Learning with the Set Covering Machine

Mario Marchand MARCHAND@SITE.UOTTAWA.CA
School of Information Technology and Engineering, University of Ottawa, Ottawa, Ont., Canada, K1N-6N5

John Shawe-Taylor JOHN@DCS.RHBNC.AC.UK
Department of Computer Science, Royal Holloway, University of London, Egham, UK, TW20-0EX

Abstract

We generalize the classical algorithms of Valiant and Haussler for learning conjunctions and disjunctions of Boolean attributes to the problem of learning these functions over arbitrary sets of features; including features that are constructed from the data. The result is a general-purpose learning machine, suitable for practical learning tasks, that we call the Set Covering Machine. We present a version of the Set Covering Machine that uses *generalized balls* for its set of data-dependent features and compare its performance to the famous Support Vector Machine. By extending a technique pioneered by Littlestone and Warmuth, we bound its generalization error as function of the amount of data compression it achieves during training.

1. Motivation

We may attribute the effectiveness of Support Vector Machines (Vapnik, 1998; Cristianini & Shawe-Taylor, 2000) to the fact that they combine two very good ideas. First, they map the space of input vectors onto a very high-dimensional feature space in such a way that nonlinear decision functions on the input space can be constructed by using only hyperplanes on the feature space. Second, they construct the separating hyperplane on the feature space which has the largest possible margin. Theoretical results on margin classifiers (Shawe-Taylor et al., 1998) imply that good generalization is expected whenever a large margin separating hyperplane can be found. However, for a given set of features, there might exists solutions that provide better generalization than the maximum margin solution. In particular, if the function to learn happens to depend only on a very small subset of a large set of given features, then it might be better to construct a simple decision function that depends only on a small subset of features than to find the maximum margin hyperplane on the set of all features.

For a given set of (data-independent) features, Valiant (1984) has proposed a very simple learning algorithm for building a conjunction from positive examples only (or building a disjunction from negative examples only). However, the obtained function might contain a lot of features. To reduce their number, Haussler (1988) has proposed to use the classical greedy set covering algorithm (see Kearns & Vazirani (1994) for a clear exposition of these algorithms) on the remaining examples that are not used by the Valiant algorithm. It was shown by Haussler (1988) that good generalization of this algorithm is expected whenever there exists a small conjunction (or disjunction) of few relevant (data-independent) features that makes zero error with the training examples.

Because conjunctions and disjunctions are (just) subsets of the set of linearly separable functions, we might believe that the maximum margin solution will generally exhibit better generalization than a small conjunction (or disjunction) obtained from the Valiant-Haussler algorithm. To investigate this possibility we have performed the following experiment. Each (training and testing) example is a boolean vector of 80 attributes who's class (or label) is determined according to the conjunction $x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5$. Hence the function to learn is a conjunction of 5 attributes out of 80. Each positive example was generated by fixing x_i to 1 for $i = 1 \cdots 5$ and by choosing a value for x_i in $\{0, 1\}$ uniformly and independently for $i = 6 \cdots 80$. Each negative example was generated by first choosing a value for x_i in $\{0, 1\}$ uniformly and independently for $i = 1 \cdots 80$ and accepting it only if at least one x_i is set to 0 for some $i \in \{1 \cdots 5\}$. The accuracy of both the maximum margin solution and the Valiant-Haussler solution was measured with respect to a testing set of 10000 examples—half of which are positive examples. The results are plotted on figure 1 for various training

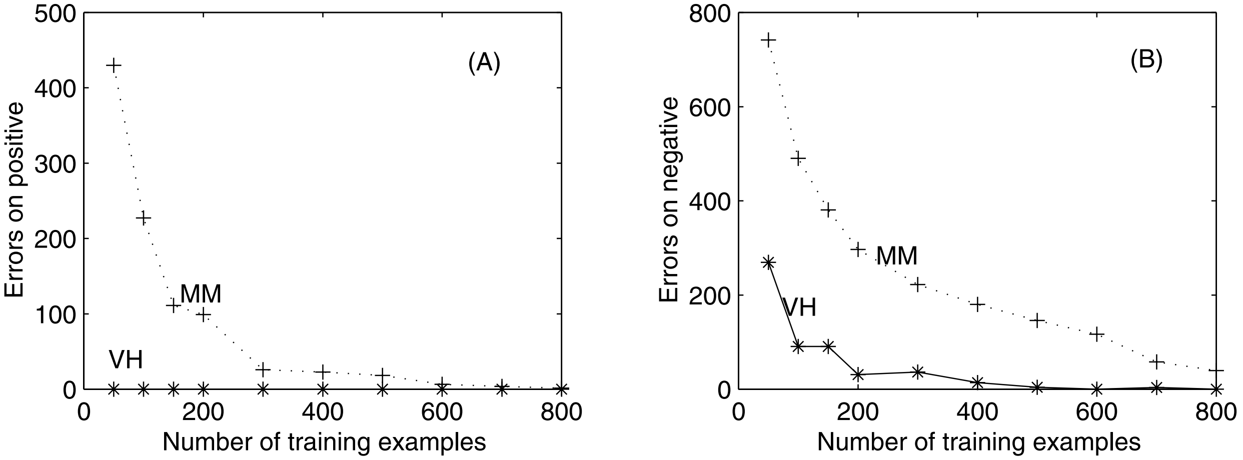

Figure 1. Comparison of the maximum margin solution (MM) with the Valiant-Haussler set covering greedy solution (VH)

sets sizes—each containing an equal amount of positive and negative examples.

