Journal of Machine Learning Research X (2005) X-X Submitted 5/2005; Published X/XX

A Framework for Learning Predictive Structures from Multiple Tasks and Unlabeled Data

Rie Kubota Ando RIE1@US.IBM.COM
IBM T.J. Watson Research Center
Yorktown Heights, NY 10598, U.S.A.

Tong Zhang TZHANG@YAHOO-INC.COM
Yahoo Research, New York City, U.S.A.

Editor: Peter Bartlett

Abstract

One of the most important issues in machine learning is whether one can improve the performance of a supervised learning algorithm by including unlabeled data. Methods that use both labeled and unlabeled data are generally referred to as semi-supervised learning. Although a number of such methods are proposed, at the current stage, we still don't have a complete understanding of their effectiveness. This paper investigates a closely related problem, which leads to a novel approach to semi-supervised learning. Specifically we consider learning predictive structures on hypothesis spaces (that is, what kind of classifiers have good predictive power) from multiple learning tasks. We present a general framework in which the structural learning problem can be formulated and analyzed theoretically, and relate it to learning with unlabeled data. Under this framework, algorithms for structural learning will be proposed, and computational issues will be investigated. Experiments will be given to demonstrate the effectiveness of the proposed algorithms in the semi-supervised learning setting.

1. Introduction

In machine learning applications, one can often find a large amount of unlabeled data without difficulty, while labeled data are costly to obtain. Therefore a natural question is whether we can use unlabeled data to build a more accurate classifier, given the same amount of labeled data. This problem is often referred to as semi-supervised learning.

In general, semi-supervised learning algorithms use both labeled and unlabeled data to train a classifier. Although a number of methods have been proposed, their effectiveness is not always clear. For example, Vapnik introduced the notion of transductive inference (Vapnik, 1998), which may be regarded as an approach to semi-supervised learning. Although some success has been reported (e.g., see Joachims, 1999), there has also been criticism pointing out that this method may not behave well under some circumstances (Zhang and Oles, 2000). Another popular semi-supervised learning method is co-training (Blum and Mitchell, 1998), which is related to the bootstrap method used in some NLP applications (Yarowsky, 1995) and to EM (Nigam et al., 2000). The basic idea is to label part of unlabeled data using a high precision classifier, and then put the "automatically-labeled" data into the training data. However, it was pointed out by Pierce and Cardie

(2001) that this method may degrade the classification performance when the assumptions of the method are not satisfied (that is when noise is introduced into the labels through non-perfect classification). This phenomenon is also observed in some of our experiments reported in Section 5.

Another approach to semi-supervised learning is based on a different philosophy. The basic idea is to define good functional structures using unlabeled data. Since it does not bootstrap labels, there is no label noise which can potentially corrupt the learning procedure. An example of this approach is to use unlabeled data to create a data-manifold (graph structure), on which proper smooth function classes can be defined (Szummer and Jaakkola, 2002, Zhou et al., 2004, Zhu et al., 2003). If such smooth functions can characterize the underlying classifier very well, then one is able to improve the classification performance.

It is worth pointing out that smooth function classes based on graph structures do not necessarily have good predictive power. Therefore a more general approach, based on the same underlying principle, is to directly learn a good underlying smooth function class (that is, what good classifiers are like). If the learning procedure takes advantage of unlabeled data, then we obtain a semi-supervised learning method that is specifically aimed at finding structures with good predictive power.

This motivates the general framework we are going to develop in this paper. That is, we want to learn some underlying predictive functional structures (smooth function classes) that can characterize what good predictors are like. We call this problem *structural learning*. Our key idea is to learn such structures by considering multiple prediction problems simultaneously. At the intuitive level, when we observe multiple predictors for different problems, we have a good sample of the underlying predictor space, which can be analyzed to find the common structures shared by these predictors. Once important predictive structures on the predictor space are discovered, we can then use the information to improve upon each individual prediction problem. A main focus of this paper is to formalize this intuitive idea and analyze properties of structural learning more rigorously.

The idea that one can benefit by considering multiple problems together has appeared in the statistical literature. In particular, Bayesian hierarchical modeling is motivated from the same principle. However, the framework developed in this paper is under the frequentist setting, and the most relevant statistical studies are shrinkage methods in multiple-output linear models (see Section 3.4.6 of Hastie et al., 2001). In particular, the algorithm proposed in Section 3 has a form similar to a shrinkage method proposed by Breiman and Friedman (1997). However, the framework presented here (as well as the specific algorithm in Section 3) is more general than the earlier statistical studies. In the machine learning literature, related work is sometime referred to as *multi-task learning*, for example, see (Baxter, 2000, Ben-David and Schuller, 2003, Caruana, 1997, Evegniou and Pontil, 2004, Micchelli and Ponti, 2005) and references therein. We shall call our procedure structural learning since it is a more accurate description of what our method does in the semi-supervised learning setting. That is, we transfer the predictive structure learned from multiple tasks (on unlabeled data) to the target supervised problem. In the literature, this idea is also referred to as *inductive transfer*. The success of this approach depends on whether the learned structure is helpful for the target supervised problem.

It follows that although this work is motivated by semi-supervised learning, the general structural learning (or multi-task learning) problem considered in the paper is of indepen-

dent interest. For semi-supervised learning, as we shall show later, the multiple prediction problems needed for structural learning can be generated from unlabeled data. However, the basic framework can also be applied to other applications where we have multiple prediction problems that are not necessarily derived from unlabeled data (as in the earlier statistical and machine learning studies). Because of this, the first part of the paper focuses on the development of a general structural learning paradigm as well as our algorithm. The main implication is that one can reliably learn a good underlying structure if it is shared by multiple prediction problems. In the second part, we shall demonstrate how to apply the idea of learning structure to semi-supervised learning, and demonstrate the effectiveness of the proposed method in this context.

A short version of this paper, mainly reporting some empirical results, appeared in ACL (Ando and Zhang, 2005). This version includes a more complete derivation of the proposed algorithm, with theoretical analysis and several additional experimental results. In Section 2, we formally introduce the structural learning problem under the framework of standard machine learning. We then propose a specific algorithm that finds a common low-dimensional feature space shared by the multi-problems. The algorithm will be studied in Section 3, with theoretical analysis given in Appendix A. Section 4 shows how to apply structural learning in the context of semi-supervised learning. The basic idea is to use unlabeled data to generate auxiliary prediction problems that are useful for discovering important predictive structures. Such structures can then be estimated using the algorithm developed in Section 3. We will also give intuitive justifications on why the structure shared by the artificially created auxiliary problems is helpful for the supervised problem. Experiments are provided in Section 5 to illustrate the effectiveness of the algorithm proposed in Section 3 on several semi-supervised tasks. Section 6 presents a high level summary of the main ideas developed in the paper.

2. The Structural Learning Problem

This section introduces the problem of learning predictive functional structures. Although related ideas have been explored in some earlier statistical and machine learning studies, for completeness, we shall include a self-contained description. The framework considered here will be the basis of our algorithm presented in Section 3.

2.1 Supervised learning

In the standard formulation of supervised learning, we seek a predictor that maps an input vector $\mathbf{x} \in \mathcal{X}$ to the corresponding output $y \in \mathcal{Y}$. Usually, one selects the predictor from a set $\mathcal{H}$ of functions based on a finite set of training examples $\{(\mathbf{X}_i, Y_i)\}$ that are independently generated according to some unknown probability distribution $\mathcal{D}$. The set $\mathcal{H}$, often called the *hypothesis space*, consists of functions from $\mathcal{X}$ to $\mathcal{Y}$ that can be used to predict the output in $\mathcal{Y}$ of an input datum in $\mathcal{X}$. Our goal is to find a predictor f so that its error with respect to $\mathcal{D}$ is as small as possible. In this paper, we assume that the quality of the predictor p is measured by the expected loss with respect to $\mathcal{D}$:

$$R(f) = \mathbf{E}_{\mathbf{X},Y} L(f(\mathbf{X}), Y).$$

Given a set of training data, a frequently used method for finding a predictor $\hat{f} \in \mathcal{H}$ is to minimize the empirical error on the training data (often called *empirical risk minimization* or ERM):

$$\hat{f} = \arg\min_{f \in \mathcal{H}} \sum_{i=1}^{n} L(f(\mathbf{X}_i), Y_i).$$

It is well-known that with a fixed sample size, the smaller the hypothesis space $\mathcal{H}$, the easier it is to learn the best predictor in $\mathcal{H}$. The error caused by learning the best predictor from finite sample is called the *estimation error*. However, the smaller the hypothesis space $\mathcal{H}$, the less accurate the best predictor in $\mathcal{H}$ becomes. The error caused by using a restricted $\mathcal{H}$ is often referred to as the *approximation error*. In supervised learning, one needs to select the size of $\mathcal{H}$ to balance the trade-off between approximation error and estimation error. This is typically done through model selection, where we learn a set of predictors from a set of candidate hypothesis spaces $\mathcal{H}_\theta$, and then pick the best choice on a validation set.

2.2 Learning good hypothesis spaces

In practice, a good hypothesis space should have a small approximation error and a small estimation error. The problem of choosing a good hypothesis space is central to the performance of the learning algorithm, but often requires specific domain knowledge or assumptions of the world.

Assume that we have a set of candidate hypothesis spaces. If one only observes a single prediction problem $\mathcal{X} \to \mathcal{Y}$ on the underlying domain $\mathcal{X}$, then a standard approach to hypothesis space selection (or model selection) is by cross validation. If one observes multiple prediction problems on the same underlying domain, then it is possible to make better estimate of the underlying hypothesis space by considering these problems simultaneously.

We now describe a simple model for structural learning, which is the foundation of this paper. A similar point of view can also be found in (Baxter, 2000). Consider m learning problems indexed by $\ell \in \{1, \ldots, m\}$, each with n_ℓ samples $(\mathbf{X}_i^\ell, Y_i^\ell)$ indexed by $i \in \{1, \ldots, n_\ell\}$, which are independently drawn from a distribution $\mathcal{D}_\ell$. For each problem ℓ, assume that we have a set of candidate hypothesis spaces $\mathcal{H}_{\ell,\theta}$ indexed by a common structural parameter $\theta \in \Gamma$ that is shared among the problems.

Now, for the ℓ-th problem, we are interested in finding a predictor $f_\ell : \mathcal{X} \to \mathcal{Y}$ in $\mathcal{H}_{\ell,\theta}$ that minimizes the expected loss over $\mathcal{D}_\ell$. For notational simplicity, we assume that the problems have the same loss function (although the requirement is not essential in our analysis). Given a fixed structural parameter θ, the predictor for each problem can be estimated using empirical risk minimization (ERM) over the hypothesis space $\mathcal{H}_{\ell,\theta}$:

$$\hat{f}_{\ell,\theta} = \arg\min_{f \in \mathcal{H}_{\ell,\theta}} \sum_{i=1}^{n_\ell} L(f(\mathbf{X}_i^\ell), Y_i^\ell), \qquad (\ell = 1, \ldots, m). \tag{1}$$

The purpose of structural learning is to find an optimal structural parameter θ such that the expected risks of the predictors $\hat{f}_{\ell,\theta}$ (each with respect to the corresponding distribution $\mathcal{D}_\ell$), when averaged over $\ell = 1, \ldots, m$, are minimized.

If we use cross-validation for structural parameter selection, then we can immediately notice that a more stable estimate of the optimal θ can be obtained by considering multiple

learning tasks together. In particular, if for each problem ℓ, we have a validation set $(\bar{\mathbf{X}}_j^\ell, \bar{Y}_j^\ell)$ for $j = 1, \ldots, \bar{n}_\ell$, then for structural learning, the total number of validation data is $\sum_{\ell=1}^{m} \bar{n}_\ell$. Therefore effectively, we have more data for the purpose of selecting the optimal shared hypothesis space structure. This implies that even if the sample sizes are small for the individual problems, as long as m is large, we are able to find the optimal θ accurately. A PAC style analysis will be provided in Appendix A, where we can state a similar result without cross-validation.

In general, we expect that the hypothesis space $\mathcal{H}_{\ell,\theta}$ determines the functional structure of the learned predictor. The θ parameter can be a continuous parameter that encodes our assumption of what a good predictor should be like. If we have a large parameter space, then we can explore many possible functional structures. This argument (more rigorous results are given in Appendix A) implies that it is possible to discover the optimal shared structure when the number of problems m is large.

2.3 Good structures on the input space

The purpose of this section is to provide an intuitive discussion on why in principle, there exist good functional structures (good hypothesis spaces) shared by multiple tasks. Conceptually, we may consider the simple case $\mathcal{H}_{\ell,\theta} = \mathcal{H}_\theta$, where different problems share exactly the same underlying hypothesis space.

Given an arbitrary input space $\mathcal{X}$ without any known structure, we argue that it is often possible to learn what a good predictor looks like from multiple prediction problems. The key reason is that in practice, not all predictors are equally good (or equally likely to be observed). In real world applications, one usually observes "smooth" predictors where the smoothness is with respect to a certain intrinsic underlying distance on the input space. In general, if two points are close in this intrinsic distance, then the values that a good predictor produces at these points are also likely to be similar. In particular, completely random predictors are likely to be bad predictors, and are rarely observed in practical applications.

In machine learning, the smoothness condition is often enforced by the hypothesis space we select. For example, kernel methods constrain the smoothness of a function using a certain reproducing kernel Hilbert space (RKHS) norm. For such functions (in a RKHS), closeness of two points under a certain metric often implies closeness in predictive values. One may also consider more complicated smoothness conditions that explore the observed data-manifold (e.g. graph-based semi-supervised learning methods mentioned in the introduction). Such a smoothness condition will be useful if it correlates well with predictive ability.

