x) \cdot r_d)$.

Note: We remark that a somewhat weaker bound on d can be derived from the machinery of margin bounds. Margin bounds [27, 116] tell us that using $d = O(\frac{1}{\varepsilon} \lceil \frac{1}{\gamma^2} \log^2(\frac{1}{\gamma\varepsilon}) + \log \frac{1}{\delta} \rceil)$ points, with probability $1 - \delta$, any separator with margin $\geq \gamma$ over the observed data has true error $\leq \varepsilon$. Thus, the projection of the target function w into the space spanned by the observed data will have true error $\leq \varepsilon$ as well. (Projecting w into this space maintains the value of $w \cdot z_i$, while possibly shrinking the vector w , which can only increase the margin over the observed data.) The only technical issue is that we want as a conclusion for the separator not only to have a low error rate over the distribution, but also to have a large margin. However, this can be obtained from the double-sample argument used in [27, 116] by using a $\gamma/4$ -cover instead of a $\gamma/2$ -cover. Margin bounds, however, are a bit of an overkill for our needs, since we are only asking for an existential statement (the *existence* of w') and not a universal statement about all separators with large empirical margins. For this reason we are able to get a better bound by a direct argument from first principles.

Lemma 3.2.2 implies that if P is linearly separable with margin γ under K , and we draw $d = \frac{8}{\varepsilon} \lceil \frac{1}{\gamma^2} + \ln \frac{1}{\delta} \rceil$ random unlabeled examples x_1, \dots, x_n from D , then with probability at least $1 - \delta$ there is a separator w' in the ϕ -space with error rate at most ε that can be written as

$$w' = \alpha_1 \phi(x_1) + \dots + \alpha_d \phi(x_d).$$

Notice that since $w' \cdot \phi(x) = \alpha_1 K(x, x_1) + \dots + \alpha_d K(x, x_d)$, an immediate implication is that if we simply think of $K(x, x_i)$ as the i th “feature” of x — that is, if we define $F_1(x) = (K(x, x_1), \dots, K(x, x_d))$ — then with high probability the vector $(\alpha_1, \dots, \alpha_d)$ is an approximate linear separator of $F_1(P)$. So, the kernel and distribution together give us a particularly simple way of performing feature generation that preserves (approximate) separability. Formally, we have the following.

Corollary 3.2.3 *If P has margin γ in the ϕ -space, then with probability $\geq 1 - \delta$, if x_1, \dots, x_d are drawn from D for $d = \frac{8}{\varepsilon} \lceil \frac{1}{\gamma^2} + \ln \frac{1}{\delta} \rceil$, the mapping*

$$F_1(x) = (K(x, x_1), \dots, K(x, x_d))$$

produces a distribution $F_1(P)$ on labeled examples in R^d that is linearly separable with error at most ε .

Unfortunately, the above mapping F_1 may not preserve margins because we do not have a good bound on the length of the vector $(\alpha_1, \dots, \alpha_d)$ defining the separator in the new space, or the length of the examples $F_1(x)$. The key problem is that if many of the $\phi(x_i)$ are very similar, then their associated features $K(x, x_i)$ will be highly correlated. Instead, to preserve margin we want to choose an orthonormal basis of the space spanned by the $\phi(x_i)$: i.e., to do an orthogonal projection of $\phi(x)$ into this space. Specifically, let $S = \{x_1, \dots, x_d\}$ be a set of $\frac{8}{\varepsilon} \lceil \frac{1}{\gamma^2} + \ln \frac{1}{\delta} \rceil$ unlabeled examples from D as in Corollary 3.2.3. We can then implement the desired orthogonal projection of $\phi(x)$ as follows. Run $K(x, y)$ for all pairs $x, y \in S$, and let $M(S) = (K(x_i, x_j))_{x_i, x_j \in S}$ be the resulting kernel matrix. Now decompose $M(S)$ into $U^T U$, where U is an upper-triangular matrix. Finally, define the mapping $F_2 : X \rightarrow R^d$ to be $F_2(x) = F_1(x)U^{-1}$, where F_1 is the mapping of Corollary 3.2.3. This is equivalent to an orthogonal projection of $\phi(x)$ into $\text{span}(\phi(x_1), \dots, \phi(x_d))$. Technically, if U is not full rank then we want to use the (Moore-Penrose) pseudoinverse [30] of U in place of U^{-1} .

By Lemma 3.2.2, this mapping F_2 maintains approximate separability at margin $\gamma/2$ (See [22] for a full proof):

Theorem 3.2.4 *If P has margin γ in the ϕ -space, then with probability $\geq 1 - \delta$, the mapping $F_2 : X \rightarrow R^d$ for $d \geq \frac{8}{\varepsilon} \lceil \frac{1}{\gamma^2} + \ln \frac{1}{\delta} \rceil$ has the property that $F_2(P)$ is linearly separable with error at most ε at margin $\gamma/2$.*

Notice that the running time to compute $F_2(x)$ is polynomial in $1/\gamma, 1/\varepsilon, 1/\delta$ and the time to compute the kernel function K .

3.2.2 An improved mapping

We now describe an improved mapping, in which the dimension d has only a logarithmic, rather than linear, dependence on $1/\varepsilon$. The idea is to perform a two-stage process, composing the mapping from the previous section with an additional Johnson-Lindenstrauss style mapping to reduce dimensionality even further. Thus, this mapping can be thought of as combining two types of random projection: a projection based on points chosen at random from D , and a projection based on choosing points uniformly at random in the intermediate space.

In particular, let $F_2 : X \rightarrow R^{d_2}$ be the mapping from Section 3.2.1 using $\varepsilon/2$ and $\delta/2$ as its error and confidence parameters respectively. Let $\hat{F} : R^{d_2} \rightarrow R^{d_3}$ be a random projection as in Theorem 3.1.1. Then consider the overall mapping $F_3 : X \rightarrow R^{d_3}$ to be $F_3(x) = \hat{F}(F_2(x))$.

We now claim that for $d_2 = O(\frac{1}{\varepsilon}[\frac{1}{\gamma^2} + \ln \frac{1}{\delta}])$ and $d_3 = O(\frac{1}{\gamma^2} \log(\frac{1}{\varepsilon\delta}))$, with high probability, this mapping has the desired properties. The basic argument is that the initial mapping F_2 maintains approximate separability at margin $\gamma/2$ by Lemma 3.2.2, and then the second mapping approximately preserves this property by Theorem 3.1.1. In particular, we have (see [22] for a full proof):

Theorem 3.2.5 *If P has margin γ in the ϕ -space, then with probability at least $1 - \delta$, the mapping $F_3 = \hat{F} \circ F_2 : X \rightarrow R^{d_3}$, for values $d_2 = O(\frac{1}{\varepsilon}[\frac{1}{\gamma^2} + \ln \frac{1}{\delta}])$ and $d_3 = O(\frac{1}{\gamma^2} \log(\frac{1}{\varepsilon\delta}))$, has the property that $F_3(P)$ is linearly separable with error at most ε at margin $\gamma/4$.*

As before, the running time to compute our mappings is polynomial in $1/\gamma, 1/\varepsilon, 1/\delta$ and the time to compute the kernel function K .

Since the dimension d_3 of the mapping in Theorem 3.2.5 is only logarithmic in $1/\varepsilon$, this means that if P is perfectly separable with margin γ in the ϕ -space, we can set ε to be small enough so that with high probability, a sample of size $O(d_3 \log d_3)$ would be perfectly separable. That is, we could use an arbitrary noise-free linear-separator learning algorithm in R^{d_3} to learn the target concept. However, this requires using $d_2 = \tilde{O}(1/\gamma^4)$ (i.e., $\tilde{O}(1/\gamma^4)$) unlabeled examples to construct the mapping.

Corollary 3.2.6 *Given $\varepsilon', \delta, \gamma < 1$, if P has margin γ in the ϕ -space, then $\tilde{O}(\frac{1}{\varepsilon'\gamma^4})$ unlabeled examples are sufficient so that with probability $1 - \delta$, mapping $F_3 : X \rightarrow R^{d_3}$ has the property that $F_3(P)$ is linearly separable with error $o(\varepsilon'/(d_3 \log d_3))$, where $d_3 = O(\frac{1}{\gamma^2} \log \frac{1}{\varepsilon'\gamma\delta})$.*

3.3 A General Theory of Learning with Similarity Functions

As described in the previous sections of this chapter, Kernel functions have become an extremely popular tool in machine learning, with an attractive theory as well. This theory views a kernel as implicitly mapping data points into a possibly very high dimensional space, and describes a kernel function as being good for a given learning problem if data is separable by a large margin in that implicit space. However, while quite elegant, this theory does not directly correspond to one's intuition of a good kernel as a good similarity function. Furthermore, it may be difficult for a domain expert to use the theory to help design an appropriate kernel for the learning task at hand since the implicit mapping may not be easy to calculate. Finally, the requirement of positive semi-definiteness may rule out the most natural pairwise similarity functions for the given problem domain.

We describe here an alternative, more general theory of learning with similarity functions (i.e., sufficient conditions for a similarity function to allow one to learn well) that does not require reference to

implicit spaces, and does not require the function to be positive semi-definite (or even symmetric). Our results also generalize the standard theory in the sense that any good kernel function under the usual definition can be shown to also be a good similarity function under our definition (though with some loss in the parameters).

3.3.1 Formal Framework

Our goal is to give definitions for what it means for a similarity function K to be “good” for a learning problem P that (ideally) are intuitive, broad, and have the property that a good similarity function results in the ability to learn well. Note that as with the theory of kernel functions, the notion of “goodness” is with respect to a given learning problem P , and *not* with respect to a class of target functions as in the PAC framework.

We say that K is an (ϵ, γ) -good kernel function for a given learning problem P if P is separable with error at most ϵ at margin γ in the ϕ -space: i.e., there exists a vector w in the ϕ -space that has error ϵ at margin γ , where we use a normalized notion of margin, and for simplicity we consider only separators through the origin.⁷ We say that K is a γ -good kernel function if it is (ϵ, γ) -good for $\epsilon = 0$; i.e., it has zero error at margin γ . Moreover, we say that K is a normalized kernel if $K(x, x) = 1$ for all x .

For simplicity we assume throughout this section that all kernels are normalized: note that any kernel function K can be converted to a normalized one $\hat{K}(x, x') = \frac{K(x, x')}{\sqrt{K(x, x)K(x', x')}}$ without changing its margin properties.

Note A similarity function is not always a legal kernel. For example, suppose we say two documents have similarity 1 if they have either an author in common or a keyword in common, and otherwise they have similarity 0. Then you could have three documents A , B , and C , such that $K(A, B) = 1$ because A and B have an author in common, $K(B, C) = 1$ because B and C have a keyword in common, but $K(A, C) = 0$ because A and C have neither an author nor a keyword in common (and $K(A, A) = K(B, B) = K(C, C) = 1$). On the other hand, a kernel requires that if $\phi(A)$ and $\phi(B)$ are of unit length and $\phi(A) \cdot \phi(B) = 1$, then $\phi(A) = \phi(B)$, so this could not happen if K was a kernel.⁸

In the following we will use $\ell(x)$ to denote the label of example x (i.e., $\ell(x) = c^*(x)$) and use $x \sim P$ as shorthand for $(x, \ell(x)) \sim P$.

3.3.2 Sufficient Conditions for Learning with Similarity Functions

We now provide a series of sufficient conditions for a similarity function to be useful for learning, leading to our main notion given in Definition 3.3.3.

We begin with our first and simplest notion of “good similarity function” that is intuitive and yields an immediate learning algorithm, but which is not broad enough to capture all good kernel functions. Nonetheless, it provides a convenient starting point. This definition says that K is a good similarity function for a learning problem P if most examples x (at least a $1 - \epsilon$ probability mass) are on average at least γ more similar to random examples x' of the *same* label than they are to random examples x' of the opposite label. Formally,

⁷ Specifically, K is an (ϵ, γ) -good kernel function if there exists a vector w such that

$$\Pr_{(x, \ell(x)) \sim P} \ell(x) \frac{\phi(x) \cdot w}{\|\phi(x)\| \|w\|} \geq \gamma \geq 1 - \epsilon.$$

⁸ You could make such a function positive semidefinite by instead defining similarity to be the *number* of authors and keywords in common, but perhaps that is not what you want for the task at hand. Alternatively, you can make the similarity matrix positive semidefinite by blowing up the diagonal, but that would reduce the normalized margin.

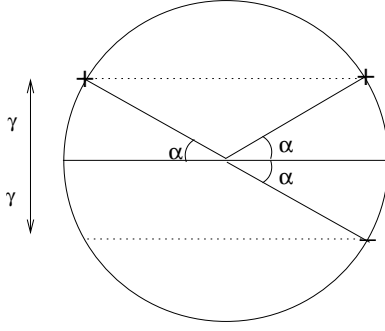


Figure 3.1: Positives are split equally among upper-left and upper-right. Negatives are all in the lower-right. For $\alpha = 30^\circ$ (so $\gamma = 1/2$) a large fraction of the positive examples (namely the 50% in the upper-right) have a higher dot-product with negative examples ($\frac{1}{2}$) than with a random positive example ($\frac{1}{2} \cdot 1 + \frac{1}{2}(-\frac{1}{2}) = \frac{1}{4}$).