For both solutions (or algorithms), the errors made on the 5000 positive testing examples are reported separately (on figure 1a) from the the errors made on the 5000 negative testing examples (on figure 1b). Each point denotes the average over 20 different training sets of the same size. We see that the VH solution is substantially better than the MM solution on both the positive and the negative testing examples. It is therefore worthwhile to explore the learning capabilities of a general purpose learning machine, that we call the Set Covering Machine (SCM), which uses the Valiant-Haussler greedy algorithm as its main optimization tool to construct a conjunction (or disjunction) of a small number of relevant features.

2. The Set Covering Machine

The learning algorithms of Valiant and Haussler are constructing a conjunction (or disjunction) *on the original set of input variables*. Since the values of input variables in many data sets are often non Boolean and since conjunctions and disjunctions are quite limited sets of functions, these algorithms cannot be used directly as general-purpose "practical" learning algorithms.

To fix this problem we will extend the power of conjunctions and disjunctions by using the same idea that was used in Support Vector Machines to extend the power of separating hyperplanes: we will map the original input space into a high-dimensional feature space and then construct simple conjunctions (or disjunctions) on that feature space. In this way, simple Boolean functions on the feature space may represent complex decision functions on the original input space. Moreover, the original input space need not be Boolean even if the high-dimensional feature space must be Boolean in order to construct conjunctions or disjunctions on that space. Finally, since the algorithms of Valiant and Haussler manipulate features directly, we will need to define the mapping that the SCM will used. This is to be contrasted with Support Vector Machines where the mapping need only to be defined from the eigenvectors of a Mercer kernel.

Therefore, if $\mathbf{x}$ denotes an arbitrary n-dimensional vector of the input space. The SCM is a classifier that first maps each $\mathbf{x}$ to a s-dimensional Boolean feature vector $\mathbf{h}(\mathbf{x}) = (h_1(\mathbf{x}), h_2(\mathbf{x}), \cdots, h_s(\mathbf{x}))$ where s may be much larger than n and where each $h_i(\mathbf{x})$ is called a (Boolean-valued) *feature*. The Boolean-valued label (or class) computed by the SCM for an input vector $\mathbf{x}$ is a conjunction (or disjunction) of some (relevant) features $h_i(\mathbf{x})$ that was found by the algorithms of Valiant and Haussler. If $\mathcal{R}$ denotes the set of features returned by these algorithms, the output $f(\mathbf{x})$ of the SCM is given by:

$$f(\mathbf{x}) = \begin{cases} \bigvee_{i\in\mathcal{R}} h_i(\mathbf{x}) & \text{for a disjunction} \\ \bigwedge_{i\in\mathcal{R}} h_i(\mathbf{x}) & \text{for a conjunction} \end{cases}$$

Throughout the paper we write that a function (or a feature) is *consistent* with an example if it correctly classifies that example. Similarly, a function is said to be consistent with a set of examples if it correctly classifies all the examples in that set.

Given a set of m training examples, the learning algorithm for the SCM must also be provided with a set $\mathcal{S}$ of s Boolean valued features. There is no restriction on the chosen set $\mathcal{S}$ except that there must exist a conjunction (or disjunction) of features in $\mathcal{S}$ which is consistent with all the training examples. We will describe such a set of features in the next section. A *good* set $\mathcal{S}$ is one for which there exist a *small* subset of features whose conjunction (or disjunction) is consistent with all the training examples.

The learning algorithm consists of two phases: the *Valiant step* followed by the *Haussler step*. We describe each of these phases for the case of a conjunction. The disjunction case immediately follows by symmetry.

The Valiant Step: *Find the set $\mathcal{C}$ of all the features in $\mathcal{S}$ which are consistent with all the positive training examples.*

This set $\mathcal{C}$ has the property that $\bigwedge_{i\in\mathcal{C}} h_i(\mathbf{x})$ is consistent with *all* the m training examples. Indeed, by construction, this conjunction is consistent with all the positive training examples. Moreover, recall that, by hypothesis, there must exist a subset $\mathcal{E}$ of $\mathcal{S}$ such that $\bigwedge_{i\in\mathcal{E}} h_i(\mathbf{x})$ is consistent with *all* the m training examples. But since $\mathcal{C}$ is the set of features that are consistent only with the positive training set, we have that $\mathcal{E} \subseteq \mathcal{C}$. Thus $\bigwedge_{i\in\mathcal{C}} h_i(\mathbf{x})$ is consistent with all the negative training examples as well.

