A bound concerning the generalization ability of a certain class of learning algorithms

Yoram Gat
University of California, Berkeley

Technical report No. 548, March 1999

Abstract

A classifier is said to have good generalization ability if it performs on test data almost as well as it does on the training data. The main result of this paper provides a sufficient condition for a learning algorithm to have good finite sample generalization ability. This criterion applies in some cases where the set of all possible classifiers has infinite VC dimension. We apply the result to prove the good generalization ability of support vector machines.

Introduction

I consider the classical problem of learning a classifier from examples which can be formalized as follows: Let $Z_i = (X_i, Y_i), i = 1, 2, \ldots$ be iid random variables taking values in $\mathcal{Z} = \mathcal{X} \times \{-1, +1\}$. The problem is predicting Y_{l+1} given $X_1, \ldots, X_{l+1}$ and $Y_1, \ldots, Y_l$.

The solution to the problem is a map $M : \mathcal{Z}^l \to \mathcal{F}$, where $\mathcal{F}$ is a space of classifier functions, i.e., each $f \in \mathcal{F}$ is a function $f : \mathcal{X} \to \{-1, +1\}$. Thus the prediction is $Y^*_{l+1} = f^*(X_{l+1})$ where $f^* = M(Z_1, \ldots, Z_l)$.

The quality of the solution may be measured using the expected error rate (also called expected risk):

$$\mathrm{EXER} = \mathbf{P}(Y^*_{l+1} \neq Y_{l+1}).$$

The solution M is usually geared toward finding a function which has low empirical error rate (also called empirical risk):

$$\mathrm{EMER} = \frac{1}{2l} \sum_{i=1}^{l} |f^*(X_i) - Y_i| \, .$$

Therefore, it is often desirable to be able to obtain bounds for the difference between the empirical and the expected error rates. The behavior of the difference will depend on the underlying, unknown probability measure. The term generalization ability is used to describe the worst-case behavior of the difference between the empirical and expected error rate for a specific algorithm. The

AMS 1991 subject classifications. Primary 62H30.
Key words and phrases. Generalization ability, support vector machines, VC dimension, perceptron algorithm.

smaller the probability for a large difference, the better is the generalization ability of the algorithm.

One map M commonly used is

$$M(Z_1, \ldots, Z_l) = \arg \min_{f \in \mathcal{F}} \sum_{i=1}^{l} |Y_i - f(X_i)| \,.$$

This is known as the Empirical Risk Minimization (ERM) method. It has been shown that the generalization ability of the algorithm can be determined by using the VC dimension of the set of functions $\mathcal{F}$ ([1] sec. 4.9).

Other learning algorithms use maps of the form

$$M(Z_1, \ldots, Z_l) = \arg \min_{f \in \tilde{M}(Z_1, \ldots, Z_l)} \sum_{i=1}^{l} |Y_i - f(X_i)| \,,$$

where $\tilde{M}$ is an auxiliary map $\tilde{M} : \mathcal{X}^l \to 2^{\mathcal{F}}$. I call this type of algorithms Restricted Empirical Risk Minimization (RERM) rules.

The main result

The following theorem guarantees the generalization ability of certain learning algorithms even when $\mathcal{F}$ has an infinite VC dimension:

Theorem 1 *Denote*

$$\begin{aligned} \overline{M}(z_1, \ldots, z_{2l}) \quad &= \quad \big\{ M(z_{i(1)}, \ldots, z_{i(l)}) : \textit{the } i(j)\textit{'s are } l \textit{ distinct indices} \\ &\qquad\qquad \textit{in the range } 1, \ldots, 2l \big\} \,. \end{aligned}$$

If

$$\sup_{z_1, \ldots, z_{2l} \in \mathcal{Z}} \left| \overline{M}(z_1, \ldots, z_{2l}) \right| = c(l),$$

then

$$\mathbf{P}(|\mathrm{EXER} - \mathrm{EMER}| > \epsilon) < 2c(l) \exp -(l\epsilon^2 - 2\epsilon).$$

Proof: Since for any Binomial variable, B, $\mathbf{P}(B > \mathbf{E}B + 1) < 0.5$, it is enough to bound

$$p_{\epsilon'} = \mathbf{P}(\left| \frac{1}{2l} \sum_{i=l+1}^{2l} |f^*(X_i) - Y_i| - \mathrm{EMER} \right| > \epsilon'),$$

where $\epsilon' = \epsilon - \frac{1}{l}$. This is done by conditioning on the values of z_i, $i = 1, \ldots, 2l$ and then taking the expectation over the different possible orders.