Definition 3.3.1 K is a **strongly** (ϵ, γ) -good similarity function for a learning problem P if at least a $1 - \epsilon$ probability mass of examples x satisfy:

$$\mathbf{E}_{x' \sim P}[K(x, x') | \ell(x) = \ell(x')] \geq \mathbf{E}_{x' \sim P}[K(x, x') | \ell(x) \neq \ell(x')] + \gamma. \quad (3.1)$$

For example, suppose all positive examples have similarity at least 0.2 with each other, and all negative examples have similarity at least 0.2 with each other, but positive and negative examples have similarities distributed uniformly at random in $[-1, 1]$. Then, this would satisfy Definition 3.3.1 for $\gamma = 0.2$ and $\epsilon = 0$, but with high probability would not be positive semidefinite.⁹

Definition 3.3.1 captures an intuitive notion of what one might want in a similarity function. In addition, if a similarity function K satisfies Definition 3.3.1 then it suggests a simple, natural learning algorithm: draw a sufficiently large set S^+ of positive examples and set S^- of negative examples, and then output the prediction rule that classifies a new example x as positive if it is on average more similar to points in S^+ than to points in S^- , and negative otherwise. Formally:

Theorem 3.3.1 If K is strongly (ϵ, γ) -good, then $(4/\gamma^2) \ln(2/\delta)$ positive examples S^+ and $(4/\gamma^2) \ln(2/\delta)$ negative examples S^- are sufficient so that with probability $\geq 1 - \delta$, the above algorithm produces a classifier with error at most $\epsilon + \delta$.

Theorem 3.3.1 implies that if K is a strongly (ϵ, γ) -good similarity function for small ϵ and not-too-small γ , then it can be used in a natural way for learning. However, Definition 3.3.1 is not sufficient to capture all good kernel functions. In particular, Figure 3.1 gives a simple example in \mathcal{R}^2 where the standard kernel $K(x, x') = x \cdot x'$ has a large margin separator (margin of $1/2$) and yet does not satisfy Definition 3.3.1, even for $\gamma = 0$ and $\epsilon = 0.49$.

Notice, however, that if in Figure 3.1 we simply ignored the positive examples in the upper-left when choosing x' , then we would be fine.

This then motivates the following intermediate notion of a similarity function K being good under a weighting function w over the input space that can downweight certain portions of that space.

Definition 3.3.2 A similarity function K together with a bounded weighting function w over X (specifically, $w(x') \in [0, 1]$ for all $x' \in X$) is a **strongly** (ϵ, γ) -good weighted similarity function for a learning

⁹In particular, if the domain is large enough, then with high probability there would exist negative example A and positive examples B, C such that $K(A, B)$ is close to 1 (so they are nearly identical as vectors), $K(A, C)$ is close to -1 (so they are nearly opposite as vectors), and yet $K(B, C) \geq 0.2$ (their vectors form an acute angle).

problem P if at least a $1 - \epsilon$ probability mass of examples x satisfy:

$$\mathbf{E}_{x' \sim P}[w(x')K(x, x')|\ell(x) = \ell(x')] \geq \mathbf{E}_{x' \sim P}[w(x')K(x, x')|\ell(x) \neq \ell(x')] + \gamma. \quad (3.2)$$

We can view Definition 3.3.2 intuitively as saying that we only require most examples be substantially more similar on average to *reasonable* points of the same class than to *reasonable* points of the opposite class, where “reasonableness” is a score in $[0, 1]$. A pair (K, w) satisfying Definition 3.3.2 can be used in exactly the same way as a similarity function K satisfying Definition 3.3.1, with the exact same proof used in Theorem 3.3.1 (except now we view $w(y)K(x, x')$ as the bounded random variable we plug into Hoeffding bounds).

Definition 3.3.2 now motivates our main definition given below. The key difference is that whereas in Definition 3.3.2 one needs the designer to construct both the similarity function K and the weighting function w , in Definition 3.3.3 we only require that such a w exist, but it need not be known a-priori. That is, we ask only that there exist a large probability mass of “reasonable” points (a weighting scheme) satisfying Definition 3.3.2, but the designer need not know in advance what that weighting scheme should be.

Definition 3.3.3 (main) *A similarity function K is an (ϵ, γ) -good similarity function for a learning problem P if there exists a bounded weighting function w over X ($w(x') \in [0, 1]$ for all $x' \in X$) such that at least a $1 - \epsilon$ probability mass of examples x satisfy:*

$$\mathbf{E}_{x' \sim P}[w(x')K(x, x')|\ell(x) = \ell(x')] \geq \mathbf{E}_{x' \sim P}[w(x')K(x, x')|\ell(x) \neq \ell(x')] + \gamma. \quad (3.3)$$

We now show two interesting properties of Definition 3.3.3. First, if K is a similarity function satisfying it, then we can use K to explicitly map the data into a space in which there is a separator with low-error (not much more than ϵ) at a large margin (not too much less than γ), and thereby convert the learning problem into a standard one of learning a linear separator. The second is that any “good kernel” (a kernel with a large margin separator in its implicit ϕ -space) must satisfy Definition 3.3.3, though with some degradation in the parameters. We prove the first statement, which is the easier of the two, in this section, and we state the second one, which has a more involved proof, in Section 3.3.3.

Theorem 3.3.2 *If K is an (ϵ, γ) -good similarity function, then if one draws a set S from P containing $d = (4/\gamma)^2 \ln(2/\delta)$ positive examples $S^+ = \{y_1, y_2, \dots, y_d\}$ and d negative examples $S^- = \{z_1, z_2, \dots, z_d\}$, then with probability at least $1 - \delta$, the mapping $\rho_S : X \rightarrow \mathbb{R}^{2d}$ defined as*

$$\rho_S(x) = (K(x, y_1), \dots, K(x, y_d), K(x, z_1), \dots, K(x, z_d))$$

has the property that the induced distribution $\rho_S(P)$ in \mathbb{R}^{2d} has a separator of error at most $\epsilon + \delta$ at margin at least $\gamma/4$.

Proof: Consider the linear separator \tilde{w} in the ρ_S space defined as $\tilde{w}_i = w(y_i)$, for $i \in \{1, 2, \dots, d\}$ and $\tilde{w}_i = -w(z_{i-d})$, for $i \in \{d+1, d+2, \dots, 2d\}$. We will show that, with probability at least $(1 - \delta)$, \tilde{w} has error at most $\epsilon + \delta$ at margin $\gamma/4$. Let **Good** be the set of x satisfying inequality $\mathbf{E}_{x' \sim P}[w(x')K(x, x')|\ell(x) = \ell(x')] \geq \mathbf{E}_{x' \sim P}[w(x')K(x, x')|\ell(x) \neq \ell(x')] + \gamma$; so, by assumption, $\Pr_{x \sim P}[x \in \text{Good}] \geq 1 - \epsilon$.

Consider some fixed point $x \in \text{Good}$. We begin by showing that for any such x ,

$$\Pr_{S^+, S^-} \left(\ell(x) \frac{\tilde{w} \cdot \rho_S(x)}{\|\tilde{w}\| \|\rho_S(x)\|} \geq \frac{\gamma}{4} \right) \geq 1 - \delta^2.$$

To do so, first notice that d is large enough so that with high probability, at least $1 - \delta^2$, we have both

$$|\mathbf{E}_{x' \in S^+}[w(x')K(x, x')] - \mathbf{E}_{x' \sim P}[w(x')K(x, x')|\ell(x') = 1]| \leq \frac{\gamma}{4}$$

and

$$|\mathbf{E}_{x' \in S^-}[w(x')K(x, x')] - \mathbf{E}_{x' \sim P}[w(x')K(x, x') | \ell(x') = -1]| \leq \frac{\gamma}{4}.$$

Let's consider now the case when $\ell(x) = 1$. In this case we have

$$\ell(x)\tilde{w} \cdot \rho_S(x) = d\left(\frac{1}{d} \sum_{i=1}^d w(y_i)K(x, y_i) - \frac{1}{d} \sum_{i=1}^d w(z_i)K(x, z_i)\right),$$

and so combining these facts we have that with probability at least $(1 - \delta^2)$ the following holds:

$$\ell(x)\tilde{w} \cdot \rho_S(x) \geq d(\mathbf{E}_{x' \sim P}[w(x')K(x, x') | \ell(x') = 1] - \frac{\gamma}{4} - \mathbf{E}_{x' \sim P}[w(x')K(x, x') | \ell(x') = -1] - \frac{\gamma}{4}).$$

Since $x \in \mathbf{Good}$, this then implies that $\ell(x)\tilde{w} \cdot \rho_S(x) \geq d\gamma/2$. Finally, since $w(x') \in [-1, 1]$ for all x' , and since $K(x, x') \in [-1, 1]$ for all pairs x, x' , we have that $\|\tilde{w}\| \leq \sqrt{2d}$ and $\|\rho_S(x)\| \leq \sqrt{2d}$, which implies

$$\Pr_{S^+, S^-} \left(\ell(x) \frac{\tilde{w} \cdot \rho_S(x)}{\|\tilde{w}\| \|\rho_S(x)\|} \geq \frac{\gamma}{4} \right) \geq 1 - \delta^2.$$

The same analysis applies for the case that $\ell(x) = -1$.

Since the above holds for any $x \in \mathbf{Good}$, it is also true for random $x \in \mathbf{Good}$, which implies by Markov's inequality that with probability $1 - \delta$, the vector \tilde{w} has error at most δ at margin $\gamma/4$ over P restricted to points $x \in \mathbf{Good}$. Adding back the ϵ probability mass of points x not satisfying

$$\mathbf{E}_{x' \sim P}[w(x')K(x, x') | \ell(x) = \ell(x')] \geq \mathbf{E}_{x' \sim P}[w(x')K(x, x') | \ell(x) \neq \ell(x')] + \gamma,$$

yields the theorem. ■

Theorem 3.3.2 states that if K is a good similarity function then with high probability there exists a low-error (at most $\epsilon + \delta$) large-margin (at least $\frac{\gamma}{4}$) separator in the transformed space under mapping ρ_S . Furthermore the dimensionality of this space is not too large, only $O(\frac{1}{\gamma^2} \log \frac{1}{\delta})$. Thus, all we need now to learn well is to draw a new, fresh sample, map it into the transformed space using ρ_S , and then apply a good algorithm for learning linear separators in the new space.¹⁰

Remark: Standard margin bounds imply that if K is a good *kernel* function, then with high probability, the points in a random sample $S = \{x_1, \dots, x_d\}$ can be assigned weights so that the resulting vector defines a low-error large-margin separator in the ϕ -space. Definition 3.3.3 can be viewed as requiring that each point x' have a weight $w(x')$ that is solely a function of the example itself and not the set S to which it belongs. The results in Section 3.3.3 below imply that if K is a good kernel, then these “sample independent” weights must exist as well.

Variations Tailored to Efficient Algorithms

Our implication is that there will exist a low-error large-margin separator in the transformed space, so that we can then run a standard linear-separator learning algorithm. Technically, however, the guarantee for algorithms such as SVM and Perceptron is not that they necessarily find the *minimum-error* separator on their data (which is NP-hard) but rather that they find the separator that minimizes the *total distance* one

¹⁰ One interesting aspect to notice is that we can use *unlabeled examples* instead of labeled examples when defining the mapping $\rho_S(x)$. However, if the data distribution is highly unbalanced, say with substantially more negatives than positives, then the mapping may no longer have the large-margin property.

would need to move points to put them on the correct side by the given margin. Thus, our worst-case guarantee for SVM and Perceptron is only that they find a separator of error $O((\epsilon + \delta)/\gamma)$, though such algorithms are known to do quite well in practice and the same issue would apply for the definition of an (ϵ, γ) -good kernel.

We can also modify our definition to capture the notion of good similarity functions for the SVM and Perceptron algorithms as follows:

Definition 3.3.4 (tailored to SVM and Perceptron) *A similarity function K is an (ϵ, γ) -good similarity function in hinge loss for a learning problem P if there exists a weighting function $w(x') \in [0, 1]$ for all $x' \in X$ such that*

$$\frac{1}{\gamma} \mathbf{E}_x \left[\max(\gamma_x, 0) \right] \leq \epsilon,$$

where $\gamma_x = \mathbf{E}_{x' \sim P} [w(x')K(x, x') | \ell(x) \neq \ell(x')] + \gamma - \mathbf{E}_{x' \sim P} [w(x')K(x, x') | \ell(x) = \ell(x')]$.

