Learning from Partially Labeled Data

by

Marcin Olof Szummer

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Sep 2002

Author . . . .
Department of Electrical Engineering and Computer Science
Sep 3rd, 2002

Certified by . . . .
Tommi S. Jaakkola
Associate Professor
Thesis Supervisor

Certified by . . . .
Tomaso A. Poggio
Professor
Thesis Supervisor

Read by . . . .
Leslie P. Kaelbling
Professor
Thesis Reader

Read by . . . .
Thomas Hofmann
Assistant Professor
Thesis Reader

Accepted by . . . .
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Learning from Partially Labeled Data
by
Marcin Olof Szummer

Submitted to the Department of Electrical Engineering and Computer Science
on Sep 3rd, 2002, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Classification with partially labeled data involves learning from a few labeled examples as well as a large number of unlabeled examples, and represents a blend of supervised and unsupervised learning. Unlabeled examples provide information about the input domain distribution, but only the labeled examples indicate the actual classification task. The key question is how to improve classification accuracy by linking aspects of the input distribution $P(\boldsymbol{x})$ to the conditional output distribution $P(y|\boldsymbol{x})$ of the classifier.

This thesis presents three approaches to the problem, starting with a kernel classifier that can be interpreted as a discriminative kernel density estimator and is trained via the EM algorithm or via margin-based criteria. Secondly, we employ a Markov random walk representation that exploits clusters and low-dimensional structure in the data in a robust and probabilistic manner. Thirdly, we introduce information regularization, a non-parametric technique based on minimizing information about labels over regions covering the domain. Information regularization provides a direct and principled way of linking $P(\boldsymbol{x})$ to $P(y|\boldsymbol{x})$, and remains tractable for continuous $P(\boldsymbol{x})$.

The partially labeled problem arises in many applications where it is easy to collect unlabeled examples, but labor-intensive to classify the examples. The thesis demonstrates that the approaches require very few labeled examples for high classification accuracy on text and image-classification tasks.

Thesis Supervisor: Tommi S. Jaakkola
Title: Associate Professor

Thesis Supervisor: Tomaso A. Poggio
Title: Professor

Acknowledgments

I am very grateful to Prof. Tommi Jaakkola and Prof. Tomaso Poggio, my outstanding duo of thesis advisors. Prof. Jaakkola has always shown me how to reach beyond what I thought was possible. He has been very helpful and generous to me and I have learned a lot from our collaboration. I cannot thank him enough. I am deeply indebted to Prof. Poggio for suggesting the topic of this thesis, and for guiding and supporting me throughout the entire journey. My thesis readers Prof. Leslie Kaelbling and Prof. Thomas Hofmann helped me see how to make the thesis clearer.

Tom Minka has been a great source of inspiration ever since we met at the beginning of graduate school. He has freely shared his passion and deep understanding of machine learning, and has made many valuable connections that have affected the development of the thesis. I will miss our imaginative discussions over dinner.

I have had the privilege of working with many extraordinary people in several superb research groups at MIT. At CBCL, I will always remember Shayan Mukherjee, Ryan "Rif" Rifkin, Tony Ezzat, Massimiliano Pontil, Theos Evgeniou, Bernd Heisele, Luis Perez-Breva, and Marypat Fitzgerald.

I have enjoyed participating in Prof. Jaakkola's weekly group meetings, and had stimulating conversations with Chen-Hsiang Yeang, Adrian Corduneanu and Jason Rennie.

I started my graduate school career at the vision and modeling group at the MIT Media Laboratory, which was an incredibly diverse and creative environment. I owe many thanks to Prof. Rosalind Picard who nurtured my first independent research ideas resulting in my Master's thesis on modeling temporal textures. Prof. Aaron Bobick continued to support my interest in computer vision. The vision and modeling team would not be complete without Yuan Qi, Flavia Sparacino, Tony Jebara, Yuri Ivanov, and Sumit Basu. I am grateful to Kris Popat for the many things he explained so lucidly to me.

The car experiments in this thesis were performed by Andrew Crane as part of his Master's thesis, and I also worked with undergraduate researchers Emina Torlak and Paul Chang. Matthias Oberländer from Daimler-Chrysler contributed the car dataset and motivated the classification task.

Finally, I must thank my mother Barbara and my father Maciej for their encouragement and patience throughout this endeavor, and my sister Karolina whose smile raised my spirits in the final stage of the thesis. *Kocham Was wszystkich mocno, mocno!*

Contents

Chapter 1

The Partially Labeled Learning Problem

1.1 An Introduction to the Partially Labeled Learning Problem

In this thesis we study machine learning algorithms that learn from examples. We refer to such algorithms simply as learners or classifiers. Our learners will be given both labeled and unlabeled examples, which are collectively referred to as partially labeled examples. The labels represent classes or categories of the examples. The task of the learner is to infer labels for the unlabeled examples, in other words, to classify these examples. We will generally assume that the number of unlabeled examples is large relative to the number of labeled examples.

For instance, the learning scenario may be a medical diagnosis task. The learner is given blood measurements from 1000 patients and a diagnosis "healthy" or "unhealthy" for ten of these patients. The task is to classify the other given 990 patients as healthy or unhealthy.

The crucial detail in the partially labeled learning scenario is that the learner is expected to learn from both the labeled as well as all the unlabeled examples when performing the task. This is why both labeled and unlabeled examples are provided to the learner before learning begins.

Learning from partially labeled data is also referred to as semi-supervised learning. The term semi-supervised refers to a blend of supervised learning and unsupervised learning, which are well-studied machine learning problems [Bis95, DHS00]. In the supervised learning problem, the learner is initially provided with only labeled examples, and the task is to learn a classification rule from this "training data." Later the classification rule can be employed to classify unlabeled data ("test data"), but at that point no learning is taking place: the classifier is trained on only the training data.

In the unsupervised learning problem, the learner is given only unlabeled examples. The task is to group similar examples according to some similarity criterion, to find "clusters" of like examples. The clusters may correspond to underlying classes or categories, but the learner cannot tell what clusters correspond to what categories, because it does not employ any labeled examples. Unsupervised learning is often applied to discover structure, regularities or categories in the data, but typically requires human analysis to determine whether the discovered regularities are interesting, and to determine the correspondence between clusters and meaningful categories.

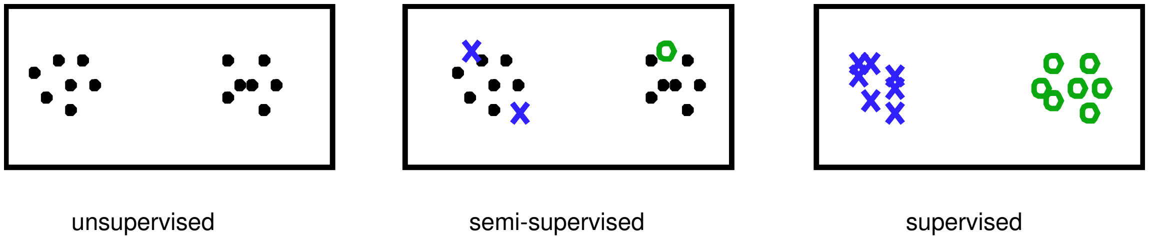


Figure 1-1: Semi-supervised learning can be viewed as a blend of the supervised and the unsupervised learning problem. There are unlabeled examples (dots) and labeled examples in two classes (crosses and circles).

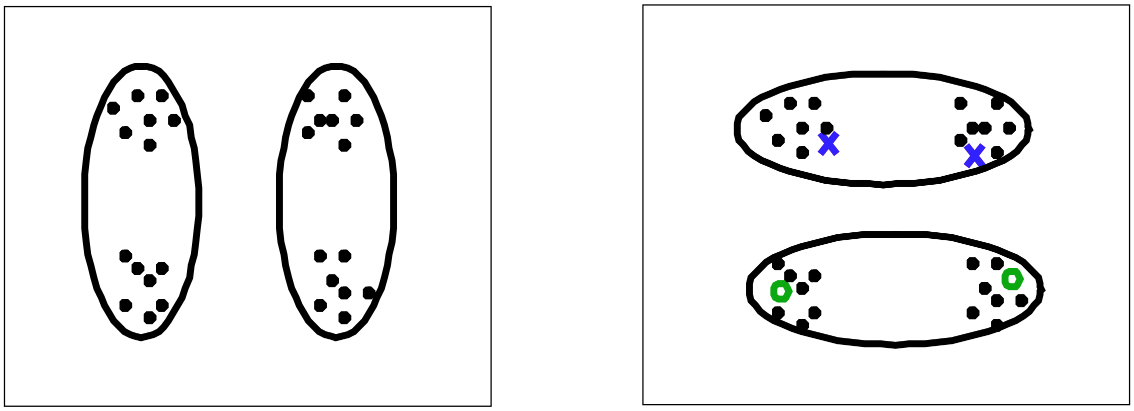


Figure 1-2: For this dataset, unsupervised learning finds two alternative clusterings (depicted as vertical and horizontal ellipses in the left and right panels.) A few labeled examples indicate that one of the clusterings is consistent with the label categories, but the other is not.

Semi-supervised learning conceptually falls between supervised and unsupervised learning. It can be seen as a supervised learning technique that supplements labeled examples by structure or regularities learned from unlabeled examples. In the terminology of supervised learning, a semi-supervised learner trains on both the training set and the *unlabeled* test set. Training on the unlabeled test set is allowed and advantageous as we shall see, but the learner must not train with labels of test examples. Labels for test examples may exist to measure the accuracy of the learner's classification, but are not available to the learner.

Alternatively, semi-supervised learning can be viewed as an unsupervised learning technique that exploits labels to help it focus on relevant structure in the data. For example, there may be two reasonable groupings of the unlabeled data. Labeled examples may be consistent with only one of the two groupings, and may also disambiguate any correspondence between clusters and classes (Figure 1-2.) To find clusterings that are consistent with the labeled examples, we can impose constraints from the labeled data during the clustering process [KKM02]. The constraints may disallow clusters that contain labeled examples from different classes, or require that all examples from the same class belong to the same cluster.

1.1.1 Partially Labeled Data Enables Accurate Learning with Few Labeled Examples

When labeled and unlabeled examples are available, we want to learn using both, because learning accuracy potentially improves with more examples. “There is no data like more data”, as the saying goes. This thesis examines how partially labeled data can enable accurate learning with few labeled examples. By accurate learning, we mean low classification error (good generalization performance) on the unlabeled examples. When compared to supervised learner on the same task, a partially labeled learner may achieve the same classification error with fewer labeled examples, or achieve a lower classification error with the same labeled examples as the supervised learner.

The interesting question is how labeled and unlabeled examples can both contribute to learning. On the lowest level, unlabeled examples yield information about the data distribution in the input domain $\mathcal{X}$. While unlabeled examples may uncover properties of the *domain* of the problem, they say nothing about the *learning task* over the domain. Labeled examples provide vital guidance about the task, but there may be too few labeled examples to reach a desired level of learning accuracy. By exploiting the structure of the domain learned from unlabeled examples, classification accuracy with few labeled examples may be improved.

For some specific intuition on how partially labeled data can be useful, consider a text classification task consisting of text documents and their labels representing topic categories. The documents are classified based on the words they contain. Unlabeled documents can be used to build an enhanced language model with more word statistics, new words and contexts of words. A new document may contain important keywords that were absent from the labeled documents but present in the unlabeled documents. To classify its topic, we can use the unlabeled documents to find other words co-occurring with these novel keywords, and then refer to the labeled documents for related documents and their labels.

1.1.2 Few Labeled Examples but Many Unlabeled Examples

In partially labeled problems of interest, there are usually few labeled examples compared to the many unlabeled examples. It is often infeasible to obtain labels for all examples in large data sets. Assigning labels can require expensive resources such as human labor or laboratory tests. In some cases ground truth labels are impossible to obtain, e.g., if the necessary measurements can no longer be made, or if the labels will be assigned only in the future. In contrast, unlabeled examples are frequently easy to obtain in large quantities, and can outnumber the amount of labeled data substantially. For instance, it is labor-intensive to collect image databases of only faces, but it is easy to collect arbitrary imagery with occasional faces, e.g., by crawling the World Wide Web, or by pointing a video camera out the window.

1.1.3 Practical Significance

The partially labeled data problem is increasingly important in practice. We generate more data with digital sensors, devices, cameras, computers, and networks. We preserve more data since storage has become cheap and plentiful. The data is of more varied and complex types, and yet we desire specific and personalized analysis. We interact more with computers, and are willing to give a few examples of our preferences, but also want computers to learn in an unobtrusive, unsupervised way. For instance, large corpora of text documents are available, but are only roughly categorized, if at all. The categories may not be suitable to our task at hand. With partially labeled learning, we could provide a few labeled examples describing our purposes and have the system use the remaining data to build a custom classifier. Other domains with partially labeled data include data mining of consumer preferences, biological data from DNA arrays, and movements of financial markets.

There are also developmental motivations for studying the process of learning from partially data. Children acquire language mainly by listening and imitating, with very limited feedback from adults. Human beings also excel at other partially labeled learning tasks, such as visual discrimination with hyperacuity [FEP95].

1.2 Formalization of the Problem

We now define the partially labeled problem more formally.

We are given a set L of labeled examples, and a set U of unlabeled examples. The sets have N_L and N_U elements respectively, for a total of N_{LU} elements. An unlabeled example can be fully described by a vector $\boldsymbol{x}$. The $\boldsymbol{x}$-value encodes the features of the example, and if there are d real-valued features, $\boldsymbol{x}$ can be represented as a point in $\mathbb{R}^d$. The labeled examples are written as $(\boldsymbol{x}, \tilde{y})$ pairs. The y-value encodes the label of the example, and the tilde indicates that the label has been observed. We focus on classification problems where y is discrete and can be modeled as an integer value (although no ordering among classes is assumed.) For binary classification problems, $y \in \{1, -1\}$.

Altogether, we can write the partially labeled data set as $\{(\boldsymbol{x}_1, \tilde{y}_1), \ldots, (\boldsymbol{x}_{N_L}, \tilde{y}_{N_L}), \boldsymbol{x}_{N_L+1}, \ldots, \boldsymbol{x}_{N_{LU}}\}$. We assume that the labeled and unlabeled examples have been drawn independently and identically distributed (IID) from the same underlying but unknown joint distribution $P(\boldsymbol{x}, y)$. This assumption will allow us to use the same probabilistic model for both labeled and unlabeled examples. However, there will typically be many more unlabeled than labeled examples, so that $N_U \gg N_L$. We must examine the mechanism that determines whether the label of an example will be observed or not.

1.2.1 Missingness Mechanism

Let $D_{\text{obs}} = L \cup U = \{(\boldsymbol{x}_1, \tilde{y}_1), \ldots, (\boldsymbol{x}_{N_L}, \tilde{y}_{N_L}), \boldsymbol{x}_{N_L+1}, \ldots, \boldsymbol{x}_{N_{LU}}\}$ be the observed data and $D_{\text{mis}} = \{y_{N_L+1}, \ldots, y_{N_{LU}}\}$ be the unobserved ("missing") labels, and let the complete data with unobserved labels filled in be $D = D_{\text{obs}} \cup D_{\text{mis}}$. Let R be a vector of indicator variables, whose values are 0 if the corresponding y labels are missing, and 1 if observed. Since we know what labels are observed and missing, R is known, and we ask whether the patterns of the missing labels contain any information that can help classification. Model this missing data mechanism by $P(R|D, \xi)$, where the conditioning is on the complete data set D and model parameters ξ. Model the data itself as $P(D|\theta)$ and note that model parameters θ describe the complete data, not just the observed subset. Overall, the observed data is R and D_{obs} and is modeled by the joint distribution $P(R, D_{\text{obs}}|\theta, \xi)$.

We will assume that the missing data mechanism is *ignorable* [LR87, Sch97], i.e., it contributes no information about the data model. Consequently, we will disregard the missingness pattern R and ξ, and model solely the observed data $P(D_{\text{obs}}|\theta)$ or posterior

$P(\theta|D_{\text{obs}})$. The two models including or excluding the missingness mechanism lead to the same probabilistic inferences when

- the missing data mechanism is *missing at random* (MAR), and
- the parameters θ and ξ are distinct (*parameter distinctness*, PD).

The missingness mechanism is MAR if $P(R|D_{\text{mis}}, D_{\text{obs}}) = P(R|D_{\text{obs}})$, in other words, when the missingness pattern is conditionally independent of the missing data given the observed data. The MAR assumption still holds when labels (values of D_{mis}) are missing more frequently for certain classes, but such a class imbalance can only be indirect and must be fully explained by D_{obs}. We now show that the model with ignored missingness mechanism leads to the same likelihood-based inferences given the MAR and parameter distinctness assumption. Write the joint likelihood of the observed data

$$\begin{aligned} P(R, D_{\text{obs}}|\theta,\xi) = \int P(R, D|\theta,\xi)\,\mathrm{d}D_{\text{mis}} \overset{\text{PD}}{=} \int P(R|D,\xi)P(D|\theta)\,\mathrm{d}D_{\text{mis}} = \\ \overset{\text{MAR}}{=} P(R|D_{\text{obs}},\xi)\int P(D|\theta)\,\mathrm{d}D_{\text{mis}} = P(R|D_{\text{obs}},\xi)P(D_{\text{obs}}|\theta). \end{aligned} \tag{1.1}$$

Maximizing the joint likelihood $P(R, D_{\text{obs}}|\theta,\xi)$ to determine θ now reduces to maximizing only $P(D_{\text{obs}}|\theta)$, namely the likelihood ignoring the missingness mechanism. The same MAR and parameter distinctness assumptions allow a similar factorization of the posterior $P(\theta,\xi|R, D_{\text{obs}})$.

Ignorability is a desirable property, and can be guaranteed in many settings when the experimenter has control over what data is missing. For example, in this thesis we will perform cross-validation experiments in which labels will be hidden completely at random, so that D_{mis} is independent of any other variable. Cross-validation thus introduces its own ignorable label missingness mechanism, separate from the missingness mechanism in the originally supplied data.

Unfortunately, in a real partially labeled problem, the missingness mechanism may not be ignorable. The labeled part of the training set is likely to come from a particular subset of data. The labeled set may over-represent examples that are easy to label. Ambiguous examples may be discarded or left unlabeled. In a text categorization task, only documents stemming from a particular source may be labeled (such as documents from a particular web-site [BM98b].)

One way to tackle non-ignorable missingness mechanisms is to collect data features that correlate with reasons why the labels are missing. By including such extra features, a mechanism that was not MAR may become MAR. The statistics literature has investigated a few techniques for directly handling non-ignorable missingness, but the techniques are problem-specific and sensitive to prior assumptions [LR87]. In practice, the ignorability assumption is widely employed, and is a valid assumption for the cross-validation experiments used in this thesis to study learning performance.

1.2.2 Predicting Labels for Specific Unlabeled Examples

The partially labeled learning task is to predict the labels of given unlabeled examples. We have already emphasized that the unlabeled examples are provided before the learning begins, so that the learner can exploit all examples. A second benefit of a priori given

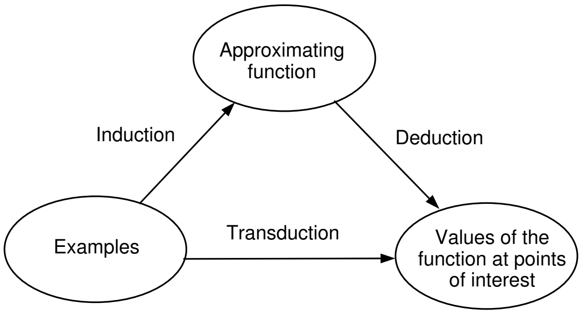


Figure 1-3: Transductive inference directly predicts labels at given points. Inductive inference first learns a decision function, capable of predicting the label of any point, and then deduces the value of the function for for particular examples. This illustration appeared in [Vap98].

examples is that the learner need not infer a classification function capable of predicting the label of an arbitrary example - the learner only needs to label the given examples.

Vapnik [Vap98] refers to the task of predicting labels of given examples as "transduction", and contrasts it with induction (inferring a classification function that predicts labels of arbitrary examples) and deduction (applying the classification function to given examples) (see Figure 1-3). The transduction task is a subset of the induction task, and Vapnik argues that transduction is an "easier" task that may requires less training data than induction followed by deduction.

Several of the proposed learning algorithms in this thesis have no functional form for the classifier decision boundary, and are transductive in this sense. Such algorithms require retraining to classify new unlabeled examples that were not provided earlier. We also introduce algorithms that do infer a complete decision function. This thesis will focus on exploiting the partially labeled data, with or without explicit decision boundaries.

The term "transduction" has loosely been used to refer to other concepts, such as minimizing the margin at unlabeled points during SVM training, and also to refer to semi-supervised learning in general. To avoid confusion, we will explicitly write "transductive margin" when referring to algorithms that minimize margin for unlabeled points.

1.3 Related Learning Problems

We have already contrasted partially labeled learning with supervised and unsupervised learning. Below we list some other related machine learning problems. Developments in the partially labeled problem are likely to benefit these other problems and vice versa.

Partially Labeled Regression

Partially labeled regression differs from partially labeled classification in that the labels y are continuous real values instead of unordered discrete categories. This problem appears to be easier in certain regards than the partially labeled classification problem. The classification problem may lead to difficult combinatorial optimization problems, which is not the case for regression. For example, maximum entropy discrimination (MED) [JMJ99] requires an approximation to make partially labeled classification computationally feasible, but has an efficient exact solution for regression [Jeb01].

Partially Labeled Learning with a Given Marginal Distribution $P(\boldsymbol{x})$

Instead of being provided with individual unlabeled examples, we may be given the true distribution of the unlabeled data $P(\boldsymbol{x})$. This problem can be seen as the limit of the standard partially labeled learning problem with infinitely many unlabeled examples. In chapter 5 we present an algorithm that addresses this problem.

Learning with Missing Data

Learning with missing data is an important problem in statistics [LR87, Sch97]. In this problem, both y values and $\boldsymbol{x}$ features may be missing for each example. The learning task may be either classification or regression, but is more frequently the estimate of a summary statistic, for example, an average household income.

Learning with Noisy or Coarse Labels

All labels may be nominally observed, but can be very noisy. In this case, the task is to reconstruct or de-noise the labels [LS01]. If we know that the noise-level for certain labels is 50% (in the case of binary classification with equal class priors), then those labels are de facto unobserved, as in the partially labeled problem.

Alternatively, the observed labels may be coarse, and the problem is to infer a fine-grained set of labels. For instance, we may have a hierarchical taxonomy of labels, and observe only the labels at the top levels of the hierarchy [MRMN98].

Learning from Positive Examples and Anomaly Detection

Sometimes we are given only positive examples and unlabeled examples [LDG00]. Positive examples and unlabeled examples may be readily available, but it can be difficult to obtain negative examples explicitly. For instance, world-wide-web bookmarks provide positive examples of interesting web pages, and unlabeled web pages are easy to collect, but negative examples (uninteresting web pages) are never labeled.

In the anomaly detection task, there are no explicit positive or negative examples, only unlabeled examples. Most unlabeled examples are assumed to be negative, and the task is to spot positive ’anomalies’ [JMJ99]. This problem is similar to estimating the support of a density [SPST$^+$01].

Query Learning and Active Learning

Query learning has the same setting with labeled and unlabeled examples as does partially labeled learning. The learner can query an oracle for labels of the unlabeled points. The challenge is to choose queries that maximize learning accuracy, to minimize the number of queries [FSST97]. We can combine active learning with partially labeled learning for a further improvement [TK01]. In the active learning problem, the learner is free to query the label of any point in the input domain, not just the unlabeled examples.

1.4 Issues, Goals and Contributions

The partially labeled learning problem has not been extensively researched yet. There exist a few proposed approaches to the problem, which will be detailed in chapter 2. However, the assumptions and principles that enable these techniques to work are often poorly understood. The techniques suffer from various limitations, and often learn less accurately than supervised learning algorithms employing only the labeled subset of the partially labeled data. Some techniques are computationally intractable for problems with more than 50 points.

This thesis develops new state-of-the-art learning algorithms for the partially labeled learning problem. The guiding design goals behind these algorithms are

Explicit principles for exploiting partially labeled data: The algorithms are based on explicit principles that make it clear how they take advantage of partially labeled data.

High learning accuracy with few labeled examples: The algorithms employ powerful models and inference techniques that make them competitive with existing partially labeled techniques as well as state-of-the art supervised learning techniques

Wide applicability: We apply the same learning algorithms to problems in different application domains, and avoid restrictive assumptions.

Computational feasibility: The algorithms are computationally feasible for large numbers of unlabeled examples.

The novel contributions of this thesis include

- a kernel classifier for partially labeled data
- a classifier for partially labeled data that lies on low-dimensional manifolds (nonlinear subspaces) embedded in a high-dimensional input domain
- parameter inference techniques based on different measures of large margins for the above partially labeled classifiers
- a regularizer and a classifier that explicitly link the marginal density $P(\boldsymbol{x})$ to the conditional density $P(y|\boldsymbol{x})$

1.5 Organization of the Thesis

The thesis consists of six chapters as follows.

Chapter 1: The Partially Labeled Learning Problem introduces the problem, formalizes it, and relates it to other learning problems.

Chapter 2: Algorithmic Framework overviews existing algorithms for the partially labeled problem, and examines what principles they use to exploit unlabeled data.

Chapter 3: Models for the Conditional and the Marginal describes two probabilistic models that link aspects of the marginal distribution $P(\boldsymbol{x})$ with the conditional distribution $P(y|\boldsymbol{x})$.

Chapter 4: Classification details how to incorporate label information into the models defined in chapter 3. Several inference techniques for training the classifiers are given.

Chapter 5: Information Regularization describes a regularizer based on mutual information that directly links the marginal with the conditional, and proposes a classifier employing this regularizer.

Chapter 6: Discussion concludes and proposes future work.

Chapters 3 and 4 are an expanded combination of the two papers [SJ01, SJ02].

Chapter 2

Algorithmic Framework

This chapter examines the assumptions necessary for successful learning with partially labeled data. We review existing algorithms for solving the problem and ask what assumptions they are using.

2.1 Assumptions

In the partially labeled learning problem, our desire is to employ both labeled and unlabeled data to improve learning accuracy. We shall study *when* and *how* unlabeled data can help in classification. However, first we illustrate that unlabeled data may not help at all.