In general, a good distance measure on $\mathcal{X}$ induces a good hypothesis space which enforces smoothness with respect to the underlying distance. However, in reality, it is often not clear what is the best distance measure in the underlying space. For example, in natural language processing, the space $\mathcal{X}$ consists of discrete points such as words, for which no appropriate distance can be easily defined. Even for continuous vector-valued input points, it is difficult to justify that the Euclidean distance is better than something else. Even after a good distance function can be selected, it is not clear whether we can define appropriate smoothness conditions with respect to the distance.

If we observe multiple tasks, then important common structures can be discovered simply by analyzing the multiple predictors learned from the data. If these tasks are very similar to the actual learning task which we are interested in, then we can benefit significantly from the discovered structures. Even if the tasks are not directly related, the discovered structures can still be useful. This is because in general, predictors tend to share similar smoothness conditions with respect to a certain distance that is intrinsic to the underlying input space.

As an example to illustrate the main argument graphically, we consider a discrete input space of six points $\mathcal{X} = \{A, B, C, D, E, F\}$. Assume we obtain estimates of three functions from three different prediction problems, and plot the obtained function values against the input points in Figure 1. In this example, we can notice that function values at points A, C, and D are similar, while function values at points F and E are similar. Therefore by observing the estimated functions, we may conclude that under some intrinsic distance metric on $\mathcal{X}$, points A, C, and D are "close" to each other, and points E and F are "close" to each other. A good function on $\mathcal{X}$ should be smooth with respect to this intrinsic distance. We will come back to the argument presented in this section using text data as a more concrete example, when we discuss semi-supervised learning in Section 4.

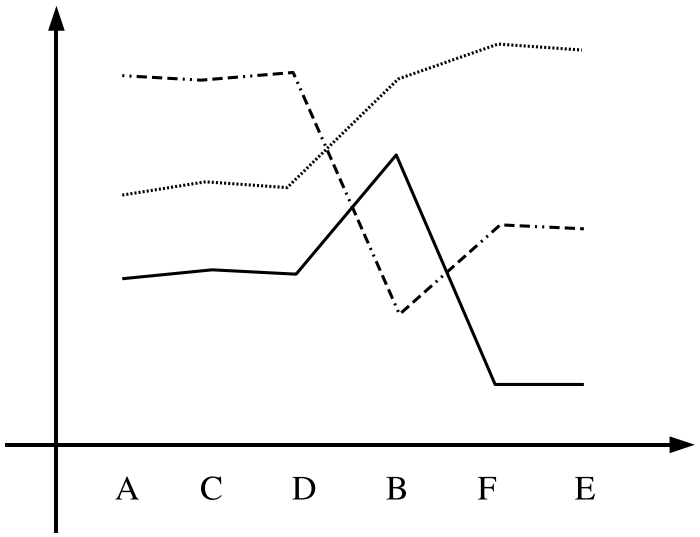

Figure 1: An Illustration of Discovering Functional Structure From Multiple Prediction Tasks

2.4 A more abstract form of structural learning

We may also pose structural learning in a slightly more abstract form, which is useful when we don't use empirical risk minimization as the learner.

Assume that for each problem ℓ, we are given a learning algorithm $\mathcal{A}_\ell$ that takes a set of training samples $S_\ell = \{(\mathbf{X}_i^\ell, Y_i^\ell)\}_{i=1,\ldots,n_\ell}$ and a structural parameter $\theta \in \Gamma$, and produce a predictor $\hat{f}_{\ell,\theta}$: $\hat{f}_{\ell,\theta} = \mathcal{A}_\ell(S_\ell, \theta)$. Note that if the algorithm estimates the predictor from a hypothesis space $\mathcal{H}_{\ell,\theta}$ by empirical risk minimization, then we have $\hat{f}_{\ell,\theta} \in \mathcal{H}_{\ell,\theta}$.

Assume further that there is a procedure that estimates the performance of the learned predictor $\hat{f}_{\ell,\theta}$ using possibly additional information T_ℓ (which for example, could be a validation set) as $\mathcal{O}_\ell(S_\ell, T_\ell, \theta)$. Then in structural learning, we find $\hat{\theta}$ by using a regularized estimator

$$\hat{\theta} = \arg\min_{\theta\in\Gamma} \left[r(\theta) + \sum_{\ell=1}^{m} \mathcal{O}_\ell(S_\ell, T_\ell, \theta) \right], \tag{2}$$

where $r(\theta)$ is a regularization parameter that encodes our belief on what θ value is preferred. The number of problems m behaves like the sample size in standard learning. This is our fundamental estimation method for structural learning. Once we obtain an estimate $\hat{\theta}$ of the structural parameter, we can use the learning algorithm $\mathcal{A}_\ell(S_\ell, \hat{\theta})$ to obtain predictor $\hat{f}_{\ell,\theta}$ for each ℓ.

As an example, assume that we estimate the accuracy of $\hat{f}_{\ell,\theta}$ using a validation set $T_\ell = \{(\bar{\mathbf{X}}_j^\ell, \bar{Y}_j^\ell)\}_{j=1,\ldots,\bar{n}_\ell}$, then we may simply let $\mathcal{O}(S_\ell, T_\ell, \theta) = \alpha_\ell \sum_{j=1}^{\bar{n}_\ell} L(\hat{f}_{\ell,\theta}(\bar{\mathbf{X}}_j^\ell), \bar{Y}_j^\ell)$, where $\alpha_\ell > 0$ are weighting parameters. It is also possible to estimate the accuracy of the learned predictor based on the training set alone using the standard learning theory for empirical risk minimization. This approach will be employed in Section 3, and leads to practical algorithms that can be formulated as optimization problems.

3. Algorithms

In this section, we develop a specific learning algorithm under the standard machine learning framework. The basis of our learner is *joint empirical risk minimization*, which will be analyzed in Appendix A. We consider linear prediction models since they have been shown to be effective in many practical applications. These methods include state-of-the-art machine learning algorithms such as kernel machines and boosting.

3.1 Joint empirical risk minimization

Based on the framework outlined in Section 2, we are interested in finding a hypothesis space $\mathcal{H}_{\cdot,\theta}$, using an estimator of the form (2). As being pointed out in Section 2, conceptually this could be achieved using a validation set. However, such an approach can lead to a quite difficult computational procedure since we have to optimize the empirical risk on the training data for each possible value of θ, and then choose an optimal θ on the validation set. Therefore for complicated structures with continuous θ parameter such as the model we consider in Section 3, this approach is not feasible.

A more natural method is to perform a joint optimization on the training set, with respect to both the predictors $\{f_\ell\}$, and the structural parameter θ. To this end, we will consider the model given by equation (1), and pose it as a joint optimization problem over

the m problems, where θ is the shared structural parameter:

$$[\hat{\theta}, \{\hat{f}_\ell\}] = \operatorname*{arg\,min}_{\theta\in\Gamma, \{f_\ell\in\mathcal{H}_{\ell,\theta}\}} \sum_{\ell=1}^{m} \frac{1}{n_\ell} \sum_{i=1}^{n_\ell} L(f_\ell(\mathbf{X}_i^\ell), Y_i^\ell). \tag{3}$$

Since the shared structural parameter θ depends on m problems, it can be more reliably estimated by joint minimization. For completeness, we include a theoretical analysis in Appendix A.

3.2 Structural learning with linear predictors

In order to derive a practical algorithm from (3), we shall consider a specific joint model which can be solved numerically. Specifically, we employ linear prediction models for the multiple tasks, and assume that the underlying structure is a shared low-dimensional subspace. Although not necessarily most general, this model leads to a simple and intuitive computational procedure. As we shall also see later, it is quite effective for semi-supervised learning problems that we are interested in.

Given the input space $\mathcal{X}$, a linear predictor is not necessarily linear on the original space, but rather can be regarded as a linear functional on a high dimensional feature space $\mathcal{F}$. We assume there is a known feature map $\mathbf{\Phi} : \mathcal{X} \to \mathcal{F}$. A linear predictor f is determined by a weight vector $\mathbf{w}$: $f(\mathbf{x}) = \mathbf{w}^T\mathbf{\Phi}(\mathbf{x})$. In order to apply the structural learning framework, we consider a parameterized family of feature maps. In this setting, the goal of structural learning may be regarded as learning a good feature map. For the specific formulation which we consider in this paper, we assume that the overall feature map contains two components: one component is with a known high-dimensional feature map, and the other component is a parameterized low-dimensional feature map. That is, the linear predictor has a form

$$f(\mathbf{x}) = \mathbf{w}^T\mathbf{\Phi}(\mathbf{x}) + \mathbf{v}^T\mathbf{\Psi}_\theta(\mathbf{x}),$$

where $\mathbf{w}$ and $\mathbf{v}$ are weight vectors specific for each prediction problem, and θ is the common structure parameter shared by all problems.

In order to simplify numerical computation, we further consider a simple linear form of feature map, where $\theta = \Theta$ is an $h \times p$ dimensional matrix, and $\mathbf{\Psi}_\theta(\mathbf{x}) = \Theta\mathbf{\Psi}(\mathbf{x})$, with $\mathbf{\Psi}$ a known p-dimensional vector function. We now can write the linear predictor as:

$$f_\Theta(\mathbf{w}, \mathbf{v}; \mathbf{x}) = \mathbf{w}^T\mathbf{\Phi}(\mathbf{x}) + \mathbf{v}^T\Theta\mathbf{\Psi}(\mathbf{x}).$$

This hypothesis space (with appropriate regularization conditions) is analyzed in Appendix A after Theorem 4. We point out there that the key idea of this formulation is to discover a shared low-dimensional predictive structure parameterized by Θ.

Applying (2) with $\mathcal{C}(S_\ell, T_\ell, \theta)$ given by regularized empirical risk, we obtain the following formulation:

$$[\{\hat{\mathbf{w}}_\ell, \hat{\mathbf{v}}_\ell\}, \hat{\Theta}] = \operatorname*{arg\,min}_{\{\mathbf{w}_\ell, \mathbf{v}_\ell\}, \Theta} \left[r(\Theta) + \sum_{\ell=1}^{m} \left(g(\mathbf{w}_\ell, \mathbf{v}_\ell) + \frac{1}{n_\ell} \sum_{i=1}^{n_\ell} L(f_\Theta(\mathbf{w}_\ell, \mathbf{v}_\ell; \mathbf{X}_i^\ell), Y_i^\ell) \right) \right], \tag{4}$$

where $g(\mathbf{w}, \mathbf{v})$ is an appropriate regularization condition on the weight vector $(\mathbf{w}, \mathbf{v})$, and $r(\Theta)$ is an appropriate regularization condition on the structural parameter Θ. In this formulation, we weight each problem equally (by dividing the number of instances n_ℓ) so that

no problem will dominate the others. One may also choose other weighting schemes. Note that the regularized ERM method in (4) has the same form as (3). The main difference is that we replaced the hard-constrained regularization (picking the predictors from a hypothesis space) by its computationally more convenient version of penalized regularization. Up to appropriately defined Lagrangian multipliers, these two formulations are equivalent.

If we consider kernel learning, and assume that the feature map $\mathbf{\Phi}(\mathbf{x})$ belongs to a reproducing kernel Hilbert space, then equation (4) can be kernelized. There are several ways to do so. One possibility is to kernelize in the $\mathbf{w}$ parameter — we simply replace the vector parameter $\mathbf{w}_\ell$ by n_ℓ dual parameters α_j^ℓ $(j = 1, \ldots, n_\ell)$, and the linear score $\mathbf{w}_\ell^T \mathbf{\Phi}(\mathbf{X}_i^\ell)$ by $\sum_{j=1}^{n_\ell} \alpha_j^\ell K(\mathbf{X}_j^\ell, \mathbf{X}_i^\ell)$. For simplicity, we do not consider kernel methods in this paper.

3.3 Alternating structure optimization

It is possible to solve (4) using general purpose optimization methods. However, in this section, we show that by exploring the special structure of the formulation, we can develop a more interesting and conceptually appealing computational procedure. In general, we should pick L and g such that the formulation is convex for fixed Θ. However, the joint optimization over $\{\mathbf{w}_\ell, \mathbf{v}_\ell\}$ and Θ will become non-convex. Therefore, one typically can only find a local minimum with respect to Θ. This usually doesn't lead to serious problems since given the local optimal structural parameter Θ, the solution $\{\mathbf{w}_\ell, \mathbf{v}_\ell\}$ will still be globally optimal for every ℓ. Moreover, the algorithm which we propose later in section uses SVD for dimension reduction. At the conceptual level, the possible local optimality of Θ is not a major issue simply because the SVD procedure itself is already good at finding globally optimal low dimensional structure.

With fixed Θ, the computation of $\{\mathbf{w}_\ell, \mathbf{v}_\ell\}$ for each problem ℓ becomes decoupled, and various optimization algorithms can be applied for this purpose. The specific choice of such algorithms is not important for the purpose of this paper. In our experiments, for convenience and simplicity, we employ stochastic gradient descent (SGD), widely used in the neural networks literature. It was recently argued that this simple method can also work well for large scale convex learning formulations (Zhang, 2004).

In the following, we consider a special case of (4) which has a simple iterative SVD solution. Let $\mathbf{\Phi}(\mathbf{x}) = \mathbf{\Psi}(\mathbf{x}) = \mathbf{x} \in R^p$ with square regularization of weight vectors. Then we have

$$[\{\hat{\mathbf{w}}_\ell, \hat{\mathbf{v}}_\ell\}, \hat{\Theta}] = \arg\min_{\{\mathbf{w}_\ell, \mathbf{v}_\ell\}, \Theta} \sum_{\ell=1}^{m} \left(\frac{1}{n_\ell} \sum_{i=1}^{n_\ell} L((\mathbf{w}_\ell + \Theta^T \mathbf{v}_\ell)^T \mathbf{X}_i^\ell, Y_i^\ell) + \lambda_\ell \|\mathbf{w}_\ell\|_2^2 \right), \quad (5)$$
$$\text{s.t.} \quad \Theta\Theta^T = I_{h\times h} \ ,$$

with given constants $\{\lambda_\ell\}$. Note that in this formulation, the regularization condition $r(\Theta)$ in (4) is absorbed into the orthonormal constraint $\Theta\Theta^T = I_{h\times h}$, and thus does not need to be explicitly included.