In other words, we are asking: on average, by how much would a random example x fail to satisfy the desired γ separation between the weighted similarity to examples of its own label and the weighted similarity to examples of the other label (this is γ_x). This quantity is then scaled by $1/\gamma$.

By applying the same analysis as in the proof of Theorem 3.3.2, one can show that given a similarity function satisfying this definition, we can use SVM in the transformed space to achieve error $O(\epsilon + \delta)$.

Combining Multiple Similarity Functions

Suppose that rather than having a single similarity function, we were instead given n functions K_1, \dots, K_n , and our hope is that some convex combination of them will satisfy Definition 3.3.3. Is this sufficient to be able to learn well? (Note that a convex combination of similarity functions is guaranteed to have range $[-1, 1]$ and so be a legal similarity function.) The following generalization of Theorem 3.3.2 shows that this is indeed the case, though the margin parameter drops by a factor of \sqrt{n} . This result can be viewed as analogous to the idea of learning a kernel matrix studied by [93] except that rather than explicitly learning the best convex combination, we are simply folding the learning process into the second stage of the algorithm.

Theorem 3.3.3 *Suppose K_1, \dots, K_n are similarity functions such that some (unknown) convex combination of them is (ϵ, γ) -good. If one draws a set S from P containing $d = (4/\gamma)^2 \ln(2/\delta)$ positive examples $S^+ = \{y_1, y_2, \dots, y_d\}$ and d negative examples $S^- = \{z_1, z_2, \dots, z_d\}$, then with probability at least $1 - \delta$, the mapping $\rho_S : X \rightarrow R^{2nd}$ defined as*

$$\rho_S(x) = (K_1(x, y_1), \dots, K_n(x, y_d), K_1(x, z_1), \dots, K_n(x, z_d))$$

has the property that the induced distribution $\rho_S(P)$ in R^{2nd} has a separator of error at most $\epsilon + \delta$ at margin at least $\gamma/(4\sqrt{n})$.

Note that the above argument actually shows something a bit stronger than Theorem 3.3.3. In particular, if we define $\alpha = (\alpha_1, \dots, \alpha_n)$ to be the mixture vector for the optimal K , then we can replace the margin bound $\gamma/(4\sqrt{n})$ with $\gamma/(4\|\alpha\|\sqrt{n})$. For example, if α is the uniform mixture, then we just get the bound in Theorem 3.3.2 of $\gamma/4$.

Multi-class Classification

We can naturally extend all our results to multi-class classification. In particular, the analog of Definition 3.3.3 is that we require most examples to have their average weighted similarity to points of their own class be at least γ greater than their average weighted similarity to *each* of the other classes. Assume for

concreteness that there are r possible labels, and denote the space of possible labels by $Y = \{1, \dots, r\}$; thus, by a *multi-class learning problem* we mean a distribution P over labeled examples $(x, \ell(x))$, where $x \in X$ and $\ell(x) \in Y$. In this case, Definition 3.3.3 is not sufficient for learning since it would allow for the situation that, say, class-labels 1 and 2 are very different from 3 and 4, but we cannot distinguish 1 from 2 or 3 from 4. Instead, a natural extension of Definition 3.3.3 that suffices for learning in this case is the following:

Definition 3.3.5 (main, multi-class) *A similarity function K is an (ϵ, γ) -good similarity function for a learning problem P if there exists a bounded weighting function w over X ($w(x') \in [0, 1]$ for all $x' \in X$) such that at least a $1 - \epsilon$ probability mass of examples x satisfy:*

$$\mathbf{E}_{x' \sim P}[w(x')K(x, x') | \ell(x) = \ell(x')] \geq \mathbf{E}_{x' \sim P}[w(x')K(x, x') | \ell(x) = i] + \gamma \text{ for all } i \in Y, i \neq \ell(x)$$

One can then learn using standard adaptations of linear-separator algorithms to the multiclass case (e.g., see [67]).

3.3.3 Good Kernels are Good Similarity Functions

Our main result in [16] shows that a good kernel in the standard sense (i.e. a kernel with a large margin separator in its implicit ϕ -space) will also satisfy Definition 3.3.3, though with some degradation of the parameters. Formally, we have shown in [16] that:

Theorem 3.3.4 *If K is a (ϵ, γ) -good kernel function, then for any $\epsilon_{acc} > 0$, K is a $(\frac{8(\epsilon+2\epsilon_{acc})}{\gamma}, \frac{1}{2M(\gamma, \epsilon_{acc})})$ -good similarity function, where $M(\gamma, \epsilon) = \frac{1}{\epsilon} \left(\frac{3}{\gamma^2} + \log(\frac{1}{\epsilon}) \right)$.*

For example, if K is a $(0, \gamma)$ -good kernel function, then for any ϵ' it is also a $(\epsilon', \tilde{O}(\epsilon'\gamma^3))$ -good similarity function.

This result was later strengthened in [23, 119], leading to a better understanding of the problem.

3.3.4 Perspective

The main contribution of our work [16] is to develop a theory of learning with similarity functions: namely, of when a similarity function is good for a given learning problem, that is more general and in terms of more tangible quantities than the standard theory of kernel functions. We provide a definition that we show is both sufficient for learning and satisfied by the usual large-margin notion of a good kernel. Moreover, the similarity properties we consider do not require reference to implicit high-dimensional spaces nor do they require that the similarity function be positive semidefinite. In this way, we provide the first rigorous explanation showing why a kernel function that is good in the large-margin sense can also formally be viewed as a good similarity function, thereby giving formal justification to the standard intuition about kernels.

Chapter 4

Mechanism Design, Machine Learning, and Pricing Problems

4.1 Generic Random Sampling Based Mechanisms for Revenue Maximization in an Unlimited Supply Setting

In this chapter we make an explicit connection between machine learning and mechanism design. In particular, we show how Sample Complexity techniques in Statistical Learning Theory can be used to reduce problems of incentive-compatible mechanism design to standard algorithmic questions, for a wide range of revenue-maximizing problems in an unlimited supply setting.

4.1.1 Problem Formulation

In recent years there has been substantial work on problems of algorithmic mechanism design. These problems typically take a form similar to classic algorithm design or approximation-algorithm questions, except that the inputs are each given by *selfish agents* who have their own interest in the outcome of the computation. As a result it is desirable that the mechanisms (the algorithms and protocol) be *incentive compatible* — meaning that it is in each agent’s best interest to report its true value — so that agents do not try to game the system. This requirement can greatly complicate the design problem.

In our work [20] we consider the design of mechanisms for one of the most fundamental economic objectives: *profit maximization*. Agents participating in such a mechanism may choose to falsely report their preferences if it might benefit them. What we show, however, is that so long as the number of agents is sufficiently large as a function of a measure of the complexity of the mechanism design problem, we can apply sample-complexity techniques from learning theory to reduce this problem to standard algorithmic questions in a broad class of settings. It is useful to think of the techniques we develop in the context of designing an auction to sell some goods or services, though they also apply in more general scenarios.

In a seminal paper Myerson [100] derives the optimal auction for selling a single item given that the bidders’ true valuations for the item come from some known *prior distribution*. His mechanism generalizes trivially to any single-parameter agent setting with arbitrary supply constraints or costs to the auctioneer for the outcome produced. Following a trend in the recent computer science literature on optimal auction design, we consider the *prior-free* setting in which there is no underlying distribution on valuations and we wish to perform well for any (sufficiently large) set of bidders. In absence of a known prior distribution we will use machine learning techniques to estimate properties of the bidders’ valuations. We consider the *unlimited supply* setting in which this problem is conceptually simpler because

there are no infeasible allocations; though, it is often possible to obtain results for limited supply or with cost functions on the outcome via reduction to the unlimited supply case [7, 64, 75]. Research in optimal prior-free auction design is important for optimal auction design because it directly links inaccurate distributional knowledge typical of small markets with loss in performance.

Implicit in mechanism design problems is the fact that the selfish agents that will be participating in the mechanism have *private information* that is known only to them. Often this private information is simply the agent's valuation over the possible outcomes the mechanism could produce. For example, when selling a single item (with the standard assumption that an agent only cares if they get the item or not and not whether another agent gets it) this valuation is simply how much they are willing to pay for the item. There may also be *public information* associated with each agent. This information is assumed to be available to the mechanism. Such information is present in structured optimization problems such as the *knapsack auction problem* [7] and *multicast auction problem* [64] and is the natural way to generalize optimal auction design for independent but non-identically distributed prior distributions (which are considered by Myerson [100]) to the prior-free setting. There are many standard economic settings where such public information is available, e.g., in the college tuition mechanism, in-state or out-of-state residential status is public; for acquiring a loan, a consumer's credit report is public information; for automobile insurance, driving records, credit reports, and the make and color of the vehicle are public information.

A fundamental building block of an incentive compatible mechanism is an *offer*. For full generality an offer can be viewed as an incentive compatible mechanism for one agent. As an example, if we are selling multiple units of a single item, an offer could be a *take-it-or-leave-it* price per unit. A rational agent would accept such an offer if it is lower than the agent's valuation for the item and reject if it is greater. Notice that if all agents are given the same take-it-or-leave-it price then the outcome is *non-discriminatory* and the same price is paid by all winners. Prior-free auctions based on this type of non-discriminatory pricing have been considered previously (see, e.g., [75]).

One of the main motivations of this work is to explore *discriminatory pricing* in optimal auction design. There are two standard means to achieve discriminatory pricing. The first, is to discriminate based on the public information of the consumer. Naturally, loans are more costly for individuals with poor credit scores, car insurance is more expensive for drivers with points on their driving record, and college tuition at state run universities is cheaper for students that are in-state residents. In this setting a reasonable offer might be a mapping from the public information of the agents to a take-it-or-leave-it price. We refer to these types of offers as *pricing functions*. The second standard means for discriminatory pricing is to introduce similar products of different qualities and price them differently. Consumers who cannot afford the expensive high-quality version may still purchase an inexpensive low-quality version. This practice is common, for example, in software sales, electronics sales, and airline ticket sales. An offer for the multiple good setting could be a take-it-or-leave it price for each good. An agent would then be free to select the good (or bundle of goods) with the (total) price that they most prefer. We refer to these types of offers as *item pricings*.

Notice that allowing offers in the form of pricing functions and item pricings, as described above, provides richness to both algorithmic and mechanism design questions. This richness; however, is not without cost. Our performance bounds are parameterized by a suitable notion of the *complexity* of the class of allowable offers. It is natural that this kind of complexity should affect the ability of a mechanism to optimize. It is easier to approximate the optimal offer from a simple classes of offers, such as take-it-or-leave-it prices for a single item, than it is for a more complex class of offers, such as take-it-or-leave-it prices for multiple items. Our prior-free analysis makes the relationship between a mechanism's performance and the complexity of allowed offers precise.

We phrase our auction problem generically as: given some class of reasonable offers, can we construct

an incentive-compatible auction that obtains profit close to the profit obtained by the optimal offer from this class? The auctions we discuss are generalizations of the random sampling auction of Goldberg et al. [74]. These auctions make use of a (non-incentive-compatible) algorithm for computing a best (or approximately best) offer from a given class for any set of consumers. Thus, we can view this construction as reducing the optimal mechanism design problem to the optimal algorithm design problem.

The idea of the reduction is as follows. Let \mathcal{A} be an algorithm (exact or approximate) for the purely algorithmic problem of finding the optimal offer in some class \mathcal{G} for any given set of consumers S with known valuations. Our auction, which does not know the valuations a priori, asks the agents to report their valuations (as bids), splits agents randomly into two sets S_1 and S_2 , runs the algorithm \mathcal{A} separately on each set (perhaps adding an additional penalty term to the objective to penalize solutions that are too “complex” according to some measure), and then applies the offer found for S_1 to S_2 and the offer found on S_2 to S_1 . The incentive compatibility of this auction allows us to assume that the agents will indeed report their true valuations. Sample-complexity techniques adapted from machine learning theory can then give a guarantee on the quality of the results if the market size is sufficiently large compared to a measure of complexity of the class of possible solutions. From an economics perspective, this can be viewed as replacing the Bayesian assumption that bidders come from a known prior distribution (e.g., as in Myerson’s work [100]) with the use of learning, over a random subset S_1 of an arbitrary set of bidders S , to get enough information to apply to S_2 (and vice versa).

It is easy to see that as the size of the market grows, the law of large numbers indicates that the above approach is asymptotically optimal. This is not surprising as conventional economic wisdom suggests that even the approach of market analysis followed by the Bayesian optimal mechanism would incur negligibly small loss compared to the Bayesian optimal mechanism which was endowed with foreknowledge of the distribution. In contrast, the main contribution of this work is to give a mechanism with upper bounds on the convergence rate, i.e., the relationship between the size of the market, the approximation factor, and the complexity of the class of reasonable offers.

