

---

# Effective Transductive Learning via PAC-Bayesian Model Selection

---

Ran El-Yaniv  
Leonid Gerzon

RANI@CS.TECHNION.AC.IL  
GLEONID@TX.TECHNION.AC.IL

Computer Science Department  
Technion - Israel Institute of Technology

## Abstract

We study a transductive learning approach based on clustering. In this approach one constructs a diversity of unsupervised models of the unlabeled data using clustering algorithms. These models are then exploited to construct a number of hypotheses using the labeled data and the learner selects an hypothesis that minimizes a transductive PAC-Bayesian error bound, which holds with high probability. Empirical examination of this approach, implemented with spectral clustering, on a suite of benchmark datasets, indicates that the new approach is effective and that on some datasets it significantly outperforms one of the best transductive learning algorithms known today.

## 1. Introduction

Recently, there has been a growing need for and interest in learning algorithms capable of utilizing both labeled and unlabeled data (see, for example, the survey by Seeger, 2002). Despite widespread confusion, a distinction between *inductive* ‘semi-supervised’ learning and *transductive* learning should be made. While both approaches attempt to utilize unlabeled data (in addition to the standard labeled training set) to accelerate the learning process, in transduction one is only interested in classifying the given unlabeled points. There is no need to construct a general hypothesis capable of classifying unseen data points. Therefore, at the outset transduction appears to be an easier problem than (semi-supervised) induction.<sup>1</sup>

---

<sup>1</sup>We can make analogies between transduction and a “take-home exam” (where the student gets to see the questions before studying), and between semi-supervised learning and a standard “classroom exam” (where the student gets to see only exam questions from previous years before studying).

Transductive inference was introduced by Vapnik more than twenty years ago (Vapnik, 1982), and through the years he has been its greatest advocate.<sup>2</sup> Although the interest in this learning framework is rising, there is still lack of *provably useful* learning principles and algorithms that are both theoretically founded as well as good performers on real world data.

In this paper we study a transductive learning scheme that was proposed in a recent NIPS paper (Derbeko et al., 2004).<sup>3</sup> In this scheme the learner applies a clustering algorithm over the unlabeled data to generate several (unsupervised) models. The learner then utilizes the labeled data to guess labels for entire clusters (so that all points in the same cluster have the same label). This way the algorithm forms a number of hypotheses, one of which is then selected based on a PAC-Bayesian error bound for transduction.

The natural idea of first clustering the unlabeled data and then assigning labels to clusters has been around for a long time and there are plenty of heuristic procedures that attempt to learn using this approach within a semi-supervised or transductive settings (see, for example, Sec. 2.1 in Seeger, 2002 and references therein). However, to the best of our knowledge, none of the existing procedures was ever theoretically justified in terms of provable reduction of the true risk.

The new approach, given here the acronym *CLUST*, is both simple and rests on solid theoretical ground. As we show here, *CLUST* is competitive with one of the best transductive learning methods. In particular, we show in Sec. 5 a rather extensive set of experiments where we study the performance of *CLUST* and compare it to the recent ‘Spectral Graph Transducer’ (SGT) algorithm proposed in (Joachims, 2003). Our re-

---

<sup>2</sup>In his 1995 book Vapnik (1995) writes: “This model [transduction] can provide the strongest contribution to the 2000 years of discussions about the essence of human reason”.

<sup>3</sup>That paper derived this scheme from a general PAC-Bayesian error bound for transduction but did not discuss any implementation detail or evaluate its practical significance.

sults indicate that CLUST fares well in this comparison. Note that the SGT algorithm is shown in (Joachims, 2003) to outperform other well known transductive learning methods such as the transductive support vector machine (TSVM) of (Joachims, 1999).

## 2. The Transduction Setting

Throughout the paper we consider a classification setting. For simplicity (and initial evaluation of the basic ideas), we focus on *binary* classification.<sup>4</sup> We rely on the transduction formulation proposed in (Vapnik, 1998, Chapt. 8). In this formulation the learner is given a *full sample*  $X_{m+u}$  of  $m+u$  unlabeled points. After observing these points, a *training set*  $X_m$  of  $m$  points is chosen uniformly at random among all  $m$ -subsets of the full sample. The labels of all the points in the training set are provided to the learner. Based on the labeled and unlabeled points, the learner's goal is to predict, as accurately as possible, the labels of the remaining unlabeled points, which constitute the *test set*,  $X_u = X_{m+u} \setminus X_m$ .

**Remark 2.1** The random choice of the training set from the full sample as described above is equivalent to choosing uniformly at random  $m$  points from the full sample, *without replacement*. Therefore, unlike the standard inductive setting, points in the training (and test) sets are *dependent*.

**Remark 2.2** It is not assumed that points in the full sample are drawn i.i.d. from some (unknown) distribution. The only distributional requirement here is that the training set is drawn uniformly at random from all possible  $m$ -subsets. Vapnik considered a second transduction setting in which the learner receives training and test sets, which are assumed to be drawn i.i.d. from some unknown distribution. This second setting is more suitable for some applications (and may appear more natural). However, Theorem 8.1 in (Vapnik, 1998) states that error bounds for the first setting imply the same bounds for the second setting. We therefore restrict our attention, throughout this paper, to the first setting and note that the error bounds and algorithms considered in this paper are applicable in the second setting as well.