Generally speaking, unlabeled data does not help classification when the unlabeled data distribution does not relate to the classification task, or the relation is of a form that the classifier is not equipped to represent or exploit. Consider an idealized partially labeled data problem, where we are given the true input distribution $P(\boldsymbol{x})$. Can we always improve the classification $P(y|\boldsymbol{x})$ given $P(\boldsymbol{x})$? This is not the case. Certain forms of $P(\boldsymbol{x})$ may contribute no information about $P(y|\boldsymbol{x})$ at all, alternatively $P(y|\boldsymbol{x})$ can be difficult to learn regardless of $P(\boldsymbol{x})$. For an example of the former, consider the case of uniform $P(\boldsymbol{x})$. By symmetry, we obtain no knowledge about the location of the decision boundary. At best, knowledge that $P(\boldsymbol{x})$ is uniform can prevent incorrect inferences based on small samples that have accidental clusters. We might otherwise have assumed that such clusters were real and corresponded to classes. For an example of the latter, the difficult to learn $P(y|\boldsymbol{x})$, consider a degenerate $P(y|\boldsymbol{x})$ that does not depend on $\boldsymbol{x}$ at all. Similarly, the dependence on $\boldsymbol{x}$ may be too intricate or essentially random. Learning is inefficient when $y = f(\boldsymbol{x})$, but f is a one-way function, such as a cryptographic hash. To be fair, such a situation is hopeless not just for semi-supervised learning, but for ordinary supervised learning too.

In theory, knowledge of $P(\boldsymbol{x})$ should at least not hurt classification performance, since we can always choose to ignore it. In practice, we face a model selection problem: how should we use $P(\boldsymbol{x})$, if at all? It has been experimentally observed that partially labeled learning with poor model selection decreases performance on some text classification tasks [NMTM00].

Thus, we need some assumptions for unlabeled data learning to help. We will subsequently translate the assumptions into principles that can be employed by algorithms, but that still abstract from computational aspects. Finally, we will discuss algorithms.

2.1.1 Assumptions Linking the Marginal to the Conditional

The general assumption necessary for partially labeled learning is that $P(\boldsymbol{x})$ tells us something about $P(y|\boldsymbol{x})$ relevant to the decision boundary. This assumption is also used in some supervised learning algorithms, as we shall see. However, partially labeled data learning is more dependent on this assumption, because more information from $P(\boldsymbol{x})$ must be applied towards $P(y|\boldsymbol{x})$ than in supervised learning. Both types of learning require further assumptions such as smoothness of the decision function. The labeled data and unlabeled data should be drawn IID from the same distribution. We also assume that the label missingness mechanism is ignorable (section 1.2.1) and need not be modeled.

Here we will focus on the assumption linking $P(\boldsymbol{x})$ to $P(y|\boldsymbol{x})$. Such an assumption is data-dependent and must be empirically tested on the data set of interest. Our goal is to find a general form of inductive bias that works effectively on several datasets from different domains (although there are limits to how general the bias can be [Wol96b, Wol96a].)

A general such assumption is that the conditional distribution $P(y|\boldsymbol{x})$ should not change much when the marginal density $P(\boldsymbol{x})$ is high. For example, regions of high $P(\boldsymbol{x})$ often correspond to clusters in the data. The assumption implies that points in a cluster belong to the same class (since the conditional cannot change much between two points in a contiguous cluster). Therefore, the assumption has also been termed the *cluster assumption* [See01b]. We now show that this type of assumption is made by a family of successful supervised algorithms, raising our hopes that it also represents a generally applicable inductive bias for semi-supervised learning.

Large-margin algorithms (e.g., support vector machines and boosting) prefer decision boundaries that lie in low-density regions. When the data is separable, the large margin criterion dictates that the decision boundary should separate the classes with a large distance. The higher the density, the smaller the possible margin in that region, the less desirable a decision boundary there, and consequently, the more constant the labeling. A similar argument holds true for non-separable datasets when a soft margin is used. The soft margin allows points to fall within the margin band around the decision boundary, but penalizes such points. Thus, this criterion still prefers to put the decision boundary within a low-density region to minimize the penalty and to allow large margins as before.

2.2 Theoretical Results in Partially Labeled Data

Theoretical insights into learning from partially labeled data are currently rather limited. Labeled data is exponentially more useful than unlabeled data under certain assumptions [CC95, CC96]. Generative formulations can generally take advantage of unlabeled data, whereas strictly discriminative formulations cannot [ZO00]. Under the co-training assumptions [BM98b], partially labeled data that is provably useful. Generalization error bounds extend to partially labeled data classifiers by counting the number of resulting equivalence classes [Vap98]. Unfortunately, little is known about general properties of unlabeled data that can be exploited for supervised learning.

2.3 Existing Algorithms for Partially Labeled Data

Several practical algorithms for partially labeled data have been proposed. Techniques based on joint density modeling are the oldest and most established, but large-margin techniques, co-training, and kernels for partially labeled data have also become popular.

2.3.1 Joint Density Models

Generative models that jointly account for both $\boldsymbol{x}$ and y can naturally take advantage of partially labeled data. Consider a joint model $P(\boldsymbol{x}, y|\boldsymbol{\theta})$ with parameters $\boldsymbol{\theta}$. Such a model contains a marginal model of $\boldsymbol{x}$, namely $P(\boldsymbol{x}|\boldsymbol{\theta}) = \sum_y P(\boldsymbol{x}, y|\boldsymbol{\theta})$. Hence, unlabeled examples can be used for estimating $\boldsymbol{\theta}$. The parameters may for instance be chosen to maximize the joint likelihood.

To apply this idea to a particular problem, we must choose an appropriate probabilistic model. Gaussian mixture models are a popular and versatile choice. Each example is assumed to be generated from some mixture component $i \in M$, and the likelihood of an example $(\boldsymbol{x}, \tilde{y})$ is $P(\boldsymbol{x}, \tilde{y}|\boldsymbol{\theta}) = \sum_i P(i|\boldsymbol{\theta}) P(\boldsymbol{x}, \tilde{y}|i, \boldsymbol{\theta})$. To maximize the joint log-likelihood, we solve the optimization problem

$$\max_{\boldsymbol{\theta}} \sum_{k \in L} \log \sum_{i \in M} P(i|\boldsymbol{\theta}) P(\tilde{y}_k|i, \boldsymbol{x}_k, \boldsymbol{\theta}) P(\boldsymbol{x}_k|i, \boldsymbol{\theta}) + \sum_{k \in U} \log \sum_{i \in M} P(i|\boldsymbol{\theta}) P(\boldsymbol{x}_k|i, \boldsymbol{\theta}). \tag{2.1}$$

The unlabeled examples $k \in U$ have no label y and hence we only maximize the marginal likelihood of $\boldsymbol{x}$ for them. The expectation-maximization algorithm (EM) provides an efficient way of solving the optimization problem. EM alternates between estimating model parameters $\boldsymbol{\theta}$ and inferring soft labels for the unlabeled examples.

The Gaussian mixture model with partially labeled data has been applied successfully for modeling satellite imagery [MU97]. The multinomial naive Bayes model has been used to classify partially labeled text documents [NMTM00].

Unfortunately, there are several disadvantages with this approach. One limitation is that most generative model make fairly restrictive assumptions about the data distribution, which may not be true and cause the models to fail. We would like to model very general classes of distributions, which is possible via discriminative modeling [RH]. A second issue is that for classification purposes, we are only interested in obtaining the conditional model $P(y|\boldsymbol{x})$. The joint estimation approach may focus too much attention on modeling aspects of $P(\boldsymbol{x})$ we do not care about, and not pay enough attention to $P(y|\boldsymbol{x})$. For example, when there are many unlabeled points, the unlabeled data may dominate the likelihood (eq. 2.1) and the labeled data may be hardly modeled at all, resulting in poor classification performance. One possible solution is to downweight the unlabeled data part of the likelihood, but then the question becomes how to allocate the weights to the two sources of data [CJ02].

In this thesis, we will employ conditional probability models rather than joint density models, which avoids the problems mentioned above.

2.3.2 Large-margin Techniques with Unlabeled Data

Transduction with support vector machines [Vap98, BD99] attempts to maximize the classification margin on both labeled and unlabeled data, while classifying the labeled data as correctly as possible. This discriminative method imposes fewer restrictions on the data model than do Gaussian mixture models. However, finding the optimal decision boundary requires searching over possible labelings of unlabeled points. The search reduces to a mixed integer programming problem that is NP-complete. The maximum entropy discrimination framework [JMJ99] also optimizes the margin based on both labeled and unlabeled data. However, instead of searching over the labelings directly, it searches over distributions of labelings for both labeled and unlabeled points. It tries to satisfy classification constraints subject to being close to a prior distribution over the labelings. The prior for labeled points peaks around 0 or 1 according to their class, whereas priors on unlabeled points are peaked at 0.5. This formulation also has a cost that is exponential in the number of unlabeled points, but a mean-field approximation makes the problem feasible.

2.3.3 Co-training

Co-training [BM98b] classifies data that exhibits multiple views, specifically when data attributes can be partitioned into groups ("views") that are individually sufficient for learning but conditionally independent given the class. Co-training works by feeding classifications made by one learner as examples for the other and vice-versa. However, in practice the individual learners are noisy, and then co-training easily veers down the wrong path.

2.3.4 Kernels with Partially Labeled Data

Unlabeled data can be used to adapt kernels and distance metrics. In the Fisher kernel approach [JH99], unlabeled data is used to train a generative model from which a kernel is derived. The kernel can then be used in any discriminative kernel method, such as a support vector machine or a nearest neighbor classifier [Hof00].

Chapter 3

Models for the Conditional and the Marginal

This chapter introduces models for partially labeled data. Our classifier takes a set of labeled and unlabeled examples, and outputs labels for the unlabeled examples. To perform this task, the classifier needs to have a model for the data. The model encodes knowledge about the structure of the input space $\mathcal{X}$, such as similarity measures or distance metrics. It also captures assumptions about how closeness in the input space relates to closeness in the output space $\mathcal{Y}$ (the labels). Finally, the model incorporates assumptions about noise in both domains. We will employ probabilistic models, because they are effective for describing uncertainty and noise, and have well-founded principles and tools of inference.

To perform the classification task, all we need is a model for the conditional $P(y|\boldsymbol{x})$. However, in the process of building this conditional model, we want to incorporate unlabeled data. It supplies no direct information about the labels y, but does tell us about the marginal $P(\boldsymbol{x})$. To benefit from the unlabeled data, we need to model aspects of marginal $P(\boldsymbol{x})$ and combine it with the conditional. The challenge is to focus on the aspects of $P(\boldsymbol{x})$ that are beneficial to determining $P(y|\boldsymbol{x})$. Other aspects of $P(\boldsymbol{x})$ may be irrelevant for the classification task.

The chapter introduces models for both the marginal and the conditional, and shows how they fit together. The marginal models take points in the input space, and build probabilistic models or representations of them. The conditional model operates on the representation, and arrives at a classification. The parameters are estimated from the labeled data during classifier training. Parameter estimation is discussed in depth in chapter 4, and here we only use a simple maximum likelihood estimator.

This chapter focuses on different models for incorporating aspects of the marginal $P(\boldsymbol{x})$. We want general models that apply to many types of data, and therefore we employ nonparametric models. These methods learn quickly from few examples, and prior knowledge can be incorporated by choosing appropriate kernels and distance metrics, and by preprocessing the data.

We introduce two models for aspects of the marginal: the kernel expansion and the Markov random walk. The kernel expansion representation is a direct nonparametric approach. It employs a general-purpose kernel density model, which often works well, but does not take advantage of any special structure in the data input space $\mathcal{X}$. The Markov random walk representation is a refined approach that can exploit certain structure in the marginal, in particular clusters at a "relevant" resolution. It also models data that approx-

Table 3.1: Notation

Symbol	Explanation
$\boldsymbol{x}$	Feature vector for a data point, $\boldsymbol{x} \in \mathbb{R}^d$ in d-dimensions
y	Label for a data point. An integer ranging over possible classes, ± 1 for binary classes.
$\tilde{y}$	Observed label.
L	Set of labeled points
U	Set of unlabeled points
C	Set of classes
N_L	Number of labeled points
N_U	Number of unlabeled points
N_{LU}	Number of labeled plus unlabeled points
N_{L_y}	Number of labeled points in class y
N_C	Number of classes
$[P(i\|\boldsymbol{x})]_{i \in L}$	Vector $[P(i=1\|\boldsymbol{x}), \ldots, P(i=N_L\|\boldsymbol{x})]$

imately occupies low-dimensional subspaces of the input space. The data may be locally Euclidean and vary only along a few dimensions. However, the relevant dimensions may change in space, so that the data approximately lies on a curved manifold. The Markov random walk can tackle such manifolds while robustly handling noisy points far from the subspace. Both the Markov random walk and the kernel expansion representations use labeled data only to a limited degree, e.g., to determine what aspects of the marginal to focus on, in terms of model scale or smoothness.

Our models are instances of general model building for partially labeled data. There have been a few other attempts along these lines, including the Fisher kernel [JH99, Hof00] and other “kernel engineering” methods [LCB$^+$02, CWS02, BM98b]. Section 2.3.4 describes this related work.

The chapter concludes by discussing sample complexity and generalization guarantees.

3.1 The Kernel Expansion Representation

We begin by using only labeled data. We review joint density estimation of $P(\boldsymbol{x}, y)$, which implicitly contains models for both the marginal $P(\boldsymbol{x})$ and the conditional $P(y|\boldsymbol{x})$. Joint density estimation can be done by putting kernels at each data point, both for $\boldsymbol{x}$ and y. For classification purposes, we want to estimate conditional probabilities. The conditional can be obtained from the joint via Bayes rule, and we get a kernel density classifier for labeled data. The form of the classifier suggests a natural model or representation for $\boldsymbol{x}$, which we call the kernel expansion representation. This representation is density-dependent.

Next, we incorporate unlabeled data into the classifier. We simply apply the kernel expansion representation including the unlabeled points. However, y values are not available for unlabeled points, but we can hypothesize labels for them, and treating them as parameters to be estimated. We then demonstrate classification behavior on synthetic data, and analyze the effect of the unlabeled data. Finally, we show real data experiments.

3.1.1 Joint Density Estimation

We start by assuming a large number of *labeled* examples $D = \{(\boldsymbol{x}_1, \tilde{y}_1), \ldots, (\boldsymbol{x}_{N_L}, \tilde{y}_{N_L})\}$ (refer to table 3.1 for notation). A joint kernel[1] density estimate can be written as

$$P(\boldsymbol{x}, y) = \frac{1}{N_L} \sum_{i \in L} K_y(y, \tilde{y}_i)\, K_x(\boldsymbol{x}, \boldsymbol{x}_i), \tag{3.1}$$

where the kernels are functions such that $\int K_x(\boldsymbol{x}, \boldsymbol{x}_i)\, \mathrm{d}\mu(\boldsymbol{x}) = 1$ and $\sum_y K_y(y, \tilde{y}_i) = 1$ for each i. A common choice is Gaussian kernels for $\boldsymbol{x}$, $K_x(\boldsymbol{x}, \boldsymbol{x}_i) = N(\boldsymbol{x};\, \boldsymbol{x}_i, \sigma^2 I\,)$, as in a Parzen windows estimator. Since y is discrete, a possible choice is $K_y(y, \tilde{y}_i) = \delta(y, \tilde{y}_i)$ (where the delta function $\delta(y, \tilde{y}_i)$=1 when y=$\tilde{y}_i$ and is 0 otherwise). By adjusting the width of the kernel appropriately as a function of the number data points N_L, we can obtain a *consistent* density estimate converging to the true joint density $P(\boldsymbol{x}, y)$ as $N_L \to \infty$ [Sco92, Sil98][2].

Given a fixed number of examples, the kernel functions $K_x(\boldsymbol{x}, \boldsymbol{x}_i)$ may be viewed as conditional probabilities $P(\boldsymbol{x}|\boldsymbol{x}_i)$, where i indexes the observed points. We shall denote a point x_i simply by its index i, and write this conditional probability as $P(\boldsymbol{x}|i)$. Likewise, the observed labels $\tilde{y}_i$ may be noisy, which we can model with soft kernel functions $K_y(y, \tilde{y}_i) = Q(y|i)$. The parameter $Q(y|i)$ is the probability that point $\boldsymbol{x}_i$ has a true label y, even though the observed label was $\tilde{y}_i$. For binary classification, we can take $Q(y|i) = q^{1-\delta(y,\tilde{y}_i)}(1-q)^{\delta(y,\tilde{y}_i)}$ for some given label noise level $q \in [0, 1]$. The resulting joint density model is

$$P(\boldsymbol{x}, y) = \frac{1}{N_L} \sum_{i \in L} Q(y|i)\, P(\boldsymbol{x}|i) \tag{3.2}$$

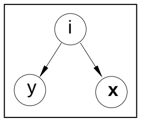


Figure: Independence diagram.

We assume that there is an equal probability of observing any of the points, and we can interpret $1/N_L$ as a uniform prior probability $P(i)$ of the index variable $i = 1, \ldots, N_L$. The resulting model conforms to the independence diagram depicted above. This is reminiscent to a mixture model with latent class variables i (e.g., the aspect model [HP98]). However, we associate a component i with each training example, and the probability $P(\boldsymbol{x}|i)$ is trivially fit to the data by centering a kernel function on i. The kernel function is only adjustable for a global scale (kernel width). This is unlike mixture models where there are only a few components i, but where $P(\boldsymbol{x}|i)$ is adjusted in both mean and covariance to fit the overall density. Our simple fitting yields a significant computational savings when training for classification, which is the objective in this thesis.

[1] The word "kernel" has two distinct meanings in this thesis: it may refer to a kernel for density estimation, or a kernel used by kernel machines. The former must integrate to 1. The latter must be a symmetric and positive semidefinite function.

[2] We also require that $\int |K_x|\, \mathrm{d}x < \infty$. For consistency, as the number of data point increases, the kernel bandwidth σ must tend to 0, yet slowly enough so that the number of data points covered by the kernel tends to infinity; formally, as $N_L \to \infty$ we have $\sigma \to 0$ and $N_L \sigma^d \to \infty$ where d is the number of dimensions.

3.1.2 Conditional Probability Estimation

The conditional probability of the label y given an example $\boldsymbol{x}$ is given by

$$P(y|\boldsymbol{x}) = \sum_{i \in L} Q(y|i) P(i|\boldsymbol{x}), \tag{3.3}$$

where $P(i|\boldsymbol{x}) = P(\boldsymbol{x}|i)P(i)/P(\boldsymbol{x})$, and $P(i)$ is uniform as before. The quality of the conditional probability depends both on how accurately $Q(y|i)$ are known as well as on the properties of the membership probabilities $P(i|\boldsymbol{x})$ (always known) that must be relatively smooth.

In the resulting conditional kernel probability estimate, each example $\boldsymbol{x}$ is represented by a vector of membership probabilities

$$[P(i = 1|\boldsymbol{x}),\ \ldots, P(i = N_L\ |\boldsymbol{x})] \tag{3.4}$$

Thus, we can view the kernel expansion as taking original points $\boldsymbol{x}$ and changing their representation, before they are used for classification. For radial kernels, the new representation is translation and rotation invariant. It always sums to 1, which makes it a competitive representation: the kernels compete to explain the data, and if one kernel is better, the others will be deemphasized [HKP91, p. 217].

Classification is performed simply by choosing the class with the maximum conditional probability: $y = \text{argmax}_c\, P(y = c|\boldsymbol{x})$. For binary classes, we can write the classifier as the sign of $f(\boldsymbol{x}) = \sum_i w_i P(i|\boldsymbol{x})$, with weights $w_i = Q(y = 1|i) - Q(y = -1|i)$. This is a kernel density classifier [HTF01, p. 184]. It differs from a linear classifier in that the weights $Q(y|i)$ and w_i are bounded, $0 \leq Q(y|i) \leq 1$ and $-1 \leq w_i \leq 1$. This limits the influence of outliers. The kernel classifier can straightforwardly be extended to kernel regression by considering continuous y values, e.g., to obtain the Naradaya-Watson estimator [Bis95].

3.1.3 Conditional Probability Estimation with Partially Labeled Data

We now assume that we have labels for only a few examples, and our training data is $\{(\boldsymbol{x}_1, \tilde{y}_1), \ldots, (\boldsymbol{x}_{N_L}, \tilde{y}_{N_L}), \boldsymbol{x}_{N_L+1}, \ldots, \boldsymbol{x}_{N_{LU}}\}$. Typically, $N_L \ll N_U$. We still want to estimate the conditional probabilities $P(y|\boldsymbol{x}_k)$ using the model defined above. Any point $\boldsymbol{x}$ is now expanded in terms of both labeled and unlabeled points. Since y is now unknown for most points, we treat $Q(y|i)$, $i = 1, \ldots, N_{LU}$ as free parameters to be estimated from the few labeled examples. The parameters must still be valid probabilities, so $Q(y|i) \in [0, 1]$ and $\sum_y Q(y|i) = 1$. A common approach is to choose the parameters to maximize the conditional log-likelihood

$$\max_{\{Q(y|i)\}} \sum_{k \in L} \log P(\tilde{y}_k|\boldsymbol{x}_k) = \max_{\{Q(y|i)\}} \sum_{k \in L} \log \sum_{i \in L \cup U} Q(\tilde{y}_k|i) P(i|\boldsymbol{x}_k), \tag{3.5}$$

where the first summation is only over the labeled examples. Importantly, the log-likelihood is a concave function of $Q(y|i)$, so the optimization has no local maxima, and is computationally feasible. The solution is readily found via the EM algorithm. The EM algorithm alternates between estimating what components i are responsible for a data point $(\boldsymbol{x}_k, \tilde{y}_k)$ (the "E" step), and then estimating the parameters $Q(y|i)$ (the "M" step). For concave objectives, EM is guaranteed to converge to a global maximum. We provide details of parameter estimation with EM and other techniques in chapter 4.

Almost all free parameters in kernel expansion are tied to the discriminative task. When used with diagonal Gaussian kernels, the representation has only one free parameter, the kernel width σ, responsible for modeling $P(\boldsymbol{x})$. All the other parameters, namely the weights $Q(y|i)$, are associated with the task of estimating $P(y|\boldsymbol{x})$.

To ensure that the kernel expansion classifier has good generalization performance on new data, we must ensure that it does not overfit the training data. The issue is that as the number of unlabeled examples increases, the kernel expansion representation gets bigger, requires fitting more parameters $Q(y|i)$, and may become overly flexible. There are two sources for regularizing the capacity of the classifier. Firstly, we impose smoothness constraints on the membership probabilities $P(i|\boldsymbol{x})$. This corresponds to making the kernels sufficiently wide and restricts the effective degrees of freedom of the classifier. Its conditionals cannot change too much from one location to another. For example, when the kernels become very wide, all data will be classified in the same class (in lemma 2, p. 40 we quantize the degrees of freedom more generally). Secondly, we can regularize the parameter estimates $Q(y|i)$. Parameters should be set to as neutral values as possible while affording good classification. If a parameter is not helpful for the task, it ought to be flat $Q(y|i) = 1/N_C$ for all classes y. In this way, unlabeled data outliers can effectively be eliminated from affecting classification decisions. We will discuss maximum entropy and other parameter estimation regularizers in chapter 4. In this chapter, we use maximum likelihood parameter estimation, which does not regularize parameter estimates, so we will rely entirely on the smoothness of the representation.

3.1.4 Classification Examples with Synthetic Data

Figure 3-1 shows classification examples with kernel expansion. There are 500 unlabeled data points, sampled from two Gaussians. Two labeled points are located at the center of each Gaussian. We apply kernel expansion with various kernel widths σ = 0.1, 0.3, 0.6, and 1.5. We show the resulting decision boundaries and level curves of the conditional density using the maximum likelihood estimator. For appropriate kernel widths (σ=0.3 or 0.6), the decision boundary curves to avoid high density regions near the labeled points. When the kernel is too narrow (σ=0.1) the maximum likelihood estimator overfits (in chapter 4 we regularize classifier weights to address this problem). For σ=1.5, the boundary is a slanted straight line, due to the very smooth kernels used by the representation. For comparison, Figure 3-2 shows the decision boundary when the Gaussians are not angled. The boundary of kernel expansion (σ=0.6) is then approximately a straight line, which is the same boundary as a support vector machine based on only labeled data would give.

3.2 How Kernel Expansion Helps Estimating the Conditional

3.2.1 Asymptotic Motivation

The kernel expansion with only labeled data (eq. 3.3) is a conditional probability estimator. When the kernels underlying $P(i|\boldsymbol{x})$ and $Q(y|i)$ are chosen appropriately, this estimator will be consistent. In other words, as the number of labeled points tends to infinity, its estimates will approach the true conditional probabilities. A classifier based on these probabilities will be Bayes optimal.

With a finite number of labeled points, but infinite number of unlabeled points, the kernels $P(i|\boldsymbol{x})$ provide the power to *represent* the Bayes optimal decision boundary, given

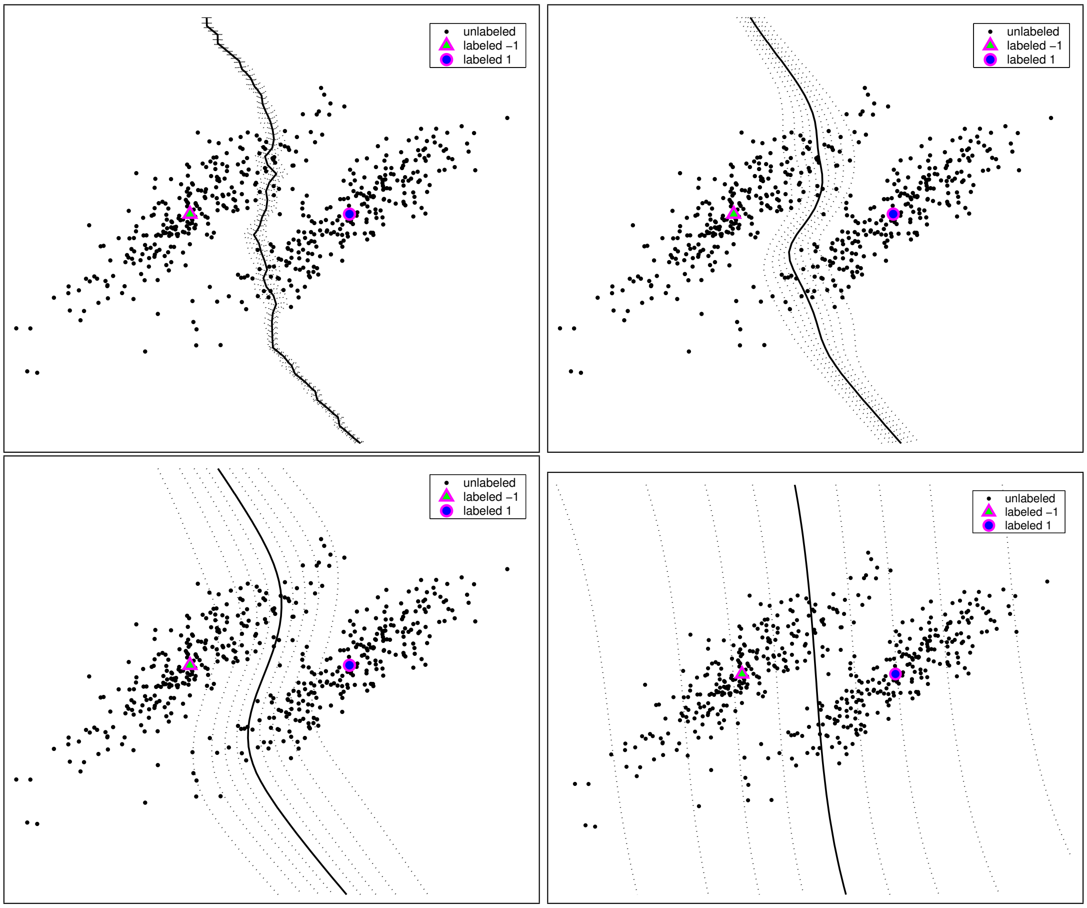


Figure 3-1: Kernel expansion decision boundaries (solid) and level curves (dashed). The level curves are for 0.2, 0.4, 0.6, 0.8 and their negations. The kernel widths are $\sigma = 0.1$ (top left), 0.3, 0.6, and 1.5 (bottom right). The box has extent 3×3.