Our contributions: In our work [20] we present a general framework for reducing problems of incentive-compatible mechanism design to standard algorithmic questions, for a broad class of revenue-maximizing pricing problems. To obtain our bounds we use and extend sample-complexity techniques from machine learning theory (see [11, 42, 88, 121]) and to design our mechanisms we employ machine learning methods such as *structural risk minimization*. In general we show that an algorithm (or β -approximation) can be converted into a $(1 + \epsilon)$ -approximation (or $\beta(1 + \epsilon)$ -approximation) for the optimal mechanism design problem when the market size is at least $O(\beta\epsilon^{-2})$ times a reasonable notion of the complexity of the class of offers considered. Our formulas relating the size of the market to the approximation factor give upper bounds on the performance loss due to unknown market conditions and we view these as bounds on the *convergence rate* of our mechanism. From a learning perspective, the mechanism-design setting presents a number of technical challenges when attempting to get good bounds: in particular, the payoff function is discontinuous and asymmetric, and the payoffs for different offers are non-uniform. For example, we develop bounds based on a different notion of *covering number* than typically used in machine learning, in order to obtain results that are more meaningful for our setting.

We instantiate our framework for a variety of problems, some of which have been previously considered in the literature, including:

Digital Good Auction Problem: The *digital good auction problem* considers the sale of an unlimited number of units of an item to indistinguishable consumers, and has been considered by Goldberg et al. [74] and a number of subsequent papers. As argued in [74] the only reasonable offers for this setting are take-it-or-leave-it prices.

The analysis techniques developed in our work give a *simple* proof that the random sampling auction

(related to that of [74]) obtains a $(1 - \epsilon)$ fraction of the optimal offer as long as the market size is at least $O(\frac{h}{\epsilon^2} \log \frac{1}{\epsilon})$ (where h is an upper bound on the valuation of any agent).

Attribute Auction Problem: The *attribute auction problem* is an abstraction of the problem using discriminatory prices based on public information (a.k.a., *attributes*) of the agents. A seller can often increase its profit by using discriminatory pricing: for example, the motion picture industry uses region encodings so that they can charge different prices for DVDs sold in different markets. Further, in many generalizations of the digital good auction problem, the agents are distinguishable via public information so the techniques exposed in the study of attribute auctions are fundamental to the study of profit maximization in general settings.

Here a reasonable class of offers to consider are mappings from the agents' attributes to take-it-or-leave-it prices. As such, we refer to these offers as *pricing functions*. For example, for one-dimensional attributes, a natural class of pricing functions might be piece-wise constant functions with k prices, as studied in [36]. In our work we give a *general* treatment that can be applied to arbitrary classes of pricing functions. For example, if attributes are multi-dimensional, pricing functions might involve partitioning agents into markets defined by coordinate values or by some natural clustering, and then offering a constant price or a price that is some other simple function of the attributes within each market. Our bounds give a $(1 + \epsilon)$ -approximation when the market size is large in comparison to ϵ^{-2} scaled by a suitable notion of the complexity of the class of offers.

Combinatorial Auction Problem: We also consider the goal of profit maximization in an unlimited-supply combinatorial auction. This generalizes the digital good auction and exemplifies the problem of discriminatory pricing through the sale of multiple products. The setting here is the following. We have m different items, each in unlimited supply (like a supermarket), and bidders have valuations over *subsets* of items. Our goal is to achieve revenue nearly as large as the best revenue that uses take-it-or-leave-it prices for each item individually, i.e., the best *item-pricing*.

For arbitrary item pricings we show that our reduction has a convergence rate of $\tilde{\Omega}\left(\frac{hm^2}{\epsilon^2}\right)$ no matter how complicated those bidders' valuations are (where the $\tilde{\Omega}$ hides terms logarithmic in n , the number of agents; m , the number of items; and h , the highest valuation). If instead the specification of the problem constrains the item prices to be integral (e.g., in pennies) or the consumers to be *unit-demand* (desiring only one of several items) or *single-minded* (desiring only a particular bundle of items) then our bound improves to $\tilde{\Omega}\left(\frac{hm}{\epsilon^2}\right)$. This improves on the bounds given by [72] for the unit-demand case by roughly a factor of m .

A special case of this setting is the problem of auctioning the right to traverse paths in a network. When the network is a tree and each user wants to reach the root (like drivers commuting into a city or a multicast tree in the Internet), Guruswami et al. [76] give an exact algorithm for the algorithmic problem to which our reduction applies as noted above.

Related Work: Several papers [36, 40] have applied machine learning techniques to mechanism design in the context of maximizing revenue in online auctions. The online setting is more difficult than the "batch" setting we consider, but the flip-side is that as a result, that work only applies to quite simple mechanism design settings where the class \mathcal{G} of allowable offers has small size and can be easily listed. Also, in a similar spirit to the goals of our work, Awerbuch et al. [14] give reductions from online mechanism design to online optimization for a broad class of revenue maximization problems. Their work compares performance to the sum of bidders' valuations, a quite demanding measure. As a result, however, their approximation factors are necessarily logarithmic rather than $(1 + \epsilon)$ as in our results.

The reader will notice that in converting an algorithm (or approximation algorithm) for finding the

best offer in \mathcal{G} into an incentive-compatible mechanism, as described above, we produce a mechanism whose outcome is not simply that of a single offer applied to all consumers. For example, even in the simplest case of auctioning a digital good to indistinguishable bidders, we compare our performance to the best take-it-or-leave-it price, and yet the auction itself does not in fact offer each bidder the same price (all bidders in S_1 get the same price, and all bidders in S_2 get the same price, but those two prices may be different). In fact, Goldberg and Hartline [73] show that this sort of behavior is necessary: it is not possible for an incentive-compatible auction to approximately maximize profit and offer all the bidders the same price.

Structure of this chapter: We briefly overview in the following our model, present a few examples, as well as our most basic mechanisms. Precise analysis of convergence rates for the exemplary examples mentioned before, as well as refined bounds and mechanisms appear in our work in [20].

4.1.2 Definitions

A *market* consists of a set of agents, $S = \{1, \dots, n\}$, and space of possible outcomes \mathcal{O} . We consider *unlimited supply* allocation problems where \mathcal{O}_i is set of possible allocations to agent i and $\mathcal{O} = \mathcal{O}_1 \times \dots \times \mathcal{O}_n$ (i.e., all possible combinations of allocations are feasible). Except where noted, we assume there is no cost to the mechanism for producing any outcome.

Let \mathcal{G} denote a class of *offers*. The precise notion of what an offer is will depend on the specific application; however, our bounds apply in the abstract setting where, given an agent's preferences, an offer maps this agent, via their preferences, to a profit obtained by the mechanism. For an offer $g \in \mathcal{G}$, we let $g(i)$ denote the profit obtained by the mechanism from making offer g to bidder i (which could be 0). The profit from a set of bidders $S' \subseteq S$, all receiving offer g , is denoted $g(S') = \sum_{i \in S'} g(i)$. Given the true preferences of S and a class of offers \mathcal{G} , a natural *algorithmic optimization problem* is to find the $g \in \mathcal{G}$ with maximum profit, i.e., $\text{opt}_{\mathcal{G}} = \text{argmax}_{g \in \mathcal{G}} g(S)$. Let $\text{OPT}_{\mathcal{G}} = \max_{g \in \mathcal{G}} g(S)$ be this maximum profit.

To make this discussion more precise we consider the setting in which an agent preference is to maximize their *quasi-linear utility*. This is one of the most studied settings in the mechanism design literature. In such a setting each agent has a valuation function v_i which gives their value for an outcome $o_i \in \mathcal{O}_i$. The utility they wish to maximize is then their valuation for the outcome, $v_i(o_i)$, minus the payment they must make. Given an offer which specifies required payments for each possible allocation, the agent prefers the one that maximizes their utility.

Each agent may also be distinguishable via publicly known attribute values. An offer may use such publicly known attributes for price discrimination. We let pub_i be the public attribute of agent i . We view an offer as a function from an agent's attribute to outcome and payment pairs. Composing this with the agent's quasi-linear preferences as described above, an offer maps each agent to a profit to the mechanism.

Consider the following examples. If we are selling a single item in unlimited supply as in the digital good auction problem, an offer might be a single take-it-or-leave-it price p (i.e., a price p on the outcome of taking the item and a price zero on the outcome of not taking it). The profit when making such an offer to an agent with value v_i for the item is zero if $p > v_i$ and p if $p \leq v_i$. For attribute-auctions, an offer might be a take-it-or-leave-it price that depends on the bidder's public information: for instance in the case of college tuition an offer might be one price for in-state students and one price for out-of-state students. If there are multiple items for sale as in combinatorial auctions, an offer might assign separate take-it-or-leave-it prices on each item, or be even more complicated as in bundle-pricing. We discuss these examples in greater detail below.

In our work [20] we focus on the *mechanism design problem* of designing an incentive-compatible auction that obtains a profit close to $\text{OPT}_{\mathcal{G}}$. Incentive compatibility constraints require that it be in each

bidder's interest to truthfully report their preferences to the mechanism. For example, in the quasi-linear case we say a mechanism is incentive compatible if there is no profile of the other bidders' bids such that a given bidder i could obtain a strictly higher utility by bidding some $b_i \neq v_i$ rather than his true valuation, v_i .

Notice that when we offer g to an agent i , we assume that the mechanism chooses the outcome and payment specified by the offer that maximizes the agent's utility given their reported preferences. As a result, the mechanism that always makes each agent a single offer g is inherently incentive compatible; there is nothing an agent can declare as their valuation to get a different offer and, given the offer, the outcome and payment selected by the mechanism are the ones that the bidder most prefers. In fact the following more general result is easy to show:

Fact 4.1.1 *A mechanism is incentive compatible if the offer made to any bidder does not depend on the bidder's bid.*

Our approach to incentive compatible mechanism design is via reduction to the algorithmic optimization problem. The algorithmic optimization problem is to determine the $g \in \mathcal{G}$ maximizing $g(S)$, for a given market, S , and some desired class of offers, \mathcal{G} (i.e., compute $\text{opt}_{\mathcal{G}}(S)$). This computational problem is interesting in its own right; however, we will not consider it here. Instead we will assume that we are given an algorithm that either exactly solves the algorithmic optimization problem or approximates it. Some of our techniques also make use of the existence of an algorithm that optimizes over the profit of an offer minus some penalty term that is related to the complexity of the offer, i.e., $\max_{g \in \mathcal{G}} g(S) - \text{pen}_g(S)$.

One final point at this level of generality: we will assume that h is an upper bound on the value of $g(i)$ for all $i \in S$ and $g \in \mathcal{G}$; that is, no individual bidder can influence the total profit by more than h . For example, this will occur if $v_i(o) \leq h$ for all bidders i and outcomes o . This term will come into our general sample-complexity bounds. Auctions that make use of the technique of structural risk minimization will need to know h in advance.

Examples The following examples precisely illustrate the relationship between the outcome of the mechanism, offers, valuations, and attributes.

Digital Good Auction: The digital good auction models an auction of a single item in unlimited supply to indistinguishable bidders. Here the possible outcomes for bidder i is $\mathcal{O}_i = \{0, 1\}$ where $o_i = 1$ represents bidder i receiving a copy of the good and $o_i = 0$ otherwise. We normalize their valuation function $v_i(0) = 0$ and opt for a simple shorthand notation of $v_i = v_i(1)$ as the bidders privately known valuation for receiving the good. As described in the introduction, in this setting the bidders have no public information. Here, a natural class of offers, \mathcal{G} , is the class of all take-it-or-leave-it prices. For bidder i with valuation v_i and offer $g_p =$ "take the good for $\$p$, or leave it" the profit is

$$g_p(i) = \begin{cases} p & \text{if } p \leq v_i \\ 0 & \text{otherwise.} \end{cases}$$

Attribute Auctions: This is the same as the digital good setting except now each bidder i is associated a public attribute, $\text{pub}_i \in \mathcal{X}$, where \mathcal{X} is the *attribute space*. We view \mathcal{X} as an abstract space, but one can envision it as \mathcal{R}^d , for example. Let \mathcal{P} be a class of pricing functions from \mathcal{X} to \mathcal{R}_+ , such as all linear functions, or all functions that partition \mathcal{X} into k markets in some natural way (say, based on distance to k cluster centers) and offer a different price in each. Let \mathcal{G} be the class of take-it-or-leave-it offers induced by \mathcal{P} . That is, if $q \in \mathcal{P}$ is a pricing function, then the offer $g_q \in \mathcal{G}$ induced by q is: "for bidder i , take good for $\$q(\text{pub}_i)$, or leave it". The profit to the mechanism

from bidder i with valuation v_i and public information pub_i is

$$g_q(i) = \begin{cases} q(pub_i) & \text{if } q(pub_i) \leq v_i, \\ 0 & \text{otherwise.} \end{cases}$$

We have analysed in [20] several interesting classes of pricing functions.