To simplify the formulas, I use below $\Delta^f(z)$ as shorthand for $|f(x)-y|/2$, where $z=(x,y)$. Thus $\Delta^f(z)$ is either 0 or 1.

$$p_{\epsilon'} \;=\; \mathbf{E}\frac{1}{(2l)!}\sum_{\sigma}\mathbf{1}(\left|\sum_{i=1}^{l}\Delta^{f^*}(Z_{\sigma(i)})-\sum_{i=l+1}^{2l}\Delta^{f^*}(Z_{\sigma(i)})\right|>l\epsilon').$$

Here, as below, $\sum_\sigma$ means summing over all permutations of the numbers $1,\ldots,2l$.

$$\begin{aligned}
p_{\epsilon'} &\leq \mathbf{E}\frac{1}{(2l)!}\sum_{\sigma}\mathbf{1}(\sup_{f\in\overline{M}(Z_1,\ldots,Z_{2l})}\left|\sum_{i=1}^{l}\Delta^{f}(Z_{\sigma(i)})-\sum_{i=l+1}^{2l}\Delta^{f}(Z_{\sigma(i)})\right|>l\epsilon')\\
&\leq \mathbf{E}\frac{1}{(2l)!}\sum_{\sigma}\sum_{f\in\overline{M}(Z_1,\ldots,Z_{2l})}\mathbf{1}(\left|\sum_{i=1}^{l}\Delta^{f}(Z_{\sigma(i)})-\sum_{i=l+1}^{2l}\Delta^{f}(Z_{\sigma(i)})\right|>l\epsilon')\\
&\leq \mathbf{E}\sum_{f\in\overline{M}(Z_1,\ldots,Z_{2l})}\frac{1}{(2l)!}\sum_{\sigma}\mathbf{1}(\left|\sum_{i=1}^{l}\Delta^{f}(Z_{\sigma(i)})-\sum_{i=l+1}^{2l}\Delta^{f}(Z_{\sigma(i)})\right|>l\epsilon')\\
&\leq c(l)\exp -l\epsilon'^2\\
&\leq c(l)\exp -(l\epsilon^2-2\epsilon).
\end{aligned}$$

The bound for the fraction of permutations giving a difference greater than ϵ' was calculated by Vapnik ([1] sec. 4.13). □

The proof above follows the argument of Theorem 4.1 of [1], which deals with the generalization ability of ERM algorithms. The main difference is the reference to the random set $\overline{M}(Z_1,\ldots,Z_{2l})$ rather than to a fixed set of functions. Two variants of the result stated in Theorem 4.1 of [1] are Theorem 4.2 of [1] and the main result of [2]. The first gives better bounds when the empirical error rate is small, and the other gives a better rate of convergence when $c(l)$ is polynomial. Both can be adapted and proven for the setup here in a manner similar to that of Theorem 1.

The next result follows immediately from Theorem 1:

Corollary 1 *For maps M of the RERM type, the bound of Theorem 1 holds provided that*

$$\sup_{z_1,\ldots,z_{2l}\in\mathcal{Z}}\left|\overline{\tilde{M}}(z_1,\ldots,z_{2l})\right|=c(l),$$

with

$$\overline{\tilde{M}}(z_1,\ldots,z_{2l})=\bigcup_{i(1),\ldots,i(l)}\tilde{M}(z_{i(1)},\ldots,z_{i(l)}).$$

where the $i(j)$'s are l distinct indices in the range $1,\ldots,2l$.

Example (r-determined rules): Corollary 1 can be used to obtain a non-trivial generalization property for any rule of the RERM type where $\tilde{M}$ is of the form

$$\tilde{M}(z_1,\ldots,z_l)=\left\{f_{z_{j(1)},\ldots,z_{j(r)}}:j(i)\in\{1,\ldots,l\},\,i=1,\ldots,r\right\},$$

since for any map M of this type, $\left|\tilde{M}(z_1,\dots,z_{2l})\right| \leq (2l)^r$. Below, I refer to such rules as r-determined rules.