In order to solve this optimization problem, we may introduce an auxiliary variable $\mathbf{u}_\ell$ for each problem ℓ such that $\mathbf{u}_\ell = \mathbf{w}_\ell + \Theta^T \mathbf{v}_\ell$. Therefore we may eliminate $\mathbf{w}$ using $\mathbf{u}$ to

obtain:

$$[\{\hat{\mathbf{u}}_\ell, \hat{\mathbf{v}}_\ell\}, \hat{\Theta}] = \arg\min_{\{\mathbf{u}_\ell, \mathbf{v}_\ell\}, \Theta} \sum_{\ell=1}^{m} \left(\frac{1}{n_\ell} \sum_{i=1}^{n_\ell} L(\mathbf{u}_\ell^T \mathbf{X}_i^\ell, Y_i^\ell) + \lambda_\ell \|\mathbf{u}_\ell - \Theta^T \mathbf{v}_\ell\|_2^2 \right), \quad \text{s.t.} \quad \Theta\Theta^T = I_{h\times h} \ . \tag{6}$$

At the optimal solution, we let $\hat{\mathbf{w}}_\ell = \hat{\mathbf{u}}_\ell - \hat{\Theta}^T \hat{\mathbf{v}}_\ell$.

In order to solve (6), we use the following alternating optimization procedure:

- Fix $(\Theta, \mathbf{v})$, and optimize (6) with respect to $\mathbf{u}$.
- Fix $\mathbf{u}$, and optimize (6) with respect to $(\Theta, \mathbf{v})$.
- Iterate until convergence.

One may also propose other alternating optimization procedures. For example, in the first step, we may fix Θ and optimize with respect to $(\mathbf{u}, \mathbf{v})$.

In the alternating optimization procedure outlined above, with a convex choice of L, the first step becomes a convex optimization problem. There are many well-established methods for solving it (as mentioned earlier, we use SGD for its simplicity). We shall focus on the second step, which is crucial for the derivation of our method. It is easy to see that the optimization of (6) with fixed $\{\mathbf{u}_\ell\} = \{\hat{\mathbf{u}}_\ell\}$ is equivalent to the following problem:

$$[\{\hat{\mathbf{v}}_\ell\}, \hat{\Theta}] = \arg\min_{\{\mathbf{v}_\ell\}, \Theta} \sum_{\ell} \lambda_\ell \|\hat{\mathbf{u}}_\ell - \Theta^T \mathbf{v}_\ell\|_2^2, \quad \text{s.t.} \quad \Theta\Theta^T = I_{h\times h}. \tag{7}$$

Using simple linear algebra, we know that with fixed Θ,

$$\min_{\mathbf{v}_\ell} \|\hat{\mathbf{u}}_\ell - \Theta^T \mathbf{v}_\ell\|_2^2 = \|\hat{\mathbf{u}}_\ell\|_2^2 - \|\Theta\hat{\mathbf{u}}_\ell\|_2^2,$$

and the optimal value is achieved at $\hat{\mathbf{v}}_\ell = \Theta\hat{\mathbf{u}}_\ell$. Now by eliminating $\mathbf{v}_\ell$ and use the above equality, we can rewrite (7) as

$$\hat{\Theta} = \arg\max_{\Theta} \sum_{\ell=1}^{m} \lambda_\ell \|\Theta\hat{\mathbf{u}}_\ell\|_2^2, \quad \text{s.t.} \quad \Theta\Theta^T = I_{h\times h}.$$

Let $\mathbf{U} = [\sqrt{\lambda_1}\hat{\mathbf{u}}_1, \ldots, \sqrt{\lambda_m}\hat{\mathbf{u}}_m]$ be an $p \times m$ matrix, we have

$$\hat{\Theta} = \arg\max_{\Theta} \ \mathrm{tr}(\Theta\mathbf{U}\mathbf{U}^T\Theta^T), \quad \text{s.t.} \quad \Theta\Theta^T = I_{h\times h},$$

where $\mathrm{tr}(A)$ is the trace of matrix A. It is well-known that the solution of this problem is given by the SVD (singular value decomposition) of $\mathbf{U}$: let $\mathbf{U} = V_1 D V_2^T$ be the SVD of $\mathbf{U}$ (assume that the diagonal elements of D are arranged in decreasing order), then the rows of $\hat{\Theta}$ are given by the first h rows of V_1^T (left singular vectors corresponding to the largest h singular values of $\mathbf{U}$). We now summarize the above derivation into an algorithm described in Figure 2, which solves (5) by alternating optimization of $\mathbf{u}$ and $(\Theta, \mathbf{v})$.

Note that since the objective value in (5) decreases at each iteration, the procedure produces parameters $\{\hat{\mathbf{w}}_\ell, \hat{\mathbf{v}}_\ell\}, \hat{\Theta}$ with converging objective values. In general, the parameters

Input: training data $\{(\mathbf{X}_i^\ell, Y_i^\ell)\}$ $(\ell = 1, \ldots, m)$
Parameters: h and $\lambda_1, \ldots, \lambda_m$
Output: $h \times p$ dimensional matrix Θ
Initialize: $\mathbf{u}_\ell = 0$ $(\ell = 1 \ldots m)$, and arbitrary Θ
iterate
  for $\ell = 1$ to m **do**
    With fixed Θ and $\mathbf{v}_\ell = \Theta \mathbf{u}_\ell$, approximately solve for $\hat{\mathbf{w}}_\ell$:
      $\hat{\mathbf{w}}_\ell = \arg\min_{\mathbf{w}_\ell} \left[\frac{1}{n_\ell} \sum_{i=1}^{n_\ell} L(\mathbf{w}_\ell^T \mathbf{X}_i^\ell + (\mathbf{v}_\ell^T \Theta) \mathbf{X}_i^\ell, Y_i^\ell) + \lambda_\ell \|\mathbf{w}_\ell\|_2^2 \right]$
    Let $\mathbf{u}_\ell = \hat{\mathbf{w}}_\ell + \Theta^T \mathbf{v}_\ell$
  endfor
  Compute the SVD of $\mathbf{U} = [\sqrt{\lambda_1}\mathbf{u}_1, \ldots, \sqrt{\lambda_m}\mathbf{u}_m]$:
    $\mathbf{U} = V_1 D V_2^T$ (with diagonals of D in descending order)
  Let the rows of Θ be the first h rows of V_1^T
until converge

Figure 2: SVD-based Alternating Structure Optimization Algorithm

$\{\hat{\mathbf{w}}_\ell, \hat{\mathbf{v}}_\ell\}, \hat{\Theta}$ will also converge (to a local optimal solution). However, in reality, it is usually sufficient to use the Θ parameter from the first iteration of the procedure. This is because the performance of our model is not very sensitive to a small perturbation of the structural parameter Θ. The main dimensional reduction effect is already well captured by SVD in the first iteration.

It is important to point out that our SVD-based alternating structure optimization (SVD-ASO) procedure is fundamentally different from the usual principal component analysis (PCA) which can be regarded as dimension reduction in the data space $\mathcal{X}$. However, the dimension reduction performed in the SVD-ASO algorithm is on the predictor (classifier) space instead of the data space. This is possible because we observe multiple predictors from multiple learning tasks. If we regard the observed predictors as sample points of the predictor distribution in the predictor space (corrupted with estimation error, or noise), then our algorithm can be interpreted as finding the "principal components" of these predictors. Consequently the method directly looks for low-dimensional structures with the highest predictive power. By contrast, the principal components of input data in the data space do not necessarily have good predictive power.

3.4 An extension of the basic SVD-ASO algorithm

One may extend (5) and the SVD-ASO procedure in various ways. For example, if $\mathbf{x}$ belongs to an infinite dimensional Hilbert space, then the SVD in Figure 2 can be replaced by the corresponding kernel principal component analysis. However, this generalization is outside the scope of the analysis given in the paper.

In our experiments, we use another extension, where features (components of $\mathbf{x}$) are grouped into different types and the SVD dimension reduction is computed separately for each group. This is important since in applications, features are not homogeneous. If we

know that some features are more similar to each other (e.g. they have the same type), then it is reasonable to perform a more localized dimension reduction among these similar features. To formulate this idea, we divide input features into G groups, and rewrite each input data-point $\mathbf{X}_i^\ell$ as $[\mathbf{X}_{i,t}^\ell]_{t=1,\ldots,G}$, where t is the feature type which specifies which group the feature is in. Each $\mathbf{X}_{i,t}^\ell \in R^{p_t}$, and thus $\mathbf{X}_i^\ell \in R^p$ with $p = \sum_{t=1}^G p_t$. We associate each group t with a structural parameter $\Theta_t \in R^{h_t \times p_t}$, which is a projection operator of this feature type into h_t dimensional space. Equation (5) can be replaced by the following structural learning method:

$$[\{\hat{\mathbf{w}}_{\ell,t}, \hat{\mathbf{v}}_{\ell,t}\}, \{\hat{\Theta}_t\}] = \underset{\{\mathbf{w}_{\ell,t}, \mathbf{v}_{\ell,t}\}\{\Theta_t\}}{\arg\min} \sum_{\ell=1}^{m} \left(\frac{1}{n_\ell} \sum_{i=1}^{n_\ell} L(\sum_{t=1}^{G} (\mathbf{w}_{\ell,t} + \Theta_t^T \mathbf{v}_{\ell,t})^T \mathbf{X}_{i,t}^\ell, Y_i^\ell) + \sum_{t=1}^{G} \lambda_{\ell,t} \|\mathbf{w}_{\ell,t}\|_2^2 \right), \quad \text{s.t. } \forall t \in \{1,\ldots,G\}: \quad \Theta_t \Theta_t^T = I_{h_t \times h_t}. \tag{8}$$

Similarly as before, we can introduce auxiliary variables $\mathbf{u}_{\ell,t} = \mathbf{w}_{\ell,t} + \Theta_t^T \mathbf{v}_{\ell,t}$, and perform alternating optimization over $\mathbf{u}$ and $(\Theta, \mathbf{v})$. The resulting algorithm is essentially the same as the SVD-ASO method in Figure 2, but with the SVD dimension reduction step performed separately for each feature group t.

Some other extensions of the basic algorithm can also be useful for certain applications. For example, we may choose to regularize only those components of $\mathbf{w}_\ell$ which correspond to the non-negative part of $\mathbf{u}_\ell$ (we may still regularize the negative part of $\mathbf{u}_\ell$, but using the corresponding components of $\mathbf{u}_\ell$ instead of $\mathbf{w}_\ell$). The reason for doing so is that the positive weights of a linear classifier are usually directly related to the target concept, while the negative components often yield much less specific information. The resulting method can be easily formulated and solved by a variant of the basic SVD-ASO algorithm. In effect, in the SVD computation, we only use the positive components of $\mathbf{u}_\ell$.

4. Semi-supervised Learning

We are now ready to illustrate how to apply the structural learning paradigm developed earlier in the paper to the semi-supervised learning setting. The basic idea is to create auxiliary problems using unlabeled data, and then employ structural learning to reveal predictive structures intrinsic to the underlying input space $\mathcal{X}$.

4.1 Learning from unlabeled data through structural learning

We systematically create multiple prediction problems from unlabeled data. We call these *created* prediction problems *auxiliary problems*, while we call the original supervised prediction problem (which we are interested in) the *target problem*.

Our method consists of the following two steps:

1. Learn a good structural parameter θ by performing a joint empirical risk minimization on the auxiliary problems, using originally unlabeled data that are automatically 'labeled' with auxiliary class labels.

2. Learn a predictor for the target problem by empirical risk minimization on the originally labeled data, using θ computed in the first step. In particular, in our bi-linear formulation (Section 3), we fix Θ and optimize (8) with respect to $\mathbf{w}$ and $\mathbf{v}$ for the target problem.

The first step seeks a hypothesis space $\mathcal{H}_\theta$ through learning the predictive functional structure shared by auxiliary predictors. If auxiliary problems are, to some degree, related to the target task, then the obtained hypothesis space $\mathcal{H}_\theta$, which improves the average performance of auxiliary predictors, will also help the target problem. Therefore, the relevancy of auxiliary problems plays an important role in our method. We will return to this issue in the next section.

An alternative to the above two-step procedure is to perform a joint empirical risk minimization on the target problem (with labeled data) and on the auxiliary problems (with unlabeled data) at once. However, in our intended applications, the number of labeled data available for the target problem is usually small. Therefore the inclusion of the target predictor in the joint ERM will not have a significant impact.

4.2 Auxiliary problem creation

Our approach to semi-supervised learning requires auxiliary problems with the following characteristics:

- *Automatic labeling*: we need to automatically generate various "labeled" data for the auxiliary problems from unlabeled data.
- *Relevancy*: auxiliary problems should be related to the target problem (that is, they share a certain predictive structure) to some degree.

We consider two strategies for automatic generation of auxiliary problems: one in a completely unsupervised fashion, and the other in a partially supervised fashion. Some of the example auxiliary problems introduced in this section are used in our experiments described in Section 5.

We have briefly discussed the relationship of PCA and SVD-ASO in Section 3. In the above mentioned framework of semi-supervised learning, the standard PCA (applied to unlabeled data) can also be roughly regarded as a result of generating k auxiliary problems from k unlabeled data points so that the i-th problem has only one positive example (the i-th data point). In general, the strategies which we will suggest below are more flexible and more effective.

For clarity, we introduce the following two mini target tasks as running examples.

Text genre categorization Consider the task of assigning one of the three categories in { SCIENCE, SPORTS, ECONOMY } to text documents. For this problem, suppose that we use frequencies of content words as features.

Word tagging Consider the task of assigning one of the three part-of-speech tags { NOUN, VERB, OTHER } to words in English sentences. For instance, the word "test" in "... a test procedure ..." should be assigned the tag NOUN, and that in "We will test it ..." should be assigned the tag VERB. For this problem, suppose that we use strings of the current and surrounding words as features.