Combinatorial Auctions: Here we have a set J of m distinct items, each in unlimited supply. Each consumer has a private valuation $v_i(J')$ for each bundle $J' \subseteq J$ of items, which measures how much receiving bundle J' would be worth to the consumer i (again we normalize such that $v_i(\emptyset) = 0$). For simplicity, we assume bidders are indistinguishable, i.e., there is no public information. A natural class of offers \mathcal{G} (studied in [76]) is the class of functions that assign a separate price to each item, such that the price of a bundle is just the sum of the prices of the items in it, called *item-pricing*. For price vector $\mathbf{p} = (p_1, \dots, p_m)$ let the offer $g_{\mathbf{p}} =$ “for bundle J' , pay $\sum_{j \in J'} p_j$ ”. The profit for bidder i on offer $g_{\mathbf{p}}$ is

$$g_{\mathbf{p}}(i) = \sum \left\{ p_j : j \in \operatorname{argmax}_{J' \subseteq J} \left[v_i(J') - \sum_{j' \in J'} p_{j'} \right] \right\}.$$

Marginal Cost Auctions with Budgets: To illustrate an interesting model with agents in a non-quasi-linear setting consider the case each bidder i 's preference is given tuple (B_i, v_i) where B_i is their budget and v_i is their value-per-unit received. Possible allocations for bidder i , \mathcal{O}_i , are non-negative real numbers corresponding to the number of units they receive. Assuming their total payment is less than their budget, bidder i 's utility is simply $v_i o_i$ minus their payment; an agent's utility when payments exceed their budget is negative infinity.

We assume that the seller has a fixed marginal cost c for producing a unit of the good. Consider the class of offers \mathcal{G} with $g_p =$ “pay $\$p$ per unit received”. A bidder i faced with offer g_p with $p < v_i$ will maximize their utility by buying enough units to exactly exhaust their budget. The payoff to the auctioneer for this bidder i is therefore B_i less c times the number of units the bidder demands. I.e.,

$$g_p(i) = \begin{cases} B_i - cB_i/p & \text{if } p \leq v_i, \\ 0 & \text{otherwise.} \end{cases}$$

This model is quite similar to one considered by Borgs et al. [44]. Though we do not explicitly analyze this setting, it is simple to apply our generic analyses to get reasonable bounds.

4.1.3 Generic Reductions

As mentioned, we are interested in reducing incentive-compatible mechanism design to the (non incentive-compatible) algorithmic optimization problem. Our reductions in [20] are based on *random sampling*. Let \mathcal{A} be an algorithm (exact or approximate) for the algorithmic optimization problem over \mathcal{G} . The simplest mechanism that we consider, which we call $\text{RSO}_{(\mathcal{G}, \mathcal{A})}$ (Random Sampling Optimal offer), is the following generalization of the random sampling digital-goods auction from [74]:

0. Bidders commit to their preferences by submitting their bids.
1. Randomly split the bidders into two groups S_1 and S_2 by flipping a fair coin for each bidder to determine its group.
2. Run \mathcal{A} to determine the best (or approximately best) offer $g_1 \in \mathcal{G}$ over S_1 , and similarly the best (or approximately best) $g_2 \in \mathcal{G}$ over S_2 .

3. Finally, apply g_1 to all bidders in S_2 and g_2 to all bidders in S_1 .

In [20] we also consider various more refined versions of $RSO_{(\mathcal{G}, \mathcal{A})}$ that discretize \mathcal{G} or perform some type of *structural risk minimization* (in which case we will need to assume \mathcal{A} can optimize over the modifications made to \mathcal{G}).

Note 1: One might think that the “leave-one-out” mechanism, where the offer made to a given bidder i is the best offer for all other bidders, i.e., $\text{opt}_{\mathcal{G}}(S \setminus \{i\})$, would be a better mechanism than the random sampling mechanism above. However, as pointed out in [74, 75], such a mechanism (and indeed, any symmetric deterministic mechanism) has poor worst-case revenue. Furthermore, even if bidders’ valuations are independently drawn from some distribution, the leave-one-out revenue can be much less stable than $RSO_{(\mathcal{G}, \mathcal{A})}$ in that it may have a non-negligible probability of achieving revenue that is far from optimal, whereas such an event is exponentially small for $RSO_{(\mathcal{G}, \mathcal{A})}$.¹

Note 2: The reader will notice that in converting an algorithm for finding the best offer in \mathcal{G} into an incentive-compatible mechanism, we produce a mechanism whose outcome is not simply that of a single offer applied to all consumers. For example, even in the simplest case of auctioning a digital good to indistinguishable bidders, we compare our performance to the best take-it-or-leave-it price, and yet the auction itself does not in fact offer each bidder the same price (all bidders in S_1 get the same price, and all bidders in S_2 get the same price, but those two prices may be different). In fact, Goldberg and Hartline [73] show that this sort of behavior is necessary: it is not possible for an incentive-compatible auction to approximately maximize profit and offer all the bidders the same price.

4.1.4 Generic Analyses

The following theorem shows that the random sampling auction incurs only a small loss in performance if the profit of the optimal offer is large in comparison to the logarithm of the number of offers we are choosing from. In [20] we present more sophisticated techniques for bounding the *effective size* (or complexity) of \mathcal{G} that can yield even stronger guarantees.

Theorem 4.1.2 *Given the offer class \mathcal{G} and a β -approximation algorithm \mathcal{A} for optimizing over \mathcal{G} , then with probability at least $1 - \delta$ the profit of $RSO_{(\mathcal{G}, \mathcal{A})}$ is at least $(1 - \epsilon)\text{OPT}_{\mathcal{G}}/\beta$ as long as*

$$\text{OPT}_{\mathcal{G}} \geq \beta \frac{18h}{\epsilon^2} \ln \left(\frac{2|\mathcal{G}|}{\delta} \right).$$

Notice that this bound holds for all ϵ and δ simultaneously as these are not parameters of the mechanism. In particular, this bound and those given by the two immediate corollaries, below, show how the approximation factor improves as a function of market size.

Corollary 4.1.3 *Given the offer class \mathcal{G} and a β -approximation algorithm \mathcal{A} for optimizing over \mathcal{G} , then with probability at least $1 - \delta$, the profit of $RSO_{(\mathcal{G}, \mathcal{A})}$ is at least $(1 - \epsilon)\text{OPT}_{\mathcal{G}}/\beta$, when $\text{OPT}_{\mathcal{G}} \geq n$ and the number of bidders n satisfies*

$$n \geq \frac{18h\beta}{\epsilon^2} \ln \left(\frac{2|\mathcal{G}|}{\delta} \right).$$

Corollary 4.1.4 *Given the offer class \mathcal{G} and a β -approximation algorithm \mathcal{A} for optimizing over \mathcal{G} then with probability at least $1 - \delta$, the profit of $RSO_{(\mathcal{G}, \mathcal{A})}$ is at least*

$$(1 - \epsilon)\text{OPT}_{\mathcal{G}}/\beta - \frac{18h\beta}{\epsilon^2} \ln \left(\frac{2|\mathcal{G}|}{\delta} \right).$$

¹For example, say we are selling just one item and the distribution over valuations is 50% probability of valuation 1 and 50% probability of valuation 2. If we have n bidders, then there is a nontrivial chance (about $1/\sqrt{n}$) that there will be the exact same number of each type ($n/2$ bidders with valuation 1 and $n/2$ bidders with valuation 2), and the mechanism will make the wrong decision on everybody. The $RSO_{(\mathcal{G}, \mathcal{A})}$ mechanism on the other hand has only an exponentially small probability of doing this poorly.

If bidders' valuations are in the interval $[1, h]$ and the take-it-or-leave-it offer of \$1 is in \mathcal{G} , then the condition $\text{OPT}_{\mathcal{G}} \geq n$ is trivially satisfied and Corollary 4.1.3 can be interpreted as giving a bound on the *convergence rate* of the random sampling auction. Corollary 4.1.4 is a useful form of our bound when considering structural risk minimization and it also matches the form of bounds given in prior work (e.g., [36]).

For example, in the digital good auction with the class of offers \mathcal{G}_ϵ consisting of all take-it-or-leave-it offers in the interval $[1, h]$ discretized to powers of $1 + \epsilon$, we have $\text{OPT}_{\mathcal{G}_\epsilon} \geq n$ (since each bidder's valuation is at least 1), $\beta = 1$ (since the algorithmic problem is easy), and $|\mathcal{G}_\epsilon| = \lceil \log_{1+\epsilon} h \rceil$. So, Corollary 4.1.3 states that $O(\frac{h}{\epsilon^2} \log \log_{1+\epsilon} h)$ bidders are sufficient to perform nearly as well as optimal (we derive better bounds for this problem in [20]).

In general we will give our bounds in a similar form as Theorem 4.1.2, knowing that bounds of the form of Corollary 4.1.3 and 4.1.4 can be easily derived. The only exceptions are the structural risk minimization results.

In the remainder of this section we prove Theorem 4.1.2. We start with a lemma that is key to our analysis.

Lemma 4.1.5 *Given S , an offer g satisfying $g(i) \leq h$ for all $i \in S$, and a profit level p , if we randomly partition S into S_1 and S_2 , then the probability that $|g(S_1) - g(S_2)| \geq \epsilon \max\{g(S), p\}$ is at most $2e^{-\frac{\epsilon^2 p}{2h}}$.*

Proof: Let Y_1, \dots, Y_n be i.i.d. random variables that define the partition of S into S_1 and S_2 : that is, Y_i is 1 with probability $\frac{1}{2}$ and Y_i is 2 with probability $\frac{1}{2}$. Let $t(Y_1, \dots, Y_n) = \sum_{i: Y_i=1} g(i)$. So, as a random variable, $g(S_1) = t(Y_1, \dots, Y_n)$ and clearly $\mathbf{E}[t(Y_1, \dots, Y_n)] = \frac{g(S)}{2}$. Assume first that $g(S) \geq p$. From the McDiarmid concentration inequality we get:

$$\Pr \left\{ \left| g(S_1) - \frac{g(S)}{2} \right| \geq \frac{\epsilon}{2} g(S) \right\} \leq 2e^{-\frac{1}{2} \epsilon^2 g(S)^2 / \sum_{i=1}^n g(i)^2}.$$

Since

$$\sum_{i=1}^n g(i)^2 \leq \max_i \{g(i)\} \sum_{i=1}^n g(i) \leq hg(S),$$

we obtain:

$$\Pr \left\{ \left| g(S_1) - \frac{g(S)}{2} \right| \geq \frac{\epsilon}{2} g(S) \right\} \leq 2e^{-\frac{\epsilon^2 g(S)}{2h}}.$$

Moreover, since $g(S_1) + g(S_2) = g(S)$ and $g(S) \geq p$, we obtain $\Pr\{|g(S_1) - g(S_2)| \geq \epsilon g(S)\} \leq 2e^{-\epsilon^2 p / (2h)}$, as desired. Consider now the case that $g(S) < p$. Again, using the McDiarmid inequality we have

$$\Pr\{|g(S_1) - g(S_2)| \geq \epsilon p\} \leq 2e^{-\frac{1}{2} \epsilon^2 p^2 / \sum_{i=1}^n g(i)^2}.$$

Since $\sum_{i=1}^n g(i)^2 \leq hg(S) \leq ph$ we obtain again that

$$\Pr\{|g(S_1) - g(S_2)| \geq \epsilon p\} \leq 2e^{-\frac{\epsilon^2 p}{2h}},$$

which gives us the desired bound. ■

It is worth noting that using tail inequalities that depend on the maximum range of the random variables rather than the sum of their squares in the proof of Lemma 4.1.5 would increase the h to an h^2 in the

exponent. Note also that if $g(i) = g'(i)$ for all $i \in S$ then they are equivalent from the point of view of the auction; we will use $|\mathcal{G}|$ to denote the number of *different* such offers in \mathcal{G} .² Lemma 4.1.5 implies that:

Corollary 4.1.6 *For a random partition of S into S_1 and S_2 , with probability at least $1 - \delta$, all offers g in \mathcal{G} such that $g(S) \geq \frac{2h}{\epsilon^2} \ln \left(\frac{2|\mathcal{G}|}{\delta} \right)$ satisfy $|g(S_1) - g(S_2)| \leq \epsilon g(S)$.*

Proof: Follows from Lemma 4.1.5 by plugging in $p = \frac{2h}{\epsilon^2} \ln \left(\frac{2|\mathcal{G}|}{\delta} \right)$ and then using the union bound over all $g \in \mathcal{G}$. ■

We complete this section with the proof of the main theorem.

Proof of Theorem 4.1.2: Let g_1 be the offer in \mathcal{G} produced by \mathcal{A} over S_1 and g_2 be the offer in \mathcal{G} produced by \mathcal{A} over S_2 . Let g_{OPT} be the optimal offer in \mathcal{G} over S ; so $g_{\text{OPT}}(S) = \text{OPT}_{\mathcal{G}}$. Since the optimal offer over S_1 is at least as good as g_{OPT} on S_1 (and likewise for S_2), the fact that \mathcal{A} is a β -approximation implies that $g_1(S_1) \geq \frac{g_{\text{OPT}}(S_1)}{\beta}$ and $g_2(S_2) \geq \frac{g_{\text{OPT}}(S_2)}{\beta}$.