Throughout the paper we use the following notations and definitions. We denote by  $\mathcal{H}$  a set of binary hypotheses consisting of functions from the input space  $\mathcal{X}$  to  $\mathcal{Y} = \{\pm 1\}$ . The learner's goal is to choose a good hypothesis from  $\mathcal{H}$ . For each  $h \in \mathcal{H}$  and a set

<sup>4</sup>The CLUST algorithm can be easily extended to handle multi-class classifications in a natural manner.

$Z = x_1, \dots, x_N$  of samples define

$$R_h(Z) \triangleq \frac{1}{N} \sum_{i=1}^N \ell(h(x_i), \phi(x_i)), \quad (1)$$

where  $\phi(x_i) \in \mathcal{Y}$  is the true label of  $x_i$  and  $\ell(\cdot, \cdot)$  is the 0/1-loss. We make use of the following quantities, which are all instances of (1). The quantity  $R_h(X_{m+u})$  is called the *full sample risk* of the hypothesis  $h$ ,  $R_h(X_u)$  is referred to as the *transduction risk* or *test error* (of  $h$ ), and  $R_h(X_m)$  is the *training error* (of  $h$ ). It is important to observe that in our transduction setting  $R_h(X_{m+u})$  is *not* a random variable, but both  $R_h(X_m)$  and  $R_h(X_u)$  are random variables, due to the random selection of the samples  $X_m$  from  $X_{m+u}$ . The goal of the learner is to choose  $h \in \mathcal{H}$  with minimal transduction risk  $R_h(X_u)$ .

## 3. Vapnik's PAC-Bayesian Bounds for Transduction

The transductive error bounds we present here are based on a slight adaptation of Vapnik's arguments presented in (Vapnik 1982; 1998). The idea is to bound the deviation between the two random variables  $R_h(X_u)$  and  $R_h(X_m)$ , which are both concentrated around their mean  $R_h(X_{m+u})$ . This deviation can be explicitly written as a tail of an hypergeometric distribution, which converges exponentially fast, and in fact, has faster convergence rates than the tail of the binomial distribution (which governs the learning rates in standard inductive learning).

Fix  $m$  and  $u$  and consider some hypothesis  $h \in \mathcal{H}$ . Suppose that  $h$  makes  $k_h$  errors on the full sample (i.e.  $k_h = (m+u)R_h(X_{m+u})$ ). Consider a random choice of the training set  $X_m$  from the full sample, and let  $B(r, k_h)$  be the probability that  $h$  makes exactly  $r$  errors over the training set  $X_m$ . This probability is by definition the hypergeometric distribution, given by

$$B(r, k_h) \triangleq \frac{\binom{k_h}{r} \binom{m+u-k_h}{m-r}}{\binom{m+u}{m}}.$$

Define

$$\begin{aligned} C(\varepsilon, k_h) &\triangleq \Pr \{R_h(X_u) - R_h(X_m) > \varepsilon\} \\ &= \Pr \left\{ \frac{k_h - r}{u} - \frac{r}{m} > \varepsilon \right\} \\ &= \sum_r B(r, k_h), \end{aligned}$$

where the summation is over all values of  $r$  such that  $\max\{k_h - u, 0\} \leq r \leq \min\{m, k_h\}$  and

$$\frac{k_h - r}{u} - \frac{r}{m} > \varepsilon. \quad (2)$$

Define

$$\Gamma_1(\varepsilon) \triangleq \max_k C(\varepsilon, k); \quad (3)$$

$$\Gamma_2(\varepsilon) \triangleq \max_k C\left(\sqrt{\frac{k}{m+u}} \cdot \varepsilon, k\right). \quad (4)$$

We now state Vapnik’s implicit bounds for transduction. The bounds are adapted slightly in order to incorporate a prior probability over  $\mathcal{H}$  (the original bounds deal with a uniform prior). This change makes them similar to McAllester’s PAC-Bayesian bounds in (McAllester, 1999). Also, we note that the original bounds in (Vapnik, 1982) are two-sided and the following theorem considers one-sided bounds.<sup>5</sup>

**Theorem 3.1 (Vapnik)** *Let  $\delta$  be given, let  $\mathbf{p}$  be a prior distribution over  $\mathcal{H}$  that may depend on the full sample  $X_{m+u}$ . Let  $\varepsilon_1^*(h)$  (resp.  $\varepsilon_2^*(h)$ ) be the minimal value of  $\varepsilon$  that satisfies  $\Gamma_1(\varepsilon) \leq \mathbf{p}(h)\delta$  (resp.  $\Gamma_2(\varepsilon) \leq \mathbf{p}(h)\delta$ ) Then, with probability at least  $1 - \delta$ , for all  $h \in \mathcal{H}$ ,*

$$R_h(X_u) < R_h(X_m) + \varepsilon_1^*(h), \quad (5)$$

and respectively,

$$R_h(X_u) < R_h(X_m) + \sqrt{R_h(X_{m+u})} \cdot \varepsilon_2^*(h). \quad (6)$$