4.2.1 Unsupervised-strategy: predicting observable sub-structures

In the first strategy, we regard some observable substructures of the input data $\mathcal{X}$ as auxiliary class labels, and try to predict these labels using other parts of the input data. For instance, for the word tagging task mentioned above, at each word position, we can create auxiliary problems by regarding the current word as auxiliary labels, which we want to predict using the surrounding words. We create one binary classification problem for each possible word value, and hence can obtain many auxiliary problems using this idea.

More generally, if we have a feature representation of the input data, then we may mask some features as unobserved, and learn classifiers to predict these 'masked' features (or some functional mapping of the masked features, e.g., bi-grams of left and current words) based on other features that are not masked. In the actual implementation, we just replace the masked feature values by zero, which has the same effect.

The automatic-labeling requirement is satisfied since the auxiliary labels are observable to us. To see why this technique may naturally meet the relevancy requirement, we note that feature components that can predict a certain masked feature are correlated to the masked feature, and thus are correlated among themselves. Therefore this technique helps us to identify correlated features with predictive power.

However, for optimal performance, it is clear that we should choose to mask (and predict) features that have good correlation to the target classes as auxiliary labels. The creation of auxiliary problems following this strategy is thus as easy or hard as designing features for usual supervised learning tasks. We can often make an educated guess based on task-specific knowledge. A wrong guess would result in adding some irrelevant features (originating from irrelevant θ-components), but it would not hurt ERM learners severely. On the other hand, potential gains from a right guess can be significant. Also note that given the abundance of unlabeled data, we have a wider range of choices than standard feature engineering in the supervised setting. For example, high-order features that suffer from the data sparseness problem in the supervised setting may be used in auxiliary problems due to the vast amount of unlabeled data that can provide more reliable statistics. The low-dimensional predictive structure discovered from the high-order features can then be used in the supervised task without causing the data-sparseness problem. This is because the rare features will be properly combined in the projection matrix Θ, so that the combined low-dimension directions will appear more frequently (and more correlated to the class-label). The example provided in Section 4.3 demonstrates this point.

The following examples illustrate auxiliary problems potentially useful for our example mini tasks.

Ex 1. Predict frequent words for text genre categorization. It is intuitive that content words that occur frequently in a document are often good indicators of the genre of that document. Let us split content words into two sets W_1 and W_2 (after removing appropriate stop words). An auxiliary task we define is as follows. Given document x, predict the word that occurs most frequently in x, among the words in set W_1. The learner only uses the words in W_2 for this prediction. This task breaks down to $|W_1|$ binary prediction problems, one for each content word in W_1.[1]

1. One may also consider variations of this idea, such as predicting whether a content word in W_1 appears more often than a certain threshold, or in the top-k most frequent list of x.

For example, let

$$W_1 = \{\text{“stadium”, “scientist”, “stock”}\},$$
$$W_2 = \{\text{“baseball”, “basketball”, “physics”, “market”}\}.$$

We treat members of W_1 as unobserved, and learn to predict whether the word “stadium” occurs more frequently in a document than “scientist” and “stock” by observing the occurrences of “baseball”, “basketball”, “physics”, and “market”. Similarly, the second problem is to predict whether “scientist” is more frequent than “baseball” and “stock”. Essentially, through this auxiliary problem, we learn the textual context in W_2 that implies that the word “stadium” occurs frequently in W_1. Assuming that “stadium” is a strong indicator of SPORTS, the problem indirectly helps to learn the correlation of W_2 members to the target class SPORTS from unlabeled data.

Ex 2. Predict word strings for word tagging. As we have already discussed above, an example auxiliary task for word tagging is to predict the word string at the current position by observing the corresponding words on the left and the right. Using this idea, we can obtain $|W|$ binary prediction problems where W is a set of all possible word strings. Another example is to predict the word on the left by observing the words at the current and right positions. The underlying assumption is that word strings (at the current and left positions) have strong correlations to the target problem – whether a word is NOUN or VERB.

4.2.2 Partially supervised-strategy: predicting the behavior of target classifier

The second strategy is motivated by co-training. We use two (or more) distinct feature maps: $\mathbf{\Phi}_1 : \mathcal{X} \to \mathcal{F}$ and $\mathbf{\Phi}_2 : \mathcal{X} \to \mathcal{F}$. First, we train a classifier for the target task, using the feature map $\mathbf{\Phi}_1$ and the labeled data. The auxiliary tasks are to predict the behavior of this classifier (such as predicted labels, assigned confidence values, and so forth), by using the other feature map $\mathbf{\Phi}_2$. Note that unlike co-training, we only use the classifier as a means of creating auxiliary problems that meet the relevancy requirement, instead of using it to bootstrap labels. The semi-supervised learning procedure following this strategy is summarized as follows.

1. Train a classifier T_1 with labeled data Z for the target task, using feature map $\mathbf{\Phi}_1$.
2. Generate labeled data for auxiliary problems by applying T_1 to unlabeled data.
3. Learn structural parameter θ by performing joint ERM on the auxiliary problems, using only the feature map $\mathbf{\Phi}_2$.
4. Train a final classifier with labeled data Z, using θ computed above and some appropriate feature map $\mathbf{\Psi}$.

Ex 3. Predict the prediction of classifier T_1. The simplest auxiliary task created by following this strategy is the prediction of the class labels proposed by classifier T_1. When the target task is c-way classification, c binary classification problems are obtained in this

manner. For example, suppose that we train a classifier using only one half of content words for the text genre categorization task. Then, one of auxiliary problems will be to predict whether this classifier would propose SPORTS category or not, based on the other half of content words only.

Ex 4. Predict the top-k choices of the classifier. Another example is to predict the combinations of k (a few) classes to which T_1 assigns the highest confidence values. Through this problem, fine-grained distinctions (related to intrinsic sub-classes of target classes) can be learned. From a c-way classification problem, $c!/(c-k)!$ binary prediction problems can be created. For instance, predict whether T_1 assigns the highest confidence values to SPORTS and ECONOMY in this order.

Ex 5. Predict the range of confidence values produced by the classifier. Yet another example is to predict the proposed labels in conjunction with the range of confidence values. For instance, predict whether T_1 would propose SPORTS with confidence greater than 0.5.

4.3 Discussions

We have introduced two strategies for creating auxiliary problems in this section. One is unsupervised, and the other partially supervised.

For the unsupervised strategy, we try to predict some sub-structures of the input using some other parts of the input. This idea is related to the discussion in Section 2.3, where we have argued that there are often good structures (or smoothness conditions) intrinsic to the input space. These structures can be discovered from auxiliary problems. For text data, some words or linguistic usages will have similar meanings. The smoothness condition is related to the fact that interesting predictors for text data (often associated with some underlying semantic meanings) will take similar values when a linguistic usage is substituted by one that is closely related. This smoothness structure can be discovered using structural learning, and specifically by the method we proposed in Section 3. In this case, the space of smooth predictors corresponds to the most predictive low dimensional directions which we may discover using the SVD-ASO algorithm. An example of computed Θ is given in Section 5.2.8, which supports the argument. It is also easy to see that this reasoning is not specific to text. Therefore the idea can be applied to other data such as images. In the following, we will briefly explain the underlying intuition on why the un-supervised auxiliary problems we create are helpful for the supervised task, and leave the development of a more rigorous and general theory to future investigation.

Suppose we split the features into two parts W_1 and W_2, and then predict W_1 based on W_2. Suppose features in W_1 are correlated to the class labels (but not necessarily correlated among themselves). Then, the auxiliary prediction problems are related to the target task, and thus can reveal useful structures of W_2. Under appropriate conditions, features in W_2 with similar predictive performance tend to map to similar low-dimensional vectors through Θ. This effect can be empirically observed in Section 5.2.8. We shall only use a simple but concrete example to illustrate the main idea. Assume that words are divided into five disjoint sets $T_j : -2 \leq j \leq 2$, with binary label $y \in \{-1, 1\}$. Assume also for simplicity that every document contains only two words x_1 and x_2, where $x_1 \in W_1 = \cup_{j=-1,0,1} T_j$,

and $x_2 \in W_2 = \cup_{j=-2,2} T_j$. Assume further that given class label $y = \pm 1$, x_1 and x_2 are independent, where x_1 is uniformly distributed over $T_0 \cup T_y$ and x_2 is uniformly distributed over T_{2y}. Then by predicting $x_1 = \ell$ based on x_2 with least squares, we obtain for each $y \in \{\pm 1\}$, identical weight-components for all words in T_{2y} (due to the exchangeability of words in each T_{2y}). Thus after dimension reduction, only two rows of Θ have non-zero singular values. The Θ-feature reduces all words in T_2 to a single vector in R^2, and all words in T_{-2} into another single vector in R^2. This gives a helpful grouping effect (words with similar predictive performance are grouped together). It is clear that in this example, we gain predictive ability by using unsupervised structure discovery. This is because the original word-spaces T_2 and T_{-2} may be extremely large, which means that one will not be able to learn very well from a small number of training examples (since each word does not occur often enough). By grouping all words in T_2 (also in T_{-2}) together, we obtain a feature that is completely correlated to the class label due to our data generation process. Therefore the grouping effect makes the originally hard problem much easier to learn. This example can be extended to a more general theory, which we shall leave to further exploration. The consequence of the example is observable in practice, as demonstrated in Section 5.2.8.

Although the above discussion implies that it is possible to find useful predictive structures even if we do not intentionally create problems to mimic the target problem, it is also clear that auxiliary problems more closely related to the target problem will be more beneficial. This is our motivation to propose the partially supervised strategy for creating auxiliary problems. Using this idea, it is always possible to create relevant auxiliary problems that are closely related to the supervised problem without knowing the effectiveness of the individual features. In practical applications, we observe that it can be desirable to create as many auxiliary problems as possible, as long as there is some reason to believe in their relevancy to the target task. This is because the risk is relatively minor, while the potential gain from a good structure is large.

Moreover, the auxiliary problems introduced above (and used in our experiments of the next section) are merely possible examples. One advantage of this approach to semi-supervised learning is that one may design a wide variety of auxiliary problems for learning various aspects of the target problem from unlabeled data. Structural learning provides a theoretical foundation and a general framework for carrying out possible new ideas.

5. Experiments

We study the performance of our structural learning-based semi-supervised method on text categorization, natural language tagging/chunking, and image classification tasks. The experimental results show that we are able to improve state-of-the-art supervised learning methods even for some problems with relatively large number of labeled data (e.g. 200K labeled data for named entity recognition).

5.1 Implementation

We experiment with the following semi-supervised learning procedure:

1. If required for auxiliary label generation, train classifiers T_i using labeled data Z and appropriate feature maps $\mathbf{\Psi}_i$.

2. For all the auxiliary problems, assign auxiliary labels to unlabeled data.

3. Compute structure matrix Θ by performing the SVD-ASO procedure (Section 3) using all auxiliary problems on the data generated above. We use the extended version to take advantage of natural feature splits, and iterate once.

4. Fix Θ and obtain the final classifier by optimizing (8) with respect to $\mathbf{w}$ and $\mathbf{v}$, using labeled data Z.

In all settings (including baseline methods), the loss function is a modification of the Huber's robust loss for regression:

$$L(p, y) = \begin{cases} \max(0, 1 - py)^2 & \text{if } py \geq -1 \\ -4py & \text{otherwise} \end{cases},$$

with square regularization ($\lambda = 10^{-4}$). It is known that the modified Huber loss works well for classification, and has some advantages, although one may select other loss functions such as SVM or logistic regression. The specific choice is not important for the purpose of this paper. The training algorithm is stochastic gradient descent (SGD) as in (Zhang, 2004). We fix h_t (dimension of Θ_t) to 50, and use it for all the settings unless otherwise specified.

5.2 Text categorization experiments

We report text categorization performance on the 20-newsgroup corpus and the Reuters-RCV1 corpus (also known as "new Reuters").

5.2.1 Feature representation

Our feature representation uses word frequencies after removing function words and common stopwords, and normalizes feature vectors into unit vectors.

5.2.2 Auxiliary problems for text categorization

We experiment with the following types of auxiliary problems:

- Freq: predicts the most frequent word by observing one half of the words (as in Section 4.2.1. Ex 1).

- Top-k: predicts combinations of the top-k choices of the classifier trained with labeled data (as in Section 4.2.2. Ex 4).

- Multi-k: for the multi-category target task, predicts the top-k choices of the classifier (trained with labeled data), regarding them as multi-category auxiliary labels. The number k is set to the average number of categories per instance, obtained from the labeled data.

Feature splits are randomly generated.

5.2.3 Data

20-newsgroup corpus The 20-newsgroup corpus is one of the standard data sets for text categorization, which consists of 20K documents from 20 newsgroups, with 1K documents per group. The task is to classify documents into these 20 newsgroups ranging over a variety of topics – computer hardware, baseball, bikes, religions, middle east issues, and so on. In pre-processing, we removed the header lines (subjects, newsgroup names, senders, and so forth) from all documents. We held out 1K documents as the test set, and arbitrarily split the rest of the corpus into the training set (2K documents) and the unlabeled data set (17K documents).

Reuters-RCV1 corpus (new Reuters) From the Reuters-RCV1 corpus, we randomly generate disjoint sets of labeled (1K), unlabeled (20K), and test (3K) examples. The Reuters-RCV1 corpus differs from the 20-newsgroup corpus in several ways. The number of categories is 102, which is five times larger than that of the 20-newsgroup corpus; the categories are organized into three-level hierarchies; each document may be assigned multiple categories — about three categories per document on average. The Reuters-RCV1 corpus preserves the natural distribution of the categories whereas the 20-newsgroup corpus has a completely uniform distribution, generated by intentionally choosing the same number of documents from each newsgroup.