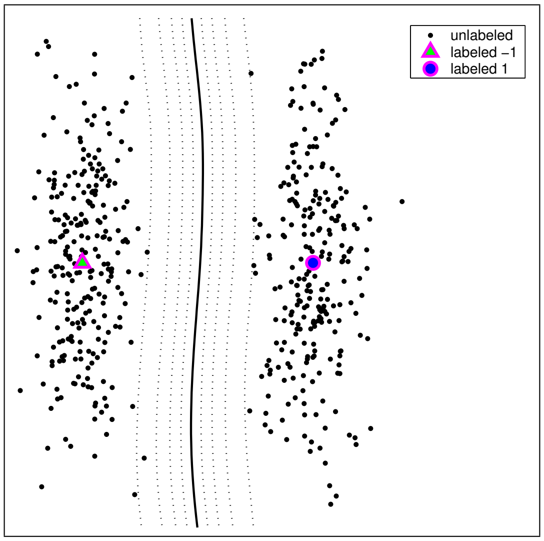


Figure 3-2: Kernel expansion decision boundaries (solid) and level curves (dashed). The level curves are for 0.2, 0.4, 0.6, 0.8 and their negations. The kernel width is $\sigma = 0.6$. The box has extent 3×3.

the right settings of $Q(y|i)$. However, we must estimate $Q(y|i)$ based on the finite labeled data, which limits accuracy.

3.2.2 Unlabeled Data Modifies the Hypothesis Space

Here we study the case of finite labeled and unlabeled data. The partially labeled data is used to obtain a *density-dependent* representation. We compare the kernel expansion using only the labeled points versus using both labeled and unlabeled points. We will show how the addition of unlabeled data modifies the hypothesis space of the kernel expansion classifier.

Kernel expansion places kernels on the data points. Each additional unlabeled point contributes an extra parameter $Q(y|i)$ influencing the conditional density around it (Gaussians and most typical kernels decrease monotonically radially.) As a result, we allow decision boundaries that are finely tailored around the unlabeled points, which were not possible before. Note that we can still express the same decision boundaries as with only labeled points, just by setting the parameters $Q(y|i)$ of unlabeled points to 0. The addition of unlabeled points increases the normalization constant of the representation. For illustration, consider a kernel expansion in terms of two labeled points, plus a third unlabeled point coinciding with the second point. The second and the third point will have the same coordinates in the representation. We use a Gaussian kernel $K(\boldsymbol{x}_1, \boldsymbol{x}_k) \propto \mathrm{e}^{-d_{1k}^2}$.

$$\text{Representation, labeled only } \; [P(i|\boldsymbol{x}_k)]_i = \frac{1}{Z_k}\left[\mathrm{e}^{-d_{1k}^2}, \quad \mathrm{e}^{-d_{2k}^2}\right] \tag{3.6}$$

$$\text{Representation, with unlabeled } \; [\bar{P}(i|\boldsymbol{x}_k)]_i = \frac{1}{V_k}\frac{1}{Z_k}\left[\mathrm{e}^{-d_{1k}^2}, \quad \mathrm{e}^{-d_{2k}^2}, \quad \mathrm{e}^{-d_{2k}^2}\right] \tag{3.7}$$

The normalization constant V_k reduces the magnitude of the conditional probabilities. However, we will be able to make the same classification decisions, because the decisions are not affected by scaling parameters of the representation.

3.3 The Relation between Kernel Expansion and Kernel Methods

Despite its name, the kernel expansion is *not* a valid kernel for kernel machines.[3] It is a *normalized* kernel, which no longer is symmetric nor positive semidefinite. Therefore, it is incorrect to train support vector machine with a kernel $K(i,k) = P(i|\boldsymbol{x}_k)$, or its symmetric version $K(i,k) = P(i|\boldsymbol{x}_k) + P(k|\boldsymbol{x}_i)$.

However, it is possible to use the features produced by kernel expansion as input features for a regularized linear classifier, e.g., an SVM. Expand the features explicitly producing vectors $\boldsymbol{\phi}(\boldsymbol{x}) = [P(i|\boldsymbol{x})]_{i\in L\cup U}$ before training the linear classifier $f(\boldsymbol{x}) = \boldsymbol{w}\cdot\boldsymbol{\phi}(\boldsymbol{x})$ (e.g., a linear SVM with no offset $\boldsymbol{b}$ term.) The SVM does not take advantage of the probabilistic interpretation of the features, and regularizes the weights $\boldsymbol{w}$ with an L_2 norm, which is not really appropriate, and does not ensure that the weights are differences of valid probabilities in $[0,1]$. This scheme can still give good performance results in practice.

[3] Perhaps a more complete name would have been conditional kernel expansion, or normalized kernel expansion.

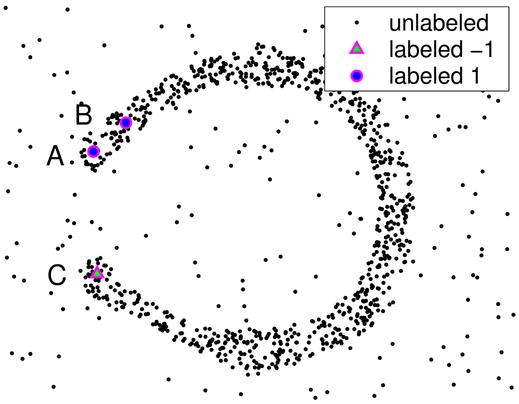


Figure 3-3: Data approximately occupying a one-dimensional curved manifold.

3.4 The Markov Random Walk Representation

Real datasets have intricate structure. The data may contain clusters, and may approximately lie in a low-dimensional subspace of the embedding space. For example, face images can be represented well with few basis functions [TP91]. Moreover, the data may be low-dimensional only locally, and the data subspace may curve nonlinearly. Figure 3-3 shows an example of data that approximately occupies a curved manifold.

Unfortunately, kernel methods (such as kernel expansion) may perform poorly when the data is locally low-dimensional [Sco92, p. 152]. Their kernels extend in all dimensions, and ignore both the low dimensionality and the curved structure of the manifold. The kernels must be rather wide to provide sufficient smoothness, but wide kernels may incorrectly span across two separate parts of the manifold. Methods that rely on generic global similarity measures may likewise fail, as data-independent global metrics do not follow the data manifold and do not take the density of the data into account.

We will now model a richer set of aspects of the marginal density than was possible with kernel expansion. We introduce a representation that employs unlabeled data to follow potentially low-dimensional data manifolds, and to capture cluster structure along the manifold. The representation is robust against noisy points located off the manifolds, and has a tunable resolution to accommodate different scales and numbers of clusters. The representation is used analogously to the kernel expansion representation, and the classifier has the same form and parameters in both cases.

3.4.1 Defining the Representation

Classifiers benefit from having an accurate global similarity measure, which allows them to determine the effects of both nearby and distant points. Data points are typically given in a global coordinate system with an associated metric. Unfortunately, while the metric may provide a reasonable local similarity measure, it is frequently inadequate as a measure of global similarity. For instance, in Figure 3-3 a Euclidean metric correctly regards points A and B as similar, but mistakenly believes that points A and C are fairly similar as well. The question is precisely how to measure similarity; we will define similarity with respect to a data model that is distinct from locality in space.

Therefore, we use the provided metric only to determine local neighbors. We represent

Given a set of points $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_{LU}}\}$ and local metric $d(\boldsymbol{x}_i, \boldsymbol{x}_k)$, form

1. A neighborhood graph: $W_{ik} = \begin{cases} \mathrm{e}^{-d(\boldsymbol{x}_i, \boldsymbol{x}_k)/\sigma} & \text{if } \boldsymbol{x}_i \text{ and } \boldsymbol{x}_k \text{ are } K\text{-nearest neighbors.} \\ & \text{Include self-loops } i{=}k. \\ 0 & \text{otherwise.} \end{cases}$

2. The local transition probability from i to k is $P_{1|0}(k|i) = \dfrac{W_{ik}}{\sum_j W_{ij}}$. Collect them into a matrix $\boldsymbol{A}$, where $\boldsymbol{A}_{ik} = P_{1|0}(k|i)$.

3. The t-step transition probability from i to k via a random walk is $[\boldsymbol{A}^t]_{ik}$ and denoted by $P_{t|0}(k|i)$.

4. Compute $\boldsymbol{A}^t\boldsymbol{Z}^{-1}$, where $\boldsymbol{Z}^{-1}$ is a diagonal matrix that normalizes the columns of $\boldsymbol{A}^t$ to sum to one. Let the k-th column be the representation vector for point k. This vector contains the starting probabilities of a random walk ending at k: $[P_{0|t}(i = 1|k), \ldots, P_{0|t}(i = N|k)]$.

Figure 3-4: Summary of the steps for constructing the Markov random walk representation.

this as a weighted graph with edges connecting the neighbors. The global similarity of two points could now be defined as the shortest distance between them in this graph. Unfortunately, shortest paths are susceptible to noise. Instead, we measure the "volume" of paths connecting two examples in the graph. This is done robustly by considering all paths via a Markov random walk. The paths follow the data manifold, and high density clusters on the manifold have more paths. The number of transitions in the Markov random walk determines the scale at which clustering can be incorporated.

We now cover the four steps in detail. Firstly, construct an adjacency graph with nodes corresponding to the points $\{\boldsymbol{x}_i\}_{i \in L \cup U}$, and with undirected edges between K-nearest neighbors, and self-edges from the points back to themselves. We will denote nodes in the graph simply by indices i and k, instead of $\boldsymbol{x}_i$ and $\boldsymbol{x}_k$. Assign a weight $W_{ik} = \exp(-d(\boldsymbol{x}_i, \boldsymbol{x}_k)/\sigma)$ to each undirected edge in the graph. The weights are symmetric ($W_{ik} = W_{ki}$), and $W_{ii} = 1$ for self-transitions. Weights for non-neighbors are 0, since we trust the metric only locally and only allow local transitions that follow the manifold. The weights decay exponentially with distance, not with the distance squared as Gaussians do. Then, the product of weights along a path relates to its total distance. This nicely reconciles the multiplicative nature of probabilities with the additive nature of distances.

Secondly, obtain the one-step transition probabilities $P_{1|0}(k|i)$ from i to k by normalizing the weights from i. (The notation $P_{t_2|t_1}(k|i)$ will denote the probability of a transition between node k at time step t_2 given node i at step t_1.) While the weights W_{ik} are symmetric, the transition probabilities $P_{1|0}(k|i)$ generally are not, because the normalization varies across nodes. Typically, $P_{1|0}(k|i) > P_{1|0}(i|k)$ whenever $\boldsymbol{x}_k$ lies in a higher-density region than $\boldsymbol{x}_i$. We can organize the one-step transition probabilities as a matrix $\boldsymbol{A}$ whose i, k-th entry is $P_{1|0}(k|i)$. The matrix $\boldsymbol{A}$ is row stochastic so that rows sum to 1.

Thirdly, follow the data manifold by performing a random walk on the graph. The probability of transitioning from point i to point k in t steps is a measure of the volume of paths between these points. Let $P_{t|0}(k|i)$ denote the t-step transition probabilities, then

$P_{t|0}(k|i) = \boldsymbol{A}^t$. If there is a high density of points directly between i and k, there will be many short paths, yielding a high transition probability. The measure is robust with respect to outliers that short-circuit the manifold (figure 3.4.2, top left). Such outliers lie in low density regions, and even if paths through outliers are shorter, there are exponentially fewer such paths and they do not contribute significantly to the overall probability.

Finally, construct the representation from the Markov random walk probabilities. Each point k is represented by a vector of conditional probabilities $[P_{0|t}(i|k)]_{i \in L \cup U}$ over the possible starting states i of a random walk ending in k. The points in this representation are close whenever they have nearly the same distribution over the starting states; in other words, when the Markov random walk makes the starting points appear indistinguishable. We calculate the starting probabilities $P_{0|t}(i|k)$ from the ending probabilities $P_{t|0}(k|i)$ and Bayes rule, assuming that the starting points are chosen uniformly at random, i.e., $P(i) = 1/N$. The normalization required by Bayes rule can be written as a matrix multiplication, so that $P_{0|t}(i|k) = [\boldsymbol{A}^t \boldsymbol{Z}^{-1}]_{ik}$, where $\boldsymbol{Z}$ is diagonal and $\boldsymbol{Z}_{kk} = \sum_i [\boldsymbol{A}^t]_{ik}$.

This representation is crucially affected by the time scale parameter t. When $t \to \infty$, all the points become indistinguishable provided that the original neighborhood graph is connected. Small values of t, on the other hand, merge points in small clusters. In this representation t, controls the resolution at which we look at the data points (cf [TS01]).

The representation is also influenced by K, σ, and the local distance metric d, which together define the one-step transition probabilities (see section 3.4.4).

Graph structures have previously been used to exploit manifold structure in the data [MS94, TdSL00, RS00], and Markov random walks have been used for clustering [TS01, MS01]. Kernel methods for graphs are developed in [KL02, CWS02]. In section 3.4.3 we show how random walks are related to spectral methods [BN02, MS01, SM00, NJW02].

3.4.2 Conditional Probability Estimation and Classification

We are now given a partially labeled data set $\{(\boldsymbol{x}_1, \tilde{y}_1), \ldots, (\boldsymbol{x}_{N_L}, \tilde{y}_{N_L}), \boldsymbol{x}_{N_L+1}, \ldots, \boldsymbol{x}_{N_{LU}}\}$, and wish to classify these points. Assume the following model for generating labels. Interpret a given point k as a sample from a random walk that ended at k after t steps. Infer what points i the walk may have started from; i.e., consider the backward random walk. All starting points i have a label parameter $Q(y|i)$ (as in kernel expansion, section 3.1.3). Assign a label to the end point k according to the weighted average of its starting point distributions:

$$P(y|k) = \sum_{i \in L \cup U} Q(y|i) P_{0|t}(i|k). \tag{3.8}$$

This is analogous to eq. 3.3 for kernel expansion. To classify the k-th point, choose the class that maximizes the conditional: $y_k = \mathrm{argmax}_c\, P(y = c|k)$. For binary classification we can write $f(\boldsymbol{x}) = \sum_i w_i P_{0|t}(i|k)$, with weights $w_i = Q(y = 1|i) - Q(y = -1|i)$. This is similar to a linear classifier but with bounded weights.

If all points were labeled, the label parameters could be set directly as $Q(y|i) = \delta(y, \tilde{y}_i)$. When most points have noisy labels or no labels, the label parameters $Q(y|i)$ are unknown, and need to be estimated. We can apply conditional likelihood estimation, or any of the techniques in chapter 4.

Classifying New Test Points

The classifier can directly process only points given at training time, because the Markov walk representation $[P_{0|t}(i|k)]_i$ is computed only for those points. There is no closed functional form for a continuous decision boundary. New test points should be incorporated by updating the random walk and retraining the classifier. The advantage is that new unlabeled data is exploited to improve classification. However, full retraining is computationally demanding, and may be unsuitable for online scenarios where new points must be immediately classified.

Retraining can be avoided by approximating representations of new test points in terms of representations of existing points. The influence of a single new point on existing representations can be bounded, due to the robustness of the representation (see [NZJ01]). Let k' be a new test point at location $\boldsymbol{x}_{k'}$. Calculate the one-step transitions $P_{1|0}(k'|j)$ from the K-nearest neighbors j as usual, and approximate

$$P_{t|0}(k'|i) \approx \sum_{j \in L \cup U} P_{t-1|0}(j|i) P_{1|0}(k'|j). \tag{3.9}$$

Here $P_{t-1|0}(j|i)$ is a precomputed random walk without the new point k', and i and j range only over the old points. Finally, $P_{0|t}(i|k') \propto P_{t|0}(k'|i)$. The new point is expressed using existing points, so the classifier can be applied without retraining. Thus, classification of a new point requires $O(N)$ operations, plus the cost of finding its nearest neighbors, which is substantially cheaper than an $O(N^3)$ recalculation. However, the new point is not exploited as partially labeled data.

Examples

Consider an example (figure 3-5) of classification with Markov random walks. We are given 2 labeled and 148 unlabeled points in an intertwining two moons pattern. This pattern has a manifold structure where distances are locally but not globally Euclidean, due to the curved arms. Therefore, the pattern is difficult to classify for traditional algorithms using global metrics, such as SVM. We use a Euclidean local metric, K=5 and σ=0.6 (the box has extent 2 × 2), and show three different timescales. At t=3 the random walk has not connected all unlabeled points to some labeled point. The parameters for unconnected points do not affect likelihood, so we assign them uniformly to both classes. The other points have a path to only one of the classes, and are therefore fully assigned to that class. At t=10 all points have paths to labeled points but the Markov process has not mixed well. Some paths do not follow the curved high-density structure, and instead cross between the two clusters. When the Markov process is well-mixed at t=30, the points are appropriately labeled. The parameter assignments are hard, but the class conditionals are weighted averages and remain soft.

3.4.3 A Comparison of Different Random Walk Models

In this section we compare two alternative random walk models, and relate random walks to models based on diffusion and to spectral clustering techniques.

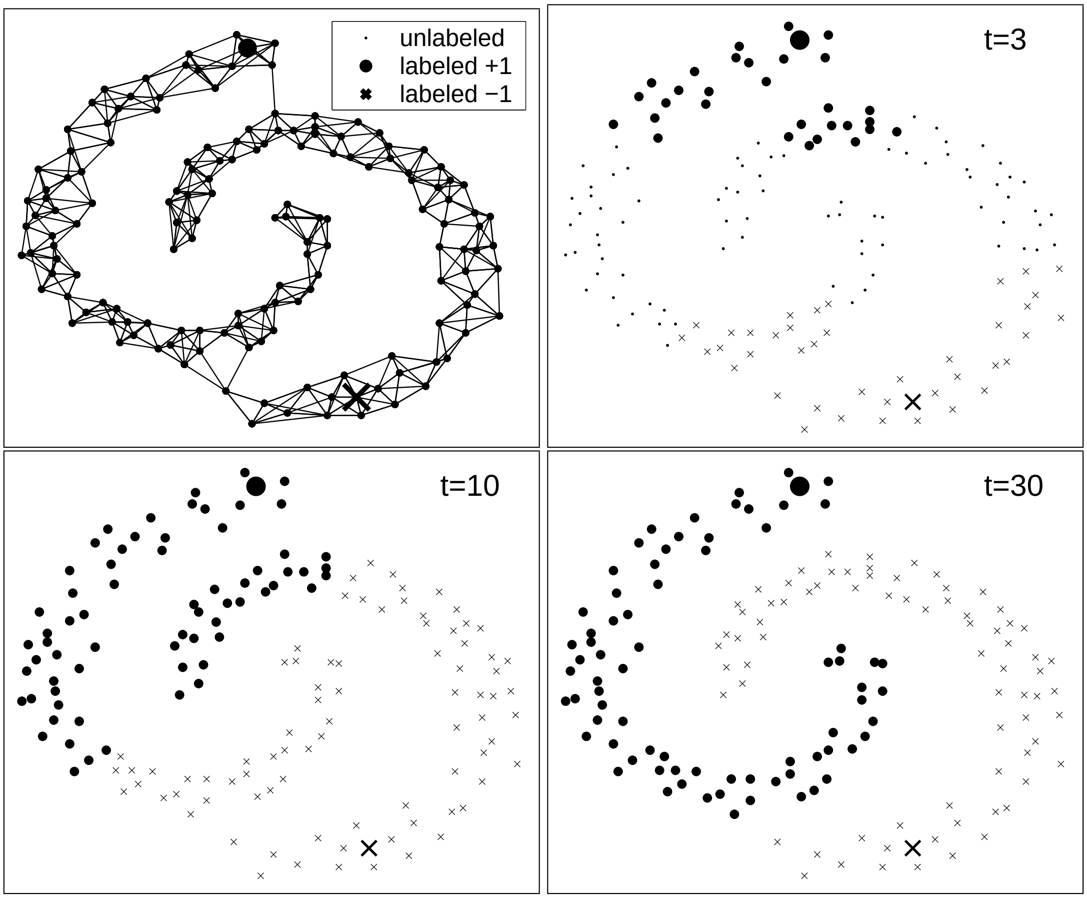


Figure 3-5: Top left: local connectivity for K=5 neighbors. Below are classifications using Markov random walks for t=3, 10, and 30 (top to bottom, left to right), estimated with maximum likelihood. There are two labeled points (large cross, circle) and 148 unlabeled points, classified (small crosses, circles) or unclassified (small dots).

A Limited Generative Model

We present a generative model for the Markov random walk. This model is limited in that it will not generate or explain the coordinates $\boldsymbol{x}_k$ of the observed examples. It will only be concerned with the *identity* or "node" k of the observed example, whereas the coordinates come from some other process.

Assume there are N possible nodes. To generate an observation,

1. Pick a starting node i uniformly at random. Draw a label y from the label distribution $Q(y|i)$ associated with the node i. However, neither the label nor the identity i will be directly observed.

2. Add t rounds of "identity noise." At each round, the node i has some probability $P_{1|0}(k|i)$ of being confused with a neighboring node k, and will then assume its identity.

3. Output the identity k at the final round as the observation.

Given an observation k, we may want to determine the original identity i and the label y. The generative model prescribes a specific inference procedure. We should find the probability $P_{0|t}(i|k)$ of the starting point given the observation, and infer the label by averaging over the label distributions of the starting points. In other words, $P(y|k) = \sum_i Q(y|i)P_{0|t}(i|k)$, just as we had before in eq. 3.8.

Classification Model

We contrast two possible label generation models

$$\text{Backward walk} \qquad P(y|k) = \sum_{i \in L \cup U} Q(y|i)P_{0|t}(i|k) \tag{3.10}$$

$$\text{Forward walk} \qquad P(y|k) = \sum_{i \in L \cup U} Q(y|i)P_{t|0}(i|k) \tag{3.11}$$

The first model (eq. 3.10) is the original one presented in section 3.4.2, and corresponds to the generative model. It employs the representation $P_{0|t}(i|k)$, which corresponds to a backward random walk. In statistics, this model may be referred to as diagnostic: to classify a given point k, we infer what points i the walk may have come from. The starting points have labels or distributions $Q(y|i)$.

The second model (eq. 3.11) is based on a forward random walk, and may be thought of as predictive: we start from a point k, and calculate probabilities of ending points $P_{t|0}(i|k)$, which have associated labels or distributions $Q(y|i)$. Spectral techniques (e.g., [MS01, CWS02]) are closely related to the forward random walk model.

We can validly represent a point either in terms of a forward or backward walk. However, if we agree with the assumptions in the generative model, we should choose the backward walk. In the forward walk, the point being classified is not viewed as an observation from a generative model.

The choice will affect the weighting of the representation. The random walk is assumed to start from a uniform distribution, but the end distribution is not uniform. Points in dense regions will generally have a higher ending probability. Asymptotically as $t \to \infty$, the two representations become

$$\text{Backward walk } P_{0|\infty}(i|k) = 1/N \qquad \text{Forward walk } P_{\infty|0}(k|i) = \pi_i^\infty, \tag{3.12}$$

where the forward walk approaches π_i^∞, the stationary distribution, which can be calculated from $\pi_i^\infty = \sum_j W_{ij} / \sum_i \sum_j W_{ij}$. With appropriate choices of the parameters, the stationary probability at i is related to the probability mass around location $\boldsymbol{x}_i$ in space. In contrast, the starting point of the random walk asymptotically approaches the prior uniform distribution.

The backward walk model has a more intuitive behavior as t is increased. Points in clusters blur together, become indistinguishable, and are weighted equally. The forward walk representation is larger at high density locations i. The corresponding parameters $Q(y|i)$ can gain in relative influence. This behavior may seem reasonable, but unfortunately, we get a double weighting effect: since the data is sampled according to the density $P(\boldsymbol{x})$, there are already proportionally more points in high density regions. The forward walk model leads to excessive weights for these regions, whereas the backward walk model is affected only once by the density.

Diffusion Model

In the limit as the number of data points goes to infinity, the discrete random walk becomes a continuous diffusion process. Instead of considering the sum over discrete paths between two points, we consider the path integral [MEP95] between the points. The path integral cannot easily be computed directly [TW01]. Fortunately, it can be computed by solving a partial differential equation, the diffusion equation [Yea02]. This remains a topic for future work.

It is also possible to design positive semidefinite kernels representing diffusion processes, which are called diffusion kernels [KL02] or heat kernels [BN02]. However, the local transition matrix must be symmetric, which does not allow a probabilistic interpretation unless all transitions have equal probability. This may still be interesting in the limit.

Eigenvalues and Spectral Clustering

Spectral techniques are successful at unsupervised clustering [SM00, Wei99], and can be employed in kernels for semi-supervised learning [CWS02, KL02, CSTK02]. They are closely related to the Markov random walk. They begin from an equivalent local transition matrix $\boldsymbol{A}$ [MS01]. Clustering is based on the eigenvectors of this matrix. The eigenvectors of the forward random walk $P_{t|0}(i|k) = \boldsymbol{A}^t$ are again the same, and are invariant to the choice of t. The random walk weights the eigenvectors by powers of their corresponding eigenvalues λ^t. For high t, only the top eigenvectors will be significant, as in spectral techniques (various other eigenvector weighting schemes are discussed in [CWS02]). Thus, the forward random walk model $P_{t|0}(i|k)$ (section 3.4.3) is very similar.

The backward random walk differs. The representation $P_{0|t}(k|i) = \boldsymbol{A}^t \boldsymbol{Z}^{-1}$ is normalized by a diagonal matrix $\boldsymbol{Z}^{-1}$, that ensures that the columns of $\boldsymbol{A}^t$ sum to 1. This normalization does not preserve the right eigenvalues or eigenvectors.