**Proof** We sketch the proof of (6).

$$\begin{aligned} & \Pr \left\{ \exists h \in \mathcal{H} : \frac{R_h(X_u) - R_h(X_m)}{\sqrt{R_h(X_{m+u})}} > \varepsilon_2^*(h) \right\} \\ & \leq \sum_{h \in \mathcal{H}} C \left( \sqrt{\frac{k_h}{m+u}} \varepsilon_2^*(h), k_h \right) \\ & \leq \sum_{h \in \mathcal{H}} \Gamma(\varepsilon_2^*(h)) \leq \sum_{h \in \mathcal{H}} \mathbf{p}(h)\delta = \delta, \end{aligned} \quad (7)$$

where (7) follows from the union bound.  $\blacksquare$

It is easy to eliminate the (unknown) quantity  $R_h(X_{m+u})$  from the right hand side of (6). Subtracting  $R_h(X_m)$  from both sides of (6), then squaring both sides and substituting  $\frac{m}{m+u}R_h(X_m) + \frac{u}{m+u}R_h(X_u)$  for  $R_h(X_{m+u})$ , we get a quadratic inequality where the “unknown” variable is  $R_h(X_u)$ . Solving for  $R_h(X_u)$  yields the following result, as in Equation (8.15) in (Vapnik, 1998).

**Corollary 3.2 (Vapnik)** *Under the conditions of*

<sup>5</sup>Specifically, in the original bound (Vapnik, 1982), the two-sided condition  $|\frac{k_h-r}{u} - \frac{r}{m}| > \varepsilon$  is used instead of condition (2).

*Theorem 3.1,*

$$\begin{aligned} R_h(X_u) & \leq R_h(X_m) + \frac{\varepsilon^*(h)^2 u}{2(m+u)} \\ & \quad + \varepsilon_2^*(h) \sqrt{R_h(X_m) + \left(\frac{\varepsilon_2^*(h)u}{2(m+u)}\right)^2}. \end{aligned} \quad (8)$$

We thus have two general bounds: The “absolute” bound (5) and the “relative” bound (8).<sup>6</sup>

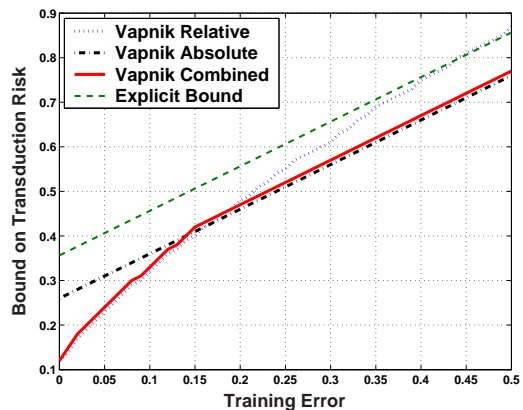


Figure 1. The Vapnik absolute, relative and combined bounds and the explicit approximation bound (9); all bounds are computed with  $m = u = 100$  and  $\mathbf{p}(h) = 1/608$ . This prior corresponds to  $C = 20$ ,  $T = 1$  and  $\tau = 5$  (see Sec. 4). The Vapnik relative and absolute bounds are computed with  $\delta = 0.05$ , and therefore the Vapnik combined bound corresponds to  $\delta = 0.1$ . The curve of the explicit bound was also calculated with  $\delta = 0.1$ .

It is possible to show that the relative bound is tighter than the absolute bound for smaller values of the empirical error  $R_h(X_m)$  and that the absolute bound is tighter for larger values of  $R_h(X_m)$  (see, for example Figure 1). We therefore use the union bound to combine the bounds into one bound by applying each of the bounds with confidence  $\delta/2$  and taking the minimum of the two at each point. Throughout the paper we refer to the combined bound as “the Vapnik (PAC-Bayesian) bound”. Note that a related result has recently been presented in (Blum & Langford, 2003).<sup>7</sup>

<sup>6</sup>In his book Vapnik only focuses on the relative bound. Both bounds are mentioned in (Bottou et al., 1994).

<sup>7</sup>In particular, Theorem 6 in (Blum & Langford, 2003) presents a comparable PAC-Bayesian bound for transduction based on a similar direct calculation of the hypergeometric tail. A brief comparison between the Vapnik bound and the bound of Theorem 6 in (Blum & Langford, 2003) indicates that the Blum-Langford bound is slightly tighter

The Vapnik PAC-Bayesian bound is rather tight and therefore, it is intriguing to see if the bound can be effectively utilized in a practical model selection, as we attempt in this paper. Possible sources of slackness in this bound are only introduced through the utilization of the union bound in (7) and the definitions of  $\Gamma_i(\varepsilon)$  in equations (3) and (4). However, note that  $\varepsilon_i^*(h)$  ( $i = 1, 2$ ) is a complicated *implicit* function of  $m$ ,  $u$ ,  $\mathbf{p}(h)$  and  $\delta$  leading to a bound that is difficult to interpret and (as noted also by Vapnik) must be computed in order to be used. A number of explicit (but looser) PAC-Bayesian bounds for transduction are presented in (Derbeko et al., 2004). For example, under the same conditions of Theorem 3.1 the following bound holds with probability at least  $1 - \delta$ , uniformly for all  $h \in \mathcal{H}$ ,

$$R_h(X_u) \leq R_h(X_m) + \sqrt{\left(\frac{m+u}{u}\right) \left(\frac{\ln \frac{1}{\mathbf{p}(h)} + \ln \frac{1}{\delta}}{m}\right)}. \quad (9)$$

Figure 1 shows the above bounds as a function of the empirical error for realistic assignments of the other problem parameters.