5.2.4 Evaluation metric

To measure the performance of the final classifier on the test sets, for singly-labeled tasks, we choose one category that produces the highest confidence value (inner product) and report classification accuracy. For multiply-labeled tasks, we choose categories that produce positive confidence values, and report the micro-averaged F-measure.

5.2.5 Text categorization performance results

20-newsgroup results Figure 3 (a) shows the accuracy results on the 20-newsgroup data in comparison with the supervised setting as the baseline. We show the averaged results over 10 runs, each with labeled examples randomly drawn from the training set. The vertical bars are 'one' standard deviations. The symbol 'semi' stands for semi-supervised, followed by the types of auxiliary problems used. The semi-supervised methods obtain significant performance improvements (up to 22.2%) over the supervised method in all settings.

Reuters-RCV1 results Figure 3 (b) shows micro-averaged F-measure on the Reuters-RCV1 data in comparison to the supervised baseline. The performance trend is similar to that of the 20-newsgroup experiments. Significant performance improvements (up to 11.6%) over the supervised method are obtained in all settings.

Auxiliary problems: unsupervised vs. partially-supervised From the results in Figure 3, we observe that when a relatively small number of labeled data are used, freq (which uses auxiliary problems created in an unsupervised manner) outperforms top-k/multi-k (partially-supervised). However it underperforms top-k/multi-k when a relatively large number of labeled data are used. Since freq learns from unlabeled data in an unsupervised fashion, its effectiveness is insensitive to the number of labeled data. In contrast,

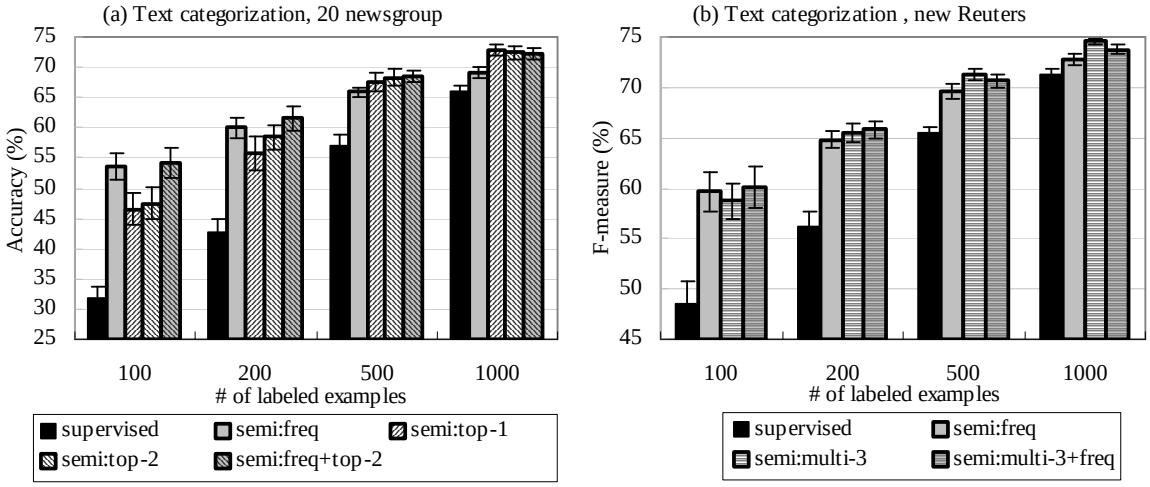

Figure 3: Text categorization performance results. Average over 10 runs. Vertical bars are standard deviations. (a) 20-newsgroup, (b) Reuters-RCV1.

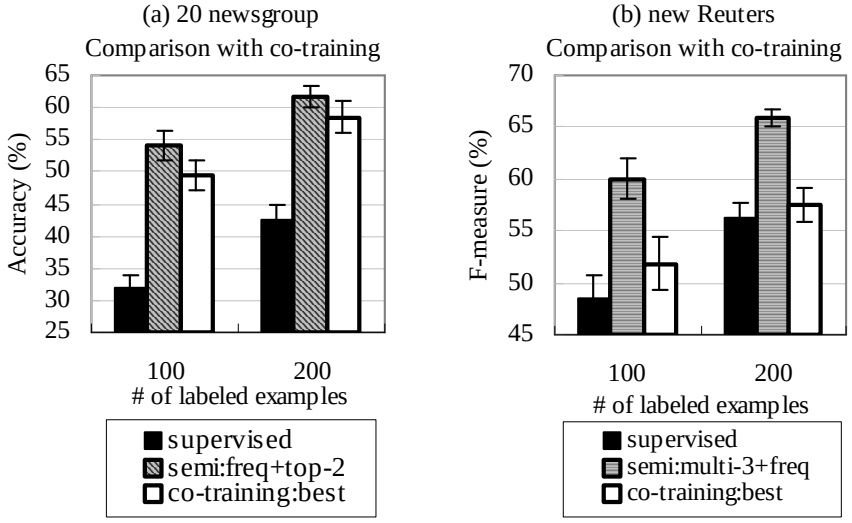

Figure 4: Comparison with co-training: averaged performance over 10 runs with standard deviations.

top-k/multi-k can take advantage of information in the labeled data when there is a reasonable amount of them. The best performance is often achieved when both types of auxiliary problems are used.

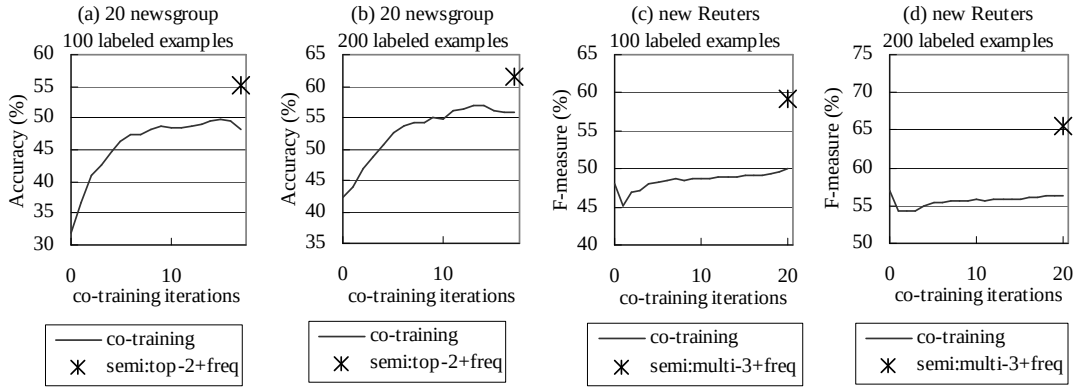

Figure 5: Co-training performance in typical runs, versus the number of iterations.

5.2.6 Performance comparison with other methods

As we have mentioned, the idea of using partially supervised auxiliary problems is motivated by co-training. Therefore we test co-training for comparison.

Co-training implementation Our implementation follows the original work (Blum and Mitchell, 1998), with the same feature splits as used in our auxiliary problems. Initial classifiers are trained with labeled instances drawn from the training sets. We maintain a pool of 10K unlabeled instances while refilling it by randomly choosing instances from the unlabeled set. The two classifiers propose labels for the unlabeled instances in this pool. For each classifier, we choose 1000 instances with high confidence while preserving the class distribution observed in the initial labeled data. This is done by choosing the class label with probabilities according to the distribution, and then the highest confident instance for that class label. The chosen instances are added to the pool of labeled data with their automatically proposed labels. The process repeats until the unlabeled instances are exhausted.

Comparison with co-training Figure 4 shows the best possible performance of co-training (the optimally stopped co-training procedure) averaged over 10 runs, with 100 and 200 labeled examples on the 20-newsgroup and the Reuters-RCV1 data. Our method outperforms the best co-training performance in all of the four settings by up to 8.4%. In Figure 5, we plot the co-training performance versus co-training iterations in typical runs.

As shown in Figure 6, our results outperform BN04 (Belkin and Niyogi, 2004)'s manifold-based semi-supervised learning method. They are also consistent with NMTM00 (Nigam et al., 2000)'s EM results. Since NMTM00 didn't report the exact numbers (we can only approximately read their results from a graph), we cannot include them in Figure 6. The performance of EM is usually similar to that of co-training as well as sel-training frequently used in NLP. Although quite successful for the 20 newsgroup data, as we shall see later, co-training and self-training do not perform very well for more difficult tasks.

# of labeled examples	BN04 best (manifold)	ASO-semi
100	39.8	54.1
200	–	61.6
500	59.9	68.5
1000	64.0	72.3

Figure 6: Comparison with similar settings in BN04 (Belkin and Niyogi, 2004) on 20 newsgroup.

5.2.7 Performance dependency on h

Recall that throughout the experiments, we fix the number of rows of Θ_t to a constant $h_t = 50$, as described in Section 5.1. Also recall that on text categorization, Θ_t is derived from auxiliary problems that use the t-th feature map. In this section, we study the performance dependency on the dimensionality h_t.

We are interested in the range of h_t roughly from 10 to 100. Figure 7 plots the performance on the 20-newsgroup and the Reuters-RCV1 corpora, in relation to $h_t = 5, 10, 15, 20, 30, \cdots, 100$. The results show that the method is insensitive to the change of dimension h_t in a relatively large range. In practice, this is a significant advantage over other dimension reduction approaches, which are typically sensitive to the choice of dimensions, or bootstrapping approaches, which are often sensitive to parameter settings.

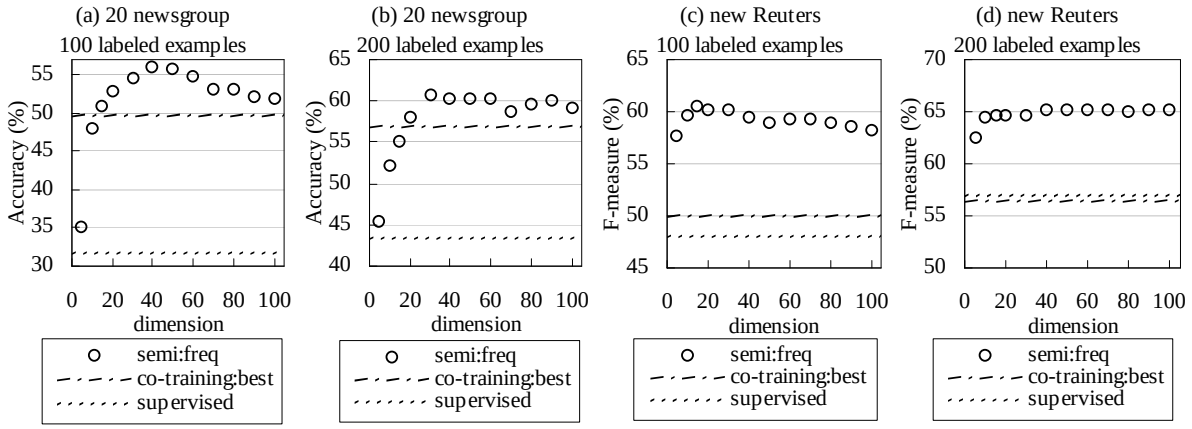

Figure 7: Performance dependency on h_t (the rank of Θ_t) in particular runs.

5.2.8 Interpretation of Θ

In order to gain some insights into the information obtained from unlabeled data, we show several significant entries of matrix Θ – the entries whose absolute values are:

- the largest in the columns (corresponding to features), and

- within the 100 largest among the positive (or negative) entries in the rows.

In the table below, we show at most ten entries chosen in this manner from the rows corresponding to the five most significant singular values. Θ was computed from the freq (unsupervised) auxiliary problems on the 20-newsgroup unlabeled data.

The second row appears to capture the distinctions between computers and religion. The third row distinguishes sports and the middle east issues. The positive entries of the fifth row appear to be about motor vehicles, and the negative entries are about printers. These topics are, indeed, relevant to the themes of the twenty newsgroups.

row#		features
2	+	pc, vesa, ibm, boards
	−	god, christian, bible, exist, doctrine, nature, worship, athos.rutgers.edu
3	+	team, detroit, series, leafs, play, cup, playoffs, played, penguins, devils
	−	israel, peace, jewish, lebanese, israelis, land, gaza, civilians, palestine, syria
4	+	files, jpeg, pov, utility, ms-windows, icon
	−	eisa, nubus, agents, attorney
5	+	oil, bikes, front, brake, rear, transmission, owner, driving, dogs, highway
	−	printer, hp, ink, appreciate, bj-200, toner, printing, bubblejet, laserjet, gcc

5.3 Named entity chunking experiments

We report named entity chunking performance on the CoNLL'03 shared-task[2] corpora (English and German). We choose this task because the original intention of this shared task was to test the effectiveness of semi-supervised learning methods (such as label bootstrap or co-training), and hence a large number of unlabeled data were made available. However, it turned out that none of the top performing systems used unlabeled data. One possible reason may be that the number of labeled data is relatively large (>200K). We show that by contrast, through our structural-learning based semi-supervised learning, it is possible to obtain results better than any of the top systems, using unlabeled data as the only additional resource. In particular, we do not use gazetteer information, which was used in all other systems.

The CoNLL corpora are annotated with four types of named entities: persons, organizations, locations, and miscellaneous names (e.g., "World Cup"). As is commonly done, we encode chunk information into word tags to cast the chunking problem to that of word tagging, and perform Viterbi-style decoding. We use the official training/development/test splits, as provided by the shared-task organizers. Our unlabeled data sets consist of 27 million words (English) and 35 million words (German), respectively. They were chosen from the same sources – Reuters and ECI Multilingual Text Corpus – as the training/development/test sets but disjoint from them.

5.3.1 Feature representation

Our feature representation is a slight modification of a simpler configuration reported in (Zhang and Johnson, 2003), which uses: token strings, parts-of-speech, character types, several characters at the beginning and the ending of the tokens, in a 5-token window

2. `http://cnts.uia.ac.be/conll2003/ner`.

around the current position; token strings in a 3-syntactic chunk window; labels of two tokens on the left to the current position; bi-grams of the current token and the label on the left; and the labels assigned to previous occurrences of the current word. These features are easily obtained without deep linguistic processing.