To follow manifolds, the markov random walk restricts local transitions to neighborhoods by using a sparse transition matrix. The same idea can be used to extend spectral clustering techniques to better respect manifolds. Similarly, spectral clustering can be used with exponential weights $W_{ij} \propto \exp(-d(i,j)/\sigma)$ instead of the currently common RBF matrix $W_{ij} = \exp(-d(i,j)^2/\sigma^2)$.

3.4.4 Choices for d, K, σ, and t

The classifier is fairly robust to choices of the parameters $d(\cdot,\cdot)$, K, and σ. We provide heuristics for choosing the parameters as follows. The local similarity measure $d(\cdot,\cdot)$ is typically given (e.g., Euclidean distance). The number of local neighbors K should be on the order of the manifold dimensionality. K must be large enough to preserve local topology: neighbors in the original space should remain neighbors in the resulting graph, so that paths on the graph can reflect direct paths in space. We set K to be at least the estimated topological dimensionality [BS98, VD95]. We also ensure that K is large enough to create a singly connected graph, yielding an ergodic Markov process. Otherwise, labeled points in one connected component cannot affect other connected components. However, K must be sufficiently small to avoid introducing edges in the neighborhood graph that span outside the manifold. For future work, we note that the local dimensionality may vary across space [Fri95], so that K may not be constant. For example, part of the data may originate from a catch-all "background" class, of higher dimensionality than more specific classes. An alternative way of constructing the neighborhood graph is to connect points that are within a fixed distance ε of each other, independent of how many such neighbors there are. The number of edges from a point is then proportional to the density at its location, which facilitates theoretical analysis. However, in practice it is easier to choose the number of neighbors K than to find a good ε.

The local scale parameter σ trades off the emphasis on shortest paths (low σ effectively ignores distant points), versus volume of paths (high σ, which effectively ignores distances).

The smoothness of the random walk representation depends on t, the number of transitions. This is a regularization parameter akin to the kernel width of a density estimator. In the limiting case t=1, we employ only the local neighborhood graph. As a special case, we obtain the kernel expansion representation [SJ01] by t=1, K=N, and $d(\cdot,\cdot)$ set to squared Euclidean distance. If all points are labeled, we obtain the K-nearest neighbors classifier by t=1, $\sigma \to \infty$. In the limiting case t=∞ the representation for each node becomes a flat distribution over the points in the same connected component: $P_{0|\infty}(i|k) = 1/N_{C_k}$ if i and k are connected in a component with N_{C_k} nodes, or 0 probability otherwise.

We can choose t based on a few unsupervised heuristics. If $t+1$ equals the diameter of a singly connected graph, then we ensure that $P_{0|t}(i|k) > 0$ so each point influences every other point. However, this scheme ignores transition probabilities. Instead, the *mixing time* of a graph measures the time it takes to approach the stationary distribution. The graph mixes faster the smaller the second largest eigenvalue λ_2 of the transition matrix $\boldsymbol{A}$ (the largest eigenvalue is always 1). To reach within ϵ in half L_1 distance from the stationary distribution, we must have [Chu97]

$$t \geq \max_i \frac{1}{1-\lambda_2}(\ln \frac{1}{\pi_i^\infty} + \ln \frac{1}{\epsilon}), \tag{3.13}$$

where π_i^∞ is the stationary probability at node i. Similarly to [TS01] we wish to choose t so that we are relatively far from the stationary distribution. An alternative unsupervised measure considers the rate of mutual information dissipation [TS01] to identify cluster development.

However, appropriate t depends on the classification task. For example, if classes change quickly over small distances, we want a sharper representation given by smaller t. Cross-validation could provide a supervised choice of t but requires too many labeled points for good accuracy. Instead, we propose to choose t that maximizes a measure of classifier confidence, namely an average margin per class on both labeled and unlabeled data. The average margin at point k is defined as $\gamma_k = P(y = \mathrm{class}(k)|k)$ (see also section 4.2.3). For labeled points, $\mathrm{class}(k) = \tilde{y}_k$, and for unlabeled points, $\mathrm{class}(k)$ is the class assigned by the classifier. Let N_c be the number of points in class c. Plot $(1/N_c) \cdot \sum_{k:\mathrm{class}(k)=c} \gamma_k$ for each class, separately for labeled and unlabeled points to avoid issues of their relative weights. See p. 53 for an illustration.

3.4.5 Individually Adaptive Time Scales

So far, we have employed a single global value of t. However, the desired smoothness may be different at different locations (akin to adaptive kernel widths in kernel density estimation). At the simplest, if the graph has multiple connected components, we can set individual t for each component. Ideally, each point has its own time scale, and the choice of time scale is optimized jointly with the classifier parameters. Here we propose a restricted version of this criterion where we find individual time scales t_k for each unlabeled point but estimate a single timescale for labeled points as before.

We attempt to find timescales such that each point is individually labeled with high confidence, but that the labels collectively have as high variability as possible. This principle ensures that the labels are confident while being balanced across classes. More precisely, define $\breve{P}(y|k)$ for any unlabeled point k as

$$\breve{P}(y|k) = \frac{1}{Z_k} \sum_{i:\tilde{y}_i = y} P_{0|t_k}(i|k), \tag{3.14}$$

where $Z_k = \sum_i P_{0|t_k}(i|k)$ and both summations are only over the labeled points. Moreover, let $P(y)$ be the overall probability over the labels across the unlabeled points or

$$P(y) = \sum\nolimits_k P(k)\breve{P}(y|k), \tag{3.15}$$

where $P(k)$ is uniform over the unlabeled points, corresponding to the start distribution. Note that $P(y)$ remains a function of all the individual time scales for the unlabeled points. With these definitions, the principle for setting the time scales reduces to maximizing the mutual information between the label and the point identity:

$$\{t_1, \ldots, t_m\} = \arg\max_{t_1,\ldots,t_m} I(y;k) = \arg\max_{t_1,\ldots,t_m} \{H(y) - \sum\nolimits_j P(k=j)H(y|k=j)\}. \tag{3.16}$$

$H(y)$ and $H(y|k)$ are the marginal and conditional entropies over the labels and are computed on the basis of $P(y)$ and $\breve{P}(y|k)$, respectively. Note that the ideal setting of the time scales would be one that determines the labels for the unlabeled points uniquely on the basis of only the labeled examples while at the same time preserving the overall variability of the labels across the points. This would happen, for example, if the labeled examples fall on distinct connected components. The criterion favors small time scales, because most labeled points reached by short random walks have the same label, and therefore provide confident labels. However, when nearby labels disagree, larger timescales are chosen.

We optimize the criterion by an axis parallel search, trying only discrete values of t_k large enough that at least one labeled point is reached from each unlabeled point. We initialize t_k to the smallest number of transitions needed to reach a labeled point. Empirically we have found that this initialization is close to the refined solution given by the objective. The objective is not concave, but separate random initializations generally yield the same answer, and convergence is rapid requiring about 5 iterations.

3.5 Generalization Guarantees Based on Smooth Representations

Both the kernel expansion and the Markov random walk representations have adjustable smoothness parameters. For the kernel expansion, it is the kernel width parameter σ, and the Markov random walk in addition has a timescale parameter t. When the parameters are set so that the representations are sufficiently smooth, we obtain generalization guarantees. We present two types of guarantees: (1) smooth representations can limit the impact of label noise in the observed labels, and (2) smoothness also restricts the capacity of our classifiers, and lets us derive a sample complexity result. The results are general and apply to any linear classifier with bounded weights that uses smooth representations of our form.

3.5.1 Robustness against Label Noise

We will assume that all examples are labeled, but that the observed labels $\{\tilde{y}_i\}$ have been sampled from the ground truth distribution $P^*(y|i)$, and therefore contain noise. We would like to guarantee with high probability that label noise in the observed labels will not substantially degrade our estimates of conditional probability. Thus, we are likely to still make correct classifications.

We need to assume sufficient smoothness, expressed as a small L_2 norm of the representation vectors $[P(i|\boldsymbol{x}_k)]_i$ for each point $\boldsymbol{x}_k$. The following result is obtained via McDiarmid's inequality:

Lemma 1 *Let $I_N = \{1, \ldots, N\}$. Given any $\delta > 0, \epsilon > 0$, and any collection of distributions $P(i|\boldsymbol{x}_k) \geq 0$, $\sum_{i \in I_N} P(i|\boldsymbol{x}_k) = 1$ for $k \in I_N$, such that $\|P(\cdot|\boldsymbol{x}_k)\|_2 \leq \epsilon/\sqrt{2\log(2N/\delta)}$, $\forall k \in I_N$, and independent samples $\tilde{y}_i$ drawn from some ground truth distribution $P^*(y|i), y \in \{-1, 1\}$, $i \in I_N$, then $P(\exists k \in I_N : |\sum_{i=1}^N \tilde{y}_i P(i|\boldsymbol{x}_k) - \sum_{i=1}^N w_i^* P(i|\boldsymbol{x}_k)| > \epsilon) \leq \delta$ where $w_i^* = P^*(y = 1|i) - P^*(y = -1|i)$ and the probability is taken over the independent samples.*

If the true distribution $P^*(y|i)$ were known, our classifier would make decisions by thresholding $\sum_i w_i^* P(i|\boldsymbol{x}_k)$, but since we only observe noisy samples, we must base decisions on $\sum_i \tilde{y}_i P(i|\boldsymbol{x}_k)$. Fortunately, the lemma proves that these quantities are close.

Both the kernel expansion and the Markov random walk representation can achieve the conditions for $P(i|\boldsymbol{x}_k)$ in the lemma. For example, consider a perfectly smooth representation $P(i|\boldsymbol{x}_k) = 1/N$ for all i. It has norm $\|P(\cdot|\boldsymbol{x}_k)\|_2 = 1/\sqrt{N}$, implying that the norm becomes arbitrarily small for large N.

Even though the lemma concerns the fully labeled case, it is still useful to the partially labeled case, by giving us a *minimum* smoothness required had the data been labeled; with less label information we will want additional smoothness.

3.5.2 Sample Size Complexity

We assume that all test examples are available at training time, so that we can consider their representations. We again rely on smoothness, this time expressed as a small L_1 norm between pairs of representation vectors $[P(i|\boldsymbol{x}_j)]_i$ and $[P(i|\boldsymbol{x}_k)]_i$ for points $\boldsymbol{x}_j$ and $\boldsymbol{x}_k$. Given these L_1 norms, we show how to calculate the V_γ-dimension [ABDCBH97] of the classifier. The V_γ-dimension is a measure of the capacity of the classifier, and is related to the logarithm of the maximum number of points that can be classified in all ways with classification margin γ. A small capacity guarantees generalization from few examples. For simplicity, we consider a binary problem with classes 1 and -1, where classification decisions are based on the sign of $f(\boldsymbol{x}_k) = \sum_{i\in L\cup U} \big(Q(y=1|i) - Q(y=-1|i)\big)P(i|\boldsymbol{x}_k)$.

Lemma 2 *Denote the L_1-norm distance between the representations of two points by $d_{jk} = \sum_{i\in L\cup U}|P(i|\boldsymbol{x}_j) - P(i|\boldsymbol{x}_k)|$. The V_γ-dimension of the binary transductive classifier $f(\boldsymbol{x}_k)$ is upper bounded by the number of connected components of a graph with N nodes and adjacency matrix $\boldsymbol{E}$, where $\boldsymbol{E}_{jk} = 1$ if $d_{jk} \leq \gamma$ and zero otherwise.*

Proof: To evaluate V_γ, we count the number of possible labelings consistent with the margin constraints $y_k f(\boldsymbol{x}_k) \geq \gamma$ for all k (labeled and unlabeled points). First, we establish that all examples j and k for which $d_{jk} \leq \gamma$ must have the same label. This follows directly from

$$|f(\boldsymbol{x}_j) - f(\boldsymbol{x}_k)| \leq \sum_{i\in L\cup U} \underbrace{|Q(y=1|i) - Q(y=-1|i))|}_{\leq 1} \cdot |P(i|\boldsymbol{x}_j) - P(i|\boldsymbol{x}_k)| \tag{3.17}$$

$$\leq \sum_{i\in L\cup U} |P(i|\boldsymbol{x}_j) - P(i|\boldsymbol{x}_k)| = d_{jk}, \tag{3.18}$$

as this difference must be larger than γ for the discriminant functions to have different signs. Since any pair of examples for which $d_{jk} \leq \gamma$ share the same label, different labels can be assigned only to examples not connected by the $d_{jk} \leq \gamma$ relation, i.e., examples in distinct connected components.■

This lemma applies more generally to any transductive classifier based on a weighted representation of examples as long as the weights are bounded.

To determine the sample size needed for a given dataset, and a desired classification margin γ, calculate the V_γ-dimension and call it r. With high probability we can correctly classify the unlabeled points given $O(r \log r)$ labeled examples [BC01]. This can also be helpful to determine timescale t since it is reflected in the V_γ, for example $V_\gamma = N$ for t=0 and $V_\gamma = 1$ for t=∞ for the full range of $\gamma \in [0, 2]$.

Chapter 4

Classification

A classifier takes a data example as input and outputs a probability that the example belongs to a particular class. In other words, the classifier is a function from the input $\boldsymbol{x}$ to the output $P(y|\boldsymbol{x})$. The probability can be viewed as a measure of class confidence. Most classifiers in this thesis use the simple functional form

$$P(y|\boldsymbol{x}) = \sum_i Q(y|i)P(i|\boldsymbol{x}). \tag{4.1}$$

This classifier applies a model to the example, to obtain a representation vector $[P(i|\boldsymbol{x})]$. The conditional probability $P(y|\boldsymbol{x})$ is a convex combination of the label distributions $Q(y|i)$, according to the representations that must sum to one.

The classifier weights the representation by a set of parameters $Q(y|i)$. It sums the result to obtain a conditional probability $P(y|\boldsymbol{x})$ for each class, and the example can be assigned to the most probable class. For binary tasks with classes $y = \pm 1$, the decision function becomes

$$\mathrm{sign} f(\boldsymbol{x}) = \mathrm{sign} \sum_i \big(Q(y=1|i) - Q(y=-1|i)\big)P(i|\boldsymbol{x}). \tag{4.2}$$

The previous chapter discussed different types of representations $P(i|\boldsymbol{x})$. The models were mainly concerned with modeling the structure of $\boldsymbol{x}$, and could operate entirely without labels y (however, the models could be improved for the purpose of classification by considering y, e.g., to determine the smoothness of the representation).

This chapter focuses on the classification task, specifically on "training" the classifier. This requires estimating the parameters $Q(y|i)$, and we elaborate on the estimation technique introduced in chapter 3. The parameters are adjusted to achieve good performance on a labeled training set. However, ultimately we are interested in accurately classifying unlabeled examples (either unlabeled examples available at training time, or new unseen examples). Thus, the parameter choices must not overfit the labeled training set, and instead must allow the classifier to generalize from the training. To prevent overfitting, we will restrict or regularize parameters to make the resulting classifier smooth.

We introduce multiple techniques for estimating the parameters $Q(y|i)$, which reflect different regularization criteria. We will contrast conditional maximum likelihood estimation (section 4.1), which has no regularization of parameters, with margin-based regularizers (section 4.2), and maximum entropy regularizers (section 4.3).

These criteria are all discriminative to varying degrees. By a discriminative classifier we mean a classifier that focuses on classification decisions, and is driven by classification mistakes. The classifier pays less attention to the confidence of those decisions. Consequently, it concentrates the learning effort on the most difficult examples, which are most likely to become mistakes. For example, conditional maximum likelihood is only weakly discriminative, because it attempts to achieve high confidence (likelihood) for the labels of all examples. On the other hand, the margin-based and maximum entropy techniques are more discriminative, and focus on achieving a given level of confidence for all labeled points, but do not care if some points achieve a confidence beyond that level.

Importantly, we ask whether unlabeled data may be helpful for classifier training. We always utilize unlabeled data to build the model $P(i|\boldsymbol{x})$ as in the last chapter. However, for classifier training, our initial algorithms use only labeled points. Specifically, they estimate parameters that minimize a performance objective (e.g., minimum loss) that involves labeled points exclusively. These training techniques are therefore no different from those used in ordinary supervised learning, although they are designed for models that give probability vectors $[P(i|\boldsymbol{x})]$, and use weights $Q(y|i)$ that must be valid probabilities.

Later in this chapter (section 4.4) we discuss *transductive loss* techniques that optimize loss on both labeled and unlabeled points. Since the labels are unknown, we must hypothesize 'pseudo-labels' in order to measure such loss. This step typically requires search over many labelings of the unlabeled points. We will search directly over discrete labelings (e.g., $y_k = \pm 1$ for all $k \in U$) using combinatorial optimization.

The computational cost of training is an important consideration. The transductive loss formulations must search a space of exponential size in the number of unlabeled points. The search can be computationally intractable, even for problems with less than 100 unlabeled points. Approximate solutions are possible, but it is unknown how good the approximations are when compared to the true solution. Fortunately, most of our non-transductive techniques (including maximum likelihood, margin-based and maximum entropy training) lead to easy convex optimization problems. We introduce an average margin criterion that even permits a closed-form solution.

4.1 Conditional Maximum Likelihood Estimation with EM

We now estimate the parameters $Q(y|i)$ in the conditional model $P(y|\boldsymbol{x}) = \sum_i Q(y|i)P(i|\boldsymbol{x})$. The maximum likelihood approach chooses parameter values that maximize the probability of the observed data. For classification, we are only interested in conditional likelihoods. We maximize the conditional likelihoods of the observed labels $\{P(\tilde{y}_k|\boldsymbol{x}_k)\}_{k\in L}$[1]

$$\max_{\{Q(y|i)\}} \sum_{k\in L} \log P(\tilde{y}_k|\boldsymbol{x}_k) = \max_{\{Q(y|i)\}} \sum_{k\in L} \log \sum_{i\in L\cup U} Q(\tilde{y}_k|i)P(i|\boldsymbol{x}_k), \tag{4.4}$$

subject to the constraints that $Q(y|i)$ are valid probabilities, namely $0 \leq Q(y|i) \leq 1$ for all i, and $\sum_y Q(y|i) = 1$. Here $P(i|\boldsymbol{x}_k)$ are fixed and known representations, such as the kernel expansion or Markov random walk representations from chapter 3. The summation over k

[1] In binary classification the objective can be written using only the parameters $Q(y = 1|i)$ for all i, since $Q(y = -1|i) = 1 - Q(y = 1|i)$, thus

$$\max_{\{Q(y=1|i)\}} \Big\{ \sum_{k:\tilde{y}_k=1} \log \sum_{i\in L\cup U} Q(\tilde{y}_k = 1|i)P(i|\boldsymbol{x}_k) + \sum_{k:\tilde{y}_k=-1} \log \sum_{i\in L\cup U} (1 - Q(\tilde{y}_k = 1|i))P(i|\boldsymbol{x}_k) \Big\}. \tag{4.3}$$

does not include unlabeled points, because they have no observed labels that could affect the conditional likelihood. This optimization problem is much easier than maximum likelihood estimation of Gaussian mixtures, where both the mixture weights and the mixture components must be fit [MU97]. Unlike for Gaussian mixtures, our objective is concave, because the log of a linear function is concave, and the sum of concave functions remains concave. Thus, there are no local maxima, only global maxima.

The optimization is easily performed via the expectation-maximization (EM) algorithm [DLR77, Min]. EM is well-suited to maximum likelihood problems with missing or hidden data. If we knew that component i was responsible for generating a particular subset of labeled points $\{(\boldsymbol{x}_k, \tilde{y}_k)\}$, then estimating the parameter $Q(y|i)$ would be easy—we would set $Q(y|i)$ to best explain the labeling of those points. The problem is that we do not know what points are associated with component i. The EM algorithm applies here. It introduces hidden assignment distributions $Q(i|\boldsymbol{x}_k, \tilde{y}_k)$, which represent the probability that the i-th component was associated with the point $(\boldsymbol{x}_k, y_k)$. It starts from some initial values of the parameter estimates $Q(y|i)$, e.g., a uniform distribution $Q(y|i) = 1/N_C$, where N_C is the number of classes. EM then iteratively alternates between

E-step: compute the probabilities of the hidden assignments given the current parameter estimates,

$$Q(i|\boldsymbol{x}_k, \tilde{y}_k) \propto Q(\tilde{y}_k|i)P(i|\boldsymbol{x}_k) \quad \forall i \in L \cup U,\ \forall k \in L, \tag{4.5}$$

normalized so that $Q(i|\cdot)$ is a valid probability.

M-step: estimate parameters to maximize the likelihood given the current hidden assignments,

$$Q(y|i) \leftarrow \frac{\sum_{k \in L : \tilde{y}_k = y} Q(i|\boldsymbol{x}_k, \tilde{y}_k)}{\sum_{k \in L} Q(i|\boldsymbol{x}_k, \tilde{y}_k)} \quad \forall i \in L \cup U,\ \forall y. \tag{4.6}$$

Stop iterating when the parameters and the conditional log-likelihood change less than some thresholds between iteration $(t-1)$ and (t), i.e., when $|Q(y|i)^{(t)} - Q(y|i)^{(t-1)}| \leq \epsilon_1$ and $|L^{(t)} - L^{(t-1)}| \leq \epsilon_2$, where $L^{(t)} = \sum_{k \in L} \log \sum_{i \in L \cup U} Q(\tilde{y}_k|i)^{(t)} P(i|\boldsymbol{x}_k)$.

The EM procedure guarantees that the likelihood increases at each iteration. Since our likelihood function is concave, EM will converge to a global maximum. The rate of convergence is linear [Wu83, XJ96], and the runtime of this algorithm is $\mathcal{O}(N_L\, N_{LU})$. Refined variants of EM for fitting component mixture weights of known components are given in [PL96].

In the absence of problem-specific knowledge, we will assume uniform priors on the class frequencies. Thus, a priori we believe that a test set sampled from this distribution has roughly equal numbers of labeled points in each class. We also assume that the cost of misclassification is the same for all classes. When the observed labeled points in a given training set have unbalanced class frequencies, we must adjust the likelihood objective to reflect the prior assumptions. Otherwise, the likelihood will be dominated by the classes with more labeled examples, and the classifier may misclassify examples from classes with fewer training examples. We balance the objective by requiring the same average log-likelihood per class

$$\max_{\{Q(y|i)\}} \sum_{y \in C} \frac{1}{N_{L_y}} \sum_{k \in L_y} \log P(\tilde{y}_k|\boldsymbol{x}_k) \tag{4.7}$$

The EM updates to this balanced objective are almost the same as before. After computing the M-step as before, we simply rescale $Q(y|i) \leftarrow Q(y|i)/N_{L_y}$ and then renormalize so that

$\sum_y Q(y|i) = 1$.

The conditional maximum likelihood will provide reasonable parameter estimates $Q(y|i)$ for classification purposes. However, the parameters are not regularized by the maximum likelihood objective, and frequently converge to the extreme values 0 or 1. The quality of the solution, as well as the potential for overfitting, depends on the smoothness of the representation $P(i|\boldsymbol{x})$. For example, in the kernel expansion representation the kernels must be sufficiently wide (section 3.5).

4.2 Margin Maximization

More discriminative formulations are also possible, which are more sensitive to actual classification decisions rather than to the probability values associated with the labels of all labeled training points. A pure discriminative learner would consider only the actual classification decisions $y = \mathrm{argmax}_y\, P(y|\boldsymbol{x})$, and ignore the magnitude of $P(y|\boldsymbol{x})$.

Unfortunately, pure discriminative learning is generally not sufficient for uniquely determining parameter values $Q(y|i)$. Multiple settings of the parameters $Q(y|i)$ yield the same labeling of the labeled training data (especially for small labeled sets), and therefore the same training error. However, they may have different test error. Thus, we need criteria other than training error, which may indicate which parameter choices give low test error. Such criteria can be based on weak generative assumptions, e.g., the notion of a margin between classes, or penalties for extreme parameter values $Q(y|i)$. We will derive such criteria, but start more abstractly from minimization of a loss function. We rederive conditional maximum likelihood, and then continue with margin-based loss functions.

4.2.1 Maximum Conditional Likelihood from a Loss Function Perspective

Previously in section 4.1, we considered maximization of the conditional likelihood, which is an instance of *conditional learning* of the probabilities $P(y|\boldsymbol{x})$. Given a dataset $\{(\boldsymbol{x}_1, \tilde{y}_1),\ldots, (\boldsymbol{x}_{N_L}, \tilde{y}_{N_L}), \boldsymbol{x}_{N_L+1},\ldots, \boldsymbol{x}_{N_{LU}}\}$, we want to train the classifier so that its conditionals $P(y|\boldsymbol{x}_k)$ are close to the observed values $\tilde{y}_k$ at the labeled points. To do so, choose the classifier parameters $\{Q(y|i)\}$ to minimize the loss between the observed labels and the classifier conditionals.

$$\min_{\{Q(y|i)\}} \mathcal{L}_{\mathrm{ML}}\big([\tilde{y}_1, \ldots, \tilde{y}_{N_L}]\,;\,[P(y|\boldsymbol{x}_1), \ldots P(y|\boldsymbol{x}_{N_L})]\big). \tag{4.8}$$

We have denoted the loss function for maximum likelihood by $\mathcal{L}_{\mathrm{ML}}(\cdot;\cdot)$, where the first argument is the true value, and the second is the model output. Both arguments are vectors, since general loss functions may measure loss jointly over multiple examples. To estimate conditional probabilities, the standard loss function is the KL-divergence from true to estimated conditionals [JB02]. The KL divergence between two distributions $P_1(\boldsymbol{y})$ and $P_2(\boldsymbol{y})$ is defined $D(P_1(\boldsymbol{y}) \parallel P_2(\boldsymbol{y})) = \sum_{\boldsymbol{y}} P_1(\boldsymbol{y}) \log P_1(\boldsymbol{y}) - \sum_{\boldsymbol{y}} P_1(\boldsymbol{y}) \log P_2(\boldsymbol{y})$. In our case, distribution $P_1(\boldsymbol{y})$ is the empirically observed conditional distribution, which equals 1 for the observed configuration of labels, and 0 otherwise. Only the second term of the KL divergence (called cross entropy) contains any parameters that will be optimized. Therefore, when we minimize KL divergence, only the second term is needed. We obtain

$$\min_{\{Q(y|i)\}} \mathcal{L}_{\mathrm{ML}}\big([\tilde{y}_1, \ldots, \tilde{y}_{N_L}]\,;\,[P(y|\boldsymbol{x}_1), \ldots P(y|\boldsymbol{x}_{N_L})]\big) = \tag{4.9}$$

$$\max_{\{Q(y|i)\}} \log P(\tilde{y}_1, \ldots, \tilde{y}_{N_L} \,|\, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_L}) = \tag{4.10}$$

$$\max_{\{Q(y|i)\}} \sum_{k \in L} \log P(\tilde{y}_k | \boldsymbol{x}_k) = \min_{\{Q(y|i)\}} \sum_{k \in L} \mathcal{L}_{\text{KL}}(\tilde{y}_k \,;\, P(y_k|\boldsymbol{x}_k)), \tag{4.11}$$

which is exactly the conditional log likelihood criterion. The loss $\mathcal{L}_{\text{KL}}$ is the KL divergence, and the summation arises from the factoring $P([\tilde{y}_1,\ldots, \tilde{y}_{N_L}] \,|\, [\boldsymbol{x}_1,\ldots, \boldsymbol{x}_{N_L}]) = \prod_{k \in L} P(\tilde{y}_k|\boldsymbol{x}_k)$.