#### 4. Transduction via Clustering

In order to utilize PAC-Bayesian bounds, such as the Vapnik bound or (9), one has to specify a prior distribution  $\mathbf{p}$  over  $\mathcal{H}$ . As observed in (Derbeko et al., 2004), within the transduction setting it is often convenient to apply PAC-Bayesian bounds because the prior distribution can be constructed *after* observing the (unlabeled) full sample. Derbeko et al. also proposed the following learning scheme, based on clustering.

Let  $\mathcal{A}(\tau, \theta)$  be a parametric clustering algorithm that can partition the full sample into any desirable number  $\tau$  of clusters. The hyper-parameter  $\theta \in \Theta$  can specify different clustering algorithms and/or their parameters (e.g., the variance in spectral clustering; see below). For  $\tau = 2, \dots, C$  ( $C \leq m$ ) and  $\theta \in \Theta$ , we apply  $\mathcal{A}(\tau, \theta)$  on the full sample and obtain a collection of partitions  $\mathcal{C}_{\tau, \theta}$ ,  $\tau = 2, \dots, C$ ,  $\theta \in \Theta$ . Setting  $T = |\Theta|$  we thus have  $(C - 1)T$  partitions of  $X_{m+u}$ . We now receive the training set  $X_m$  and would like to utilize it in order to construct  $(C - 1)T$  hypotheses, one for each partition. For each partition  $\mathcal{C}_{\tau, \theta}$ , we consider each of its  $\tau$  clusters and label all the points in the same cluster according to a majority vote among all the labeled points (from  $X_m$ ) falling within that cluster. If none of the labeled training points is in the cluster (or in case

whenever  $u \geq m$  and that they are the same when  $u < m$ . However, the differences appear to be negligible for practical purposes.

of a tie), we arbitrarily give the entire cluster a single label from  $\{\pm 1\}$ . Hence, we end up with a collection of  $(C - 1)T$  hypotheses. We consider each hypothesis  $h$ , compute its training error  $R_h(X_m)$  and then compute its Vapnik PAC-Bayesian bound based on appropriate prior  $\mathbf{p}(h)$  (see below). We select the hypothesis with the minimal error bound.

Algorithm CLUST( $X_{m+u}, X_m, \delta, \mathcal{A}$ )

**Input:**

- A full sample  $X_{m+u} = \{x_1, \dots, x_{m+u}\}$
- A training set  $\{(x_{i_1}, y_{i_1}), \dots, (x_{i_m}, y_{i_m})\}$ , where  $X_m = \{x_{i_j}\} \subset X_{m+u}$
- A confidence level  $\delta$
- A parameterized clustering algorithm  $\mathcal{A}(\tau, \theta)$ , together with  $C =$  maximum number of clusters, and  $\Theta =$  a parameter set

**Output:** A classification of the test set  $X_u = X_{m+u} \setminus X_m$  together with a bound on the classification error, which holds w.p.  $\geq 1 - \delta$

1. For each parameter  $\theta \in \Theta$  and number  $\tau$  of clusters,  $2 \leq \tau \leq C$ , apply  $\mathcal{A}(\tau, \theta)$  on  $X_{m+u}$  to generate  $(C - 1)T$  partitions  $\{\mathcal{C}_{\tau, \theta}\}$ .
2. Use the training set to generate a set  $\{h_{\tau, \theta}\}$  of  $(C - 1)T$  hypotheses corresponding to the partitions:
  - (a) For each partition  $\mathcal{C}_{\tau, \theta}$  consider each of its  $\tau$  clusters.
  - (b) For each cluster, set the labels of all its points to be the majority label of the training points falling in that cluster.
  - (c) If no labeled points fall into the cluster (or in case of ties), label the cluster according to the majority class in  $X_m$ . The resulting hypothesis is  $h_{\tau, \theta}$ .
3. For each hypothesis  $h_{\tau, \theta}$  calculate its empirical error  $R_{h_{\tau, \theta}}(X_m)$  and calculate its Vapnik (combined) bound based on the prior (10).
4. Output a classification of  $X_u$  using any hypothesis with the smallest Vapnik bound

Figure 2. CLUST pseudo-code

We now discuss the construction of the prior. Initially  $\mathcal{H}$  consists of all  $2^{m+u}$  possible dichotomies of the full sample. Let  $\mathcal{H}_{\tau, \theta} \subset \mathcal{H}$  be the set of all binary hypotheses that assign an identical label to all points in the same cluster in partition  $\mathcal{C}_{\tau, \theta}$ . Clearly,  $|\mathcal{H}_{\tau, \theta}| = 2^\tau$  for each  $\theta$ , because  $\mathcal{H}_{\tau, \theta}$  contains all possible dichotomies over  $\tau$  clusters. Without any prior information about the quality of the clustering as a function of the param-

eter  $\theta$ , each hypothesis in  $\mathcal{H}_\tau \triangleq \cup_\theta \mathcal{H}_{\tau,\theta}$  is assigned the same prior, which will be proportional to  $1/(T \cdot 2^\tau)$ . Also, without any additional information we do not know the “correct” number of clusters. Therefore, each of the  $C - 1$  subsets  $\mathcal{H}_\tau$ ,  $\tau = 2, \dots, C$ , is assigned the same probability mass,  $1/(C - 1)$ . The final prior distribution is, thus,

$$\mathbf{p}(h) = \begin{cases} \frac{1}{(C-1)T2^\tau}, & \text{if } h \in \mathcal{H}_\tau, \text{ for some } 2 \leq \tau \leq C; \\ 0, & \text{otherwise.} \end{cases} \quad (10)$$