5.3.2 Auxiliary problems for named entity chunking

We use four types of auxiliary problems and their combinations:

- Word prediction: predicts the word at the current (or left or right) position, using the features derived from the other tokens.
- Top-2: predicts the top-2 choices of the classifier. We split features into "left-context vs. the others" and "right-context vs. the others". The rest is the same as Ex 4 in Section 4.2.2.

SVD is applied to each of the feature types separately. As for the word-prediction auxiliary problems, we only consider the instances whose current words are either nouns or adjectives since named entities mostly consist of these types. Also, we leave out all but 1000 auxiliary problems of each type that have the largest numbers of positive examples. This is to ensure that auxiliary predictors can be adequately learned from unlabeled data.

5.3.3 Performance results on the CoNLL English/German corpora

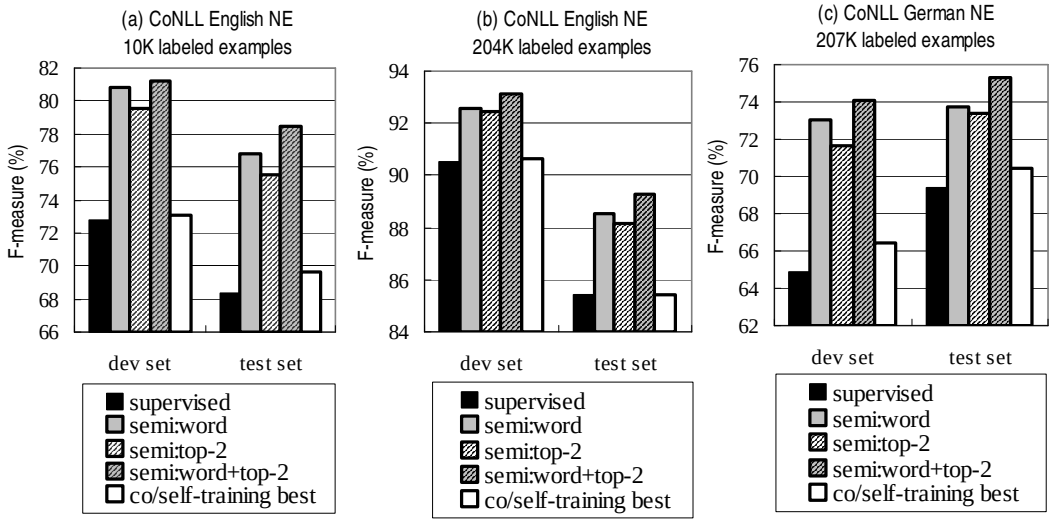

Figure 8: Named entity chunking F-measure performance. Without any gazetteer. For co- and self-training, the performance best among all the parameter settings (including the number of iterations) is shown. (a) CoNLL English corpus, 10K labeled examples. (b) CoNLL English corpus, 204K (the entire) labeled examples. (c) CoNLL German corpus, 207K (the entire) labeled examples.

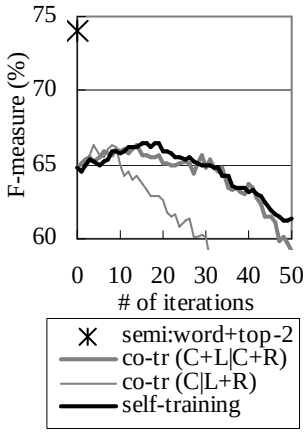

Figure 9: Co- and self-training named entity chunking performance in typical runs, versus the number of iterations. Tested on the German development set. In the legend, C, L, and R stand for current words, left context, and right context, respectively.

Figures 8 (a) and (b) show the English F-measure results – (a) with small (10K-word) labeled data, and (b) with the entire training set (204K words). German results using the entire training set are shown in Figure 8 (c). Precision and recall results in the same settings are found in Figure 15.

Note that to facilitate comparisons with the supervised baselines, we do *not* use any gazetteers or any name lexicons. Thus, there are only two kinds of information sources: labeled data and unlabeled data. We confirm that performance improvements gained by unlabeled data are significant in all of the semi-supervised settings: up to 10.10% gains with small English labeled data, up to 3.86% with larger English labeled data, and up to 9.22% improvements on the German data.

We note that word-prediction (unsupervised) auxiliary problems are particularly effective when the number of labeled examples is relatively small or the training data differ significantly from the test data. (The English test set is known to be less similar to the training set than the development set is, apparently because of the time periods from which the articles were drawn.) The best performance is achieved by combining all of the auxiliary problems. This performance trend is in line with that in the text categorization experiments.

Comparison with co- and self-training For comparison, we test co-training exploring parameter settings: pool size {50K,100K}, increment size {50, 100, 50K, 100K}, and commonly-used feature splits "current+left-context vs. current+right-context" and "current vs. context". Single-view bootstrapping is sometimes called *self-training*. In addition, we test the basic self-training, which replaces multiple classifiers in the co-training procedure with a single classifier that employs all the features. The co- and self-training performance shown in Figures 8 and 15 is the best possible performance among all the parameter settings (including the number of iterations). Co- and self-training at their best improve recall but often degrades precision. Consequently, their F-measure improvements are relatively low,

which demonstrates that it is not easy to benefit from unlabeled data on this task. Moreover, as shown in Figure 9, co- and self-training may rather degrade performance severely unless the iteration is optimally stopped. Such performance degradation (caused by contamination of automatically assigned labels) has also been observed in previous co-training studies on NLP tasks (e.g., Pierce and Cardie, 2001).

5.3.4 Comparison with previous best results

English test set

System	F-measure	Additional resources
semi:word+top-2	*89.31*	*unlabeled data*
FIJZ03(Florian et al., 2003)	88.76	gazetteers; 1.7M-word labeled data
CN03(Chieu and Ng, 2003)	88.31	gazetteers (also very elaborated features)
KSNM03(Klein et al., 2003)	86.31	rule-based post processing

German test set

Systems	F-measure	Additional resources
semi:word+top-2	*75.27*	*unlabeled data*
FIJZ03	72.41	gazetteers
KSNM03	71.90	rule-based post processing
ZJ03(Zhang and Johnson, 2003)	71.27	gazetteers

Figure 10: Comparison with previous best results on CoNLL'03 shared task

In Figure 10, we compare our performance results with those of the previous top systems among the CoNLL'03 shared-task participants.

On both English and German data, we are able to achieve performance better than those of the top participants, although they used more elaborated features. We note that the previous best English results were achieved with the help of knowledge intensive resources – such as gazetteers provided by the organizer plus additional gazetteers of a large number of names (FIJZ03, CN03); and a large amount (1.7 million words) of labeled data annotated with finer-grained named entities (FIJZ03); and rule-based post processing (KSNM03). Recall that the study of semi-supervised learning is motivated by the potential unavailability of such labor intensive resources. Hence, we feel that our results, which were obtained by using unlabeled data as the *only* additional resource, are very encouraging.

5.4 Part-of-speech tagging

We report part-of-speech (POS) tagging results on the Brown corpus. This corpus, annotated with 46 parts-of-speech, is one of the standard corpora for POS tagging research. We arbitrarily split the corpus into the labeled set (23K words), unlabeled set (1M words), and the test set (60K words).

The same auxiliary problems and feature representation (as in the named entity chunking experiments) are used, except for part-of-speech and syntactic chunk information. Following

the convention, we use error rate to measure the performance. It can be seen from Figure 11 that over 20% error reductions are achieved by learning from unlabeled data.

supervised	8.9
semi:left+curr	7.0 (21.3%)
semi:top-1	6.9 (22.5%)
semi:left+curr+top-1	6.9 (22.5%)

Figure 11: Part-of-speech tagging error rates (%). The numbers in parentheses are error reduction ratio with respect to the supervised baseline.

5.5 Hand-written digit image classification

This experiment uses the MNIST data downloaded from http://yann.lecun.com/exdb/mnist/. It consists of a training set (60K examples) and a test set (10K examples) of 28-by-28 gray-scale hand-written digits. The task is to classify the image data into 10 digits,'0'– '9'.

We use a feature representation composed of location-sensitive bags of pixel blocks, similar to the bag-of-word model in text categorization. It consists of normalized counts of pixel blocks of various shapes in the four regions (top-left, top-right, bottom-right, bottom-left). (Normalization was done by scaling the vector for each shape/region into a unit vector.) The pixel blocks are black-white patterns of 16 pixels in the shape of: squares(4×4), rectangles(2×8, 8×2), crossing lines (from top-left to bottom-right; from top-right to bottom-left), and dotted lines (horizontal and vertical). Using these features and trained with the entire training set (60K examples), the error rate in the supervised setting is 0.82%. This matches/surpasses state-of-the-art algorithms on the same data (reported on the MNIST data website) without additional image processing or transformation such as distortion or deskewing.

Auxiliary problems we used are partially-supervised. Feature splits were made by halving each image: features derived from top-left+top-right regions vs. those from bottom-left+bottom-right; top-left+bottom-left vs. top-right+bottom-right; and top-left+bottom-right vs. top-right+bottom-left.

In each run, labeled examples were randomly chosen from the training set, with the remaining training set used as unlabeled data. ASO-semi (Figure 12) consistently produced significant performance improvements over the supervised baseline.[3] It also outperforms a manifold-based semi-supervised learning method BN04 (Belkin and Niyogi, 2004) except when the number of labeled data is 100. The method in BN04 performs well for small labeled data. However, a disadvantage is that their method (which also requires dimension reduction) is more sensitive to the number of reduced dimensions. For example, with 100 labeled data, they achieved an error rate of 6.4 with 20 dimensions, but an error rate of 22.0 with 10 dimensions, and an error rate of 14.4 with 50 dimensions.

3. By contrast, co-training (with the same feature splits) sometimes rather degraded performance.

#labeled	supervised	ASO-semi	BN04 best (2nd best)
100	14.22 ± 2.90	9.13 ± 1.95	6.4 (14.4)
500	3.93 ± 0.22	3.05 ± 0.20	3.5 (3.6)
1000	2.83 ± 0.16	2.26 ± 0.11	3.2 (3.4)
5000	1.64 ± 0.07	1.47 ± 0.07	2.7 (2.9)

Figure 12: Error rates (%); average over 10 runs and standard deviation. MNIST hand-written digit image classification results on the test set. BN04 results (Belkin and Niyogi, 2004) are on the unlabeled portion of the training set.

(a) 20-newsgroup

# of labeled examples	100		200		500		1000	
supervised	32.0		42.7		56.9		66.0	
semi:freq	53.6	(+21.6)	60.0	(+17.3)	65.8	(+8.9)	69.1	(+3.1)
semi:top-1	46.6	(+14.6)	55.9	(+13.2)	67.6	(+10.7)	72.9	(+6.9)
semi:top-2	47.4	(+15.4)	58.4	(+15.7)	68.3	(+11.4)	*72.5*	(+6.5)
semi:top-2+freq	*54.1*	(+22.1)	*61.6*	(+18.9)	*68.5*	(+11.6)	72.3	(+6.3)

(b) Reuters-RCV1 corpus

# of labeled examples	100		200		500		1000	
supervised	48.5		56.3		65.4		71.4	
semi:freq	59.6	(+11.1)	64.8	(+8.5)	69.6	(+4.2)	72.8	(+1.4)
semi:multi-3	58.7	(+10.2)	65.5	(+9.2)	*71.2*	(+5.8)	*74.6*	(+3.2)
semi:multi-3+freq	*60.1*	(+11.6)	*65.8*	(+9.5)	70.7	(+5.3)	73.7	(+2.3)

Figure 13: Text categorization. Average over 10 runs. For each run, labeled examples were randomly drawn from the training set. (a) Accuracy on the 20-newsgroup corpus, (b) F-measure (micro-averaged) on Reuters-RCV1 corpus. Numbers in parentheses are performance improvements obtained from unlabeled data. The best performance in each column is italicized.

Data set	20-newsgroup				Reuters-RCV1			
# of labeled examples	100		200		100		200	
co-training:highest	49.6	(+17.6)	58.5	(+15.8)	51.9	(+2.4)	57.4	(+1.1)
co-training:lowest	34.5	(+2.5)	45.5	(+2.8)	46.8	(−1.7)	53.9	(−2.4)

Figure 14: Co-training text categorization performance. The highest and lowest performance among the iterations, averaged over 10 runs. Accuracy on 20-newsgroup and micro-averaged F-measure on Reuters-RCV1 are shown. The numbers in parentheses are improvements over the supervised settings.

(a) English (10K labeled examples)

	development set				test set			
	prec.	recall	$F_{\beta=1}$		prec.	recall	$F_{\beta=1}$	
supervised	72.04	73.46	72.74		70.52	66.25	68.32	
co/self best	72.36	73.85	73.10	(+0.36)	71.12	68.20	69.63	(+1.31)
semi:word	*82.16*	79.45	80.78	(+8.04)	78.66	74.96	76.77	(+8.45)
semi:top-2	80.78	78.31	79.52	(+6.78)	77.60	73.58	75.54	(+7.22)
semi:word+top-2	82.06	*80.46*	*81.25*	*(+8.51)*	*79.91*	*76.98*	*78.42*	*(+10.10)*

(b) English (204K labeled examples)

	development set				test set			
	prec.	recall	$F_{\beta=1}$		prec.	recall	$F_{\beta=1}$	
supervised	91.59	89.48	90.53		86.34	84.58	85.45	
co/self best	91.63	89.68	90.64	(+0.11)	86.30	84.53	85.40	(−0.05)
semi:word	93.45	91.75	92.60	(+2.07)	89.04	88.05	88.54	(+3.09)
semi:top-2	93.05	91.89	92.46	(+1.93)	88.49	87.80	88.14	(+2.69)
semi:word+top2	*93.84*	*92.48*	*93.15*	*(+2.62)*	*89.54*	*89.09*	*89.31*	*(+3.86)*

(c) German (207K labeled examples)

	development set				test set			
	prec.	recall	$F_{\beta=1}$		prec.	recall	$F_{\beta=1}$	
supervised	74.61	57.33	64.84		78.65	62.07	69.39	
co/self best	72.02	61.72	66.47	(+1.63)	77.39	64.66	70.45	(+1.06)
semi:word	*82.04*	65.80	73.03	(+8.19)	82.23	66.78	73.71	(+4.32)
semi:top-2	82.00	63.60	71.64	(+6.80)	82.74	65.91	73.37	(+3.98)
semi:word+top2	82.01	*67.52*	*74.06*	*(+9.22)*	*83.29*	*68.66*	*75.27*	*(+5.88)*

Figure 15: Named entity chunking results on the CoNLL'03 corpus. Without any gazetteer. For co-training and self-training (baseline), the best performance among all their parameter settings (including the number of iterations) is shown. (a) English, 10K labeled examples. (b) English, 204K (entire) labeled examples. (c) German, 207K (entire) labeled examples. The best performance in each column is italicized.