However, $|\mathcal{C}|$ might be very large and, as we will see below, the generalization error is expected to be small only for a classifier that contains few relevant features. Hence to reduce their number, first note that each feature in $\mathcal{C}$ is consistent with all the positive training examples and *some* negative training examples. Let Q_i denote the set of negative training examples that are consistent with feature h_i. Since $\bigwedge_{i\in\mathcal{C}} h_i(\mathbf{x})$ is consistent with the set $\mathcal{N}$ of all the negative training examples, the union $\bigcup_{i\in\mathcal{C}} Q_i$ must be equal to $\mathcal{N}$. We say that $\bigcup_{i\in\mathcal{C}} Q_i$ is a *cover* of $\mathcal{N}$. Therefore, the problem of finding the *smallest* subset of features in $\mathcal{C}$ whose conjunction is consistent with $\mathcal{N}$ is identical to the problem of finding the *smallest* collection $\mathcal{V}$ of sets Q_i for which $\bigcup_{i\in\mathcal{V}} Q_i$ is a cover of $\mathcal{N}$. This is the well known (and NP-complete) *Set Cover Problem*. Hence, it is hard to find the set cover of the minimum size but, fortunately, the well known *set cover greedy algorithm* has a good worst-case bound. If z denotes the smallest number of sets Q_i that covers $\mathcal{N}$, then the set cover greedy algorithm will always find a cover of at most $z\ln(|\mathcal{N}|)+1$ sets (Chvatal, 1979; Kearns & Vazirani, 1994). Note that this bound has no dependence on the number of subsets (or features) in $\mathcal{C}$ and has only a logarithmic dependence on $|\mathcal{N}|$. We will therefore use this algorithm to find a small number of features in $\mathcal{C}$ that covers $\mathcal{N}$ (hence the name: SCM).

The set covering greedy algorithm is a very simple algorithm: first choose the set Q_i which covers the largest number of elements in $\mathcal{N}$, remove from $\mathcal{N}$ and each $Q_{j\neq i}$ the elements that are in Q_i, then repeat this process of finding the set Q_k of largest cardinality and updating $\mathcal{N}$ and each $Q_{j\neq k}$ until there are no more elements in $\mathcal{N}$.

The Haussler Step: *Recall that $\mathcal{C}$ is the set of features returned by the Valiant step. Use the set covering greedy algorithm to find a small subset $\mathcal{R} \subseteq \mathcal{C}$ of features that covers all the negative training examples.*

The hypothesis returned by the SCM, $\bigwedge_{i\in\mathcal{R}} h_i(\mathbf{x})$ will be consistent with all the training examples and will contain a small set of features whenever *there exists* a small set of features in the initial set $\mathcal{S}$ of chosen features whose conjunction is consistent with all the training examples.

The SCM can be set to construct a disjunction (instead of a conjunction) with minimal changes. For the Valiant step, we just find the set $\mathcal{C}$ of all the features in $\mathcal{S}$ that are consistent with the training set of *negative* examples. By using the same arguments as above, we find that $\bigvee_{i\in\mathcal{C}} h_i(\mathbf{x})$ will be consistent with all the training examples. For the Haussler step, we use the greedy set covering algorithm to find a small subset $\mathcal{R} \subseteq \mathcal{C}$ of features that covers all the *positive* training examples. Then, again, we find that $\bigvee_{i\in\mathcal{R}} h_i(\mathbf{x})$ will be consistent with all the training examples.

3. A Set Covering Machine using generalized balls

The above description of the SCM applies for any chosen set $\mathcal{S}$ of Boolean-valued features. The only restriction on $\mathcal{S}$ is that there exists a conjunction (or disjunction) of some features in $\mathcal{S}$ which is consistent with all the training examples. Let us examine a simple data-dependent set of features which has this property.

For each training example $\mathbf{x}_i$ with label $y_i \in \{0,1\}$ and (real-valued) radius ρ, we define a feature $h_{i,\rho}$ to be the following ball centered on $\mathbf{x}_i$:

$$h_{i,\rho}(\mathbf{x}) \overset{\text{def}}{=} h_\rho(\mathbf{x},\mathbf{x}_i) = \begin{cases} y_i & \text{if } d(\mathbf{x},\mathbf{x}_i) \le \rho \\ \overline{y}_i & \text{otherwise} \end{cases}$$

Where $\overline{y}_i$ denotes the Boolean complement of y_i and $d(\mathbf{x},\mathbf{x}_i)$ denotes the distance between these two points. Since any metric can be used for d, we call $h_{i,\rho}$ a *generalized ball*. So far we have used only the L_1, L_2 and L_∞ metrics but it is certainly worthwhile to try to use other metrics that actually incorporate some knowledge about the learning task.