To summarize, each of the partitions  $\mathcal{C}_{\tau,\theta}$  induces one hypothesis  $h_{\tau,\theta}$ , through the use of the training set. Clearly,  $\sum_{\tau,\theta} \mathbf{p}(h_{\tau,\theta}) = 1$ , and therefore, the Vapnik PAC-Bayesian bound applies. Similarly, using the prior (10) in (9) we have, that with probability at least  $1 - \delta$  (over random choices of the training set from the full sample), for all  $h \in \mathcal{H}_\tau$ ,

$$R_h(X_u) \leq R_h(X_m) + \sqrt{\left(\frac{m+u}{u}\right) \left(\frac{\tau + \log((C-1)T) + \ln \frac{1}{\delta}}{m}\right)}. \quad (11)$$

It can be shown that the right hand side of (11) upper bounds the Vapnik bound. From (11) it follows that if we are lucky enough to compute a “quality clustering” of the data, which gives rise to an hypothesis with a small training error, based on a small number of clusters, then we have a guarantee that the transductive test error of the corresponding hypothesis will be small. A nice property of this bound is that the divergence between the true and empirical risks increases only logarithmically with the number of models we consider (i.e.,  $(C - 1)T$ ). The pseudo-code in Figure 2 summarizes the CLUST algorithm.

#### 4.1. Which Clustering Algorithm?

A successful implementation of the CLUST algorithm may critically depend on the choice of a clustering algorithm  $\mathcal{A}(\tau, \theta)$ . Although the area of unsupervised learning is very active and offers a great many types of algorithms, the task of selecting a “suitable” clustering algorithm for a given data is ill-defined and cannot have a principled solution without considering the objective function. However, the strength of the CLUST algorithm should be exactly in its ability to consider *many* clusterings simultaneously and automatically choose a relatively good one! Therefore, we seek an algorithm that is reasonably fast, and flexible in the sense that it can generate a diversity of models based on a diversity of metrics. Particularly convenient candidates are the *spectral clustering* (or kernel PCA) algorithms, which are reasonably fast and well-motivated, and flexible in the sense that they use a

kernel to assign the initial pairwise dissimilarities. For our implementation we (rather arbitrarily) selected the spectral algorithm of (Ng et al., 2002). This algorithm uses an RBF kernel to assign pairwise dissimilarities and has one parameter  $\sigma$  (which here is interchangeably referred to as  $\theta$ ). It can also cluster the data into any desired number  $\tau$  of clusters. Due to lack of space we refer the reader to (Ng et al., 2002) for the details of this algorithm.

In all the experiments described below we applied the spectral clustering algorithm with  $C = 15$  and we fixed  $\Theta = \{10^{-3}, 10^{-2}, 10^{-1}, 10^0, 10^1\}$ . That is, we clustered the data to  $\tau = 2, \dots, 15$  clusters and for each  $\tau$  we attempted five different metrics. These parameter sets were chosen and fixed before we applied the algorithm to any of the datasets.<sup>8</sup> Overall, the CLUST algorithm attempted  $5 \times 14 = 70$  unsupervised models (using the spectral clustering algorithm) in each of the applications described below. In all cases most of these unsupervised models were very unsuccessful, but for most datasets this diversity of models often included a small number of high quality models.

#### 4.2. Example

Before we present our empirical study of the CLUST algorithm, we describe in some detail one run of the algorithm. Consider Figure 3 (see the figure’s caption for a description of the five panels). The figure shows a particularly successful application of the algorithm, where the final selection is of the best hypothesis among all the 70 hypotheses that were generated.<sup>9</sup> The selected hypothesis (circled) does not have the smallest empirical (train) error and we see that the Vapnik bound did protect the algorithm from selecting a number of hypotheses with a lower (or similar) training error, but with a significantly larger test error. Another striking fact is that most of the clusterings were extremely bad and did not yield useful hypotheses (for example, see the ones corresponding to small  $\sigma$ ). The algorithm easily avoided all those “irrelevant” hypotheses. On the down side, we see that the Vapnik bound was not sufficiently tight to provide the final hypothesis with a useful “performance guarantee” (w.h.p.). In particular, the bound corresponding to the best hypothesis is

<sup>8</sup>However, we clearly used some “prior knowledge” in selecting  $\Theta$ . Some experience with SVMs indicates that such a logarithmic “grid” ( $\Theta$ ) of possible values for the RBF “width” ( $\sigma$ ) is often sufficient to achieve high quality results when applying (supervised) SVMs on datasets of medium sized dimension, as those we consider.

<sup>9</sup>This run corresponds to one of the folds (among the five; see below). Another fold was also particularly successful and in the other three the algorithm’s performance was less than perfect.