The above loss depends on all labeled training points. If a single point fails to be explained by the model and has zero likelihood, the overall loss will be infinity regardless of the likelihoods of the other points. Due to the logarithmic scale, the loss is sensitive to low probability points, such as outliers.

4.2.2 Minimum margin

The loss in the maximum likelihood approach is dominated by labeled points with low confidence. We can extrapolate this property in a more discriminative classifier, whose overall loss depends solely on the worst loss incurred at a labeled point. Let

$$\min_{\{Q(y|i)\}} \mathcal{L}_{\text{MM}}\big([\tilde{y}_1, \ldots, \tilde{y}_{N_L}] \,;\, [P(y|\boldsymbol{x}_1), \ldots P(y|\boldsymbol{x}_{N_L})]\big) = \tag{4.12}$$

$$\min_{\{Q(y|i)\}} \max_{k \in L} \mathcal{L}_{\text{KL}}(\tilde{y}_k \,;\, P(y_k|\boldsymbol{x}_k)) = \max_{\{Q(y|i)\}} \min_{k \in L} \log P(\tilde{y}_k|\boldsymbol{x}_k) \overset{\max}{=} \max_{\{Q(y|i)\}} \min_{k \in L} P(\tilde{y}_k|\boldsymbol{x}_k). \tag{4.13}$$

In other words, we consider only the likelihood of the lowest confidence point or points. (In the last step, we have written $\overset{\max}{=}$ to mean equality of the parameters $Q(y|i)$ achieving the maximum; their solution is the same without the log in the objective.) We define the margin at a labeled point to be the conditional likelihood $P(\tilde{y}_k|\boldsymbol{x}_k)$, and call the above loss the minimum margin criterion. When the margin is large, the classifier is very confident of its decision. The same definition of margin is also used by boosting [SS99], but support vector machines use a different margin of Euclidean origin.

To maximize the minimum margin over all labeled points, we solve the linear program

$$\max_{\{Q(y|i)\},\, \gamma} \quad \gamma \tag{4.14a}$$

$$\text{s.t.} \quad P(\tilde{y}_k|\boldsymbol{x}_k) \geq \gamma \qquad \forall k \in L \tag{4.14b}$$

$$0 \leq Q(y|i) \leq 1, \quad \textstyle\sum_y Q(y|i) = 1 \quad \forall i \in L \cup U,\ \forall y. \tag{4.14c}$$

The program spends effort only on labeled training examples whose classification is uncertain. Examples already classified correctly with a margin larger than γ are effectively ignored. In section 3.5.2, (p. 40) we showed that large margins restrict the capacity of a classifier. This training procedure prefers large margins at the labeled points, and we experimentally see good generalization to accurate classification of the unlabeled points.

The linear program can be solved very efficiently (even the basic simplex algorithm is solves thousand point problems in minutes). In contrast, support vector machines require solving quadratic programs, which may be more computationally expensive.

The minimum margin criterion is good when the training set does not contain any mislabeled points. Unfortunately, the criterion is sensitive to noisy labels, as it focuses all effort on maximizing the margin at the most difficult labeled point. For larger labeled sets with noise, the linear program may have an empty feasible region, and no solution. We therefore recommend this approach only for low-noise data with few labels.

4.2.3 Average margin

In the previous sections, we have implicitly assumed that the dataset constitutes one N_{LU}-point sample from some dataset distribution. We have defined the loss as a joint function of all labeled points in the dataset.

Here, we take a different view. We will assume that each point is an individual sample from some dataset distribution. Then, the loss is defined per point, and the expected loss is the average of the losses of the individual samples, as follows:

$$\min_{\{Q(y|i)\}} \mathcal{L}_{\mathrm{AM}}\big([\tilde{y}_1, \ldots, \tilde{y}_{N_L}]\,;\,[P(y|\boldsymbol{x}_1), \ldots P(y|\boldsymbol{x}_{N_L})]\big) \tag{4.15}$$

$$= \min_{\{Q(y|i)\}} \frac{1}{N_L} \sum_{k \in L} \mathcal{L}_{\mathrm{AM1}}(\tilde{y}_k\,;\,P(y_k|\boldsymbol{x}_k)) = \frac{1}{N_L} \max_{\{Q(y|i)\}} \sum_{k \in L} P(\tilde{y}_k|\boldsymbol{x}_k) \tag{4.16}$$

We have chosen the loss at a single point ($\mathcal{L}_{\mathrm{AM1}}$) to be the negative conditional likelihood $P(\tilde{y}_k|\boldsymbol{x}_k)$ of its label. The objective averages the conditional likelihoods over points. We again refer to these likelihoods as margins, hence the "average margin" name. Compare this objective to that of maximum likelihood (eq. 4.11), which is the sum of *log* likelihoods. If all points have high likelihoods (i.e., $P(\tilde{y}_k|\boldsymbol{x}_k)$ is close to 1 for all k), then the two objectives behave similarly, because $\log(x) \approx x - 1$, hence $\mathrm{argmax}_{\{Q(y|i)\}} \sum_k \log(P(\tilde{y}_k|\boldsymbol{x}_k)) \approx \mathrm{argmax}_{\{Q(y|i)\}} \sum_k P(\tilde{y}_k|\boldsymbol{x}_k)$. However, the average margin criterion is much more robust to outliers with low likelihoods, because their losses will not be magnified by the log as in the maximum likelihood approach. The average margin will effectively ignore points with almost zero likelihood. One possible concern is that the average margin could instead be dominated by a high likelihood point. However, the margins are bounded between 0 and 1, limiting the influence of any single point on the average.

To maximize the average margin, we formulate the linear program

$$\max_{\{Q(y|i)\}, \{\gamma_k\}} \quad \frac{1}{N_L} \textstyle\sum_{k \in L} \gamma_k \tag{4.17a}$$

$$\text{s.t.} \quad P(\tilde{y}_k|\boldsymbol{x}_k) \geq \gamma_k \qquad \forall k \in L \tag{4.17b}$$

$$0 \leq Q(y|i) \leq 1, \quad \textstyle\sum_y Q(y|i) = 1 \quad \forall i \in L \cup U,\ \forall y. \tag{4.17c}$$

The solution to this program has the closed form

$$Q(y|i) = \begin{cases} 1 & \text{if } y = \mathrm{argmax}_y \sum_{k \in L: \tilde{y}_m = y} P(i|\boldsymbol{x}_m), \\ 0 & \text{otherwise,} \end{cases} \tag{4.18}$$

and the classifier confidence can be written as

$$P(y|\boldsymbol{x}_k) = \sum_{i \in L \cup U,\ i:\, Q(y|i) = 1} P(i|\boldsymbol{x}_k). \tag{4.19}$$

This solution has a nice interpretation. View $P(i|\boldsymbol{x}_m)$ as a similarity between point i and

m. The parameter $Q(y|i)$ is integer-valued $\{0, 1\}$, and chosen to be 1 for the class of labeled points that are most similar to point i. Thus, $Q(y|i)$ can be seen as a weighted neighbor classification of i, based on labeled points only. The final output is a weighted neighbor classification of k, including both labeled and unlabeled points, with the classes of these points set to $Q(y|i)$ as previously from their labeled neighbors. For binary classification, we can write

$$f(\boldsymbol{x}_k) = \sum_{i \in L \cup U} \left(\operatorname{sign} \sum_{m \in L} \tilde{y}_m P(i|\boldsymbol{x}_m) \right) P(i|\boldsymbol{x}_k). \tag{4.20}$$

The closed form solution makes training very fast, and enables efficient cross-validation for setting various parameters, such as the smoothness of the representations.

The average margin solution yields hard parameters at 0 or 1, just as the minimum margin did. Again, the risk of overfitting is mitigated by a smooth representation $P(i|\boldsymbol{x})$, and the large average margin experimentally shows better generalization performance than the maximum likelihood criterion.

A similar average margin objective has been proposed in [GHS$^+$99], although for weights regularized with a non-probabilistic L1-norm. Their formulation does not have a closed form solution, and requires solving a linear program.

4.2.4 Unbalanced and Multiple Classes

We assume uniform class priors and equal misclassification costs for all classes. When the number of labeled points differs across classes in the training set, we should adjust the average margin objective. In other words, we adjust the average margin objective (eq. 4.17a) to

$$\max \sum_{y \in C} \frac{1}{N_{L_y}} \sum_{k \in L_y} \gamma_k, \tag{4.21}$$

where C is the set of classes, and N_{L_y} is the number of labeled points in class y. The solution then changes to

$$Q(y|i) = \begin{cases} 1 & \text{if } y = \text{argmax}_y \, \frac{1}{N_{L_y}} \sum_{k \in L : \tilde{y}_m = y} P(i|\boldsymbol{x}_m), \\ 0 & \text{otherwise.} \end{cases} \tag{4.22}$$

The minimum margin criterion is not as sensitive to unbalanced classes as the average margin criterion, and performs well unadjusted, as described earlier.

Both the minimum margin and average margin formulations can directly handle classification with multiple classes as presented. One could also use them as the binary classifiers in a one-versus-all combination scheme, or in an error-correcting code scheme [DB95, ASS00].

4.3 Maximum Entropy Discrimination

In the previous section, we chose parameters that maximized a margin at the labeled points. Large margins serve to reduce classifier capacity. However, there may be several sets of parameters that achieve the same margin, and there is no criterion to choose among them. Here we introduce a regularizer that prefers uninformative parameter values (in an information-theoretic sense), while still achieving large margins.

We employ the maximum entropy discrimination (MED) framework [JMJ00]. Consider a binary classifier with parameters $\boldsymbol{y} = [y_i]_{i \in L \cup U}$, where $y_i = \pm 1$ for all i, corresponding to each component i of the model. These parameters should be distinguished from observed labels denoted by $\tilde{y}_k$. Instead of estimating the parameters directly, we estimate a distribution $Q(\boldsymbol{y})$ over them. The classifier will average over this parameter distribution, reminiscent of a Bayesian approach:

$$f(\boldsymbol{x}_k) = \sum_{\boldsymbol{y} \in \{\pm 1, \ldots, \pm 1\}} \sum_{i \in L \cup U} Q(\boldsymbol{y}) y_i P(i|\boldsymbol{x}_k) = \sum_{i \in L \cup U} \underbrace{\left(\sum_{y_i \in \pm 1} Q_i(y_i) y_i \right)}_{w_i} P(i|\boldsymbol{x}_k), \tag{4.23}$$

where only the marginals $Q_i(y_i)$ of the joint distribution are needed. Moreover, the effective weights of this classifier are $w_i = \sum_{y_i = \pm 1} Q_i(y_i) y_i = E_Q[y_i] \in [-1, 1]$. This is formally similar to our previous classifiers, which for binary classification had weights $Q(y = 1|i) - Q(y = -1|i) \in [-1, 1]$ (see eq. 4.2). Intuitively, we may think of the distribution $Q_i(y_i)$ as the parameter $Q(y|i)$.

The maximum entropy discrimination approach selects the distribution $Q(\boldsymbol{y})$ of maximum entropy, subject to classifying the labeled points correctly. It encodes the principle that parameter assignments should remain uninformative to the extent possible. Since the classifier only requires marginals $Q_i(y_i)$ the maximum entropy distribution will factor: $Q(\boldsymbol{y}) = \prod_i Q_i(y_i)$. To find the distribution, we solve the constrained nonlinear program

$$\begin{aligned}
\max_{\{Q_i(y_i)\}, \{\xi_k\}} \quad & \sum_{i \in L \cup U} H(y_i) - C \sum_{k \in L} \xi_k && && \text{(4.24a)} \\
\text{s.t.} \quad & \tilde{y}_k \Big[\sum_{i \in L \cup U} \sum_{y_i = \pm 1} y_i \, Q_i(y_i) P(i|\boldsymbol{x}_k) \Big] + \xi_k \geq \gamma && \forall k \in L && \text{(4.24b)} \\
& \xi_k \geq 0 && \forall k \in L && \text{(4.24c)} \\
& 0 \leq Q_i(y_i) \leq 1, \quad \textstyle\sum_{y_i} Q_i(y_i) = 1 && \forall i \in L \cup U. && \text{(4.24d)}
\end{aligned}$$

where $H(y_i)$ is the entropy of y_i relative to its marginal distribution, namely, $H(y_i) = -Q_i(y_i) \log Q_i(y_i) - (1 - Q_i(y_i)) \log(1 - Q_i(y_i)$. Here γ specifies a target margin ($\gamma \in [0, 1]$) and the slack variables ξ_k permit deviations from the target to ensure that a solution always exists. The C parameter is a slack penalty, and can be set via cross-validation (C=$40N_L$ works well for our problems). We typically set the target γ to equal the maximum achievable margin without requiring the use of any slack. This is the γ found by the minimum margin approach in eq. 4.14. However, for problems with many noisy labels, the maximum achievable margin may become very small, and it may be preferable set the target margin to a higher value, (such as γ = 0.55 for binary class problems).

In the context of the maximum entropy estimation, if a parameter is not helpful for achieving the classification constraints, then entropy is maximized for $Q_i(y_i = \pm 1) = 0.5$, implying a classifier weight $w_i = Q_i(y_i = 1) - Q_i(y_i = -1) = 0$, so that the component $P(i|\boldsymbol{x})$ corresponding to the parameter has no effect on the boundary.

4.3.1 The Maximum Entropy Solution

The maximum entropy optimization is concave because the entropy is a concave function, and it is easy to determine a globally optimal solution. The solution is found by introducing Lagrange multipliers $\boldsymbol{\lambda} = \{\lambda_k\}_{k \in L}$ for the classification constraints. and optimizing the dual. The solution is unique, and has a well-known form [CT91, p. 267]:

$$Q(\boldsymbol{y}) = \frac{1}{Z(\boldsymbol{\lambda})} \exp\left(-\sum_{k \in L} \lambda_k \gamma + \sum_{k \in L} \lambda_k \tilde{y}_k \sum_{i \in L \cup U} y_i P(i|\boldsymbol{x}_k) \right). \tag{4.25}$$

The multipliers satisfy $\lambda_k \in [0, C]$, where the lower bound comes from the classification constraints and the upper bound from the linear margin penalties being minimized. To find the optimal setting of λ_k, we must evaluate the partition function $Z(\boldsymbol{\lambda})$ that normalizes the maximum entropy distribution:

$$\begin{aligned} Z(\boldsymbol{\lambda}) &= \mathrm{e}^{-\sum_{k \in L} \lambda_k \gamma} \sum_{y_1, \ldots, y_N} \prod_{i \in L \cup U} \mathrm{e}^{\sum_{k \in L} \tilde{y}_k \lambda_k y_i P(i|\boldsymbol{x}_k)} & (4.26) \\ &= \mathrm{e}^{-\sum_{k \in L} \lambda_k \gamma} \prod_{i \in L \cup U} \left(\mathrm{e}^{\sum_{k \in L} \tilde{y}_k \lambda_k P(i|\boldsymbol{x}_k)} + \mathrm{e}^{-\sum_{k \in L} \tilde{y}_k \lambda_k P(i|\boldsymbol{x}_k)} \right). & (4.27) \end{aligned}$$

Minimizing the jointly convex log-partition function $\log Z(\boldsymbol{\lambda})$ with respect to the Lagrange multipliers leads to the optimal setting $\{\lambda_k^*\}$. This optimization is readily done via an axis parallel line search (e.g., the bisection method [PTVF93]). Finding the solution involves $\mathcal{O}({N_L}^2\, N)$ operations. The required gradients are given by

$$\frac{\partial \log Z(\boldsymbol{\lambda})}{\partial \lambda_k} = -\gamma + \sum_{i \in L \cup U} \underbrace{\tanh\left(\sum_{m \in L} \tilde{y}_m \lambda_m^* P(i|\boldsymbol{x}_m) \right)}_{w_i^*} \tilde{y}_k P(i|\boldsymbol{x}_k). \tag{4.28}$$

The gradient has the same form as the classification constraint (eq. 4.24b). By comparing these equations, we have identified the optimal weight w_i^*, which is an expectation with respect to the maximum entropy distribution Q^*, namely $w_i^* = E_{Q^*}[y_i]$. This weight is used by the resulting classifier. The components of the maximum entropy distribution are $Q_i^*(y_i) \propto \exp\left(\sum_{k \in L} \tilde{y}_k \lambda_k^* y_i P(i|\boldsymbol{x}_k) \right)$.

Often the numbers of positive and negative training labels are imbalanced. The MED formulation can be adjusted by defining the margin penalties as $C^+ \sum_{k:\tilde{y}_k = 1} \xi_k + C^- \sum_{k:\tilde{y}_k = -1} \xi_k$ (analogously to SVMs). We choose C^+ and C^- to satisfy ${N_L}^+ C^+ = {N_L}^- C^-$, to equalize the mean slack penalties per class. The coefficients C^+ and C^- can also be modified adaptively during the estimation process to balance the rate of misclassification errors across the two classes.

4.4 Margin Maximization with Transductive Loss

The previously presented parameter estimation techniques minimized loss only at the labeled examples. The loss was a function of the observed label and the classifier conditional. Transductive loss techniques minimize the loss over both labeled and unlabeled points, and typically use the same loss function for both sets.

The main challenge is that the unlabeled points have no observed labels to measure the loss of. Transductive techniques approach this issue by treating the labels of unlabeled points as parameters, which can then be optimized or integrated out.

Here we formulate a transductive version of the average margin criterion for binary classification. The average margin will be measured at both labeled and unlabeled points. We introduce binary parameters $\{y_k \in \pm 1\}_{k \in U}$ for the labels of the unlabeled points, and maximize the average margin objective with respect to these parameters.

$$
\begin{aligned}
\max_{\{y_k\}_{k\in U}, \{Q(y|i)\}, \{\gamma_k\}} \quad & \frac{1}{N_L^+} \sum\nolimits_{k \in L^+} \gamma_k + \frac{1}{N_L^-} \sum\nolimits_{k \in L^-} \gamma_k + \frac{1}{N_U} \sum\nolimits_{k \in U} \gamma_k & & \text{(4.29a)} \\
\text{s.t.} \quad & P(\tilde{y}_k|\boldsymbol{x}_k) \geq \gamma_k & \forall k \in L \quad & \text{(4.29b)} \\
& P(y_k|\boldsymbol{x}_k) \geq \gamma_k & \forall k \in U \quad & \text{(4.29c)} \\
& 0 \leq Q(y|i) \leq 1, \quad \sum\nolimits_y Q(y|i) = 1 & \forall i \in L \cup U,\ \forall y. \quad & \text{(4.29d)}
\end{aligned}
$$

We have optimized the average per class margin for the labeled points, to account for unbalanced classes. For unlabeled points, we have lumped the classes together, since the optimization would be more difficult otherwise.

This optimization problem is combinatorial due to the integer constraints on the parameters $y \in \pm 1$. It can be solved exactly using integer programming, but the size of the search space is exponential in the number of unlabeled examples. A brute force approach would try all possible label configurations. Alternatively, we can relax some of the integer constraints, replacing them by real-valued constraints $y \in [-1, 1]$, and solve the resulting linear program. In this way, we can rule out many integer settings of the parameters as suboptimal, thereby quickly reducing the search space. We have specifically used the branch-and-bound search technique and various search heuristics, as implemented by the CPLEX mixed integer programming code [ILO01]. Unfortunately, the search space is still exponential in the number of unlabeled examples, and we have only been able to find exact solutions to problems with less than 50 unlabeled examples.

It is likewise possible to solve transductive loss versions of the minimum margin and the maximum entropy discrimination formulations. Transductive MED is presented in [JMJ00, Jeb01]. It solves for distributions over the labels, which are real-valued, thereby avoiding the need for integer programming. The optimization problem is convex but must still be approximated to become computationally tractable.

4.5 Experimental Results on Real Data

4.5.1 Experimental Protocol

To determine the behavior of the algorithms on real data, we have run several experiments. For experimental purposes, we have started from fully labeled datasets and made them partially labeled simply by randomly removing some of the labels. This procedure guarantees that some assumptions we made for the partially labeled problem are valid: the labeled and unlabeled examples come from the same distribution, and the missingness mechanism is ignorable (section 1.2).

We study the learning accuracy as a function of the number of labeled and unlabeled examples, and we present two types of plots. The first type is a learning curve where we vary the number of labeled examples (N_L), but keep the total number of labeled plus unlabeled examples (N_{LU}) fixed. All available examples are used as unlabeled data. Learning accuracy is evaluated on a fixed test set, which is included as unlabeled data during training. To reduce the variance of the accuracy estimate, we average the accuracy across 20 randomly drawn sets of labeled and unlabeled examples, but always keep the test set the same. Both the labeled examples and the test examples are chosen so that all classes are represented in equal proportions (however, if the data set has a designated train-test split, then we use the given test set.) The unlabeled data can be somewhat unbalanced across classes, but that generally has not been a cause of concern.

In the second type of learning curve, we vary the number unlabeled examples for a fixed number of labeled examples. For transductive algorithms that require all test points to be available at training time (e.g., the Markov random walk and the transductive SVM), the test set changes with the number of unlabeled examples and may be quite small when the algorithm is tested with few unlabeled examples.

The learning algorithms contain a few parameters that are not estimated as part of training, and must be tuned some other way. For example, most data representations have smoothness parameters (e.g., the kernel width σ for kernel expansion and support vector machines, and σ and t for Markov random walks.) Some training procedures contain additional parameters (the SVM and MED have a slack penalty weight C). We report our results with appropriate settings of these parameters we found for the fixed test set. To reduce the risk of overtraining, we try only a few parameter values and set smoothness parameters to be consistent across algorithms when possible. (Unfortunately, many data sets are too small to allow separate validation sets for tuning parameters.) We tune parameter values for a large number of unlabeled examples and a moderate number of labeled examples. We then keep the parameters constant across the learning curves when varying the number of labeled and unlabeled examples. It would be better to adjust parameter values when varying the number of labeled and unlabeled examples, but we generally found that the constant setting is sufficient for comparing the algorithms.

4.5.2 Experiments with Kernel Expansion on Genomic Data

We have employed the kernel expansion representation for two genomic classification tasks. In the first task, we want to detect splice sites in DNA base-pair strings. Splice sites are the boundaries between introns and exons in the genome. The dataset (from [JMJ99]) consists of 500 examples with 100-dimensional feature vectors encoding DNA strings of length 25, and the task is to determine whether each DNA string contains a splice site or not.

We show the learning curve with varying numbers of labeled examples, keeping the total number of examples at 500. The kernel width σ was fixed to the median distance to the fifth nearest neighbor from the opposite class. One experimental question is whether the kernel expansion trained with maximum likelihood overfits, since it introduces a free parameter $Q(y|i)$ for every example, and the maximum likelihood estimation criterion does not regularize the parameters. This does not appear to be the case, as shown by Figure 4-1a). The error approaches the asymptotic error exponentially fast with the number of labeled examples, as indicated by the linear trend in the semilog plot.

Figure 4-1b) compares the accuracy of the kernel expansion classifier with a support vector machine that does not use unlabeled examples. The kernel expansion is more accurate with few labeled examples on this problem. Training via the maximum entropy criterion is beneficial and provides lower error than the maximum likelihood approach.

We have also tested the kernel expansion representation on a different genomic task, namely cancer classification from DNA microarray data. We classify two types of leukemia given in the `leukemia` dataset from the UCI repository [BM98a]. This dataset is good for determining whether the conditional kernel density estimate is applicable to small datasets in high dimensions. Each input vector consists of the expression levels of over 7000 genes and the dataset has only 38 training examples and 34 designated test examples. Figure 4-2 shows that the kernel expansion approach learns quickly on this problem; with 16 or more labeled examples, it makes roughly 1 mistake on average for the 34 test examples (corresponding to 3% error), which is the same error achieved by a support vector machine using all 38 labeled training examples.

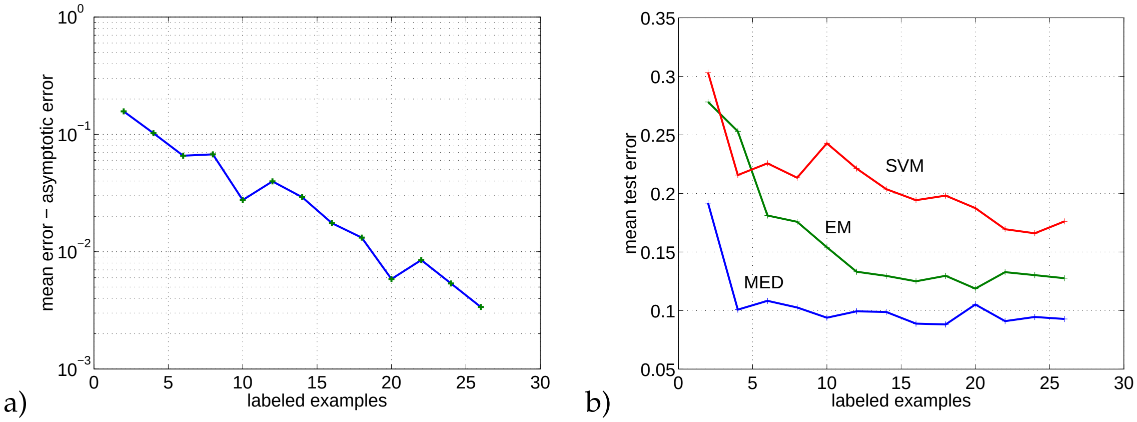


Figure 4-1: a) A semilog plot of the test error rate for the maximum likelihood formulation less the asymptotic rate as a function of labeled examples. The linear trend in the figure implies that the error rate approaches the asymptotic error exponentially fast. b) The mean test errors for maximum likelihood (EM), MED and SVM as a function of the number of labeled examples. SVM does not use unlabeled examples.