Hence, given a set $\mathcal{M}$ of m training examples, our initial set of features consists, in principle, of $\mathcal{S} = \bigcup_{i\in\mathcal{M}} \bigcup_{\rho\in[0,\infty[} h_{i,\rho}$. But obviously, for each training example $\mathbf{x}_i$, we need only to consider the set of $m-1$ distances $\{d(\mathbf{x}_i, \mathbf{x}_j)\}_{j\neq i}$. This reduces our initial set $\mathcal{S}$ to $O(m^2)$ features. There always exist a conjunction (or disjunction) of some features in that set $\mathcal{S}$ which is consistent with all the training examples as long as there does not exists two (contradictory) examples in $\mathcal{M}$ having $y_i \neq y_j$ and $\mathbf{x}_i = \mathbf{x}_j$.

In fact, we can reduce the size of $\mathcal{S}$ to exactly m features by exploiting the fact that the Haussler step is going to be executed after the Valiant step. Indeed, consider the conjunction case. Each feature in the set $\mathcal{C}$ of features returned by the Valiant step must be consistent with all the positive training examples. This implies that a generalized ball centered on a positive training example must have a radius large enough to enclosed all the positive training examples. For one such radius value, the set of negative training examples that are consistent with this ball defines the set Q_i of (negative) training examples that are covered by that ball and that will be used by the Haussler step. Hence, a larger radius that covers less negative training examples will always be penalized by the set covering greedy algorithm compared to a smaller radius that cover the same negative examples plus a few extra more. Hence, for each ball centered on a positive training example, only one radius value will be chosen by the Haussler step: the smallest radius that includes all the positive training examples. Similarly, for each negative training example, we need only to consider one radius value: the largest radius that excludes all the positive training examples. The analysis for the disjunction case gives a similar result: we need to consider only one radius for each training example; the radius of a ball centered on a positive training example is the largest one which excludes all the negative training examples and the radius of a ball centered on a negative training example is smallest one which includes all the negative training examples. Hence, for m training examples, we only need to consider exactly m generalized balls in the initial set $\mathcal{S}$ of features.

4. Constructing smaller SCMs

The SCM that we have described so far always return a function which is consistent with all the training examples (provided that we have no contradictory examples). However, in a noisy environment, the structural risk minimization principle (Vapnik, 1998) tells us (for our case) that a small conjunction which makes a few errors on the training set might give better generalization than a larger conjunction (with more features) which makes zero training error.

One way to include this flexibility into the SCM is to stop the set covering greedy algorithm when there remains a few more training examples to be covered. In this case, the SCM will contain fewer features and will make errors on those training examples that are not covered. But these examples all belong to the same class: negative for conjunctions, positive for disjunctions. This might be desired for non-symmetric loss functions but, in general, we do need to be able to make errors on training examples of both classes. In these situations, we propose to combine the early stopping of the greedy algorithm with the following heuristic.

First define $\mathcal{P}$ to be the subset of training examples for which each feature returned by the Valiant step must be consistent with. This is the set of positive training examples for the conjunction case or the set of negative training examples in the disjunction case. Similarly $\mathcal{N}$ is defined to be the set of examples that needs to be covered by the greedy set covering algorithm (this is the set of negative examples for the conjunction case). We want to allow a feature to make a few errors on $\mathcal{P}$ provided that much more examples in $\mathcal{N}$ can be covered by this feature. Hence, for a feature h, let us denote by Q_h the set of examples in $\mathcal{N}$ covered by feature h and by R_h the set of examples in $\mathcal{P}$ for which h makes an error on. Given that each example in $\mathcal{P}$ misclassified by h should decrease by some fixed *penalty* p the "usefulness" of feature h, we define the *value* V_h of feature h to be the number $|Q_h|$ of examples in $\mathcal{N}$ that h covers minus the number $|R_h|$ of examples in $\mathcal{P}$ misclassified by h times some penalty p:

$$V_h \overset{\text{def}}{=} |Q_h| - p \times |R_h|$$

Hence, the learning algorithm for the SCM is modified as follows. The Valiant step is skipped: we consider all the features in our initial set $\mathcal{S}$, including the features which make some errors on $\mathcal{P}$. Each feature $h \in \mathcal{S}$ is covering a subset Q_h of $\mathcal{N}$ and makes an error on a subset R_h of $\mathcal{P}$. For the Haussler step, we modify the set covering greedy algorithm in the following way. Instead of using the feature that covers the largest number of examples in $\mathcal{N}$, we use the feature h that has the highest value V_h. We removed from $\mathcal{N}$ and each $Q_{g\neq h}$ the elements that are in Q_h and we removed from each $R_{g\neq h}$ the elements that are in R_h. (We update each such set R_g because a feature g that makes an error on an example in $\mathcal{P}$ does not increase the error of the machine if another feature h is already making an error on that example.) We repeat this process of finding the feature h of largest value V_h and updating $\mathcal{N}$ and each $Q_{g\neq h}$, and each $R_{g\neq h}$ until only a few elements remain in $\mathcal{N}$ (early stopping the greedy). The best stopping point and the best value for the penalty parameter p are determined by cross-validation.