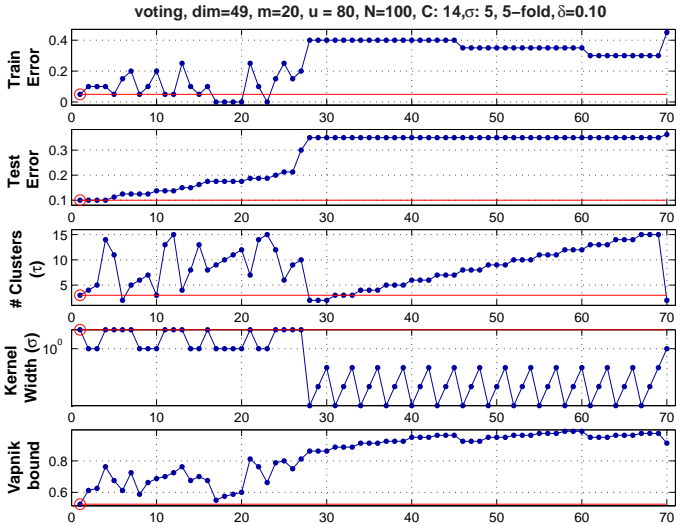


Figure 3. Application of the CLUST algorithm on the Voting dataset with  $m = 20$  and  $u = 80$ . The five panels show (from top to bottom), for all the  $14 \times 5 = 70$  hypotheses generated, their training error, test error, number of clusters ( $\tau$ ) and kernel width ( $\sigma$ ) used by the spectral clustering algorithm, and the Vapnik bound. The hypotheses are ordered (on the  $x$ -axis of each panel) according to increasing magnitude of the transduction risk (test error). The hypothesis selected by CLUST is circled.

slightly larger than 0.5. Unfortunately, this behavior was typical in all the experiments we conducted and is partly due to the very small sample size we used ( $m = 20$  and  $u = 80$  in the current example) and the relatively large number of hypotheses, which increases the slack of the bound (due to the use of the union bound in Equation (7)).

## 5. Experiments

We tested the CLUST algorithm on standard learning problems from the UCI repository. In order to minimize the risk of over-fitting due to “dataset selection,” we committed ourselves before the start of the experiment to the entire collection of datasets used by (Blum & Chawla, 2001) in their paper on the transductive ‘graph-mincut’ algorithm.<sup>10</sup> This benchmark consists of eight datasets from the UCI repository. Some properties of these sets are summarized in Table 1. Since the spectral clustering algorithm we employ assumes numerical vectors, we transformed nominal features to numerical ones. This was done in a standard way

<sup>10</sup>We have not tested the CLUST algorithm on other datasets and we report below on all our results.

Table 1. The datasets. (‘MUSH’ is the “mushroom” dataset.) The dimension is the original number of features. In cases where the original features are nominal (indicated in last column) the effective dimension, after transforming to numerical features, appears in parenthesis; ‘Bias’ is the proportion of the majority class as measured on the full sample  $X_{m+u}$ .

DATASET	DIMENSION	BIAS	NOMINAL?
MUSH	21 (117)	48%	✓
TAE	5 (57)	20%	✓
VOTING	16 (48)	38%	✓
MUSK	166	39%	×
PIMA	8	35%	×
IONOSPHERE	34	42%	×
BUPA	6	41%	×
MI	7 (17)	47%	✓

through the use of binary indicator attributes. We also normalized each attribute independently so that the dynamic range of the attributes was  $[0, 1]$ . No other preprocessing was used.

Using this collection of problems we conducted a rather extensive experiment, focusing on a very hard setting with extremely small sample size. For each dataset, we examined the performance of the CLUST algorithm over nine  $m:u$  partitions that were constructed from a full sample of  $m + u = 100$  points selected uniformly at random from the original dataset. The  $m:u$  partitions we examined are: 10:90, 20:80, ..., 90:10. For each  $m:u$  partition we provided the algorithm with the unlabeled full sample consisting of 100 points together with a labeled training set of size  $m$  that was selected randomly and uniformly among all  $m$ -subsets. We repeated this random choice of the training set five times and all the results reported here are averages over these five folds.

### 5.1. Benchmark Algorithms

As our main benchmark algorithm we selected the Spectral Graph Transducer (SGT) algorithm presented recently in (Joachims, 2003). This sophisticated algorithm appears to be one of the best transductive learning algorithms known today, as judged by the empirical study presented by Joachims. In particular, SGT clearly shows an advantage over inductive learning algorithms and in many cases it beats other well known transductive algorithms such as the transductive support vector machine (TSVM) variant of (Joachims, 1999), the graph-mincut algorithm of (Blum & Chawla, 2001) and the co-training algorithm of (Blum & Mitchell, 1998), which can be viewed as a special case of SGT.

The SGT algorithm has a number of parameters. As indicated by a sensitivity analysis presented in (Joachims, 2003), all but one of these parameters are not critical. However, one parameter ( $k$ , which determines the number of nearest neighbors of each point, for the purpose of graph construction) is critical. Joachims discovered a nice heuristic to determine a good value for this parameter, based on a solution for the optimization problem solved by the SGT algorithm. In all the experiments we conducted with SGT we assigned the value of  $k$  using the same heuristic as used by Joachims in his paper. We set the other parameters to their recommended values.<sup>11</sup> The assignment for  $k$  was chosen to be the best one (according to Joachims’ heuristic) among all 100 possible choices in  $\{1, 2, \dots, 100\}$  (recall that the size of our full sample is always 100).

In order to gain some perspective on the results we also tested and report on the test errors achieved by two extreme algorithms. The first is a trivial algorithm that labels the test set according to the majority class *as observed on the training set*. We denote this algorithm by NAIVE. Clearly any useful transductive learning algorithm must not consistently underperform NAIVE. The second benchmark is the best hypothesis *in hindsight* generated by algorithm CLUST. This benchmark is denoted by CLUST\*. We also report on the best hypothesis *in hindsight* considered by SGT (among all 100 models) and denote it by SGT\*.