6. Discussions

This paper presents a general framework for learning predictive functional structures from multiple tasks. The idea is based on the concept that if multiple problems share a common predictive structure, then the structure can be more reliably estimated by considering these problems together. The process of learning the shared functional structure is referred to as structural learning. In the learning theory framework, structural learning is to discover a common structure of the hypothesis spaces shared by the problems. The main theoretical justification of this approach is that the shared structural parameter can be reliably estimated when m is large. Using the optimally estimated structural parameter, better generalization performance (averaged over the problems) can be achieved.

Moreover, we showed that the framework of structural learning can be applied to semi-supervised learning. This is achieved by creating auxiliary problems from unlabeled data that can reveal important underlying predictive structures of the data. Some examples of auxiliary problems were provided, and experimental results demonstrated that the discovered structures are very useful. Rigorously speaking, the theory we developed in Appendix A does not directly apply to semi-supervised learning. This is because in our theory, the performance is measured by averaged generalization ability over multiple prediction problems. However, in the setting of semi-supervised learning, we are only interested in the performance of the original supervised task, and not any of the auxiliary problems. For semi-supervised learning, a more relevant consequence of our analysis is that the shared structure can be stably estimated from multiple tasks. The usefulness of the shared structure is a different issue which is not directly answered by Appendix A. An intuitive justification of auxiliary problems we created is given in Section 4.3, although a more complete theory requires further investigation.

In summary, our approach to semi-supervised learning makes a bet on the existence of a shared predictive structure useful both for the supervised problem and for the auxiliary problems. The method proposed in Section 3 is robust since even if the discovered structure does not help on the supervised problem, the only potential negative effect is the introduction of some non-predictive features. Using typical discriminative learning methods with appropriate regularization, a small number of bad features only have a minor impact on the performance. However, if some of the features discovered from the auxiliary problems are useful, then the performance improvement can be significant.

The method derived in Section 3 has the intuitive interpretation of discovering low dimensional predictive structures on the classifier space. In our model, the most predictive dimensions correspond to the principal components of the multiple classifiers. Although our algorithm is based on the joint empirical risk minimization method which has a strong foundation in learning theory (see Appendix A), in principle, we can consider a more general approach of mining structures in the classifier space. Based on this general principle, one can design other structural learning algorithms that are not necessarily based on the joint empirical risk minimization method proposed in the paper. In fact, this general principle, which we may call *structural mining*, is the heart of our analysis. We shall thus conclude this paper by comparing some underlying concepts of structural mining to those of data-mining in Figure 16. In the table, a predictor can be regarded as a real-valued function defined on the data-space. The final row points out that we may also consider a data point

$\mathbf{x}$ as a predictor on the predictor space by associating with each predictor p the functional value $\mathbf{x}(p) := p(\mathbf{x})$. In this sense, data-mining can be viewed as a special structural mining.

	data-mining	structural-mining
space of interest	data space	predictor space
instances	data-points	predictors from multiple tasks
uncertainty	measurement error	estimation error
goal	find patterns in data	find structures of the predictors
predictive power	maybe	yes
duality	a data point is a predictor of points in the predictor-space	

Figure 16: Data mining versus structural mining

Acknowledgments

The authors would like to thank Trevor Hastie and Robert Tibshirani for helpful discussions and for pointing out related statistical studies. Part of the work was supported by ARDA under the NIMD program PNWD-SW-6059.

Appendix A. Analysis of Structural Learning

We include a theoretical analysis of the joint empirical risk minimization method (3) for structure learning. The main purpose is to demonstrate that by joint minimization, the shared structure θ can be more reliably estimated. We consider the idealized case, where the performance of interests is the averaged loss over the m tasks. In particular, we are interested in the behavior when m becomes large. Our bound shows that using the joint empirical risk minimization method, it is possible to estimate the shared hypothesis space $\mathcal{H}_{\cdot,\theta}$ more reliably as $m \to \infty$.

Note that in practice, we are often interested in the performance on one particular task instead of the averaged performance over multiple tasks. This non-idealized scenario is not directly covered by our analysis. In particular, in the semi-supervised learning setting, additional theoretical analysis is needed to show that structure shared by the artificially created tasks can improve the performance of the supervised task (see Section 4.3). Still, the analysis presented here is relevant because it implies that the shared structure can be reliably estimated by the joint empirical risk minimization method, which we employ.

Instead of providing the most general analysis with the tightest possible generalization bounds, we adopt a relatively simple approach. Our purpose is to illustrate the main benefit of structural learning, that is, the ability to obtain an accurate estimate of the best hypothesis class $\mathcal{H}_{\cdot,\theta}$ ($\theta \in \Gamma$) when the number of problems m is large. The analysis is closely related to that of Baxter (2000) (also see Ben-David and Schuller, 2003). We use a different (although related) technical approach with a different covering number definition. The modifications are necessary to make our results directly applicable to the specific method proposed in Section 3.

For clarity, in the following analysis, we simplify (3) as

$$[\hat{\theta}, \{\hat{f}_\ell\}] = \arg\min_{\theta\in\Gamma, \{f_\ell\in\mathcal{H}_\theta\}} \sum_{\ell=1}^{m}\sum_{i=1}^{n} L(f_\ell(\mathbf{X}_i^\ell), Y_i^\ell), \tag{9}$$

where we only consider the case $n_1 = n_2 = \cdots = n_m = n$, and $\mathcal{H}_{1,\theta} = \mathcal{H}_{2,\theta} = \cdots = \mathcal{H}_{m,\theta} = \mathcal{H}_\theta$. This simplification is not critical, but allows us to consider the behavior of (9) as $m \to \infty$ under fixed n.

For simplicity, we use a covering-number approach in our analysis. The treatment is very similar to the case of $m = 1$, which is the standard empirical risk minimization. We need to introduce some definitions in order to state the main theorem.

Definition 1 *Consider a set V with a distance function $d : V \times V \to \{0\} \cup R^+$. Given $\epsilon > 0$, the ϵ-covering number of V, denoted by $\mathcal{N}(\epsilon, V, d(\cdot,\cdot))$, is the minimal number of balls $B(f) = \{g : d(f,g) \le \epsilon\}$ of radius ϵ needed to cover V.*

Definition 2 *Let $S^{(n)} = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)\}$ be a set of n points. We define the $\ell_2(S^{(n)})$ distance between any two functions $f(\mathbf{x}, y)$ and $g(\mathbf{x}, y)$ on $S^{(n)}$ as*

$$\ell_2(S^{(n)})(f,g) = \left(\frac{1}{n}\sum_{i=1}^{n} |f(\mathbf{X}_i, Y_i) - g(\mathbf{X}_i, Y_i)|^2\right)^{1/2}.$$

Let $\mathcal{F}$ be a class of functions of $(\mathbf{x}, y)$. The empirical ℓ_2-covering number *of $\mathcal{F}$ is the covering number $\mathcal{N}(\epsilon, \mathcal{F}, \ell_2(S^{(n)}))$ of $\mathcal{F}$ with respect to the $\ell_2(S^{(n)})$ distance. The* uniform ℓ_2 covering number *is given by*

$$\mathcal{N}_2(\epsilon, \mathcal{F}, n) = \sup_{S^{(n)}} \mathcal{N}(\epsilon, \mathcal{F}, \ell_2(S^{(n)})),$$

where the supremum is over all samples $S^{(n)}$ of size n.

Definition 3 *Define distance d_∞ between hypothesis spaces H_θ ($\theta \in \Gamma$) as*

$$d_\infty(\theta_1, \theta_2) = \sup_{f\in\mathcal{H}_{\theta_2}} \inf_{g\in\mathcal{H}_{\theta_1}} \sup_{\mathbf{x},y} |f(\mathbf{x}, y) - g(\mathbf{x}, y)|.$$

We define the d_∞-covering number of Γ as $\mathcal{N}(\epsilon, \Gamma, d_\infty)$.

The following theorem gives a (one-sided) uniform convergence result for the joint ERM method (9).

Theorem 4 *For each $\ell = 1, \ldots, m$, let $S_\ell = \{(\mathbf{X}_i^\ell, Y_i^\ell), \ldots, (\mathbf{X}_n^\ell, Y_n^\ell)\}$ be a set of n points for problem ℓ, independently drawn from a distribution D_ℓ. Assume that $L(f(\mathbf{x}), y)$ is a bounded Lipschitz function of $f(\mathbf{x}) \in \mathcal{H}_\theta$. That is, there are θ-independent constants γ and M such that $\forall\theta_1, \theta_2 \in \Gamma$ and $\forall f_1 \in \mathcal{H}_{\theta_1}, f_2 \in \mathcal{H}_{\theta_2}$:*

$$\forall(\mathbf{x}, y) : |L(f_1(\mathbf{x}), y) - L(f_2(\mathbf{x}), y)| \le \gamma |f_1(\mathbf{x}) - f_2(\mathbf{x})|,$$
$$\forall(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2) : |L(f_1(\mathbf{x}_1), y_1) - L(f_1(\mathbf{x}_2), y_2)| \le M.$$

Then there is a universal constant C such that $\forall \eta \in [0,1]$, with probability $1-\eta$, we have $\forall \hat{\theta}, \{\hat{f}_\ell \in H_{\hat{\theta}}\}$:

$$\frac{1}{m}\sum_{\ell=1}^{m} R_\ell(\hat{f}_\ell) \le \frac{1}{m}\sum_{\ell=1}^{m} \hat{R}_\ell(\hat{f}_\ell, S_\ell) + \gamma C \inf_{\epsilon_0 \ge 0}\left[\epsilon_0 + \int_{\epsilon_0}^{\infty} \sqrt{\frac{\ln \mathcal{N}(\epsilon)}{n}}\, d\epsilon\right] + M\sqrt{\frac{\ln \frac{1}{\eta}}{nm}},$$

where R_ℓ and $\hat{R}_\ell$ are the true and empirical risks for problem ℓ:

$$R_\ell(\hat{f}_\ell) = \mathbf{E}_{(\mathbf{X}^\ell, Y^\ell)\sim D_\ell} L(\hat{f}_\ell(\mathbf{X}^\ell), Y^\ell), \quad \hat{R}_\ell(\hat{f}_\ell, S_\ell) = \frac{1}{n}\sum_{i=1}^{n} L(\hat{f}_\ell(\mathbf{X}_i^\ell), Y_i^\ell),$$

and

$$\ln \mathcal{N}(\epsilon) = \sup_\theta \ln \mathcal{N}_2(\epsilon, \mathcal{H}_\theta, n) + \frac{1}{m}\ln \mathcal{N}(\epsilon, \Gamma, d_\infty).$$

We shall delay the proof to the end of this appendix, and discuss the implications of the theorem first. In summary, this result justifies the joint ERM method (9), which minimizes the empirical risk on the right hand side of Theorem 4. The theorem implies that this method implicitly minimizes an upper bound of the true risk (averaged over the m problems) on the left hand side, which leads to a theoretical guarantee of the performance of this method.

The statistical complexity of the joint ERM method depends on the joint entropy $\ln \mathcal{N}(\epsilon)$, which has two components: the first term $\sup_\theta \ln \mathcal{N}_2(\epsilon, \mathcal{H}_\theta, n)$ is the learning complexity associated with individual estimation problems (with fixed θ). The second term $\frac{1}{m}\ln \mathcal{N}(\epsilon, \Gamma, d_\infty)$ is the complexity of estimating the best structural parameter θ. The most important consequence of our analysis is that the complexity of the structural space Γ, measured by the discounted entropy $\frac{1}{m}\ln \mathcal{N}(\epsilon, \Gamma, d_\infty)$, approaches zero when $m \to \infty$. This implies that we are able to find a near optimal (as measured by the generalization bound) shared structural parameter θ when m is large.

This theorem can be used to analyze the method we propose in Section 3, where in (4) and (5), a bi-linear structural model of the following form is used:

$$\begin{aligned}\mathcal{H}_\Theta =& \left\{\mathbf{w}^T\mathbf{\Phi}(\mathbf{x}) + \mathbf{v}^T\Theta\mathbf{\Psi}(\mathbf{x}) : \|\mathbf{w}\|_2 \le \frac{A}{\sup_{\mathbf{x}}\|\mathbf{\Phi}(\mathbf{x})\|_2}, \|\mathbf{v}\|_2 \le \frac{B}{\sup_{\mathbf{x}}\|\mathbf{\Psi}(\mathbf{x})\|_2}\right\},\\ \Gamma =& \{\Theta \in R^{h\times p} : \Theta\Theta^T = I_{h\times h}\},\end{aligned}$$

where $\mathbf{\Phi}(\mathbf{x})$ and $\mathbf{\Psi}(\mathbf{x})$ are pre-defined vector functions (feature maps) of $\mathbf{x}$; $\mathbf{v}$ is an h-dimensional vector; $\mathbf{\Psi}(\mathbf{x})$ is a p-dimensional vector; and Θ is an orthonormal $h \times p$ dimensional matrix. We also use $I_{h\times h}$ to denote the h-dimensional identity matrix.