5. Bounds on the generalization error

We now develop the necessary framework to enable estimates of the generalization performance of the SCM that uses generalized balls for its set of features. We follow the probably approximately correct (pac) framework, which posits an underlying distribution generating the training data and defines the error of a hypothesis h as the probability $\mathrm{er}(h)$ that a randomly drawn test point is misclassified by h. In this framework bounds on the generalization error are required to hold with high probability, hence bounding the tail of the distribution of possible errors.

We seek a bound sensitive to the hypothesis selected. Bounds of this type are data-dependent in that they depend on properties of the solution obtained, in our case the number of features it uses. A general framework for analyzing data-dependent structural risk minimization is given in Shawe-Taylor et. al. (1998), where the so-called luckiness framework is introduced. The assessment of the generalization of the SCM will require a data-dependent analysis since the features used depend on the particular training examples. The analysis, however, can be made by extending an older result due to Littlestone and Warmuth (1986). See also Floyd and Warmuth (1995).

Littlestone and Warmuth show that if we can *reconstruct* our hypothesis from a small subset of the training examples – the so-called *compression set* – then with high confidence the generalization can be bounded in terms of the size of this set. Here we extend the Littlestone and Warmuth technique by using a compression scheme that allows to specify a subset of the compression set that can be treated separately for the reconstruction of the hypothesis. We begin by defining our expanded compression scheme.

Definition 5.1 *Consider an input space X and set H of Boolean valued functions on X. For a fixed integer m, let*

$$\mathcal{X} = (X \times \{0,1\})^m$$

be the set of training sets of size m with inputs from X. Given a learning algorithm

$$A : \mathcal{X} \longmapsto H,$$

which given a training set $S \in \mathcal{X}$ returns a hypothesis in the set H, a compression scheme for A is a map

$$\Lambda : S \longmapsto \Lambda(S) = (\Lambda'(S), \Lambda''(S)),$$

$$\textit{where } \Lambda'(S) \subset \Lambda''(S) \subset S$$

together with a reconstruction function

$$\Phi : (S', S'') \longmapsto h \in H$$

such that for any set $S \in \mathcal{X}$,

$$A(S) = \Phi(\Lambda(S))$$

Note that the definition implies that we can simply define $A(S)$ by the rule $A(S) = \Phi(\Lambda(S))$, but it is useful to distinguish the learning algorithm from the reconstruction scheme.

Theorem 5.1 *Let Λ be a compression scheme relative to a reconstruction function Φ for a learning algorithm A, with the property that the compression set is correctly classified by its reconstruction. For a training set S of m examples, let $d' = |\Lambda'(S)|$, and $d'' = |\Lambda''(S)|$. Let $k = \mathrm{er}_S(A(S))$ be the number of examples of S that are misclassified by the function $A(S)$. Then with probability $1-\delta$ over random training sets of size m, the generalization error $\epsilon(A(S))$ of the function $A(S)$ is bounded by*

$$\begin{aligned}\epsilon(A(S)) &\leq 1 - \exp\left\{\frac{-1}{m-d''-k}\left(\ln\binom{m}{d''}\right.\right.\\ &\qquad \left.\left. + \ln\binom{d''}{d'} + \ln\binom{m-d''}{k} + \ln\frac{m^2 d''}{\delta}\right)\right\}\\ &\leq \frac{1}{m-d''-k}\left\{d''\ln\left(\frac{em}{d''}\right) + d'\ln\left(\frac{ed''}{d'}\right)\right.\\ &\qquad \left. + k\ln\left(\frac{e(m-d'')}{k}\right) + \ln\frac{m^2 d''}{\delta}\right\}\end{aligned}$$

provided that $m > d'' + k$.

Proof. First consider a fixed set of integers, $d' < d'' < m$, and $k < m$. We will use the notation $\mathbf{i} = (i_1, \ldots, i_d)$ for a sequence of strictly increasing indices, $0 < i_1 < i_2 < \cdots < i_d \leq m$, where with $|\mathbf{i}|$ we denote the length d of the sequence. We denote with $\mathcal{I}$ the set of all such sequences. If the sequence $\mathbf{i}'$ contains all the indices in $\mathbf{i}$, we write $\mathbf{i} \subset \mathbf{i}'$. For $S \in \mathcal{X}$, $S_{\mathbf{i}} = ((x_{i_1}, y_{i_1}), \ldots, (x_{i_d}, y_{i_d}))$. Finally, let $\mathbf{i}(d) = (1, 2, \ldots, d)$.