4.5.3 Text Classification

We applied the Markov random walk approach to partially labeled text classification, with few labeled documents but many unlabeled ones. Text documents are represented by high-dimensional vectors but only occupy low-dimensional manifolds, so we expect Markov random walk to be beneficial. We used the `mac` and `windows` subsets from the 20 newsgroups dataset [Lan95][2]. There were 958 and 961 examples in the two classes, with 7511 dimensions. We estimated the manifold dimensionality to exceed 7, and a histogram of the distances to the 10 nearest neighbor is peaked at 1.3. We chose a Euclidean local metric, K=10, which leads to a single connected component, and σ=0.6 for a reasonable falloff. To choose the timescale t (see 38), we plot the average margin as a function of t. We want large margins for both classes simultaneously, so t=8 is a good choice, and also gave the best cross-validation accuracy.

We trained both the EM and the margin-based formulations, using between 2 and 128 labeled points, treating all remaining points as unlabeled. We trained on 20 random splits

[2] Processed as 20news-18827, `http://www.ai.mit.edu/˜jrennie/20Newsgroups/`, removing rare words, duplicate documents, and performing tf-idf mapping.

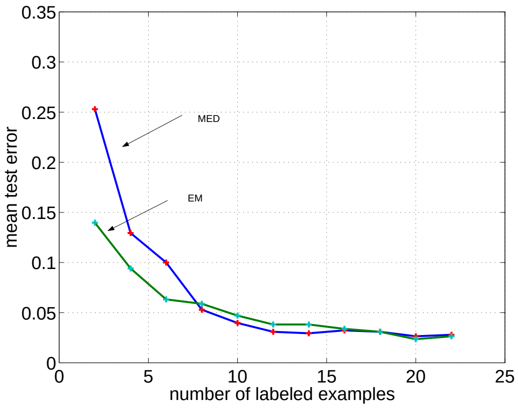


Figure 4-2: The mean test errors for the leukemia classification problem as a function of the number of randomly chosen labeled examples. Results are given for both EM (lower line) and MED (upper line) formulations.

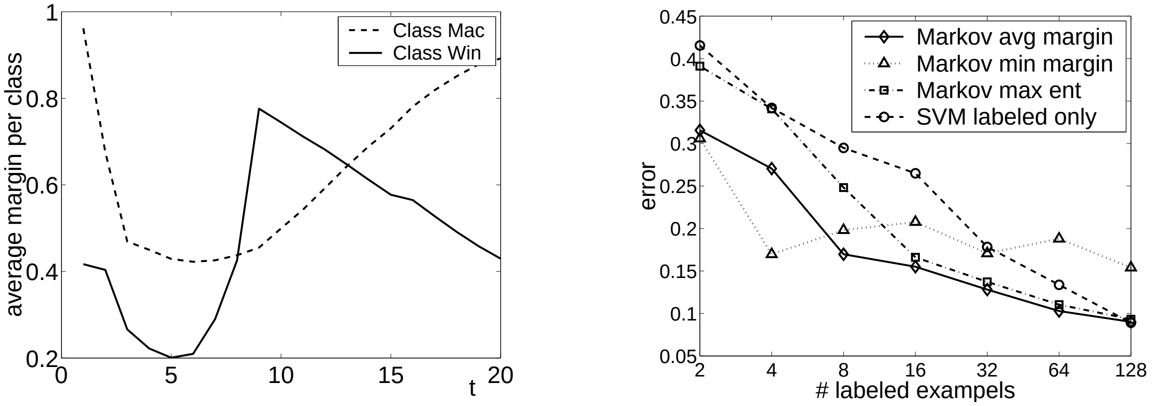


Figure 4-3: Windows vs. Mac text data. Left: Average per class margins for different t, 16 labeled documents. Right: Classification accuracy, between 2 and 128 labeled documents, for Markov random walks and best SVM.

balanced for class labels, and tested on a fixed separate set of 987 points. Results in figure 4-3 show that Markov random walk based algorithms have a clear advantage over the best SVM using only labeled data (which had a linear kernel and C=3), out of linear and Gaussian kernels, different kernel widths and values of C. The advantage is especially noticeable for few labeled points, but decreases thereafter. The average margin classifier performs best overall. It can handle outliers and mislabeled points, unlike the maximum min margin classifier that stops improving once 8 or more labeled points are supplied.

The adaptive timescale criterion (section 3.4.5) favors relatively small timescales for this dataset. For 90% of the unlabeled points, it picks the smallest timescale that reaches a labeled point, which is at most 8 for any point. As the number of labeled points increases, shorter times are chosen. For a few points, the criterion picks a maximally smooth representation (the highest timescale considered here, t=12), possibly to increase the $H(y)$ criterion. However, our experiments suggest that the adaptive time scales do not give a special classification advantage for this dataset.

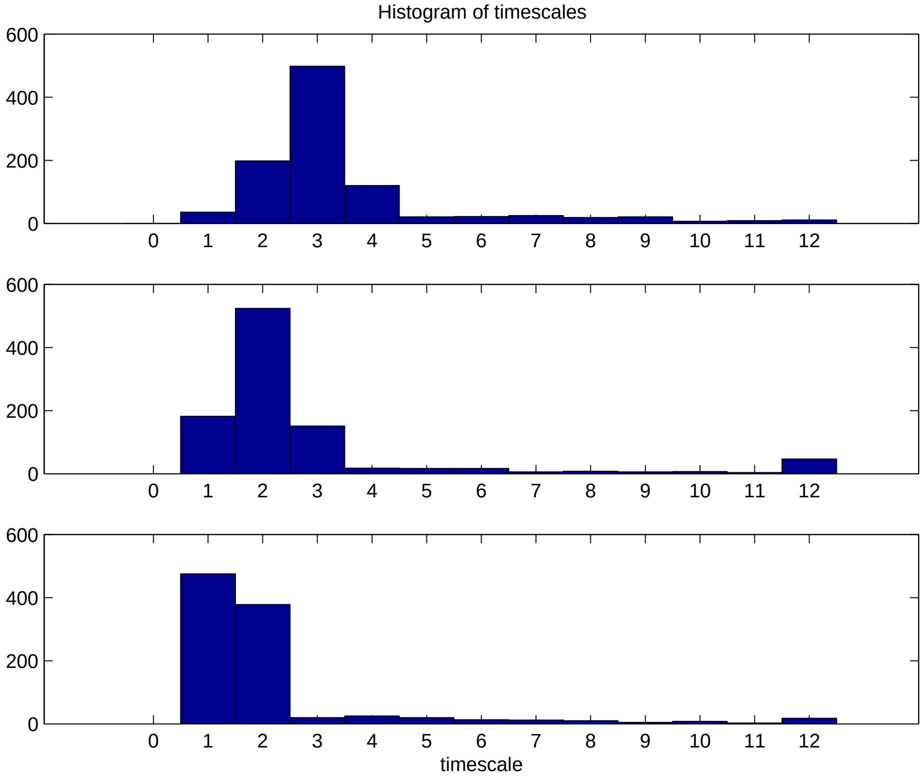


Figure 4-4: Histogram of adaptive timescales for unlabeled points given 2, 16 and 64 labeled points (top to bottom), 20 newsgroups dataset.

4.5.4 Object Detection

Object detection is a challenging computer vision problem. Our specific task is to detect cars in an image. The problem is difficult because cars have varying shapes and colors, are illuminated differently, and are set against changing backgrounds. We have simplified the task somewhat by attempting to detect only frontal cars (although slight angles are allowed) at a given scale, and without occlusion.

We[3] used a dataset from the Daimler-Chrysler research laboratory[4]. The database contains a mixture of still images and image sequences of approaching cars, recorded from a moving car. We extracted 2500 gray-level images of size 128×128 pixels. Half of the images are cars, which are centered in the image and scaled so that the front bumper occupies 64 pixels. The other half of the images are randomly chosen windows from the same road scenes the cars appeared in, and contain roads, buildings, trees, portions of cars, and whole cars which are offset in the image or appear at a different scales.

We cast the object detection problem as a classification task. We begin by extracting image features which are easier to classify than the original pixels in the images. We have employed Haar wavelet features [OPS$^+$97] which can sense edges in the image and are somewhat illumination invariant. We use horizontal, vertical and diagonal wavelet orientations, computed in subwindows at two scales 16×16 and 32×32 (see [Pap99] for details). The total number of features is 3030 per image.

We have tested both the kernel expansion representation and the Markov random walk with different parameter estimation criteria. The kernel expansion representation improves both with more labeled examples and with more unlabeled examples (Figure 4-5). We have set the kernel width adaptively at each point to equal 0.2 times the distance to the 0.05*N_{LU} nearest neighbor of the point (where N_{LU} is the total number of labeled and unlabeled examples.) Thus, the kernel width is smaller in high density regions, and smaller when there are more labeled and unlabeled points available. In the density estimation field, this type of adaptive kernel width is used by the Breiman-Meisel-Purcell (BMP) density estimator [Sco92]. Adaptive kernel widths improve classification performance compared to constant kernel widths, and also behave more appropriately when the number of examples varies.

The kernel expansion representation trained with the maximum entropy criterion leads to more accurate classifiers than when trained with the maximum likelihood approach (compare top and bottom rows of Figure 4-5). The maximum entropy criterion continues to improve accuracy with large numbers of unlabeled examples, whereas the maximum likelihood approach appears to level off.

The Markov random walk representation works extremely well on this dataset (Figure 4-6.), and requires many fewer labeled examples than kernel expansion. It achieves an error rate of 4–4.5% with only two labeled examples, and reaches 3.75% error with sixteen labeled examples. We show results for the average margin estimation criterion, and very similar results are obtained with the maximum entropy criterion. The minimum margin criterion does not do as well, and has problems handling large numbers of labeled points probably due to its noise-intolerance.

We found two crucial ingredients that contribute to the excellent performance of the Markov random walk. Firstly, the original Euclidean distance metric is only used among K

[3] Andy Crane worked on the object detection task as part of his Master's thesis [Cra02] under my supervision.

[4] Thanks to Joachim Gloger and Matthias Oberländer for supplying the car database.

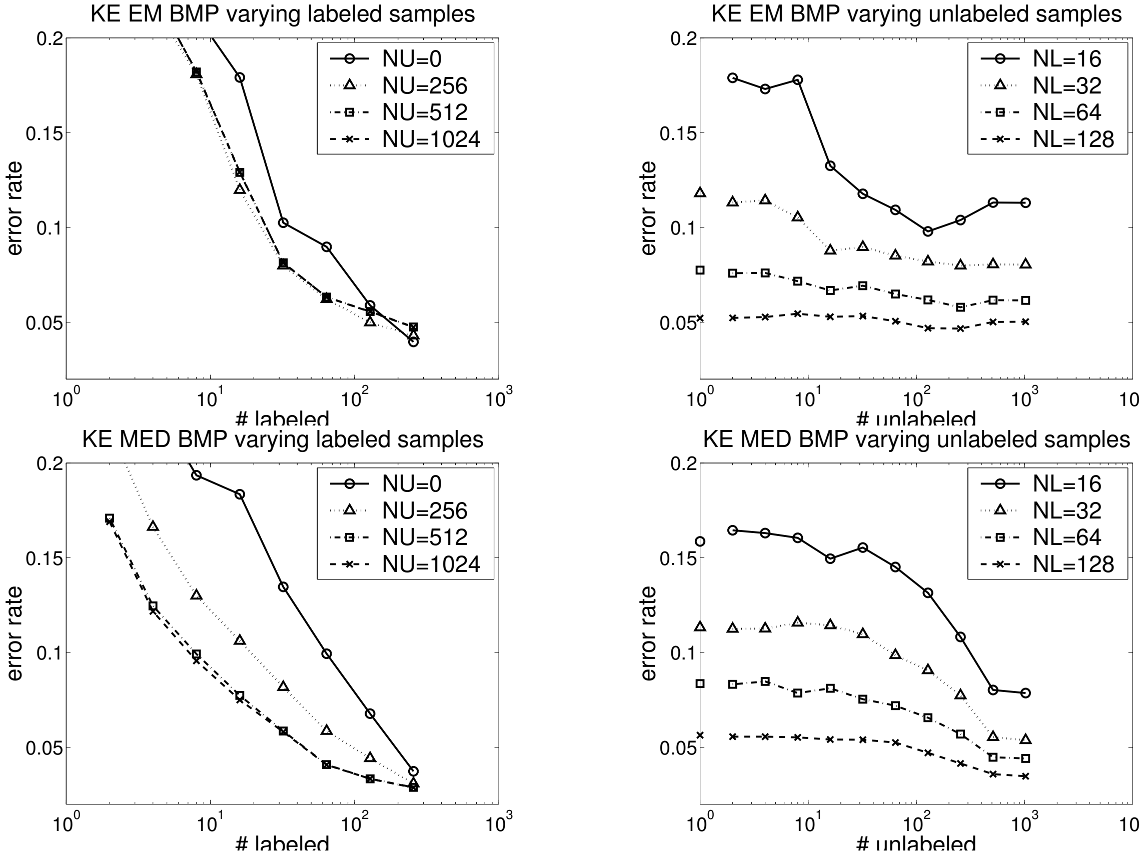
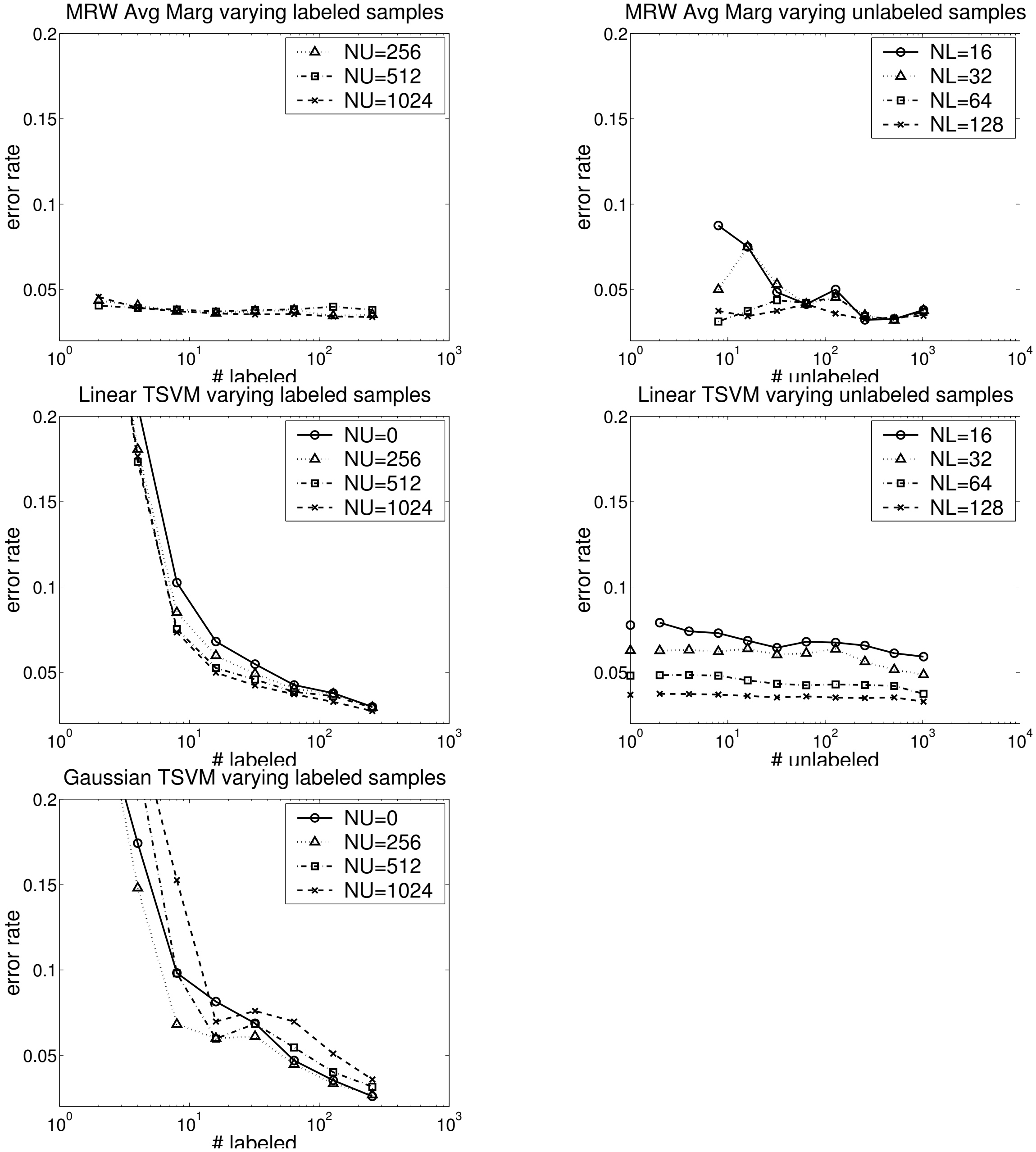


Figure 4-5: Classification error on the car detection task. The kernel expansion classifier is trained via EM (top row) and via maximum entropy discrimination MED (bottom row). Along the x-axis, we vary the number labeled points NL (left column) and the number of unlabeled points NU (right column). The kernel width σ is set adaptively per point using the BMP method. MED uses a margin parameter γ=0.1 and C=1000 $/N_L$.

nearest neighbors. The performance is dramatically poorer for values of K larger than 500 (out of 2500 examples). It appears that the original distance metric fails when used globally, and this may explain the inferior performance of kernel expansion (which implicitly uses K=2500.)

Secondly, the Markov random walk measures similarity by the average volume of paths of length t between two points. The best value of t depends on K, and is lower for high values of K. But even t=2 provides a substantial performance improvement upon t=1 (kernel expansion implicitly uses t=1.) For example, K=500, t=2 and σ=2 yields a respectable error rate of 6% with only two labeled examples. For K=5, good results can be obtained with t ranging from 6 to at least 15. The value of σ (which regulates the rate of decay of the one-step transitions with distance) appears to have little impact. Even when σ is set so high (σ=20) that all one-step transitions to K nearest neighbors are almost equally likely does the representation work as well as for moderate values of σ.

For comparison we show experiments with a transductive support vector machine, as implemented by SVM$^{\text{light}}$ (middle and bottom rows of Figure 4-6). This approach reduces to an ordinary SVM when used with zero unlabeled examples (denoted as NU=0). When used with a linear kernel, the approach improves consistently with additional data. For


MRW Avg Marg varying labeled samples

MRW Avg Marg varying unlabeled samples

Linear TSVM varying labeled samples

Linear TSVM varying unlabeled samples

Gaussian TSVM varying labeled samples

Figure 4-6: Classification error on the car detection task, for the Markov random walk representation (top row), and the transductive SVM with a linear kernel (middle row) and with a gaussian kernel (bottom row). Along the x-axis, we vary the number labeled points NL (left column) and the number of unlabeled points NU (right column). The Markov random walk uses parameters t=8, K=5, σ=3. The Gaussian kernel in the transductive SVM has σ=3.

Gaussian kernels, we have observed worsening accuracy with increasing numbers of unlabeled examples. This behavior is most likely due to a fixed choice of the kernel width (which is not appropriate), although other researchers have reported that the TSVM may become overwhelmed by too many unlabeled data points. The Gaussian kernel does perform better than the linear kernel when the number of unlabeled examples is limited to 256. Compared to kernel expansion, TSVM is worse when only two or four labeled examples are available, but has similar performance for eight labeled examples, and beats kernel expansion with sixteen or more labeled examples. TSVM is less accurate than classifiers based on the Markov random walk representation with up to 64 labeled examples.

We have performed other experiments with partially labeled learning on this car detection task, and also on a car versus truck classification task, which are described in [Cra02].

4.6 Discussion

This chapter has introduced several parameter estimation criteria, and we now summarize our experience from applying these techniques to different datasets. In general, we have found the average margin criterion to be very useful for initial experiments. Its closed form solution allows very fast training without any issues of convergence. Sometimes the average margin criterion is not sufficiently discriminative, and tends to misclassify more labeled training examples than the other criteria. Despite this fact, it usually performs reasonably well on test examples.

The minimum margin criterion is sensitive to noisy labels, and is only appropriate for training with small numbers of labeled points. However, when very few labeled points are available, this technique frequently achieves the best accuracy of all the estimation criteria, perhaps because it is the most discriminative criterion.

The maximum entropy criterion is more versatile than the minimum and average margin criteria, and allows specifying a desired target margin and a slack penalty for handling noisy points. In our experiments, it has performed the best overall.

The maximum likelihood criterion rarely achieves as good classification accuracy as the maximum entropy approach. The EM algorithm converges rather slowly in our setting, but is a well-understood and established technique.

Chapter 5

Information Regularization

5.1 Introduction

When learning with partially labeled data, we should make a link between the marginal density $P(\boldsymbol{x})$ and the conditional density $P(y|\boldsymbol{x})$. The marginal density over examples $\boldsymbol{x}$ does not depend on labels, and we can take advantage of unlabeled data to better model the marginal, or certain aspects of the marginal. However, the classifier decision boundary is based only on the conditional $P(y|\boldsymbol{x})$. A link between $P(\boldsymbol{x})$ and $P(y|\boldsymbol{x})$ enables unlabeled data to affect classifier decisions. For instance, we may assume that all points in one cluster belong to the same class (call this the "cluster link".) The cluster can be identified from unlabeled examples, namely as a contiguous region of high-density. The cluster link allows any labeled example in the cluster to influence the classification of the whole region. More formally, the cluster link asserts that $P(y|\boldsymbol{x})$ should have a derivative of low magnitude (so that $P(y|\boldsymbol{x})$ is almost constant) where $P(\boldsymbol{x})$ is high.

The link between $P(\boldsymbol{x})$ and $P(y|\boldsymbol{x})$ may be implicit or explicit. Many discriminative methods do not directly model or incorporate information from the marginal $P(\boldsymbol{x})$, but do have an implicit link. Discriminative methods often employ a notion of margin, which effectively relates $P(\boldsymbol{x})$ to $P(y|\boldsymbol{x})$: the decision boundary preferentially lies in low density regions of $P(\boldsymbol{x})$ so that the margin is large and few points fall inside the margin band. The classifiers based on kernel expansions and random walks (chapter 3) make such implicit assumptions as well. The random walk has conditionals of form $P(y|\boldsymbol{x}) = \sum_i Q(y|i)P_{0|t}(i|\boldsymbol{x})$ where the representation $P_{0|t}(i|\boldsymbol{x})$ captures cluster structure in $P(\boldsymbol{x})$. In addition, these classifiers can employ the margin assumption (see chapter 4).

In this chapter we employ information theory to explicitly constrain the conditional $P(y|\boldsymbol{x})$ on the basis of the marginal $P(\boldsymbol{x})$. We use a regularization framework, and require $P(y|\boldsymbol{x})$ to be smooth as a function of $\boldsymbol{x}$. We will impose more smoothness when the marginal density is high than when it is low. As a consequence, the conditional will not change much within a high-density cluster, and all points in the cluster will be classified the same.

The idea is in broad terms related to a number of previous approaches including maximum entropy discrimination [JMJ99], data clustering by information bottleneck [TS01], and minimum entropy data partitioning [RHD01]. A good discussion of the link between marginal and conditional densities is given in [See01a].

5.2 Information Regularization

For classification of partially labeled data, we want to learn a conditional $P(y|\boldsymbol{x})$ that both agrees with the observed labeled examples and is a smooth function of $\boldsymbol{x}$. The assumption of smoothness expresses our prior belief that small changes in $\boldsymbol{x}$ should correspond to small changes in y, so that similar examples are classified similarly. This assumption is crucial to allow the classifier to generalize to unseen examples. We will impose smoothness within a regularization framework: we look for a conditional $P(y|\boldsymbol{x})$ that has low loss on the labeled points and has a low regularization penalty.

Our regularizer will be a function of both $P(y|\boldsymbol{x})$ and $P(\boldsymbol{x})$. The regularizer will penalize changes in $P(y|\boldsymbol{x})$ more in regions where $P(\boldsymbol{x})$ is high than where it is low. The marginal $P(\boldsymbol{x})$ is assumed to be given, and may be available directly in terms of a continuous density, or as an empirical density $P(\boldsymbol{x}) = 1/N_{LU} \cdot \sum_{i \in L \cup U} \delta(\boldsymbol{x}, \boldsymbol{x}_i)$ corresponding to a set of points $\{\boldsymbol{x}_i\}$ that need not have labels. The indicator function $\delta(\boldsymbol{x}, \boldsymbol{x}_i) = 1$ when $\boldsymbol{x} = \boldsymbol{x}_i$ and 0 otherwise.

We begin by showing how to regularize a small region of the domain $\mathcal{X}$. We will later cover the domain with multiple small regions, and describe criteria that ensure regularization of the whole domain using the regularizations of the individual regions.

5.2.1 Regularizing a Single Region

Consider a small contiguous region Q in the domain $\mathcal{X}$ (e.g., an ϵ-ball). We will regularize the conditional probability $P(y|\boldsymbol{x})$ by penalizing the amount of label information contained in the region. To measure the amount of information, we employ the notion of mutual information [CT91].

The mutual information $I_Q(\boldsymbol{x}; y)$ measures the average number of bits of information that the $\boldsymbol{x}$-value of an example in region Q contains about its label y (see Figure 5-1.) In other words, mutual information quantifies the reduction of uncertainty (entropy) in predicting the label that comes from knowing *which* point in Q we are predicting versus knowing only that we are predicting a randomly drawn point in Q. The measure depends both on the marginal density $P(\boldsymbol{x})$ (specifically its restriction to $\boldsymbol{x} \in Q$ namely $P(\boldsymbol{x}|Q) = P(\boldsymbol{x}) / \int_Q P(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}$) as well as the conditional $P(y|\boldsymbol{x})$. Equivalently, we can interpret mutual information as a measure of disagreement among $P(y|\boldsymbol{x})$, $\boldsymbol{x} \in Q$. When $P(y|\boldsymbol{x})$ does not vary as a function of $\boldsymbol{x}$, the measure is zero. More precisely, the mutual information relates the densities $P(\boldsymbol{x}|Q)$ to $P(y|\boldsymbol{x})$ for a region Q according to

$$I_Q(\boldsymbol{x}; y) = \sum_y \int_{\boldsymbol{x} \in Q} P(\boldsymbol{x}|Q) P(y|\boldsymbol{x}) \log \frac{P(y|\boldsymbol{x})}{P(y|Q)}\, \mathrm{d}\boldsymbol{x} \text{ where } P(y|Q) = \int_{\boldsymbol{x} \in Q} P(\boldsymbol{x}|Q) P(y|\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}. \tag{5.1}$$

The densities conditioned on Q are normalized to integrate to 1 within the region Q. For discrete domains $\mathcal{X}$ the integral reduces to a sum. Note that the mutual information is invariant to permutations of the elements of $\mathcal{X}$ within Q, which suggests that all regions we use must be small to preserve locality properties of the domain.