The support-vector setup

The support-vector machine (SVM) ([1]) creates a linear discriminant classifier in a ball within a high dimensional, or an infinite dimensional, Euclidean space:

$$\begin{aligned}\mathcal{X} &= \{x \in \mathcal{R}^n : |x| \leq 1\}\,,\\ \mathcal{F} &= \{f_{a,b}(x) = \mathrm{sign}(a \cdot x - b) : a \in \mathcal{R}^n, b \in \mathcal{R}, |a| = 1\}\,.\end{aligned}$$

To put the definition of an SVM into the framework presented here, I introduce the following definitions:

Definition 1 *The set $S(x_1,\dots,x_l,t_1,\dots,t_l)$, $x_i \in \mathcal{X}, t_i \in \{-1,+1\}$ is the set of classifiers $f_{a,b} \in \mathcal{F}$ such that $f_{a,b}(x_i) = 1$ iff $t_i = 1$ for all $i = 1,\dots,l$.*

In other words, the set $S(x_1,\dots,x_l,t_1,\dots,t_l)$ is the set of classifiers which predict $Y = t_i$ when presented with $X = x_i$, for all $i = 1,\dots,l$.

Definition 2 *The margin of a classifier $f_{a,b} \in \mathcal{F}$ with respect to a set of points $x_1,\dots,x_l \in \mathcal{X}$ is defined as*

$$\min_{i=1,\dots,l} |a \cdot x_i + b|\,.$$

The maximum margin classifier (MMC) is the member, $f_{a,b}$, of the set S with the property that its margin is the largest in the set.

The value of the margin of the MMC is denoted by $\mathbf{marg}(x_1,\dots,x_l,t_1,\dots,t_l)$.

Using the definitions above, the SVM can now be defined as a RERM type rule with:

$$\begin{aligned}\tilde{M}(z_1,\dots,z_l) \;=\; \{f_{a,b} = s(x_1,\dots,x_l,t_1,\dots,t_l) : t_i \in \{-1,+1\}, i = 1,\dots,l,\\ \mathbf{marg}(x_1,\dots,x_l,t_1,\dots,t_l) \geq h\}\,,\end{aligned}$$

where $s(x_1,\dots,x_l,t_1,\dots,t_l)$ is some member of $S(x_1,\dots,x_l,t_1,\dots,t_l)$ and h is some fixed constant.

The set $\tilde{M}(z_1,\dots,z_l)$ may or may not contain a representative from the set $S(x_1,\dots,x_l,y_1,\dots,y_l)$. If it does contain such a representative, f, then f will have zero empirical error rate, and therefore $M(z_1,\dots,z_l) = f$ will hold. If such a representative is not in $\tilde{M}(z_1,\dots,z_l)$ then $M(z_1,\dots,z_l) = s(x_1,\dots,x_l,t_1,\dots,t_l)$ for some $t_1,\dots,t_l$ and the empirical error rate of the algorithm will be equal to the cardinality of the set $\{i : t_i \neq y_i\}$.

Based on heuristic appeal and experimental results, s is usually chosen to be equal to the MMC. Here, however, I propose a different way to select a representative, for which the generalization ability can be determined. Note that the empirical risk achieved is the same for any choice of a representative.

The algorithm below, known as the perceptron algorithm ([3]), may be used to obtain a member of $S(x_1, \ldots, x_l, t_1, \ldots, t_l)$. Let the representative, s, be the one produced by the algorithm. This algorithm had been previously considered in this context by Freund and Schapire ([4]).

- **Initialization:** Set $a \leftarrow 0, b \leftarrow 0, k \leftarrow 1$
- **Update:** If $t_k(a \cdot x_k + b) > 0$ then go to step **Loop**
- **Correction:** Set $a \leftarrow a + t_k x_k, b \leftarrow b + t_k$
- **Loop:** If a **Correction** step was not carried out in the last l loops, stop. Otherwise, set $k \leftarrow k + 1(\text{mod } l)$ and go to step **Update**

The Perceptron Convergence Theorem ([3]) states that if the points x_i all lie inside the unit sphere, and $\mathbf{marg}(x_1, \ldots, x_l, t_1, \ldots, t_l) \geq h$ then the algorithm will execute at most $\lfloor 1/h^2 \rfloor$ corrections, after which the resulting a, b parameters will provide a member $f_{a,b}$ of $S(x_1, \ldots, x_l, t_1, \ldots, t_l)$. By construction the resulting classifier is r-determined with $r \leq \lfloor 1/h^2 \rfloor$.