We bound the following probability

$$P\Big\{S \in \mathcal{X} \quad : \quad |\Lambda'(S)| = d', |\Lambda''(S)| = d'',$$

$$
\begin{aligned}
& \qquad\qquad \mathrm{er}_S(A(S)) = k, \mathrm{er}(A(S)) \geq \epsilon \Big\} \\
&= P\Big\{ S \in \mathcal{X} : \bigcup_{\mathbf{i}' \subset \mathbf{i}'' \in \mathcal{I} : |\mathbf{i}'| = d', |\mathbf{i}''| = d''} \Big\{ \\
& \Lambda(S) = (S_{\mathbf{i}'}, S_{\mathbf{i}''}), \mathrm{er}_S(A(S)) = k, \mathrm{er}(A(S)) \geq \epsilon \Big\}\Big\} \\
&\leq \sum_{\mathbf{i}' \subset \mathbf{i}'' \in \mathcal{I} : |\mathbf{i}'| = d', |\mathbf{i}''| = d''} P\Big\{ S \in \mathcal{X} : \\
& \quad \Lambda(S) = (S_{\mathbf{i}'}, S_{\mathbf{i}''}), \mathrm{er}_S(A(S)) = k, \mathrm{er}(A(S)) \geq \epsilon \Big\} \\
&= \binom{m}{d''}\binom{d''}{d'} P\Big\{ S \in \mathcal{X} : \Lambda(S) = (S_{\mathbf{i}(d')}, S_{\mathbf{i}(d'')}), \\
& \qquad\qquad \mathrm{er}_S(A(S)) = k, \mathrm{er}(A(S)) \geq \epsilon \Big\} \\
&\leq \binom{m}{d''}\binom{d''}{d'}\binom{m-d''}{k} \\
& \qquad \times (1 - \mathrm{er}(A(S))^{m-d''-k} \mathrm{er}(A(S))^k \\
&\leq \binom{m}{d''}\binom{d''}{d'}\binom{m-d''}{k} (1-\epsilon)^{m-d''-k} \\
&\leq \left(\frac{em}{d''}\right)^{d''} \left(\frac{ed''}{d'}\right)^{d'} \left(\frac{e(m-d'')}{k}\right)^{k} \\
& \qquad \times \ \exp(-\epsilon(m-d''-k)),
\end{aligned}
$$

where line 3 follows from line 2 by the union bound, line 4 from line 3 by the invariance of the probabilities to permutations of the data, and line 5 from the probability that a function with error $\mathrm{er}(A(S))$ makes exactly k mistakes. We wish the sum of these probabilities over different choices of k, d' and d'' to be less than δ. The two bounds can therefore be obtained by taking the last two expressions, setting them equal to $\delta/(m^2 d'')$, and solving for ϵ. □

Let us now apply theorem 5.1 to the learning algorithm for the SCM with generalized balls. To phrase the next theorem in a form which applies to both conjunctions and disjunctions we define a $\mathcal{P}$-example to be a positive example in the conjunction case but a negative example in the disjunction case. A $\mathcal{N}$-example is just the opposite.

Theorem 5.2 *Suppose a SCM finds a solution using R features, R^+ of which are centered on $\mathcal{P}$-examples, with k training errors on a sample of size $m > 2R + k$. Then with probability $1-\delta$ over random draws of training sets, the generalization ϵ of the resulting classifier can be bounded by*

$$
\begin{aligned}
\epsilon \quad &\leq \quad 1 - \exp\left\{ \frac{-1}{m-2R-k} \left(\ln\binom{m}{2R} + \ln\binom{2R}{R^+} \right.\right. \\
& \qquad \left.\left. + \ln\binom{m-2R}{k} + \ln\frac{2m^2R}{\delta} \right) \right\} \\
&\leq \quad \frac{1}{m-2R-k} \left\{ 2R\ln\left(\frac{em}{2R}\right) + R^+ \ln\left(\frac{2eR}{R^+}\right) \right. \\
& \qquad \left. + k\ln\left(\frac{e(m-2R)}{k}\right) + \ln\frac{2m^2R}{\delta} \right\}.
\end{aligned}
$$

Proof. The proof relies on showing that, for the SCM learning algorithm A, there exists a compression scheme Λ and a reconstruction function Φ that satisfy:

1. $A(S) = \Phi(\Lambda(S))$, for all samples S, and
2. $|\Lambda''(S)| \leq 2R$, $|\Lambda'(S)| = R^+$.
3. The reconstruction correctly classifies the compression set.

The result will then follow from an application of Theorem 5.1.