Let $p = \frac{18h}{\epsilon^2} \ln \left(\frac{2|\mathcal{G}|}{\delta} \right)$. Using Lemma 4.1.5 (applying the union bound over all $g \in \mathcal{G}$), we have that with probability $1 - \delta$, every $g \in \mathcal{G}$ satisfies $|g(S_1) - g(S_2)| \leq \frac{\epsilon}{3} \max[g(S), p]$. In particular, $g_1(S_2) \geq g_1(S_1) - \frac{\epsilon}{3} \max[g_1(S), p]$, and $g_2(S_1) \geq g_2(S_2) - \frac{\epsilon}{3} \max[g_2(S), p]$.

Since the theorem assumes that $\text{OPT}_{\mathcal{G}} \geq \beta p$, summing the above two inequalities and performing a case-analysis³ we get that the profit of $\text{RSO}_{(\mathcal{G}, \mathcal{A})}$, namely the sum $g_1(S_2) + g_2(S_1)$, is at least $(1 - \epsilon) \frac{\text{OPT}_{\mathcal{G}}}{\beta}$. More specifically, assume first that $g_1(S) \geq p$ and $g_2(S) \geq p$. This implies that $g_1(S_2) \geq g_1(S_1) - \frac{\epsilon}{3} g_1(S)$ and $g_2(S_1) \geq g_2(S_2) - \frac{\epsilon}{3} g_2(S)$, and therefore $(1 + \frac{\epsilon}{3}) g_1(S_2) \geq (1 - \frac{\epsilon}{3}) g_1(S_1)$ and $(1 + \frac{\epsilon}{3}) g_2(S_1) \geq (1 - \frac{\epsilon}{3}) g_2(S_2)$. So, the profit of $\text{RSO}_{(\mathcal{G}, \mathcal{A})}$ in this case is at least

$$\frac{1 - \frac{\epsilon}{3}}{1 + \frac{\epsilon}{3}} (g_1(S_1) + g_2(S_2)) \geq \frac{1 - \frac{\epsilon}{3}}{1 + \frac{\epsilon}{3}} \frac{\text{OPT}_{\mathcal{G}}}{\beta} \geq (1 - \epsilon) \frac{\text{OPT}_{\mathcal{G}}}{\beta}.$$

If both $g_1(S) < p$ and $g_2(S) < p$, then $g_1(S_2) \geq g_1(S_1) - \frac{\epsilon}{3} p$ and $g_2(S_1) \geq g_2(S_2) - \frac{\epsilon}{3} p$, and so the profit of $\text{RSO}_{(\mathcal{G}, \mathcal{A})}$ in this case is at least $\frac{\text{OPT}_{\mathcal{G}}}{\beta} - \frac{2\epsilon}{3} p$ which is at least $(1 - \epsilon) \frac{\text{OPT}_{\mathcal{G}}}{\beta}$ by our assumption that $\text{OPT}_{\mathcal{G}} \geq \beta p$. Finally, assume without loss of generality that $g_1(S) \geq p$ and $g_2(S) < p$. This implies that $g_1(S_2) \geq g_1(S_1) - \frac{\epsilon}{3} g_1(S)$ and $g_2(S_1) \geq g_2(S_2) - \frac{\epsilon}{3} p$. The former inequality implies that $(1 + \frac{\epsilon}{3}) g_1(S_2) \geq (1 - \frac{\epsilon}{3}) g_1(S_1)$, and so $g_1(S_2) \geq (1 - \frac{2\epsilon}{3}) g_1(S_1)$, and the latter inequality implies that $g_2(S_1) \geq g_2(S_2) - \frac{\epsilon}{3} \frac{\text{OPT}_{\mathcal{G}}}{\beta}$. Together we have that

$$g_1(S_2) + g_2(S_1) \geq \left(1 - \frac{2\epsilon}{3}\right) \frac{g_{\text{OPT}}(S_1)}{\beta} + \frac{g_{\text{OPT}}(S_2)}{\beta} - \frac{\epsilon}{3} \frac{\text{OPT}_{\mathcal{G}}}{\beta} \geq (1 - \epsilon) \frac{\text{OPT}_{\mathcal{G}}}{\beta},$$

as desired. ■

4.1.5 Structural Risk Minimization

In many natural cases, \mathcal{G} consists of offers at different “levels of complexity” k . In the case of attribute auctions, for instance, \mathcal{G} could be an offer class induced by pricing functions that partition bidders into k markets and offer a constant price in each market, for different values of k . One natural approach to such a setting is to perform *structural risk minimization* (SRM): that is, to assign a penalty term to offers based

²Notice that in our generic reduction, $|\mathcal{G}|$ only appears in the analysis and we do not actually have to know whether two offers are equivalent with respect to S when running the auction.

³Note that if $\beta = 1$, then the conclusion follows easily. The case analysis is only need to deal with the case $\beta > 1$.

on their complexity and then to run a version of $\text{RSO}_{(\mathcal{G}, \mathcal{A})}$ in which \mathcal{A} optimizes profit minus penalty. Specifically, let $\bar{\mathcal{G}}$ be a series of offers classes $\mathcal{G}_1, \mathcal{G}_2, \dots$, and let pen be a penalty function defined over these classes. We then define the procedure $\text{RSO-SRM}_{(\bar{\mathcal{G}}, \text{pen})}$ as follows:

1. Randomly partition the bidders into two sets, S_1 and S_2 , by flipping fair coin for each bidder.
2. Compute g_1 to maximize $\max_k \max_{g \in \mathcal{G}_k} [g(S_1) - \text{pen}(\mathcal{G}_k)]$ and similarly compute g_2 from S_2 .
3. Use the offer g_1 for bidders in S_2 and the offer g_2 for bidders in S_1 .

We can now derive a guarantee for the $\text{RSO-SRM}_{(\bar{\mathcal{G}}, \text{pen})}$ mechanism as follows:

Theorem 4.1.7 *Assuming that we have an algorithm for solving the optimization problem required by $\text{RSO-SRM}_{(\bar{\mathcal{G}}, \text{pen})}$, then for any given value of n, ϵ , and δ , with probability at least $1 - \delta$, the revenue of $\text{RSO-SRM}_{(\bar{\mathcal{G}}, \text{pen})}$ for $\text{pen}(\mathcal{G}_k) = \frac{8h_k}{\epsilon^2} \ln\left(\frac{8k^2|\mathcal{G}_k|}{\delta}\right)$ is*

$$\max_k ((1 - \epsilon) \text{OPT}_k - 2\text{pen}(\mathcal{G}_k)),$$

where h_k is the maximum payoff from \mathcal{G}_k and $\text{OPT}_k = \text{OPT}_{\mathcal{G}_k}$.

Proof: Using Corollary 4.1.6 and a union bound over the values $\delta_k = \delta/(4k^2)$, we obtain that with probability at least $1 - \delta$, simultaneously for all k and for all offers g in \mathcal{G}_k such that $g(S) \geq \frac{8h_k}{\epsilon^2} \ln(8k^2|\mathcal{G}_k|/\delta) = \text{pen}(\mathcal{G}_k)$, we have $|g(S_1) - g(S_2)| \leq \frac{\epsilon}{2}g(S)$. Let k^* be the optimal index, namely let k^* be the index such that $(1 - \epsilon) \text{OPT}_{k^*} - 2\text{pen}(\mathcal{G}_{k^*}) = \max_k ((1 - \epsilon) \text{OPT}_k - 2\text{pen}(\mathcal{G}_k))$, and let k_i be the index of the best offer (according to our criterion) over S_i , for $i = 1, 2$. By our assumption that g_1 and g_2 were chosen by an optimal algorithm, we have $g_i(S_i) - \text{pen}(\mathcal{G}_{k_i}) \geq g_{\text{OPT}_{k^*}}(S_i) - \text{pen}(\mathcal{G}_{k^*})$, for $i = 1, 2$.

We will argue next that $g_1(S_2) \geq \frac{1 - \frac{\epsilon}{2}}{1 + \frac{\epsilon}{2}} (g_{\text{OPT}_{k^*}}(S_1) - \text{pen}(\mathcal{G}_{k^*}))$. First, if $g_1(S_1) < \text{pen}(\mathcal{G}_{k_1})$, then the conclusion is clear since we have $0 > g_1(S_1) - \text{pen}(\mathcal{G}_{k_1}) \geq g_{\text{OPT}_{k^*}}(S_1) - \text{pen}(\mathcal{G}_{k^*})$. If $g_1(S_1) \geq \text{pen}(\mathcal{G}_{k_1})$, then as argued above we have $|g_1(S_1) - g_1(S_2)| \leq \frac{\epsilon}{2}g_1(S)$ and so

$$g_1(S_2) \geq \frac{1 - \frac{\epsilon}{2}}{1 + \frac{\epsilon}{2}} g_1(S_1) \geq \frac{1 - \frac{\epsilon}{2}}{1 + \frac{\epsilon}{2}} (g_{\text{OPT}_{k^*}}(S_1) - \text{pen}(\mathcal{G}_{k^*})).$$

Similarly, we can prove that we have $g_2(S_1) \geq \frac{1 - \frac{\epsilon}{2}}{1 + \frac{\epsilon}{2}} (g_{\text{OPT}_{k^*}}(S_2) - \text{pen}(\mathcal{G}_{k^*}))$. All these together imply that the profit of the mechanism $\text{RSO-SRM}_{(\bar{\mathcal{G}}, \text{pen})}$, namely $g_1(S_2) + g_2(S_1)$, is at least

$$\frac{1 - \frac{\epsilon}{2}}{1 + \frac{\epsilon}{2}} (g_{\text{OPT}_{k^*}}(S) - 2\text{pen}(\mathcal{G}_{k^*})) \geq ((1 - \epsilon) \text{OPT}_{k^*} - 2\text{pen}(\mathcal{G}_{k^*})),$$

as desired. ■

4.1.6 Improving the Bounds

The results above say, in essence, that if we have enough bidders so that the optimal profit is large compared to $\frac{h}{\epsilon^2} \log(|\mathcal{G}|)$, then our mechanism will perform nearly as well as the best offer in \mathcal{G} . In these bounds, one should think of $\log(|\mathcal{G}|)$ as a measure of the complexity of the offer class \mathcal{G} ; for instance, it can be thought of as the number of bits needed to describe a typical offer in that class. However, in many cases one can achieve a better bound by adapting techniques developed for analyzing generalization performance in machine learning theory. In our work, we discuss a number of such methods that can produce better bounds. These include both *analysis* techniques (such as using appropriate forms of *covering numbers*), where we do not change the mechanism but instead provide a stronger guarantee, and *design* techniques (like *discretizing*), where we modify the mechanism to produce a better bound. For details see [20].

4.2 Algorithms for Pricing Problems

Algorithmic pricing problems in the context of *revenue maximization*, especially for certain special cases of combinatorial auctions, have generated a great deal of interest recently; for a comprehensive survey see the chapter “Profit Maximization in Mechanism Design” of Hartline and Karlin in [107]. Much of the previous algorithmic and mechanism design work concerning combinatorial auctions in the context of revenue maximization has focused on *item pricing* [8, 16, 47, 62, 76, 77].

For consumers with *single-minded preferences*, [76] gives a simple $O(\log mn)$ approximation algorithm. Demaine et al. [62] show the problem to be hard to approximate to better than a $\log^\delta(m)$ factor for some $\delta > 0$.

In the case when the cardinality of the desired bundles are bounded by k , Briest and Krysta [47] give an $O(k^2)$ approximation algorithm. Independently of their work, we have provided in [17] a much *simpler* algorithm with an improved $O(k)$ guarantee.

Chapter 5

Work in Progress, Open Problems, and Future Work

The current work leads to open questions and directions of significant theoretical and practical interest. I will present the most important ones below, as well as describe recent ongoing work in the context of Clustering.

5.1 Similarity Based Learning and Clustering

Learning via Similarity Functions. Our algorithms in [22] and [16] suggest a natural way to use kernels or other similarity functions in learning problems for which one also wishes to use the native features of the examples. For instance, consider the problem of classifying a stream of documents arriving one at a time. Rather than running a kernelized learning algorithm, one can simply take the native features (say the words in the document) and augment them with a small number of additional features representing the similarity of the current example with each of a pre-selected set of initial documents. One can then feed the augmented example into a standard (possibly attribute efficient ¹) unkernelized online learning algorithm. It would be interesting to explore this idea further.