Table 2. 5-fold average errors ( $\pm$  standard error of the mean) for  $m = 20$  and  $u = 80$ . The lowest error among the three algorithms (CLUST, SGT and NAIVE) appears in boldface. Also, the best among CLUST\* and SGT\* appears in boldface.

	CLUST	SGT	NAIVE	CLUST*	SGT*
MUSH	<b>.11<math>\pm</math>.04</b>	.12 $\pm$ .04	.53 $\pm$ .01	.07 $\pm$ .00	<b>.06<math>\pm</math>.01</b>
TAE	<b>.17<math>\pm</math>.01</b>	.28 $\pm$ .03	.19 $\pm$ .01	<b>.17<math>\pm</math>.00</b>	.18 $\pm$ .01
VOTING	.13 $\pm$ .01	.13 $\pm$ .02	.37 $\pm$ .01	.10 $\pm$ .00	<b>.08<math>\pm</math>.01</b>
MUSK	.45 $\pm$ .03	<b>.42<math>\pm</math>.01</b>	.45 $\pm$ .05	.32 $\pm$ .03	<b>.31<math>\pm</math>.02</b>
PIMA	<b>.31<math>\pm</math>.03</b>	.39 $\pm$ .01	.36 $\pm$ .01	<b>.27<math>\pm</math>.02</b>	.29 $\pm$ .01
IONO	<b>.24<math>\pm</math>.03</b>	.33 $\pm$ .04	.48 $\pm$ .04	<b>.16<math>\pm</math>.04</b>	.25 $\pm$ .02
BUPA	.43 $\pm$ .03	<b>.42<math>\pm</math>.02</b>	.46 $\pm$ .04	.38 $\pm$ .02	<b>.31<math>\pm</math>.02</b>
MI	.47 $\pm$ .02	<b>.41<math>\pm</math>.01</b>	.51 $\pm$ .02	.35 $\pm$ .01	<b>.32<math>\pm</math>.02</b>

## 5.2. Empirical Observations

As mentioned, overall we conducted,  $9 \times 8 = 72$  experiments, with 9 different  $m:u$  partitions and 8 datasets. Each experiment included 5 folds.

Table 2 specifies the average errors (over the 5-folds)

<sup>11</sup>Specifically, we used  $D = 80$  and  $C = 3200$ . Also, note that we used Joachims’ code to run the SGT algorithm. This code can be downloaded from Joachims’ web site.

Table 3. “Total” average errors ( $\pm$  averaged standard errors of the mean). Each entry is an average of the 9 5-fold averages, corresponding to the nine  $m : u$  partitions. Best results appear in boldface as described in Table 2.

	CLUST	SGT	NAIVE	CLUST*	SGT*
MUSH	.09 $\pm$ .02	.09 $\pm$ .02	.54 $\pm$ .02	.07 $\pm$ .02	<b>.05<math>\pm</math>.01</b>
TAE	<b>.16<math>\pm</math>.03</b>	.23 $\pm$ .04	.17 $\pm$ .02	.13 $\pm$ .01	<b>.12<math>\pm</math>.01</b>
VOTING	.14 $\pm$ .02	<b>.13<math>\pm</math>.03</b>	.39 $\pm$ .05	.10 $\pm$ .01	<b>.06<math>\pm</math>.01</b>
MUSK	.44 $\pm$ .04	<b>.38<math>\pm</math>.03</b>	.42 $\pm$ .05	.29 $\pm$ .04	<b>.24<math>\pm</math>.02</b>
PIMA	<b>.29<math>\pm</math>.03</b>	.39 $\pm$ .03	.35 $\pm$ .03	.25 $\pm$ .03	<b>.24<math>\pm</math>.02</b>
IONO	<b>.16<math>\pm</math>.03</b>	.33 $\pm$ .03	.43 $\pm$ .03	<b>.08<math>\pm</math>.02</b>	.21 $\pm$ .02
BUPA	.43 $\pm$ .03	<b>.41<math>\pm</math>.03</b>	.42 $\pm$ .03	.33 $\pm$ .02	<b>.24<math>\pm</math>.01</b>
MI	.41 $\pm$ .03	<b>.31<math>\pm</math>.02</b>	.53 $\pm$ .03	.25 $\pm$ .02	<b>.19<math>\pm</math>.02</b>

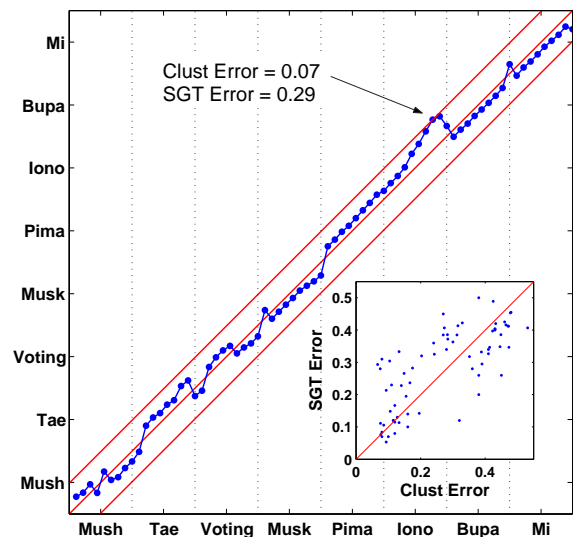


Figure 4. Two scatter plots. Each point represent one experiment among the 72 experiments conducted. The subplot is a standard scatter plot of the algorithms’ errors. For a description of the main plot, see the text.

suffered by the various algorithms, for train/test partitions with  $m = 20$  and  $u = 80$ . Such partitions, with very small training sets, are often of practical interest. We see that CLUST is significantly better than SGT in 3 datasets (TAE, PIMA and IONO). SGT is significantly better than CLUST in one dataset (MI). In most cases both CLUST and SGT are significantly better than NAIVE. In one case (MUSK) CLUST is no better than NAIVE but SGT is significantly worse than NAIVE in two cases (TAE and PIMA). In the majority of sets SGT\* achieved lower test error than CLUST\*.