The solution $A(S)$ involves R features centered on the sample points $\mathbf{x}_{s_1}, \ldots, \mathbf{x}_{s_R}$ (not necessarily in increasing index order). Assume without loss of generality that the first R^+ in the list are $\mathcal{P}$-examples. For each feature, the radius of the generalized ball is determined by a $\mathcal{P}$-example – in the case of the center being a $\mathcal{P}$-example, it is the $\mathcal{P}$-example point furthest from the center, while in the case of a $\mathcal{N}$-example center it is the $\mathcal{P}$-example point nearest to it. For the center $\mathbf{x}_{s_i}$, we will denote the point that determines the radius of its feature by $\mathbf{x}_{t_i}$, for $i = 1, \ldots, R$. The compression scheme is now defined by

$$
\Lambda''(S) = \{\mathbf{x}_{s_i}, \mathbf{x}_{t_i} : i = 1, \ldots, R\} \text{ and}
$$
$$
\Lambda'(S) = \{\mathbf{x}_{s_i} : i = 1, \ldots, R^+\}.
$$

The sizes of these sets are bounded by the required numbers. The reconstruction function Φ given a pair (S'', S') creates features by taking all the $\mathcal{N}$-examples in S'' as feature centers and setting the corresponding ρ by the nearest $\mathcal{P}$-example from S''. Features with a $\mathcal{P}$-example center are created for each point in S' with the radius set to the minimum that contains all the $\mathcal{P}$-examples in S''.

Suppose now that $\Lambda(S) = (S'', S')$. It is clear that the same centers are used in $\Phi(\Lambda(S))$ as were used in $A(S)$ and that furthermore they have the same radii. Hence, we have shown that $A(S) = \Phi(\Lambda(S))$ as required. □

6. Results on natural data

We have compared the practical performance of the SCM using the set of generalized balls with nearest-neighbor classifier (NNC) and the Support Vector Machine (SVM) equipped with a Gaussian kernel (also called the Radial Basis Function kernel) of variance $1/\gamma$. We have used the SVM program that is distributed by the Royal Holloway University of London. The data sets used and the results obtained for NNCs and SVMs are reported on table 1. All these data sets where obtained from the machine learning repository at UCI, except the Glass data set which was obtained from Rob Holte at the University of Ottawa. For each data set, we have removed all examples that contained attributes with unknown values (this has reduced substantially the "votes" data set) and we have removed examples with contradictory labels (this occurred only for a few examples in the Haberman data set). The remaining number of examples for each data set is reported in table 1. For all these data sets, we have used the 10-fold cross validation error as an estimate of the generalization error. The values reported are expressed as the total number of errors (*i.e.* the sum of errors over all testing sets). We have ensured that each training set and each testing set, used in the 10-fold cross validation process, was the same for each learning machine (*i.e.* each machine was trained on the same training sets and tested on the same testing sets).

For the SVM, the values of the kernel parameter γ and the soft margin parameter C are the ones that gave the smallest 10-fold cross validation error among an *exhaustive* scan of *many* values for these parameters. We have also reported the average number of support vectors (in the "size" columns) contained in machines obtained from the 10 different training sets. Note that SVMs performed significantly better than NNCs only for the soft margin case (finite values of C).

The results for the SCM are reported in table 2. Despite the fact that we have observed that results with the L_1 metric are often better, we have only reported here the results for the L_2 metric. This was done to obtain a fair comparison with SVMs because the argument of the Radial Basis Function kernel is given by the L_2 metric between two input vectors. The L_2 metric was also used for NNCs.

We have reported separately the results for *Consistent SCMs*, *Truncated SCMs*, and *Inconsistent SCMs*. A Consistent SCM is a machined that was trained until it made zero training errors. We see that these are the largest machines (the size is expressed as the number of generalized balls) and that they often have the largest generalization error. A Truncated SCM is a machine that was trained with an early stopping of the set covering greedy algorithm. Such a machine makes training errors only on the set $\mathcal{N}$ (negative examples for the conjunction case). We see that this is often effective in reducing both the machine size and its generalization error. Finally, an Inconsistent SCM is a machine that was permitted to make some training errors on the set $\mathcal{P}$ (positive examples for the conjunction case) and for which the set covering greedy algorithm was early stopped. We have reported the optimal value of the penalty parameter for such machines. We see that this is often effective in reducing further both the machine size and its generalization error. For Consistent SCMs and Truncated SCMs, there is a substantial difference in the generalization error of disjunctions and conjunctions on some data sets (Pima, Haberman, Bupa). But this difference is substantially reduced for Inconsistent SCMs.

We see that the optimal generalization error that we have obtained for SCMs was often very close to the one obtained for SVMs. The performance of both machines was very similar on these data sets and both of them were significantly better than the NNC (except for the Glass and Votes data sets). However, the size of SCMs is substantially smaller than SVMs.