It would also be interesting to provide a *formal separation* between learning with Similarity functions and learning with kernel functions. Note that in fact our Theorem 3.3.2 in Section 3.3.2 implies that if K is an (ϵ, γ) -good similarity function, then for any δ the distribution dependent similarity functions \tilde{K}_δ is in fact an $(\epsilon + \delta, \gamma/4)$ -good kernel function; here $\tilde{K}_\delta(x, x') = \sum_{i=1}^d K(x, y_i) \cdot K(x, z_i)$, where y_1, y_2, \dots, y_d are the positive landmarks, z_1, z_2, \dots, z_d are the negative landmarks and $d = (4/\gamma)^2 \ln(2/\delta)$. So, to make this a well posed question we need to consider a *family* of learning problems. A possible formulation is the following. Does there exist an (interesting) family of learning problems such that one single similarity function K is good for all of them, but no single kernel is good for all of them?

Clustering via Similarity Functions. In recent ongoing work [24] we use our approach for analyzing similarity functions in the context of *clustering*. Specifically, we provide a generic framework for analyzing what properties of a similarity function are sufficient to allow it to be useful for clustering, under different levels of relaxation of the clustering objective. We describe here briefly the main ideas and type of results in our framework.

¹A classic example of an *attribute efficient* learning algorithm is Winnow algorithm [97].

The problem of clustering data from pairwise similarity information is typically view the similarity information as ground-truth and then design algorithms to (approximately) optimize various graph-based objective functions. However, in most applications, this similarity information is merely based on some heuristic: the *true goal* is to cluster the points correctly rather than to optimize any specific graph property. In our work [24], we initiate a theoretical study of the design of similarity functions for clustering from this perspective. In particular, motivated by our work in the context of classification that asks “what natural properties of a similarity function are sufficient to be able to learn well?” we ask “what natural properties of a similarity function are sufficient to be able to *cluster* well?”

We provide a generic framework for analyzing what properties of a similarity function are sufficient to allow it to be useful for clustering, under different levels of relaxation of the clustering objective. We propose a measure of the *clustering complexity* (analogous to notions of *capacity* in learning theory) of a given property that characterizes its information-theoretic usefulness for clustering, and analyze this complexity for a broad class of properties, as well as develop efficient algorithms that are able to take advantage of them. We consider two natural clustering objectives: (a) list clustering: analogous to the notion of list-decoding, the algorithm can produce a small list of clusterings (which a user can select from) and (b) hierarchical clustering: the desired clustering is some pruning of this tree (which a user could navigate). Our algorithms for the most general properties hierarchical clusterings combine top-down and bottom-up methods in a spirit similar to methods of [52]. We also show how our algorithms can be extended to the inductive case, i.e., by using just a constant-sized sample, as in property testing.

This work can be viewed both in terms of providing formal advice to the *designer* of a similarity function for a given clustering task (such as clustering web-pages by topic) and in terms of advice about what *algorithms* to use given certain beliefs about the relation of the similarity function to the clustering task. Abstractly speaking, our notion of a *property* parallels that of a data-dependent concept class [122] (such as large-margin separators) in the context of classification.

Our work also provides the first formal framework for analyzing clustering with limited (non-interactive) feedback. A concrete implication of our work is a better understanding of when (in terms of the relation between the similarity measure and the ground-truth clustering) different hierarchical linkage-based algorithms will fare better than others.

5.2 Semi-Supervised Learning and Active Learning

Semi-Supervised Learning Most of the results in [15, 51] deal purely with sample complexity rather than computational efficiency, or are only computationally efficient under strong assumptions on the underlying distribution. It would be interesting to further develop *computationally efficient* algorithms in this semi-supervised PAC model. A specific question here would be getting additional positive results for Co-training with linear separators. As mentioned in section 2 it is known that the consistency problem is NP-hard; nonetheless we present in [15] an efficient algorithm for the special case when the underlying distribution satisfies the independence given the label property [15]. Even if one cannot solve the problem efficiently in general, a natural question is whether one can at least weaken the independence-given-the-label assumption and still get an efficient algorithm for this class.

Active Learning While our A^2 procedure is computationally efficient in the realizable case, it remains an open problem to make it efficient in the general case. It would also be interesting to analyze on what other (hypothesis spaces, distribution) pairs A^2 algorithm (or a modification of it) provides exponential speedups.

Another very important open problem here is what are the “right” quantities (complexity measures) that characterize sample complexity of *Active Learning* both in the agnostic and the realizable case. It would finally be interesting to consider and analyze an Active Semi-Supervised Learning model.

5.3 Mechanism Design, Machine Learning, and Pricing Problems

Generic Mechanisms for Limited Supply Setting An important extension of our work in [20] would be relaxing the assumption that the items for sale are available in unlimited supply. It is worth noting that what is particularly challenging in the limited supply setting is defining *the right benchmark*.

Profit Maximization Mechanisms for General Combinatorial Auctions It is also interesting to further study the possibility of obtaining good approximation algorithms and mechanisms for Profit Maximization Mechanisms for general Combinatorial Auctions.

Bibliography

- [1] <http://www.kernel-machines.org/>. 1.2
- [2] *6th Kernel Machines Workshop*. NIPS, 2002. <http://www-stat.ucdavis.edu/nello/nips02.html>. 1.2
- [3] *The Seventh Workshop on Kernel Machines*. COLT, 2003. <http://learningtheory.org/colt2003>. 1.2
- [4] *Workshop Graphical Models and Kernels*. NIPS, 2004. <http://users.rsise.anu.edu.au/smola/workshops/nips04>. 1.2
- [5] *Kernel Methods and Structured Domains*. NIPS, 2005. <http://nips2005.kyb.tuebingen.mpg.de>. 1.2
- [6] D. Achlioptas. Database-friendly random projections. *Journal of Computer and System Sciences*, 66(4): 671–687, 2003. 3.1
- [7] G. Aggarwal and J. Hartline. Knapsack Auctions. In *Proceedings of the 17th ACM-SIAM Symposium on Discrete Algorithms*, 2006. 4.1.1
- [8] G. Aggarwal, T. Feder, R. Motwani, and A. Zhu. Algorithms for multi-product pricing. In *Proceedings of the International Colloquium on Automata, Languages, and Programming*, pages 72–83, 2004. 4.2
- [9] D. Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1998. 2.2
- [10] D. Angluin. Queries revisited. *Theoretical Computer Science*, 313(2):175–194, 2004. 2.2
- [11] M. Anthony and P. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999. 4.1.1
- [12] M. Anthony and P. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999. 1.2, 3
- [13] R. I. Arriaga and S. Vempala. An algorithmic theory of learning, robust concepts and random projection. pages 616–623, 1999. 3.1, 3.1, 3.1.1, 3.1, 3.2
- [14] B. Awerbuch, Y. Azar, and A. Meyerson. Reducing truth-telling online mechanisms to online optimization. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, 2003. 4.1.1
- [15] M.-F. Balcan and A. Blum. A PAC-style model for learning from labeled and unlabeled data. In *Proceedings of the Annual Conference on Computational Learning Theory*, 2005. 1.1, 2, 2.1, 2.1.2, 2.1.2, 2.1.2, 5.2
- [16] M.-F. Balcan and A. Blum. On a theory of learning with similarity functions. In *International Conference on Machine Learning*, 2006. 1.2, 3, 3.3.3, 3.3.4, 4.2, 5.1
- [17] M.-F. Balcan and A. Blum. Approximation Algorithms and Online Mechanisms for Item Pricing. In *Proceedings of the 7th ACM Conference on Electronic Commerce*, 2006. 4.2
- [18] M.-F. Balcan and A. Blum. An augmented PAC-model for semi-supervised learning. Book chapter in "Semi-Supervised Learning", O. Chapelle, B. Schölkopf, and A. Zien, eds., MIT press, 2006. 1.1, 2, 2.1, 2.1.2, 2.1.2, 2.1.2
- [19] M. F. Balcan, A. Blum, and K. Yang. Co-training and expansion: Towards bridging theory and practice. In *Advances in Neural Information Processing Systems*, 2004. 1.1, 2.1.2, 2.1.2
- [20] M.-F. Balcan, A. Blum, J. Hartline, and Y. Mansour. Mechanism Design via Machine Learning. In *46th Annual IEEE Symposium on Foundations of Computer Science*, pages 605 – 614, 2005. Invited to appear in

- a special issue on Learning Theory of the Journal of Computer and System Sciences. 1.3, 4.1.1, 4.1.2, 4.1.2, 4.1.3, 4.1.3, 4.1.4, 4.1.4, 4.1.6, 5.3
- [21] M.-F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *International Conference on Machine Learning*, 2006. Invited to appear in a special issue on Learning Theory of the Journal of Computer and System Sciences. 1.1, 2, 1, 2.2, 2.2
- [22] M.-F. Balcan, A. Blum, and S. Vempala. On kernels, margins and low-dimensional mappings. *Machine Learning Journal*, 2006. 1.2, 3, 3.2, 3.2, 3.2.1, 3.2.2, 5.1
- [23] M.-F. Balcan, A. Blum, and N. Srebro. Learning with similarity functions. 2007. Manuscript. 3.3.3
- [24] M.-F. Balcan, A. Blum, and S. Vempala. A theory of similarity functions for clustering. 2007. 5.1
- [25] M.-F. Balcan, A. Broder, and T. Zhang. Margin based active learning. 2007. COLT. 1.1, 2
- [26] P. Bartlett and S. Mendelson. Rademacher and Gaussian Complexities Risk Bounds and Structural Results. *Journal of Machine Learning Research*, 54(3):463–482, 2002. 2.1
- [27] P. Bartlett and J. Shawe-Taylor. Generalization performance of support vector machines and other pattern classifiers. In *Advances in Kernel Methods: Support Vector Learning*. MIT Press, 1999. 2, 3.2.1
- [28] E. Baum and K. Lang. Query learning can work poorly when a human oracle is used. In *International Joint Conference on Neural Networks*, 1993. 2.2
- [29] E. B. Baum. Polynomial time algorithms for learning neural nets. In *Proceedings of the third annual workshop on Computational learning theory*, pages 258 – 272, 1990. 2.1
- [30] A. Ben-Israel and T.N.E. Greville. *Generalized Inverses: Theory and Applications*. Wiley, New York, 1974. 3.2.1
- [31] G.M. Benedek and A. Itai. Learnability with respect to a fixed distribution. *Theoretical Computer Science*, 86:377–389, 1991. 2.1
- [32] T. De Bie and N. Cristianini. Convex transduction with the normalized cut. Manuscript, 2004. 2.1.3
- [33] H.D. Block. The perceptron: A model for brain functioning. *Reviews of Modern Physics*, 34:123–135, 1962. Reprinted in *Neurocomputing*, Anderson and Rosenfeld. 1
- [34] A. Blum. Machine learning theory. *Essay*, 2007. 1.1
- [35] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proc. ICML*, pages 19–26, 2001. 1.1, 2.1, 2.1.1, 2.1.3
- [36] A. Blum and J. Hartline. Near-Optimal Online Auctions. In *Proceedings of the 16th ACM-SIAM Symposium on Discrete Algorithms*, pages 1156 – 1163, 2005. 4.1.1, 4.1.4
- [37] A. Blum and R. Kannan. Learning an intersection of k halfspaces over a uniform distribution. *Journal of Computer and Systems Sciences*, 54(2):371–380, 1997. 2.1
- [38] A. Blum and T. M. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT*, 1998. 1.1, 2.1, 2.1.1, 2.1.2, 2.1.2
- [39] A. Blum, A. Frieze, R. Kannan, and S. Vempala. A polynomial-time algorithm for learning noisy linear threshold functions. *Algorithmica*, 22:35–52, 1998. 2.1.2
- [40] A. Blum, V. Kumar, A. Rudra, and F. Wu. Online Learning in Online Auctions. In *Proceedings of the 14th ACM-SIAM Symposium on Discrete Algorithms*, pages 137 – 146, 2003. 4.1.1
- [41] A. Blum, J. Lafferty, R. Reddy, and M. R. Rwebangira. Semi-supervised learning using randomized mincuts. In *ICML '04*, 2004. 2.1.3
- [42] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Learnability and the Vapnik-Chervonenkis Dimension. *Journal of the ACM*, 36:929–965, 1989. 4.1.1
- [43] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Learnability and the Vapnik Chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989. 2.1
- [44] C. Borgs, J. T. Chayes, N. Immorlica, M. Mahdian, and A. Saberi. Multi-unit auctions with budget-