To regularize the choice of the conditional $P(y|\boldsymbol{x})$ within a single region Q, we introduce the following regularization principle:

Information regularization
penalize $(M_Q/V_Q) \cdot I_Q(\boldsymbol{x}; y)$, which is the total information about the labels

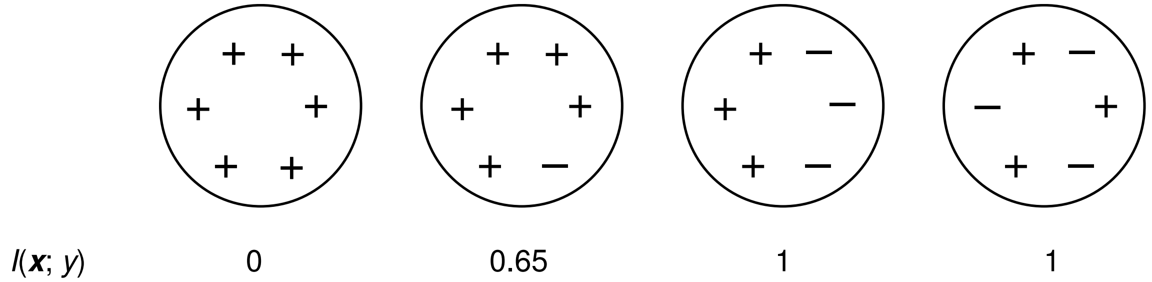


Figure 5-1: Mutual information $I(\boldsymbol{x}; y)$ measured in bits for four regions with different configurations of labels y= {+,-}. The marginal $P(\boldsymbol{x})$ is discrete and uniform across the points. The mutual information is low when the labels are homogenous in the region, and high when labels vary. The mutual information is invariant to the spatial configuration of points within the neighborhood.

> within a local region Q, normalized by the variance V_Q of the marginal in the region.

Here M_Q is the overall probability mass in the region, namely $M_Q = \int_{\boldsymbol{x} \in Q} P(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$. The mutual information $I_Q(\boldsymbol{x}; y)$ measures the information *per point*, and to obtain the total mutual information contained in a region, we must multiply by the probability mass M_Q. Thus, the regularization will be stronger for regions with high probability mass.

The V_Q is a measure of variance of the marginal $P(x|Q)$ restricted to the region. The normalization by V_Q ensures that the information is measured on a "bits per unit variance" scale. In one dimension, $V_Q = \mathrm{var}(x|Q)$. When the region is small, then the marginal will be close to uniform over the region. Then $V_Q \propto R^2$, where R is a scale measure of the region, such as the radius for spherical regions. We leave a careful analysis of the full d-dimensional case for future work, but believe that we can choose $V_Q = \mathrm{tr}\,\Sigma_Q$ or $V_Q = \sqrt[d]{\det \Sigma_Q}$, where the covariance $\Sigma_Q = \int_{x \in Q} (\boldsymbol{x} - E_Q(\boldsymbol{x}))(\boldsymbol{x} - E_Q(\boldsymbol{x}))^T P(\boldsymbol{x}|Q)\,\mathrm{d}\boldsymbol{x}$ (both formulas involving either trace or determinant give measures proportional to the squared scale of the region Q, so that doubling the scale of $\boldsymbol{x}$ will quadruple V_Q.) We now show a limiting argument that provides more insight into why the V_Q factor is necessary.

5.2.2 Limiting behavior

When a region is scaled down in size to approach a point, the mutual information it contains will approach zero (because there will remain zero uncertainty about the $\boldsymbol{x}$-value inside a point-sized region, regardless of what the y-value is). In this section we will define a related information quantity that has a well-defined and non-zero limit as the region shrinks.

For simplicity, we present the argument for a one-dimensional domain $\mathcal{X}$. Consider an infinitesimal region Q. Within this region, we can under mild assumptions approximate $P(y|x)$ by a Taylor expansion around some point $x_0 \in Q$.

$$P(y|x) \approx P(y|x_0) + \left.\frac{\mathrm{d}P(y|x)}{\mathrm{d}x}\right|_{x_0} (x - x_0) = P(y|x_0) \left(1 + \left.\frac{\mathrm{d}\log P(y|x)}{\mathrm{d}x}\right|_{x_0} (x - x_0) \right) \quad (5.2)$$

We can conveniently choose the point of the expansion x_0 to be the mean of the region

$x_0 = \int_Q P(x|Q)x \,\mathrm{d}x$. Abbreviate $s = \left.\frac{\mathrm{d}\log P(y|x)}{\mathrm{d}x}\right|_{x_0}$, noting that s is a function of $\boldsymbol{x}_0$ and y but constant with respect to x. Calculate $P(y|Q)$ by applying the Taylor expansion at each point $x \in Q$,

$$P(y|Q) = \int_Q P(y|x)P(x|Q)\,\mathrm{d}x \approx \int_Q P(x|Q)P(y|x_0)(1 + s(x - x_0))\,\mathrm{d}x \tag{5.3}$$

$$= P(y|x_0) + P(y|x_0)s\int_Q P(x|Q)(x - x_0)\,\mathrm{d}x = P(y|x_0), \tag{5.4}$$

where we have used $\int_Q P(x|Q)(x - x_0)\,\mathrm{d}x = 0$ since x_0 is the mean. We simplify the following term, which is part of the mutual information, by substituting the above expressions

$$\log\frac{P(y|x)}{P(y|Q)} = \log\frac{P(y|x_0)(1 + s(x - x_0))}{P(y|x_0)} = \log(1 + s(x - x_0)) \approx s(x - x_0) - \frac{1}{2}s^2(x - x_0)^2, \tag{5.5}$$

where we used that in the limit, $z = s(x - x_0)$ will be small, so that $\log(1 + z) \approx z - z^2/2$. The mutual information can now be written

$$I_Q(x;y) = \sum_y \int_{x\in Q} P(x|Q)P(y|x)\log\frac{P(y|x)}{P(y|Q)}\,\mathrm{d}x \tag{5.6}$$

$$\approx \sum_y \int_{x\in Q} P(x|Q)P(y|x_0)(1 + s(x - x_0))\left(s(x - x_0) - \frac{1}{2}s^2(x - x_0)^2\right)\mathrm{d}x \tag{5.7}$$

$$= \sum_y P(y|x_0)\int_{x\in Q} P(x|Q)\left(s(x - x_0) + \frac{1}{2}s^2(x - x_0)^2 - \frac{1}{2}s^3(x - x_0)^3\right)\mathrm{d}x \tag{5.8}$$

Recall that s does not scale with the size of the region Q so that $\int_{x\in Q} P(x|Q)s^n(x - x_0)^n\,\mathrm{d}x \to 0$ for all n. Now assume that the third moment of $P(x|Q)$ is neglible compared to the second in the limit, and apply $\int_Q P(x|Q)(x - x_0)\,\mathrm{d}x = 0$.

$$\approx \sum_y P(y|x_0)\frac{1}{2}s^2\int_{x\in Q} P(x|Q)(x - x_0)^2\,\mathrm{d}x \tag{5.9}$$

$$= \frac{1}{2}\underbrace{\operatorname{var}(x|Q)}_{\text{size-dependent}}\underbrace{\sum_y P(y|x_0)\left.\frac{\mathrm{d}\log P(y|x)}{\mathrm{d}x}\right|_{x_0}^2}_{\text{size-independent}} \tag{5.10}$$

We have obtained an expression for the mutual information for an infinitesimal region. The expression consists of a part $\operatorname{var}(x|Q)$ that is dependent on the size (and shape) of Q, and another part that is independent of size (and shape.) We want to regularize mutual information relative to an appropriate measure of size of the region. Equation 5.10 indicates that $\operatorname{var}(x|Q)$ is a good notion of size, because $I_Q/\operatorname{var}(x|Q)$ has a non-zero limit as the region shrinks. We drop the constant $1/2$.

The size-independent part is a Fisher information [CT91]. We can think of $P(y|x)$ as a model for y parameterized by x. The Fisher information describes how much information the parameter x contributes about the label y. The expression $\mathrm{d}\log P(y|x)/\mathrm{d}x$ is the Fisher score of y.

5.2.3 Regularizing the Whole Domain

We want to regularize the conditional $P(y|x)$ across the whole domain $\mathcal{X}$. Since individual regions must be relatively small to preserve locality, we need multiple regions to cover the domain. The cover is the set $\mathcal{C}$ of these regions. Regularization occurs only within each region, and to get regularization across regions, the regions must overlap. Without overlap, labeled points would only affect the conditional in their respective containing regions. When regions overlap, all intersecting regions will influence the conditional, and compete to assign the conditionals (Figure 5-2).

The cover should have the following properties. All areas of the domain with significant marginal density $P(\boldsymbol{x})$ should be covered, or will not be regularized (areas of zero marginal density contribute zero mutual information and need not be included.) The cover should generally be connected, so that there exists a path between any two points within the cover. Labeled points in connected covers influence the whole covered area, which is desirable when there are few labels. The amount of overlap between regions in the cover determines how strongly the regions affect each other: more overlap gives a more homogeneous regularization across the domain. However, the regions must be small to preserve locality. Ideally, we would want small yet highly overlapping regions, requiring a very large number of regions to cover the domain. In practice, we must limit the number of regions for computational reasons. The best trade-off between amount of overlap (homogenous regularization) and region size (preserving locality of the domain) remains a topic of future work. Fortunately, an initial analysis suggests that the limit of infinitesimal regions approaching 100% overlap is well-defined.

5.3 Classification with Information Regularization

Given a cover of the domain, we wish to obtain conditionals with minimal information per region in the cover, subject to correct classification of labeled points. Information regularization across multiple regions can be performed by minimizing the maximum information per region. Specifically, we constrain each region in the cover ($Q \in \mathcal{C}$) to carry information at most γ. Here γ can be thought of as an information margin measuring the maximum information contributed by any region.

$$\begin{aligned}
&\min_{P(y|\boldsymbol{x}_k),\,\gamma} && \gamma && && \text{(5.11a)}\\
&\text{s.t.} && (M_Q/V_Q)\cdot I_Q(\boldsymbol{x};y) \le \gamma && \forall Q \in \mathcal{C} && \text{(5.11b)}\\
& && P(y|\boldsymbol{x}_k) = \delta(y, \tilde{y}_k) && \forall k \in L && \text{(5.11c)}\\
& && 0 \le P(y|\boldsymbol{x}_k) \le 1, \quad \textstyle\sum_y P(y|\boldsymbol{x}_k) = 1 && \forall k \in L \cup U,\ \forall y. && \text{(5.11d)}
\end{aligned}$$

We have incorporated the labeled points by constraining their conditionals to the observed values (eq. 5.11c) (see below for other ways of incorporating labeled information). The solution $P(y|\boldsymbol{x})$ to this optimization problem is unique in regions that achieve the information constraint with equality (as long as $P(\boldsymbol{x}) > 0$). (Uniqueness follows from the strict convexity of mutual information as a function of $P(y|\boldsymbol{x})$ for nonzero $P(\boldsymbol{x})$).

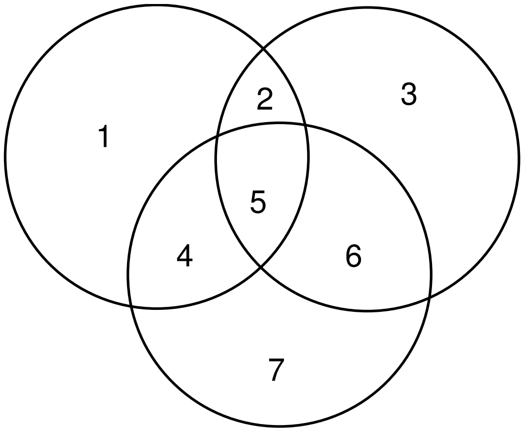


Figure 5-2: Three intersecting regions, and their atomic subregions (numbered). $P(y|\boldsymbol{x})$ for unlabeled points will be constant in atomic subregions.

Define an *atomic* subregion as a non-empty intersection of any number of regions that cannot be further intersected by any region (Figure 5-2). All unlabeled points in an atomic subregion belong to the same set of regions, and therefore participate in exactly the same constraints. They will be regularized the same way, and since mutual information is a convex function, will achieve the lowest value if all the conditionals $P(y|\boldsymbol{x})$ are equal in the atomic subregion. We can therefore parsimoniously represent conditionals of atomic subregions, instead of individual points, merely by treating such atomic subregions as "merged points" and weighting the associated constraint by the probability mass contained in the subregion.

5.3.1 Incorporating Noisy Labels

Labeled points also have conditionals $P(y|\boldsymbol{x})$, and participate in the information regularization in the same way as unlabeled points. However, their conditionals have additional constraints, which incorporate the label information. In equation 5.11c we used the constraint $P(y|\boldsymbol{x}_k) = \delta(y, \tilde{y}_k)$ for all labeled points. This constraint does not permit noise in the labels (and cannot be used when two points at the same location have disagreeing labels.) Alternatively, we can apply either of the constraints

`(fix-lbl):` $P(y|\boldsymbol{x}_i) = (1-b)^{\delta(y,\tilde{y}_i)} b^{1-\delta(y,\tilde{y}_i)}, \quad \forall i \in L$

`(exp-lbl):` $E_{P(i)}[P(\tilde{y}_i|\boldsymbol{x}_i)] \geq 1 - b.$ The expectation is over the labeled set L, where $P(i) = 1/N_L$.

The parameter $b \in [0, 0.5)$ models the amount of label noise, and is determined from prior knowledge or can be optimized via cross-validation.

Constraint `(fix-lbl)` is written out for the binary case for simplicity. The conditionals of the labeled points are directly determined by their labels, and are treated as fixed constants. Since $b < 0.5$, the thresholded conditional classifies labeled points in the observed class. In constraint `(exp-lbl)`, the conditionals for labeled points can have an average error at most b, where the averaged is over all labeled points. Thus, a few points may have conditionals that deviate significantly from their observed labels, giving robustness against mislabeled points and outliers.

To obtain classification decisions, we simply choose the class with the maximum posterior $y_k = \mathrm{argmax}_y P(y|\boldsymbol{x}_k)$. Working with binary valued $P(y|\boldsymbol{x}) \in 0, 1$ directly would yield a more difficult combinatorial optimization problem.

5.3.2 Continuous densities

Information regularization is also computationally feasible for continuous marginal densities, known or estimated. For example, we may be given a continuous unlabeled data distribution $P(\boldsymbol{x})$ and a few discrete labeled points, and regularize across a finite set of covering regions. The conditionals at points can still be merged inside atomic subregions, requiring estimates of only a finite number of conditionals.

5.3.3 Implementation

Firstly, we choose appropriate regions forming a cover, and find the atomic subregions. The choices differ depending on whether the data is all discrete or whether continuous marginals $P(\boldsymbol{x})$ are given. Secondly, we perform a constrained optimization to find the conditionals.

If the data is all discrete, create a spherical region centered at every labeled and unlabeled point (or over some reduced set still covering all the points). We have used regions of fixed radius R, but the radius could also be set adaptively at each point to the distance of its K-nearest neighbor. The union of such regions is our cover, and we choose the radius R (or K) large enough to create a connected cover. The cover induces a set of atomic subregions, and we merge the parameters $P(y|\boldsymbol{x})$ of points inside individual atomic subregions (atomic subregions with no observed points can be ignored). The marginal of each atomic subregion is proportional to the number of (merged) points it contains.

If continuous marginals are given, they will put probability mass in all atomic subregions where the marginal is non-zero. To avoid considering an exponential number of subregions, we can limit the overlap between the regions by creating a sparser cover.

Given the cover, we now regularize the conditionals $P(y|\boldsymbol{x})$ in the regions, according to eq. 5.11a. This is a convex minimization problem with a global minimum, since mutual information is convex in $P(y|\boldsymbol{x})$. It can be solved directly in the given primal form, using a quasi-Newton BFGS method. For eq. 5.11a, the required gradients of the constraints for the binary class ($y = \{\pm 1\}$) case (region Q, atomic subregion r) is:

$$\frac{M_Q}{V_Q} \frac{\mathrm{d}I_Q(\boldsymbol{x}; y)}{\mathrm{d}P(y=1|\boldsymbol{x}_r)} = \frac{M_Q}{V_Q} P(\boldsymbol{x}_r|Q) \left(\log \frac{P(y=1|\boldsymbol{x}_r)}{P(y=-1|\boldsymbol{x}_r)} \frac{P(y=-1|Q)}{P(y=1|Q)} \right). \tag{5.12}$$

The Matlab BFGS implementation `fmincon` can solve 100 subregion problems in a few minutes. We have also derived the dual optimization problem (in the Appendix), which indicates the form of the solution, and may be useful for implementation purposes.

5.4 Results and Discussion

We have experimentally studied the behavior of the regularizer with different marginal densities $P(\boldsymbol{x})$. Figure 5-3 shows the one-dimensional case with a continuous marginal density (mixture of two Gaussians), and two discrete labeled points. We choose N_Q=40 regions centered at uniform intervals of $[-1, 1]$, overlapping each other half-way, creating

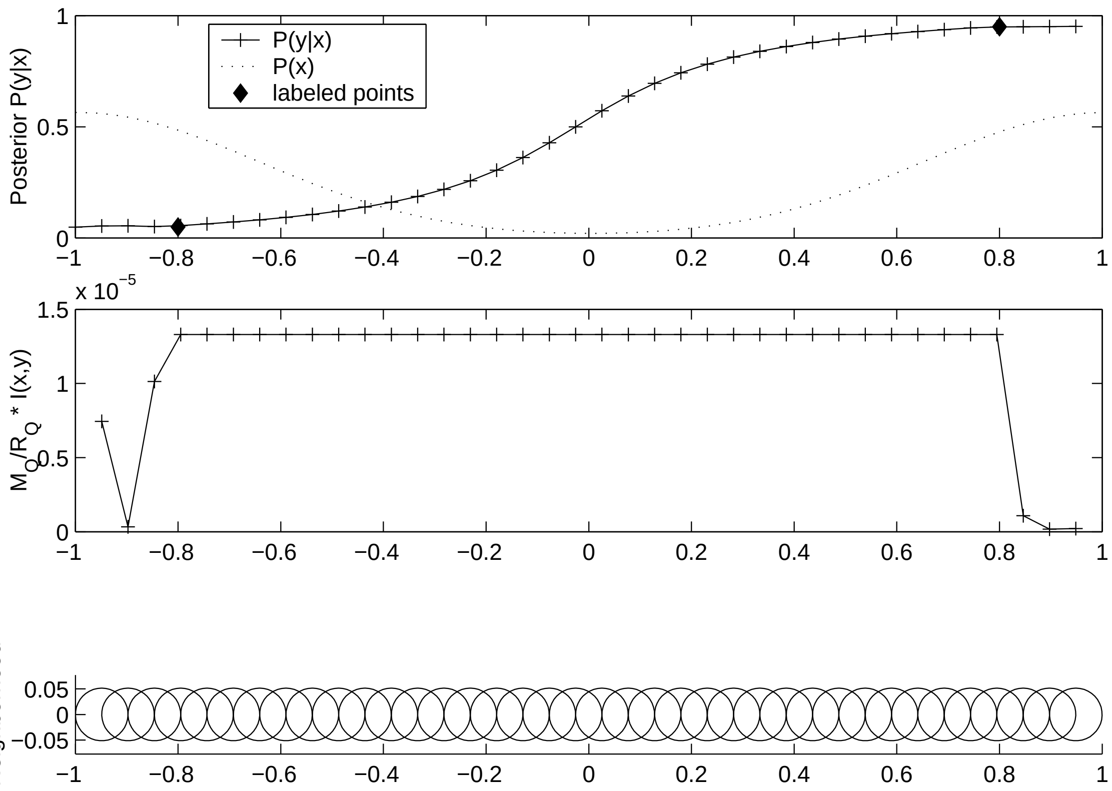


Figure 5-3: Top: conditional (solid line) for a continuous marginal $P(\boldsymbol{x})$ (dotted line) consisting of a mixture of two continuous Gaussian and two labeled points at (x=-0.8,y=-1) and (x=0.8,y=1). Middle: mutual information content of regions. Bottom: region structure - rendering of actual one-dimensional intervals.

$N_Q + 1$ atomic subregions of uniform $P(y|\boldsymbol{x})$. There are two labeled points. We show the solution attained by minimizing the maximum information (eq. 5.11a), and using the `(fix-lbl)` constraint with label noise $b = 0.05$.

The conditional varies smoothly between the labeled points of opposite classes. Note the dependence on the marginal density $P(\boldsymbol{x})$. The conditional is smoother in high-density regions, and changes more rapidly in low-density regions, as expected.

The information constraints (eq. 5.11b) are achieved with equality in all regions between labeled points of opposite classes. However, at the edges there is no pressure from the labels and the information constraints are not attained, so $P(y|\boldsymbol{x})$ is unconstrained there as long as it changes slowly.

5.4.1 Scaling Behavior

We compare solutions calculated for different resolutions of the cover. If the original cover is of sufficient resolution, the solution should remain almost the same when the resolution is increased further. The left figure 5-5 shows the solutions for covers with 10, 20, and 100 regions. The solutions with 20 or more regions are practically identical, suggesting that even relatively few regions give a solution close to the limit with infinitely many regions.

The value of $1/V_Q \cdot I_Q(\boldsymbol{x}; y)$ for any region Q should also be independent of the resolu-

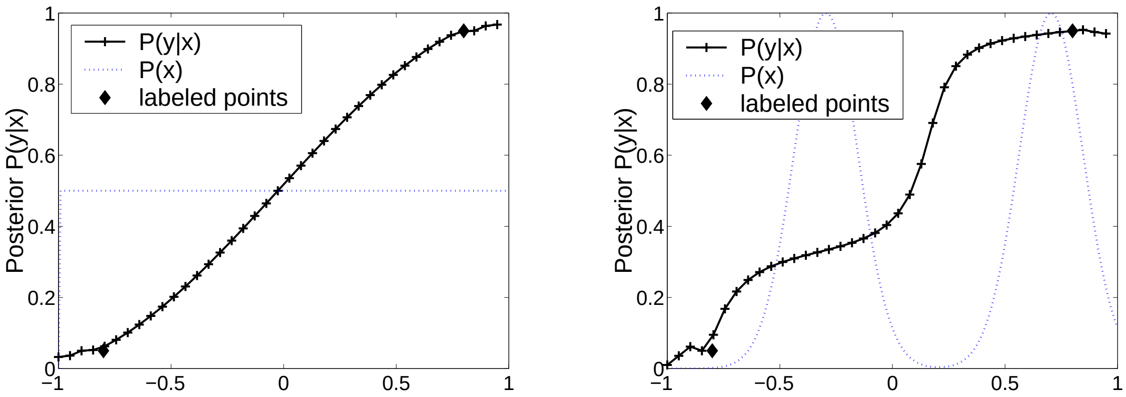


Figure 5-4: Conditionals (solid lines) for two continuous marginals (dotted lines). Left: the marginal is uniform, and the conditional approaches a straight line. Right: the marginal is a mixture of two Gaussians (with lower variance and shifted compared to Figure 5-3.) The conditional changes slowly in regions of high density.

tion when the resolution is high enough. This implies that the attained information margin $\gamma = (M_Q/V_Q) \cdot I_Q(\boldsymbol{x}; y)$ should scale as M_Q. In one dimension, M_Q scales with the radius R of the region Q, hence we should have $\gamma \propto R$. The right figure 5-5 shows that this relation holds accurately for low R corresponding to high resolutions.

5.5 Conclusion

We have presented an information theoretic regularization framework for combining conditional and marginal densities in a semi-supervised estimation setting. The framework admits both discrete and continuous (known or estimated) densities. The tractability is largely a function of the number of chosen covering regions.

The principle extends beyond the presented scope. It provides flexible means of tailoring the regularizer to particular needs. The shape and structure of the regions give direct ways of imposing relations between particular variables or values of those variables. The regions can be easily defined on low-dimensional data manifolds.

In future work we will test the regularizer on large high-dimensional datasets and explore theoretical connections to network information theory.

5.6 Appendix - dual of optimization problem

We derive the dual form of eq. 5.11a, with label noise model `(avg-lbl)`. Note that the dual is also a convex optimization problem [BSS93], and therefore relatively easy to solve, despite the large expressions. We consider the binary case with $y = \{\pm 1\}$. Form the Lagrangian U by introducing multipliers $\{\lambda_Q\}$ and β for the constraints. Define the indicator function $1_L(i) = 1$ if $i \in L$ and 0 otherwise. Capital subscripts S, Q, V, W range over the regions, and L is the set of labeled points with N_L elements. Abbreviate $c = (1 - b)N_L$, (where b is the label noise level). Also abbreviate $P(\boldsymbol{x}|Q)$ and $P(y|Q)$ by $P_Q(x)$ and $P_Q(y)$

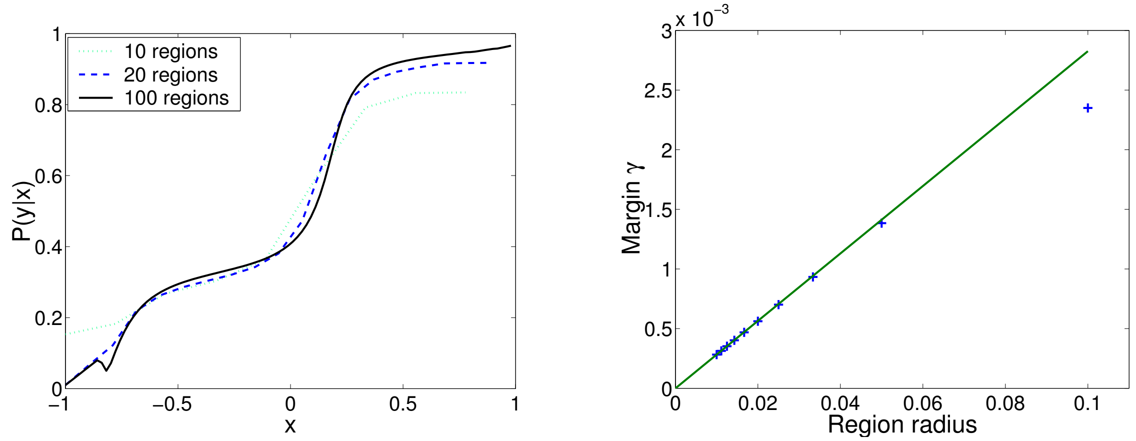


Figure 5-5: Scaling behavior for the classification task from Figure 5.4 right. Left: Conditionals $P(y|\boldsymbol{x})$ computed using 10, 20 and 100 regions. The solutions get increasingly closer. Right: The plus marks show the attained margin γ as a function of radius of the regions (corresponding to 10, 20, ..., 100 regions from right to left). The margin scales linearly when the radius is sufficiently small (a straight line is drawn for comparison.)

respectively.

$$
\begin{aligned}
U = \max_{\beta} \max_{\lambda} \min_{P(y|\boldsymbol{x}_i),\gamma} \quad & \gamma + \textstyle\sum_Q \lambda_Q (M_Q/V_Q \cdot I_Q(\boldsymbol{x};y) - \gamma) - \beta(\textstyle\sum_{i \in L} P(\tilde{y}_i|\boldsymbol{x}_i) - c) \\
& \lambda_i \geq 0, \quad \beta \geq 0, \quad 0 \leq P(y|\boldsymbol{x}_i) \leq 1 \quad \forall i \, \forall y
\end{aligned}
\tag{5.13}
$$

We obtain the constraint $\sum_Q \lambda_Q = 1$ by differentiating the Lagrangian with respect to γ. To decouple $P_Q(y)$ from $P(y|\boldsymbol{x})$, we employ the standard variational form of mutual information [CT91]. Introduce a label distribution $R_Q(y)$, and

$$
I_Q(\boldsymbol{x};y) = \sum_{i \in Q;y} P_Q(\boldsymbol{x}_i) P(y|\boldsymbol{x}_i) \log \frac{P(y|\boldsymbol{x}_i)}{P_Q(y)} = \min_{R_Q(\cdot)} \sum_{i \in Q;y} P_Q(\boldsymbol{x}_i) P(y|\boldsymbol{x}_i) \log \frac{P(y|\boldsymbol{x}_i)}{R_Q(y)}. \tag{5.14}
$$

During the optimization, update $R_Q(y) = \sum_{i \in Q} P(y|\boldsymbol{x}_i) P_Q(\boldsymbol{x}_i)$, so that $R_Q(y)$ is the average label of neighborhood Q, and at the optimum, $R_Q(y) = P_Q(y)$. Differentiating with respect to $P(y|\boldsymbol{x}_i)$ and solving we obtain

$$
P(y|\boldsymbol{x}_i) = \frac{1}{Z_{i,\lambda}} \left(\mathrm{e}^{-\beta 1_L(i)(2\delta(y,\tilde{y}_i)-1)} \prod_{Q:i \in Q} R_Q(y)^{\lambda_Q \frac{M_Q}{V_Q} P_Q(\boldsymbol{x}_i)} \right)^{\frac{1}{\sum_{S:i \in S} \lambda_S \frac{M_S}{V_S} P_S(\boldsymbol{x}_i)}} \tag{5.15}
$$

This equation has a simple interpretation. $P(y|\boldsymbol{x}_i)$ is a weighted geometric mean of $R_Q(y)$, times a term present for only labeled points. If $\lambda_Q = 0$ (inactive constraint), then the Q-th neighborhood does not contribute to setting $P(y|\boldsymbol{x}_i)$. The $Z_{i,\lambda}$ is a normalization constant such that the conditional sums to 1, and its value can be written explicitly.