We have also reported, in the "bound" columns of table 2, the bound on the generalization error obtained by computing the r.h.s. of the first inequality of theorem 5.2 (with $\delta = .05$), for each of the 10 different training sets involved in 10-fold cross validation, and multiplying that bound with the size of each testing sets. We see that, although the bound is not tight, it is nevertheless non-trivial. This is to be contrasted with the VC dimension bounds which cannot even be applied for our case since the set of functions supported by the SCM depends on the training data. It would be interesting to investigate the extent to which our bound can perform SCM model selection.

7. Conclusion

Our theoretical and experimental results indicate that the SCM is a good candidate for practical machine learning tasks. We could try to extend the SCM by considering strictly larger classes of functions like decision lists of few relevant attributes (Dhagat & Hellerstein, 1994) or even linear threshold functions of few relevant attributes (Littlestone, 1988). However, conjunctions and disjunctions might give us all the power we need if the set of chosen features is sufficiently rich. So far, the SCM was tested only for generalized balls. The fact that our bounds on the generalization error only depends on the amount of data compression achieved by the learning algorithm (and the existence of a reconstruction function) suggests that it is worthwhile to consider larger sets of features if the size of the Set Covering Machine can be significantly reduced

Table 1. Data sets, NNC results, and SVM results.

Data Set	Number of examples		NNC	SVM ($C = \infty$)			SVM (finite C)			
	positive	negative	error	γ	size	error	γ	C	size	error
BreastWisc	239	444	29	0.07	219.7	27	0.005	2	57.7	19
Votes	18	34	7	0.05	16.6	5	0.05	15	18.2	**3**
Pima	269	499	247	0.004	543.7	243	0.002	1	526.2	203
Haberman	219	75	107	0.02	92.5	111	0.01	0.6	146.4	**71**
Bupa	145	200	124	0.02	290.4	121	0.002	0.2	265.9	107
Glass	87	76	36	0.2	47.2	42	0.8	2	91.8	34
Credit	296	357	214	0.001	406.9	205	0.0006	32	423.2	**190**

Table 2. SCM results for conjunctions (c) and disjunctions (d).

Data Set	Type	Consistent SCM			Truncated SCM			Inconsistent SCM			
		size	error	bound	size	error	bound	penalty	size	error	bound
BreastWisc	c	13.9	26	216	1	23	150	1.8	2	**15**	109
	d	16.8	34	236	10	34	189	0.5	1	17	117
Votes	c	3.7	7	33	1	6	25	1.2	1	6	25
	d	3.2	7	32	3	7	30	0.9	1	6	25
Pima	c	147.2	262	763	153	262	763	1.1	3	**189**	657
	d	124.8	230	751	20	208	672	4.1	16	204	670
Haberman	c	45.5	104	280	1	76	244	1.4	1	**71**	232
	d	46.6	128	283	39	127	273	1.4	12	**71**	251
Bupa	c	80.6	144	344	58	142	341	1.7	20	122	317
	d	68.8	131	342	46	118	332	2.8	9	**106**	307
Glass	c	19	45	141	7	40	119	0.85	4	**33**	113
	d	15.9	36	131	11	36	114	3.8	16	36	126
Credit	c	137.4	255	651	121	253	648	0.8	4	197	600
	d	123.5	225	647	79	217	628	1.2	4	194	590

by using these additional features.

References

Chvatal V. (1979) A Greedy Heuristic for the Set Covering Problem. *Mathematics of Operations Research* Vol. 4 (pp. 233–235).

Cristianini N., & Shawe-Taylor J. (2000). *An Introduction to Support Vector Machines*, Cambridge University Press.

Dhagat A., & Hellerstein L. (1994). PAC Learning with Irrelevant Attributes. *Proceedings of the 35th IEEE Conference on the Foundations of Computer Science (FOCS)* (pp. 64–74).

Floyd S., & Warmuth M (1995). Sample compression, learnability, and the Vapnik-Chervonenkis dimension. *Machine Learning* Vol. 21, (pp. 269–304).

Haussler D. (1988). Quantifying Inductive Bias: AI Learning Algorithms and Valiant's Learning Framework. *Artificial Intelligence* Vol. 36, (pp. 177–221).

Kearns M.J., & Vazirani U.V. (1994). *An Introduction to Computational Learning Theory.* Cambridge: MIT Press.

Littlestone N. (1988). Learning quickly when irrelevant attributes abound: A new linear threshold algorithm. *Machine Learning* Vol. 2, (pp. 285–318).

Littlestone N. & Warmuth M. (1986). Relating Data Compression and Learnability. Technical report, University of California Santa Cruz.

Shawe-Taylor J., Bartlett P., Williamson R., & Anthony M. (1998). Structural Risk Minimization over Data-Dependent Hierarchies. *IEEE Transactions on Information Theory* Vol. 44, (pp. 1926-1940).

Valiant L.G. (1984). A Theory of the Learnable. *Comm. ACM* Vol. 27, (pp. 1134–1142).

Vapnik V.N. (1998). *Statistical Learning Theory*, New York: Wiley.