Table 3 provides a general view on the entire set of experiments. The table specifies the average errors of the various algorithms, where the average is taken with respect to the errors obtained in all nine  $m:u$

partitions. Here we see that both CLUST and SGT significantly beat each other exactly three times. We also see that both CLUST and SGT lose twice to NAIVE. A rough conclusion from this table is that no algorithm is better than the other and that each of the algorithms excels in different learning problems. We also see that the performance of SGT\* is significantly better than that of CLUST\*. Except for the IONO set, where SGT\* (and SGT) are particularly bad, on all other sets SGT\* achieves smaller error rates. Perhaps this is not so surprising because for the computation of SGT\* we considered 100 different models whereas for CLUST\*, only 70 models. In any case, the relative success of SGT\* (and of CLUST\*) may indicate that there is still room for significant improvements of both algorithms!

The two scatter plots in Figure 4 provide somewhat different general views of the experiments. Each point in the main scatter plot corresponds to one of the 72 experiments (and by itself represent an average over five folds). The experiments are ordered by datasets as indicated by the labels of the  $x$ -axis (and the  $y$ -axis). Within each dataset range, the 9 experiments are ordered in increasing size of  $m$  (each corresponding to one of the  $m:u$  partitions). Suppose that in the  $i$ th experiment (in this order) the test error of CLUST is  $a$  and the error of SGT is  $b$ . If  $a < b$  then the point corresponding to this experiment is located at the coordinate  $(i, i + b/a)$ . Otherwise, the point is located at  $(i, i - a/b)$ . Thus, a point below the diagonal represents an advantage for SGT and vice versa. The subplot in Figure 4 is a standard scatter plot. Here again there is a point for each experiment and the location is simply determined by the error rates. From these plots we see again that each of the algorithms is an overall winner on 3 different datasets and there is a mixed behavior on the other two sets. On the IONO set we observe a major advantage of CLUST over SGT. There is no such major advantage of SGT on other datasets. From the standard scatter (sub)plot we can see that there is rather weak positive correlation between the behaviors of the two algorithms over the set of experiments.

## 6. Concluding Remarks

We studied a new learning scheme for transduction based on clustering. Unlike many other transductive learning heuristics the new scheme is theoretically well-founded. Our empirical study indicates that the CLUST algorithm achieves state-of-the-art performance, and we conclude that it is competitive with the the recent SGT algorithm (while being considerably simpler than SGT).

Our study concerned a straightforward implementa-

tion of the CLUST approach. We made no attempts to improve the basic scheme. It would be interesting to consider more sophisticated prior constructions, perhaps ones which are based on more extensive knowledge that can be derived from the full sample geometry. Also, it would be interesting to see if the SGT algorithm can benefit from PAC-Bayesian model selection.

Transductive inference is an intriguing type of learning that is not well understood. We believe that transduction offers new opportunities for machine learning research. In any case, while the precise relation between the inductive and transductive settings remains unclear, the interest in transduction should not be underestimated. If transduction allows for faster (and/or computationally more efficient) learning than the best possible inductive learning, then much more research attention should be devoted to transduction.

## References

- Blum, A., & Chawla, S. (2001). Learning from labeled and unlabeled data using graph mincuts. *ICML-2001*.
- Blum, A., & Langford, J. (2003). PAC-MDL Bounds. *COLT-2003* (pp. 344–357).
- Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. *COLT-1998*.
- Bottou, L., Cortes, C., & Vapnik, V. (1994). *On the effective vc dimension* (Technical Report). Neuroprose (<ftp://archive.cis.ohio-state.edu/pub/neuroprose>). Also on <http://leon.bottou.com/publications>.
- Derbeko, P., El-Yaniv, R., & Meir, R. (2004). Error bounds for transductive learning via compression and clustering. *NIPS 16, to appear*.
- Joachims, T. (1999). Transductive inference for text classification using support vector machines. *ICML-1999*.
- Joachims, T. (2003). Transductive learning via spectral graph partitioning. *ICML-2003*.
- McAllester, D. (1999). Some PAC-Bayesian theorems. *Machine Learning*, 37(3), 355–363.
- Ng, A., Jordan, M., & Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. *NIPS 14* (pp. 849–856).
- Seeger, M. (2002). *Learning with labeled and unlabeled data* (Technical Report). Available at <http://www.dai.ed.ac.uk/homes/seeger/papers/>.
- Vapnik, V. N. (1982). *Estimation of Dependences Based on Empirical Data*. New York: Springer Verlag.
- Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. New York: Springer Verlag.
- Vapnik, V. N. (1998). *Statistical Learning Theory*. New York: Wiley Interscience.