Now maximize the dual by taking derivatives of the Lagrangian with respect to λ_V.

$$
\frac{\mathrm{d}U}{\mathrm{d}\lambda_V} = -\gamma + \frac{M_V}{V_V} \sum_{i \in V;y} P_V(\boldsymbol{x}_i) P(y|\boldsymbol{x}_i) \log \frac{P(y|\boldsymbol{x}_i)}{R_V(y)} \tag{5.16}
$$

$$+ \sum_{Q\colon Q\cap V\neq\varnothing} \lambda_Q \frac{M_Q}{R_Q} \sum_{i\in Q\cap V;y} P_Q(\boldsymbol{x}_i)\frac{\partial P(y|\boldsymbol{x}_i)}{\partial \lambda_V}(1+\log\frac{P(y|\boldsymbol{x}_i)}{V_Q(y)}) - \beta \sum_{i\in L\cap V} \frac{\partial P(\tilde{y}_i|\boldsymbol{x}_i)}{\partial \lambda_V}$$

$$\frac{\mathrm{d}U}{\mathrm{d}\beta} = \sum_{Q\colon Q\cap L\neq\varnothing} \lambda_Q \frac{M_Q}{V_Q} \sum_{i\in Q\cap L;y} P_Q(\boldsymbol{x}_i)\frac{\partial P(y|\boldsymbol{x}_i)}{\partial \beta}(1+\log\frac{P(y|\boldsymbol{x}_i)}{R_Q(y)}) \tag{5.17}$$

$$- \beta \sum_{i\in L} \frac{\partial P(\tilde{y}_i|\boldsymbol{x}_i)}{\partial \beta} - (\sum_{i\in L} P(\tilde{y}_i|\boldsymbol{x}_i) - c)$$

$$\frac{\partial P(y|\boldsymbol{x}_i)}{\partial \lambda_V} = P(y|\boldsymbol{x}_i)(1-P(y|\boldsymbol{x}_i))\frac{1}{\sum\limits_{S:i\in S}\lambda_S}\cdot \tag{5.18}$$

$$\left(\sum_{W:i\in W}\frac{\lambda_W}{\sum\limits_{S:i\in S}\lambda_S}\log\frac{1-R_W(y)}{R_W(y)} - \frac{\beta 1_L(i)(2\delta(y,\tilde{y}_i)-1)}{\sum\limits_{S:i\in S}\lambda_S} - \log\frac{1-R_V(y)}{R_V(y)}\right)$$

$$\frac{\partial P(y|\boldsymbol{x}_i)}{\partial \beta} = P(y|\boldsymbol{x}_i)(1-P(y|\boldsymbol{x}_i))\frac{1}{\sum\limits_{S:i\in S}\lambda_S} 1_L(i)(2\delta(y,\tilde{y}_i)-1) \tag{5.19}$$

The above defines an optimization problem in dual variables λ_Q, β and $R_Q(y)$, whereas the primal variables $P(y|\boldsymbol{x})$ have been eliminated in the sense of parameterization. We maximize it by following the gradient with respect to λ and β, and update $R_Q(y)$ after each step. However, we multiply the λ-gradient by a projection matrix to respect the constraint $\sum_Q \lambda_Q = 1$, and also ensure that $\lambda \geq 0$. The margin variable γ is eliminated by the projection, and need only be calculated at the end of the optimization.

Chapter 6

Discussion and Future Work

This thesis has examined the learning from partially labeled data problem. In order to learn from both labeled and unlabeled data, the key challenge has been to formulate mechanisms that link aspects of the marginal $P(\boldsymbol{x})$ to the conditional $P(y|\boldsymbol{x})$. In chapter 2 we discussed the links used by a few existing algorithms. In chapter 3, we presented two data representations and built a classifier of the form $P(y|\boldsymbol{x}) = \sum_i Q(y|i)P(i|\boldsymbol{x})$. The link is implicit in this equation and depends on whether the kernel expansion or Markov random walk is used for the representation $P(i|\boldsymbol{x})$. In chapter 5 we introduced a direct and explicit link based on restricting information about labels over small neighborhoods. Unlike earlier approaches, this link does not make any parametric assumptions about the dependence between the marginal and conditional, and may therefore be more generally applicable.

The link between $P(\boldsymbol{x})$ and $P(y|\boldsymbol{x})$ provides an inductive bias for the partially labeled data problem. What link works best depends on the particular dataset. Our goal has been to find links that encode an inductive bias that works well across several different application domains. The wide applicability is demonstrated by substantial classification accuracy improvements in several domains, including text and image classification. Compared to a state-of-the-art fully supervised approach (the support vector machine),

- kernel expansion reduces the mean classification error by 50% in splice site classification of genetic data (for 5–25 labeled examples and 500 unlabeled examples) (p. 52).
- the Markov random walk representation includes the kernel expansion representation as a special case, and has achieved promising performance on several additional datasets. For a text classification task, the reduction in error ranges from 20–40% in the range of 2–32 labeled examples (with 1900 unlabeled examples) (p. 53). For an image classification task, the Markov random walk reduces error by more than 30% with 2–32 labeled examples (with 2500 unlabeled examples) (p. 57).

Our approaches are competitive with another leading semi-supervised technique, namely the transductive support vector machine (TSVM) as implemented by SVM$^{\text{light}}$ [Joa99]. The Markov random walk method has superior accuracy on the image classification task with 2–32 labeled examples. The classifier accuracy improves with large numbers of unlabeled examples, whereas the TSVM technique becomes unstable and accuracy degrades with more than 200 unlabeled examples on this dataset.

Apart from achieving high classification accuracy, the techniques in this thesis remain computationally tractable with large numbers of unlabeled examples. In chapter 4 we presented several parameter inference techniques for estimating the parameters $Q(y|i)$ of the classifier $P(y|\boldsymbol{x}) = \sum_i Q(y|i)P(i|\boldsymbol{x})$. The common feature of the different training criteria in our context (expectation maximization, minimum margin, average margin and MED) is that they all lead to convex and continuous optimization problems that can be solved relatively quickly. The average margin criterion even permits a closed form solution which allows very fast cross-validation. The criteria differ in their level of discriminativeness, regularization of parameters, and robustness against noisy labels. On the data sets we have tried, none of the inference techniques is universally better than the others.

Practitioners may ask if they should use partially labeled learning techniques and how much classification accuracy can be gained. The approaches in this thesis are fairly straightforward to apply and do not require any new skills beyond those needed to apply supervised learning. When few labeled examples are available, the accuracy improvements from considering unlabeled examples can be substantial. Our recommendation is therefore that partially labeled learning should be tried when few labeled examples are available and when it is difficult to collect additional labeled examples.

However, when many labeled examples are available, current partially labeled learning techniques may offer little improvement and frequently perform worse than fully supervised techniques. Disappointing performance may be partly due to the relative immaturity of partially labeled data techniques, and the performance gap is likely to narrow with future research. The gap may not disappear entirely, however, since semi-supervised learning inherently relies on an assumed link between $P(\boldsymbol{x})$ and $P(y|\boldsymbol{x})$. Supervised learning may also employ the same type of link, but does not need to exploit it to the same extent. When the assumed link is not quite accurate, semi-supervised learning will be hurt more. Conversely, semi-supervised learning will gain more when the assumption is correct.

Many questions about the partially labeled learning problem remain a subject of future research. Most importantly, we should continue to look for general links between $P(\boldsymbol{x})$ and $P(y|\boldsymbol{x})$ that work in multiple domains. This search can be guided by successful inductive biases in other learning problems (such as the large margin concept for supervised learning). An intriguing question is whether new powerful links discovered in the context of semi-supervised learning will enable improved supervised and unsupervised learning.

Once we have a wider set of links and other assumptions for partially labeled learning, we will need diagnostics to test the appropriateness of their assumptions for particular datasets. For example, if we believe that clusters in the input domain correspond to classes, we can verify this on a dataset by clustering the data and checking the homogeneity of the class labels in the clusters. In this thesis, we have relied upon cross-validation to test the validity of the assumptions embedded in the learning algorithms. Unfortunately, cross-validation is not a very good diagnostic, as it is inaccurate when only a few labeled points are available, and is also computationally demanding. A better approach is to base diagnostics on bounds on generalization error. For instance, the $\xi\alpha$-leave-one-out bounds have been applied to transductive SVMs by computing the bounds separately on the labeled and unlabeled data [Joa00]. These bounds can indicate when SVM transduction has likely failed, and in that case it may be safer to apply an ordinary SVM without unlabeled data. Diagnostics of this type will be useful for model and algorithm selection.

There are numerous extensions to the approaches detailed in the thesis. For the kernel expansion and Markov random walk representations, it is interesting to analyze the limit as the number of unlabeled examples increases without bound. We have seen that kernel expansion can asymptotically represent the Bayes optimal decision boundary, but the parameters weighting the representation must still be inferred. The Markov random walk representation leads to path integrals and diffusion equations in the limit. Study of the limiting case can provide better insight into the finite data case, and may for instance suggest better ways of choosing smoothness parameters of the representations.

We have demonstrated information regularization on small one-dimensional examples with continuous marginals $P(\boldsymbol{x})$. In high dimensions, the set of covering regions must be defined carefully to avoid a very large number of non-empty atomic subregions, which would render the computation intractable. In future work, we may be able to replace the regularization of covering regions by regularization of the conditional as a continuous function. The continuous regularizer can be derived by taking the limit of infinitely many almost completely overlapping regions. A continuous solution to the regularization problem may then be found through a variational optimization.

We have barely scratched the surface of the partially labeled learning problem. With time, we believe that this type of learning will become widespread and enable many applications that have not been possible due to the scarcity of labeled examples. Partially labeled learning will undoubtedly be an important ingredient in empowering computers to learn the way human beings do.

Bibliography

[ABDCBH97] Noga Alon, Shai Ben-David, Nicolò Cesa-Bianchi, and David Haussler. Scale-sensitive dimensions, uniform convergence, and learnability. *Jrnl. of the ACM*, 44(4):615–631, 1997.

[ASS00] Erin L. Allwein, Robert E. Schapire, and Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. In *Proc. 17th Intl. Conf. on Machine Learning (ICML)* [ICM00].

[BC01] Avrim Blum and Shuchi Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proc. 18th Intl. Conf. on Machine Learning (ICML)* [ICM01], pages 19–26.

[BD99] Kristin Bennett and A. Demiriz. Semi-supervised support vector machines. In *Advances in Neural Information Processing Systems (NIPS)* [NIP99]. `http://www.rpi.edu/~bennek/`.

[Bis95] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford Univ. Press, 1995.

[BM98a] C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998. `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

[BM98b] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the 1998 Conference on Computational Learning Theory*, July 1998. `http://www.cs.cmu.edu/afs/cs.cmu.edu/user/avrim/www/Papers/cotrain.ps.gz`.

[BN02] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. Technical Report TR 2002-01, Univ. Chicago, Dept. Comp. Sci. and Statistics, January 2002.

[BS98] J. Bruske and G. Sommer. Intrinsic dimensionality estimation with optimally topology preserving maps. *IEEE Trans. Pattern Analysis and Mach. Intell. (PAMI)*, 20(5):572–575, May 1998.

[BSS93] Mokhtar Bazaraa, Hanif Sherali, and C Shetty. *Nonlinear programming: theory and algorithms*. Wiley, 2nd edition, 1993.

[CC95] V. Castelli and T. Cover. On the exponential value of labeled samples. *Pattern Recognition Letters*, 16(1):105–111, 1995.

[CC96] V. Castelli and T. Cover. The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing paramter. *IEEE Trans. Info. Theory*, 42:2102–2117, 1996.

[Chu97] Fan Chung. *Spectral graph theory*. American math. soc., 1997.

[CJ02] Adrian Corduneanu and Tommi Jaakkola. Continuation methods for mixing heterogeneous sources. In *18th Conf. on Uncertainty in AI*, 2002.

[Cra02] Andrew S. Crane. Object recognition with partially labeled examples. Master's thesis, Massachusetts Inst. of Technology, 2002.

[CSTK02] Nello Cristianini, John Shawe-Taylor, and Jaz Kandola. Spectral kernel methods for clustering. In *Advances in Neural Information Processing Systems (NIPS)* [NIP02].

[CT91] Thomas Cover and Joy Thomas. *Elements of Information Theory*. Wiley, 1991.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[CWS02] Olivier Chapelle, Jason Weston, and Bernhard Schölkopf. Cluster kernels for semi-supervised learning. submitted to NIPS, Vol. 15, 2002.

[DB95] Thomas G. Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *Jrnl. of Artificial Intelligence Research*, 2:263–286, 1995.

[DHS00] Richard Duda, Peter Hart, and David Stork. *Pattern Classification*. John Wiley, 2000.

[DLR77] A. Dempster, N. Laird, and D. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *Jrnl. of the Royal Statistical Society, Series B*, 39:1–38, 1977.

[Efr75] Bradley Efron. The efficiency of logistic regression compared to normal discriminant analysis. *Jrnl. of the American Statistical Assoc*, 70(352):892–898, December 1975.

[FEP95] M. Fahle, S. Edelman, and T. Poggio. Fast perceptual learning in visual hyperacuity. *Vision Research*, 35:3003–3013, 1995.

[Fri95] Bernd Fritzke. A growing neural gas network learns topologies. In *Advances in Neural Information Processing Systems (NIPS)*, volume 7, pages 625–632. MIT Press, 1995.

[FSST97] Y. Freund, S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28:133–168, 1997.

[GHS+99] Thore Graepel, Ralf Herbrich, Bernhard Schölkopf, Alex Smola, Peter Bartlett, Klaus-Robert Müller, Klaus Obermayer, and Robert Williamson. Classification on proximity data with LP-machines. In *9th Intl. Conf. on Artificial Neural Nets*, pages 304–309, 1999.

[HKP91] John Hertz, Anders Krogh, and Richard G. Palmer. *Introduction to the theory of neural computation*. Addison-Wesley, 1991.

[Hof00] Thomas Hofmann. Learning the similarity of documents. In *Advances in Neural Information Processing Systems (NIPS)* [NIP00].

[HP98] Thomas Hofmann and Jan Puzicha. Unsupervised learning from dyadic data. Technical Report TR-98-042, Intl. Comp. Sci. Inst., Berkeley, CA, 1998. `http://www.icsi.berkeley.edu/~hofmann/Papers/Hofmann-ICSI-TR98-042.ps`.

[HTF01] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2001.

[ICM98] *Proc. 15th Intl. Conf. on Machine Learning (ICML)*, July 1998.

[ICM00] *Proc. 17th Intl. Conf. on Machine Learning (ICML)*, 2000.

[ICM01] *Proc. 18th Intl. Conf. on Machine Learning (ICML)*, 2001.

[ICM02] *Proc. 19th Intl. Conf. on Machine Learning (ICML)*, 2002.

[ILO01] ILOG. *CPLEX User's Manual*, 7.1 edition, 2001.

[JB02] Michael I. Jordan and Christopher Bishop. An introduction to graphical models. Book to be published, 2002.

[Jeb01] Tony Jebara. *Discriminative, Generative and Imitative Learning*. PhD thesis, Massachusetts Inst. of Technology Media laboratory, Dec 2001.

[JH99] Tommi Jaakkola and David Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems (NIPS)* [NIP99], pages 487–493.

[JMJ99] Tommi Jaakkola, Maria Meila, and Tony Jebara. Maximum entropy discrimination. Technical Report AITR-1668, Massachusetts Inst. of Technology AI lab, 1999. `http://www.ai.mit.edu/`.

[JMJ00] Tommi Jaakkola, Marina Meila, and Tony Jebara. Maximum entropy discrimination. In *Advances in Neural Information Processing Systems (NIPS)* [NIP00], pages 470–476.

[Joa99] Thorsten Joachims. Transductive inference for text classification using support vector machines. In *Proc. 16th Intl. Conf. on Machine Learning (ICML)*, 1999. `http://www-ai.cs.uni-dortmund.de/DOKUMENTE/joachims_99c.ps.gz`.

[Joa00] Thorsten Joachims. *The Maximum-Margin Approach to Learning Text Classifiers: Methods, Theory, and Algorithms*. PhD thesis, Universität Dortmund, Germany, 2000. Fachbereich Informatik.

[KKM02] Dan Klein, Sepandar Kamvar, and Christopher Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *Proc. 19th Intl. Conf. on Machine Learning (ICML)* [ICM02], pages 307–314.

[KL02] Risi Kondor and John Lafferty. Diffusion kernels on graphs and other discrete structures. In *Proc. 19th Intl. Conf. on Machine Learning (ICML)* [ICM02].

[Lan95] Kenneth Lang. Learning to filter netnews. In *Proc. 12th Intl. Conf. on Machine Learning (ICML)*, pages 331–339, July 1995.

[LCB$^+$02] G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. In *Proc. 19th Intl. Conf. on Machine Learning (ICML)* [ICM02].

[LDG00] Fabien Letouzey, François Denis, and Rémi Gilleron. Learning from positive and unlabeled examples. In *11-th Intl. Conf. on Algorithmic Learning Theory (ALT)*, Sydney, Australia, December 2000.

[LR87] Roderick J. Little and Donald B. Rubin. *Statistical analysis with missing data*. Wiley, New York, 1987.

[LS01] Neil D. Lawrence and Bernhard Schölkopf. Estimating a kernel fisher discriminant in the presence of label noise. In *Proc. 18th Intl. Conf. on Machine Learning (ICML)* [ICM01].

[MEP95] Daniel V. Schroeder Michael E. Peskin. *An Introduction to Quantum Field Theory*. Perseus Publishing, 1995.

[Min] Thomas P. Minka. The expectation-maximization algorithm for MAP estimation. tutorial note,`http://www.media.mit.edu/~tpminka/`.

[MRMN98] A. McCallum, R. Rosenfeld, T. Mitchell, and A. Ng. Improving text classification by shrinkage in a hierarchy of classes. In *Proc. 15th Intl. Conf. on Machine Learning (ICML)* [ICM98], pages 359–367.

[MS94] Thomas Martinetz and Klaus Schulten. Topology representing networks. *Neural Networks*, 7(3):507–522, 1994.

[MS01] Marina Meilă and Jianbo Shi. A random walks view of spectral segmentation. In *AI and Statistics 2001, Intl. Workshop on Artificial Intelligence and Statistics, Key West, Florida*, 2001.

[MU97] D. Miller and T. Uyar. A mixture of experts classifer with learning based on both labelled and unlabelled data. In *Advances in Neural Information Processing Systems (NIPS)*, volume 9, pages 571–577. MIT Press, 1997.

[NIP99] *Advances in Neural Information Processing Systems (NIPS)*, volume 11. MIT Press, 1999.

[NIP00] *Advances in Neural Information Processing Systems (NIPS)*, volume 12. MIT Press, 2000.

[NIP01] *Advances in Neural Information Processing Systems (NIPS)*, volume 13. MIT Press, 2001.

[NIP02] *Advances in Neural Information Processing Systems (NIPS)*, volume 14. MIT Press, 2002.

[NJW02] Andrew Ng, Michael Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems (NIPS)* [NIP02].

[NMTM00] Kamal Nigam, Andrew McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2):103–134, 2000. `http://www.cs.cmu.edu/~knigam/papers/emcat-mlj99.ps`.

[NZJ01] Andrew Ng, Alice Zheng, and Michael Jordan. Link analysis, eigenvectors, and stability. In *IJCAI*, 2001.

[OPS$^+$97] Michael Oren, Constantine Papageorgiou, Pawan Sinha, Edgar Osuna, and Tomaso Poggio. Pedestrian detection using wavelet templates. In *Proc. Comp. Vision Pattern Recogn. (CVPR)*, pages 193–199, Puerto Rico, 1997.

[Pap99] Constantine Papageorgiou. *A Trainable System for Object Detection in Images and Video Sequences*. PhD thesis, Massachusetts Inst. of Technology, 1999.

[PL96] Ramani Pilla and Bruce Lindsay. Faster EM methods in high-dimensional finite mixtures. In *Proc. of the Statistical Computing Section*, pages 166–171. American Statistical Association, 1996.

[PTVF93] William Press, Saul Teukolsky, William Vetterling, and Brian Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 2nd edition, 1993.

[RH] Y. Dan Rubinstein and Trevor Hastie. Discriminative vs informative learning.

[RHD01] Stephen Roberts, C. Holmes, and D. Denison. Minimum-entropy data partitioning using reversible jump Markov chain Monte Carlo. *IEEE Trans. Pattern Analysis and Mach. Intell. (PAMI)*, 23(8):909–914, 2001.

[RS00] Sam Roweis and Lawrence Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, December 2000.

[Sch97] J. L. Schafer. *Analysis of Incomplete Multivariate Data*. Chapman & Hall, 1997.

[Sco92] David Scott. *Multivariate density estimation: theory, practice, and visualization*. Wiley, 1992.

[See01a] Matthias Seeger. Input-dependent regularization of conditional density models. Unpublished. `http://www.dai.ed.ac.uk/homes/seeger/`, 2001.

[See01b] Matthias Seeger. Learning with labeled and unlabeled data. Unpublished. `http://www.dai.ed.ac.uk/homes/seeger/`, February 2001.

[Sil98] B. Silverman. *Density estimation for statistics and data analysis*. Chapman & Hall, 1998.

[SJ01] Martin Szummer and Tommi Jaakkola. Kernel expansions with unlabeled examples. In *Advances in Neural Information Processing Systems (NIPS)* [NIP01], pages 626–632. `http://www.ai.mit.edu/people/szummer/papers/kernelexp-nips00.ps.gz`.

[SJ02] Martin Szummer and Tommi Jaakkola. Partially labeled classification with markov random walks. In *Advances in Neural Information Processing Systems (NIPS)* [NIP02], pages 945–952. `http://www.ai.mit.edu/people/szummer/`.

[SM00] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Analysis and Mach. Intell. (PAMI)*, 22(8):888–905, August 2000.

[SPST$^+$01] Bernhard Schölkopf, John Platt, John Shawe-Taylor, Alex Smola, and Robert Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

[SS99] Robert E. Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

[TdSL00] Joshua Tenenbaum, Vin de Silva, and John Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, December 2000.

[TK01] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *Jrnl. of Machine Learning Research*, 2:45–66, November 2001.

[TP91] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[TS01] Naftali Tishby and Noam Slonim. Data clustering by markovian relaxation and the information bottleneck method. In *Advances in Neural Information Processing Systems (NIPS)* [NIP01], pages 640–646.

[TW01] Joseph F. Traub and Henryk Wozniakowski. Path integration on a quantum computer. Technical Report 01-10-055, Santa Fe Institute, 2001. Working paper, `http://www.santafe.edu/sfi/publications/Working-Papers/01-10-055.pdf`.

[Vap98] Vladimir Vapnik. *Statistical Learning Theory*. Wiley, 1998.

[VD95] Peter Verveer and Robert Duin. An evaluation of intrinsic dimensionality estimators. *IEEE Trans. Pattern Analysis and Mach. Intell. (PAMI)*, 17(1):81–86, January 1995.

[Wei99] Yair Weiss. Segmentation using eigenvectors: a unifying view. In *Intl. Conf. Computer Vision*, pages 975–982, 1999.

[Wol96a] David H. Wolpert. The existence of a priori distinctions between learning algorithms. *Neural Computation*, 8(7):1391–1420, October 1996.

[Wol96b] David H. Wolpert. The lack of a priori distinctions between learning algorithms. *Neural Computation*, 8(7):1341–1390, October 1996.

[Wu83] C. F. Jeff Wu. On the convergence properties of the EM algorithm. *Annals of Statistics*, 11(1):95–103, March 1983.

[XJ96] Lei Xu and Michael Jordan. On convergence properties of the EM algorithm for gaussian mixtures. *Neural Computation*, 8:129–151, 1996.

[Yea02] Chen-Hsiang Yeang. Path integral approaches on statistical problems. Draft note and personal communication, May 2002.

[ZO00] T. Zhang and F. Oles. A probability analysis on the value of unlabeled data for classification problems. In *Proc. 17th Intl. Conf. on Machine Learning (ICML)* [ICM00].