- constrained bidders. In *Proceedings of the 6th ACM Conference on Electronic Commerce*, pages 44–51, 2005. 4.1.2
- [45] S. Boucheron, G. Lugosi, and P. Massart. A sharp concentration inequality with applications. *Random Structures and Algorithms*, 16:277–292, 2000. 2.1.2
- [46] O. Bousquet, S. Boucheron, and G. Lugosi. Theory of Classification: A Survey of Recent Advances. *ESAIM: Probability and Statistics*, 2005. 2.1, 2.1.2
- [47] P. Briest and P. Krysta. Single-Minded Unlimited Supply Pricing on Sparse Instances. In *Proceedings of the 17th ACM-SIAM Symposium on Discrete Algorithms*, 2006. 4.2
- [48] N. H Bshouty and N. Eiron. Learning monotone dnf from a teacher that almost does not answer membership queries. *Journal of Machine Learning Research*, 3:49–57, 2002. 2.2
- [49] V. Castelli and T.M. Cover. On the exponential value of labeled samples. *Pattern Recognition Letters*, 16: 105–111, 1995. 2.1, 2
- [50] V. Castelli and T.M. Cover. The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing parameter. *IEEE Transactions on Information Theory*, 42(6):2102–2117, 1996. 2.1, 2
- [51] O. Chapelle, B. Schlkopf, and A. Zien. *Semi-Supervised Learning*. MIT press, 2006. 1.1, 2, 2.1, 3, 2.1.2, 6, 2.1.2, 5.2
- [52] D. Cheng, R. Kannan, S. Vempala, and G. Wang. A divide-and-merge methodology for clustering. In *Proc. of the ACM Symposium on Principles of Database Systems*, 2005. 5.1
- [53] D. Cohen, L. Atlas, and R. Ladner. Improving generalization with active learning. 15(2):201–221, 1994. 1.1, 2, 1, 2.2
- [54] M. Collins and Y. Singer. Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 189–196, 1999. 2.1.1
- [55] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273 – 297, 1995. 1.2
- [56] N. Cristianini, J. Shawe-Taylor, Andre Elisseeff, and J. Kandola. On kernel target alignment. In *Advances in Neural Information Processing Systems*, 2001. 1.2
- [57] S. Dasgupta. Experiments with random projection. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 143–151, 2000. 3.1
- [58] S. Dasgupta. Coarse sample complexity bounds for active learning. In *Advances in Neural Information Processing Systems*, 2005. 1.1, 2
- [59] S. Dasgupta and A. Gupta. An elementary proof of the Johnson-Lindenstrauss Lemma. *Random Structures & Algorithms*, 22(1):60–65, 2002. 3, 3.1
- [60] S. Dasgupta, M. L. Littman, and D. McAllester. Pac generalization bounds for co-training. In *Advances in Neural Information Processing Systems 14*, 2001. 2.1.2
- [61] S. Dasgupta, A. Kalai, and C. Monteleoni. Analysis of perceptron-based active learning. In *COLT*, 2005. 2.2
- [62] E. Demaine, U. Feige, M.T. Hajiaghayi, and M. Salavatipour. Combination Can Be Hard: Approximability of the Unique Coverage Problem . In *Proceedings of the 17th ACM-SIAM Symposium on Discrete Algorithms*, 2006. 4.2
- [63] L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, 1996. 2.1, 2.1.2, 2.1.2
- [64] A. Fiat, A. Goldberg, J. Hartline, and A. Karlin. Competitive Generalized Auctions. In *Proceedings 34th ACM Symposium on the Theory of Computing*, pages 72 – 81, 2002. 4.1.1
- [65] A. Flaxman. Personal communication, 2003. 2.1.2
- [66] D. Fradkin and D. Madigan. Experiments with random projections for machine learning. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*,

- pages 517–522, 2003. 3.1
- [67] Y. Freund and R. E. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277 – 296, 1999. 3.3.2
- [68] Y. Freund and R.E. Schapire. Large margin classification using the Perceptron algorithm. *Machine Learning*, 37(3):277–296, 1999. 1
- [69] Y. Freund, H.S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2-3):133–168, 1997. 2.2
- [70] R. Ghani. Combining labeled and unlabeled data for text classification with a large number of categories. In *Proceedings of the IEEE International Conference on Data Mining*, 2001. 2.1.1
- [71] N. Goal, G. Bebis, and A. Nefian. Face recognition experiments with random projection. In *Proceedings SPIE Vol. 5779*, pages 426–437, 2005. 3.1
- [72] A. Goldberg and J. Hartline. Competitive Auctions for Multiple Digital Goods. In *Proceedings of the 9th Annual European Symposium on Algorithms*, pages 416 – 427, 2001. 4.1.1
- [73] A. Goldberg and J. Hartline. Competitiveness via Consensus. In *Proceedings of the 14th ACM-SIAM Symposium on Discrete Algorithms*, pages 215 – 222, 2003. 4.1.1, 4.1.3
- [74] A. Goldberg, J. Hartline, and A. Wright. Competitive Auctions and Digital Goods. In *Proceeding of the 12th ACM-SIAM Symposium on Discrete Algorithms*, pages 735–744, 2001. 1.3, 4.1.1, 4.1.3, 4.1.3
- [75] A. Goldberg, J. Hartline, A. Karlin, M. Saks, and A. Wright. Competitive Auctions and Digital Goods. *Games and Economic Behavior*, 2006. 1.3, 4.1.1, 4.1.3
- [76] V. Guruswami, J. Hartline, A. Karlin, D. Kempe, C. Kenyon, and F. McSherry. On Profit-Maximizing Envy-Free Pricing. In *Proceedings of the 16th ACM-SIAM Symposium on Discrete Algorithms*, pages 1164 – 1173, 2005. 4.1.1, 4.1.2, 4.2
- [77] J. Hartline and V. Koltun. Near-Optimal Pricing in Near-Linear Time . In *9th Workshop on Algorithms and Data Structures*, pages 422 – 431, 2005. 4.2
- [78] B. Heisele, P. Ho, and T. Poggio. Face recognition with support vector machines: Global versus component-based approach. In *International Conference on Computer Vision*, 2001. 1.2
- [79] R. Herbrich. *Learning Kernel Classifiers*. MIT Press, Cambridge, 2002. 1.2, 3
- [80] R. Hwa, M. Osborne, A. Sarkar, and M. Steedman. Corrected co-training for statistical parsers. In *ICML-03 Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, Washington D.C., 2003. 1.1, 2.1
- [81] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 604–613, 1998. 3.1
- [82] J. Jackson. An efficient membership-query algorithm for learning dnf with respect to the uniform distribution. *Journal of Computer and System Sciences*, 57(3):414–440, 1995. 2.2
- [83] T. Joachims. *Learning to Classify Text Using Support Vector Machines: Methods, Theory, and Algorithms*. Kluwer, 2002. 1.2, 3
- [84] T. Joachims. Transductive learning via spectral graph partitioning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2003. 2.1.3
- [85] T. Joachims. Transductive inference for text classification using support vector machines. In *Proc. ICML*, pages 200–209, 1999. 1.1, 2.1, 2.1.1
- [86] W. B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Conference in Modern Analysis and Probability*, pages 189–206, 1984. 3.1
- [87] M. Kearns. Efficient noise-tolerant learning from statistical queries. In *Journal of the ACM (JACM)*, pages 983 – 1006, 1998. 7
- [88] M. Kearns and U. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, 1994. 2.1, 2.1.2,

- [89] J. Kleinberg. Detecting a network failure. In *Proceedings of the 41st IEEE Symposium on Foundations of Computer Science*, pages 231–239, 2000. 2.1.3
- [90] J. Kleinberg, M. Sandler, and A. Slivkins. Network failure detection and graph connectivity. In *Proceedings of the 41st IEEE Symposium on Foundations of Computer Science*, pages 231–239, 2004. 2.1.3
- [91] A. R. Klivans, R. O’Donnell, and R. Servedio. Learning intersections and thresholds of halfspaces. In *Proceedings of the 43rd Symposium on Foundations of Computer Science*, pages 177–186, 2002. 2.1
- [92] V. Koltchinskii. Rademacher Penalties and Structural Risk Minimization. *IEEE Transactions of Information Theory*, 54(3):1902–1914, 2001. 2.1
- [93] G. R. G. Lanckriet, N. Cristianini, P. L. Bartlett, L. El Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, (5):27–72, 2004. 1.2, 3.3.2
- [94] B. Leske. The value of agreement: A new boosting algorithm. In *Proceedings of the Annual Conference on Computational Learning Theory*, 2005. 2.1.1
- [95] A. Levin, P. Viola, and Y. Freund. Unsupervised improvement of visual detectors using co-training. In *Proc. 9th Int. Conf. Computer Vision*, pages 626–633, 2003. 1.1, 2.1, 2.1.1
- [96] N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, fourier transform, and learnability. In *Proceedings of the Thirtieth Annual Symposium on Foundations of Computer Science*, pages 574–579, Research Triangle Park, North Carolina, October 1989. 2.1
- [97] N. Littlestone. Learning when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1987. 1
- [98] Nick Littlestone. From on-line to batch learning. In *COLT ’89: Proceedings of the 2nd Annual Workshop on Computational Learning Theory*, pages 269–284, 1989. 1
- [99] S. Mendelson and P. Philips. Random subclass bounds. In *Proceedings of the 16th Annual Conference on Computational Learning Theory (COLT)*, 2003. 2.1.3
- [100] R. Meyerson. Optimal Auction Design. *Mathematics of Operations Research*, 6:58–73, 1983. 4.1.1
- [101] M. Minsky and S. Papert. *Perceptrons: An Introduction to Computational Geometry*. The MIT Press, 1969. 1
- [102] T. Mitchell. The discipline of machine learning. *CMU-ML-06 108*, 2006. 1.1
- [103] K. R. Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12:181 – 201, 2001. 1.2
- [104] K. Nigam. Using unlabeled data to improve text classification. Technical Report Doctoral Disertation, CMU-CS-01-126, Carnegie Mellon University, 2001. 3
- [105] K. Nigam and R. Ghani. Analyzing the effectiveness and applicability of Co-training. In *Proc. ACM CIKM Int. Conf. on Information and Knowledge Management*, pages 86–93, 2000. 2.1.1
- [106] K. Nigam, A. McCallum, S. Thrun, and T.M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Mach. Learning*, 39(2/3):103–134, 2000. 1.1, 2.1, 3
- [107] N. Nisan, T. Rougharden, E. Tardos, and V. Vazirani (Eds.). *Algorithmic Game Theory*. 2006. To appear. 4.2
- [108] A. B. J. Novikoff. On convergence proofs on perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata*, 1962. 1
- [109] S. Park and B. Zhang. Large scale unstructured document classification using unlabeled data and syntactic information. In *PAKDD 2003*, LNCS vol. 2637, pages 88–99. Springer, 2003. 2.1.1
- [110] S.-B. Park and B.-T. Zhang. Co-trained support vector machines for large scale unstructured document classification using unlabeled data and syntactic information. *Information Processing and Management*, 40(3):421 – 439, 2004. 1.1, 2.1
- [111] D. Pierce and C. Cardie. Limitations of Co-Training for natural language learning from large datasets. In

- Proc. Conference on Empirical Methods in NLP*, pages 1–9, 2001. 2.1.1
- [112] D. Rosenberg and P. L. Bartlett. The Rademacher Complexity of Co-Regularized Kernel Classes. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2007. 2.1.1
- [113] B. Scholkopf, K. Tsuda, and J.-P. Vert. *Kernel Methods in Computational Biology*. MIT Press, 2004. 1.2, 3
- [114] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004. 1.2, 3
- [115] J. Shawe-Taylor, P. L. Bartlett, R. C. Williamson, and M. Anthony. Structural risk minimization over data-dependent hierarchies. *IEEE Transactions on Information Theory*, 44(5):1926–1940, 1998. 2.1.3
- [116] J. Shawe-Taylor, P. L. Bartlett, R. C. Williamson, and M. Anthony. Structural risk minimization over data-dependent hierarchies. *IEEE Transactions on Information Theory*, 44(5):1926–1940, 1998. 1.2, 3, 2, 3.2.1
- [117] John Shawe-Taylor. Rademacher Analysis and Multi-View Classification. 2006. <http://www.gla.ac.uk/external/RSS/RSScomp/shawe-taylor.pdf>. 2.1.1
- [118] A. J. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans. *Advances in Large Margin Classifiers*. MIT Press, 2000. 1.2
- [119] N. Srebro. How Good is a Kernel as a Similarity Function. *COLT*, 2007. 3.3.3
- [120] L.G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984. 2.1
- [121] V. Vapnik. *Statistical Learning Theory*. Springer-Verlag, 1998. 4.1.1
- [122] V. N. Vapnik. *Statistical Learning Theory*. John Wiley and Sons Inc., 1998. 1.2, 2.1, 2.1.2, 3, 5.1
- [123] S. Vempala. A random sampling based algorithm for learning the intersection of half-spaces. In *Proceedings of the 38th Symposium on Foundations of Computer Science*, pages 508–513, 1997. 2.1
- [124] K. A. Verbeurgt. Learning DNF under the uniform distribution in quasi-polynomial time. In *COLT*, pages 314–326, 1990. 2.1
- [125] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Meeting of the Association for Computational Linguistics*, pages 189–196, 1995. 2.1, 2.1.1
- [126] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Scholkopf. Learning with local and global consistency. In *NIPS*, 2004. 2.1.3
- [127] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proc. ICML*, pages 912–912, 2003. 1.1, 2.1, 2.1.1, 2.1.3
- [128] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning: From gaussian fields to gaussian processes. Technical report, Carnegie Mellon University, 2003. 2.1.3