Applying the bound for r-determined rules leads to the following conclusion: For any fixed h, if a support-vector method is employed and a classifier with a margin of h and empirical error rate R is found, then there exists an $r(h)$-determined classifier for which the following statement holds:

$$\mathbf{P}(\text{EXER} > R + \epsilon) < 2(2l)^{\lfloor 1/h^2 \rfloor} \exp -(l\epsilon^2 - 2\epsilon). \tag{1}$$

The perceptron algorithm can be used to obtain such a classifier.

An important point about the perceptron algorithm is that it can be executed without reference to the training vectors themselves but rather making use only of the inner products between training vectors. The importance of this property stems from the fact that often in applications of the support vector machine calculating inner products between training vectors is feasible, but any explicit representation of the vectors is prohibitively expensive.

Equation (1) can be converted into a $1 - \delta$ upper confidence bound. With probability of at least $1 - \delta$, the following inequality holds:

$$\text{EXER} < R + \sqrt{\frac{1}{l}\left(\frac{\log 2l}{h^2} + \log\frac{1}{\delta} + \log 2e^2\right)}. \tag{2}$$

The upper confidence bound (2) holds under the assumption that h is fixed in advance. It is common practice, however, to have h random. This is, for example, the case when the empirical error rate is pre-specified (e.g. zero).

A result suitable for the case of a random h will have the form of simultaneous upper confidence bounds for $r = \frac{1}{h^2} = 1, \ldots, l$. This is obtained by simply replacing δ by δ/l in (2), obtaining a $1-\delta$ an upper confidence bound of the following form:

$$\text{EXER} < R + \sqrt{\frac{1}{l}\left(\frac{\log 2l}{h^2} + \log\frac{l}{\delta} + \log 2e^2\right)}. \tag{3}$$

Since insisting on a pre-specified empirical error rate may lead to a large upper confidence bound, different procedures may be followed. One such procedure would be an adaptation of the perceptron algorithm:

- **Initialization:** Set $a(0) \leftarrow 0, b(0) \leftarrow 0, k \leftarrow 1, j \leftarrow 0, R(0) \leftarrow l$
- **Update:** If $t_k(a(j)\cdot x_k + b(j)) > 0$ go to step **Loop**
- **Correction:** Set $a(j+1) \leftarrow a(j) + t_k x_k, b(j+1) \leftarrow b(j) + t_k$, $j \leftarrow j+1, R(j) \leftarrow |\{i : t_k(a\cdot x_k + b) \leq 0\}|$
- **Loop:** If $R(j) = 0$ or $j = l$, go to step **Optimization**. Otherwise, set $k \leftarrow k + 1(\text{mod } l)$ and go to step **Update**
- **Optimization:** Set

$$j^* = \arg\min_{0\leq i\leq j} R(i) + \sqrt{\frac{1}{l}\left(i\log 2l + \log\frac{l}{\delta} + \log 2e^2\right)}.$$

Set $f^* = f_{a(j^*),b(j^*)}$. Stop

At the termination of the algorithm, f^* is a classifier with empirical error rate $R(j^*)$, and which with probability of at least $1-\delta$ has expected error rate no greater than

$$R(j^*) + \sqrt{\frac{1}{l}\left(j^*\log 2l + \log\frac{l}{\delta} + \log 2e^2\right)}.$$

Experimental results The use of variants of the perceptron algorithm in the support vector context had been previously suggested and implemented by Freund and Schapire ([4]). They carried out experiments using the perceptron algorithm for classifying images of handwritten digits and report error rates which are somewhat larger than those obtained with maximum margin classifiers.

Acknowledgments I thank Peter Bickel for pointing out the problem which resulted in this paper, for his valuable comments and for having reviewed the paper.

References

[1] Vapnik, V. N. (1998). *Statistical Learning Theory.* John Wiley & Sons, Inc., New York.

[2] Devroye, L. (1982). Bounds for the uniform deviation of empirical measures. *J. Multivariate Anal.*, **12** 72-79.

[3] Minsky, M. L. and Papert S. A. (1988). *Perceptrons.* The MIT Press, Cambridge.

[4] Freund, Y. and Schapire, R. E. (1998). Large margin classification using the perceptron algorithm. *COLT '98: Proceedings of the eleventh annual conference on computational learning theory*, 209-217.

Yoram Gat
University of California, Berkeley
367 Evans Hall
Berkeley, CA, 94720
E-mail: yoram@stat.berkeley.edu