For this model, the matrix Θ, shared by the different prediction problems, is the structural parameter. When we fix Θ, the hypothesis space $\mathcal{H}_\Theta$ is parameterized by weight vectors $\mathbf{w}$ and $\mathbf{v}$, where $\mathbf{w}$ can be a high dimensional vector (regularized using A), and $\mathbf{v}$ is a low dimensional vector (of dimensionality h). The idea of this model is to find a common low-dimensional predictive structure (shared by the m problems) parameterized by the projection matrix Θ. If we can discover such a structure, then we only need to use

a very small A to regularize the $\mathbf{w}$ vector, which leads to improved generalization performance. In other words, there is a trade-off between the dimensionality h of the common low-dimensional predictive structure, and the regularization size A. The optimal trade-off is through the shared structural parameter Θ, which can be more reliably estimated using structural learning formulation (3) when m is large.

This intuitive argument can be more rigorously justified using Theorem 4 with appropriate covering number estimates. Specifically, it can be shown that there are universal constants C_1, C_2 and C_3 such that

$$\sup_\theta \ln \mathcal{N}_2(\epsilon, \mathcal{H}_\theta, n) \leq \frac{C_1 A^2}{\epsilon^2} + C_2 h \ln\left(1 + \frac{B}{\epsilon}\right), \quad \ln \mathcal{N}(\epsilon, \Gamma, d_\infty) \leq C_3 hp \ln\left(1 + \frac{B}{\epsilon}\right).$$

We shall not include a detailed proof of these estimates (which is not essential for the purpose of this paper) but only outline the basic ideas used in our derivation. The term $C_1 A^2/\epsilon^2$ follows from a simple estimate of the Rademacher complexity of the sub function class in $\mathcal{H}_\Theta$ corresponding to $\mathbf{w}^T \mathbf{\Phi}(\mathbf{x})$ as $A/\sqrt{n}$, and a straight-forward application of Sudak's minoration (e.g., see Ledoux and Talagrand, 1991, Chapter 12). The two $\ln(1 + B/\epsilon)$ terms can be obtained by explicit discretization of the corresponding finite dimensional parameter spaces — one for the h-dimensional sub function class in $\mathcal{H}_\Theta$ corresponding to $\mathbf{v}^T \Theta \mathbf{\Psi}(\mathbf{x})$ (with fixed Θ and variable $\mathbf{v}$), and the other for a direct discretization of the hp-dimensional variable Θ. We simply note that a bounded set in a d-dimensional parameter space can be covered by $O(\epsilon^{-d})$ grid points with width no greater than ϵ in every direction.

By using the above covering number estimates, the complexity term in Theorem 4 becomes

$$\ln \mathcal{N}(\epsilon) \leq \frac{C_1 A^2}{\epsilon^2} + C_2 h \ln\left(1 + \frac{B}{\epsilon}\right) + \frac{1}{m} C_3 hp \ln\left(1 + \frac{B}{\epsilon}\right).$$

The third term is the complexity of estimating the structural parameter Θ, which vanishes as $m \to \infty$. The first and second terms characterize the trade-off between the regularization size A for the $\mathbf{w}$ parameter, and the dimensionality h for the $\mathbf{v}$ parameter. With the estimated Θ, the model approximates the underlying true predictor better for a fixed regularization size A (and thus a fixed complexity term $\ln \mathcal{N}(\epsilon)$ in Theorem 4), which implies better generalization behavior.

Proof of Theorem 4 Given training data $S = \cup_\ell S_\ell$, we define a vector function class on S:

$$\mathcal{F}_S = \{f = [f_\ell] : f(\mathbf{X}_i^\ell) = f_\ell(\mathbf{X}_i^\ell), f_\ell \in H_\theta, \theta \in \Gamma\},$$

where we use the notation $[f_\ell] = [f_\ell]_{\ell=1,\ldots,m} = [f_1, \ldots, f_m]$. Similarly, define

$$\mathcal{F}_S^L = \{[L(f(X_i^\ell), Y_i^\ell)]_{\ell=1,\ldots,m} : f \in \mathcal{F}_S\}.$$

We introduce two lemmas.

Lemma 5 *We have the following bounds:*

$$\ln \mathcal{N}(2\gamma\epsilon, \mathcal{F}_S^L, \ell_2(S)) \leq \ln \mathcal{N}(2\epsilon, \mathcal{F}_S, \ell_2(S)) \leq \sum_{\ell=1}^m \sup_\theta \ln \mathcal{N}(\epsilon, \mathcal{H}_\theta, \ell_2(S_\ell)) + \ln \mathcal{N}(\epsilon, \Gamma, d_\infty).$$

Proof The first inequality is a direct consequence of the Lipschitz condition in Theorem 4. We shall prove the second inequality. Consider an ϵ-cover of Γ in the d_∞ metric. For simplicity, we denote the cover by $\bar{\theta}_1, \ldots, \bar{\theta}_{N_\Gamma}$, where $N_\Gamma = \mathcal{N}(\epsilon, \Gamma, d_\infty)$. For each $\bar{\theta}_j$, we can find an ϵ-cover of $H_{\bar{\theta}_j}$ on S_ℓ as $\bar{f}_{\ell,j,1}, \ldots, \bar{f}_{\ell,j,N_\ell}$, where $N_\ell = \mathcal{N}(\epsilon, \mathcal{H}_\theta, \ell_2(S_\ell))$.

Now given any $f = [f_\ell] \in \mathcal{F}_S$, where $f_\ell \in \mathcal{H}_\theta$ for some $\theta \in \Gamma$, we can find $1 \leq J \leq N_\Gamma$, and $f' = [f'_\ell] \in \mathcal{F}_S$ such that each $f'_\ell \in \mathcal{H}_{\bar{\theta}_J}$ and $\ell_2(S)(f, f') \leq \epsilon$. We can further approximate each f'_ℓ by a $\bar{f}_{\ell,J,K_\ell}$ where $1 \leq K_\ell \leq N_\ell$ such that $\ell_2(S_\ell)(f'_\ell, \bar{f}_{\ell,J,K_\ell}) \leq \epsilon$. It follows that if we let $\bar{f} = [\bar{f}_{\ell,J,K_\ell}]_{\ell=1,\ldots,m}$, then $\ell_2(S)(f', \bar{f}) \leq \epsilon$. Therefore we have $\ell_2(S)(f, \bar{f}) \leq 2\epsilon$. This means that $\mathcal{F}_S$ has a 2ϵ-cover of the form $\bar{f} = [\bar{f}_{\ell,J,K_\ell}]$ ($J = 1, \ldots, N_\Gamma$, $K_\ell = 1, \ldots, N_\ell$ for $\ell = 1, \ldots, m$). The size of this cover is $N_\Gamma \prod_\ell N_\ell$. ■

Lemma 6 *Let*

$$Q(S) = \sup_{[f_\ell] \in \mathcal{F}_S} \frac{1}{m} \sum_{\ell=1}^{m} (R_\ell(f_\ell) - \hat{R}_\ell(f_\ell, S_\ell)),$$

then $\forall \eta \in [0, 1]$, *with probability* $1 - \eta$*:*

$$Q(S) \leq \mathbf{E}_S\, Q(S) + M \sqrt{\frac{\ln \frac{1}{\eta}}{mn}}.$$

Proof For a given $1 \leq \bar{\ell} \leq m$ and $1 \leq \bar{i}$, we create a new dataset $\bar{S} = \cup_\ell \bar{S}_\ell$ by changing the $\bar{i}$-th datum of the $\bar{\ell}$-th problem in $S = \cup_\ell S_\ell$ from $(X_{\bar{i}}^{\bar{\ell}}, Y_{\bar{i}}^{\bar{\ell}})$ to $(\bar{X}_{\bar{i}}^{\bar{\ell}}, \bar{Y}_{\bar{i}}^{\bar{\ell}})$ (and keep all the other data points identical). Then it is easy to verify that

$$Q(S) - Q(\bar{S}) \leq \sup_\theta \sup_{f \in \mathcal{H}_\theta} \frac{1}{mn} |L(f(X_{\bar{i}}^{\bar{\ell}}), Y_{\bar{i}}^{\bar{\ell}}) - L(f(\bar{X}_{\bar{i}}^{\bar{\ell}}), \bar{Y}_{\bar{i}}^{\bar{\ell}})| \leq \frac{M}{mn}.$$

The lemma is a direct consequence of McDiarmid's concentration inequality (McDiarmid, 1989). ■

We are now ready to prove the main theorem. Consider a sequence of binary random variables $\sigma = \{\sigma_i^\ell\}$ such that each $\sigma_i^\ell = \pm 1$ is independent with probability $1/2$. The *Rademacher complexity* of $\mathcal{F}_S^L$ under empirical sample S, is given by

$$R(\mathcal{F}_S^L, S) = \mathbf{E}_\sigma \sup_{f \in \mathcal{F}_S} \left(\frac{1}{mn} \sum_{\ell=1}^{m} \sum_{i=1}^{n} \sigma_i^\ell L(f(\mathbf{X}_i^\ell), Y_i^\ell) \right).$$

It is well known that there exists a universal constant C (a variant of Corollary 2.2.8 in van der Vaart and Wellner, 1996):

$$R(\mathcal{F}_S^L, S) \leq \frac{C}{2} \inf_{\epsilon_0} \left[\epsilon_0 + \frac{1}{\sqrt{mn}} \int_{\epsilon_0}^{\infty} \sqrt{\ln \mathcal{N}_2(2\epsilon, \mathcal{F}_S^L, nm)} d\epsilon \right].$$

Applying Lemma 5, we obtain

$$R(\mathcal{F}_S^L, S) \leq \frac{C}{2} \inf_{\epsilon_0} \left[\epsilon_0 + \int_{\epsilon_0}^{\infty} \sqrt{\frac{\ln \mathcal{N}(\epsilon/\gamma)}{n}} d\epsilon \right] = \frac{\gamma C}{2} \inf_{\epsilon_0} \left[\epsilon_0 + \int_{\epsilon_0}^{\infty} \sqrt{\frac{\ln \mathcal{N}(\epsilon)}{n}} d\epsilon \right].$$

Using the standard symmetrization argument (for example, see Lemma 2.3.1 of (van der Vaart and Wellner, 1996)), we have

$$\mathbf{E}_S \, Q(S) \leq 2\mathbf{E}_S \, R(\mathcal{F}_S^L, S) = \gamma C \inf_{\epsilon_0} \left[\epsilon_0 + \int_{\epsilon_0}^{\infty} \sqrt{\frac{\ln \mathcal{N}(\epsilon)}{n}} d\epsilon \right].$$

The theorem is now a direct consequence of Lemma 6.

References

Rie Kubota Ando and Tong Zhang. A high-performance semi-supervised learning method for text chunking. In *ACL 05*, 2005.

J. Baxter. A model for inductive bias learning. *Journal of Artificial Intelligent Research*, pages 149–198, 2000.

Mikhail Belkin and Partha Niyogi. Semi-supervised learning on Riemannian manifolds. *Machine Learning*, Special Issue on Clustering:209–239, 2004.

S. Ben-David and R. Schuller. Exploiting task relatedness for multiple task learning. In *COLT 03*, 2003.

Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100, 1998.

L. Breiman and J. Friedman. Predicting multivariate responses in multiple linear regression. *J. Roy. Statist. Soc. B.*, 59:3–37, 1997. with discussion.

Rich Caruana. Multi-task learning. *Machine Learning*, pages 41–75, 1997.

Hai Leong Chieu and Hwee Tou Ng. Named entity recognition with a maximum entropy approach. In *Proceedings CoNLL-2003*, pages 160–163, 2003.

T. Evegniou and M. Pontil. Regularized multi–task learning. In *Proc. Conf. on Knowledge Discovery and Data Mining*, 2004.

Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. Named entity recogintion through classifier combination. In *Proceedings CoNLL-2003*, pages 168–171, 2003.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

Thorsten Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 200–209, 1999.

Dan Klein, Joseph Smarr, Huy Nguyen, and Christopher D. Manning. Named entity recognition with character-level models. In *Proceedings CoNLL-2003*, pages 188–191, 2003.

Michel Ledoux and Michel Talagrand. *Probability in Banach spaces*. Springer-Verlag, Berlin, 1991. ISBN 3-540-52013-9. Isoperimetry and processes.

C. McDiarmid. On the method of bounded differences. In *Surveys in Combinatorics*, pages 148–188. Cambridge University Press, 1989.

C. A. Micchelli and M. Ponti. Kernels for multi–task learning. In *NIPS 2004*, 2005. to appear.

Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, Special issue on information retrieval:103–134, 2000.

David Pierce and Claire Cardie. Limitations of co-training for natural language learning from large datasets. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing (EMNLP-2001)*, 2001.

M. Szummer and T. Jaakkola. Partially labeled classification with Markov random walks. In *NIPS 2001*, 2002.

Aad W. van der Vaart and Jon A. Wellner. *Weak convergence and empirical processes*. Springer Series in Statistics. Springer-Verlag, New York, 1996. ISBN 0-387-94640-3.

V.N. Vapnik. *Statistical learning theory*. John Wiley & Sons, New York, 1998.

D. Yarowsky. unsupervised word sense disambiguation rivaling supervised methods. In *proceedings of ACL 95*, pages 189–196, 1995.

Tong Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *ICML 04*, pages 919–926, 2004.

Tong Zhang and David E. Johnson. A robust risk minimization based named entity recognition system. In *Proceedings CoNLL-2003*, pages 204–207, 2003.

Tong Zhang and Frank J. Oles. A probability analysis on the value of unlabeled data for classification problems. In *ICML 00*, pages 1191–1198, 2000.

D. Zhou, O. Bousquet, T.N. Lal, J. Weston, and B. Schlkopf. Learning with local and global consistency. In *NIPS 2003*, pages 321–328, 2004.

Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML 2003*, 2003.