Learning and Generalization: Theoretical Bounds

Ralf Herbrich*

Microsoft Research

CB3 0FB Cambridge

United Kingdom

rherb@microsoft.com

Robert C. Williamson

Australian National University

Canberra 0200

Australia

Bob.Williamson@anu.edu.au

*Corresponding author. Tel: +44 1223 479803, Fax: +44 1223 479999

Introduction

The fundamental difference between a system that learns and one that merely memorizes is that the learning system *generalizes* to unseen examples. In order to understand the performance of learning machines, and to gain insight that helps to design better ones, it is helpful to have theoretical bounds on the generalization ability of the machines. The determination of such bounds is the subject of the present article. In order to formulate them it is necessary to formalize the learning problem and turn the question of how well a machine generalizes into a mathematical question. Below we introduce one possible formalization — the one adopted in the field of statistical learning theory.

Formalization of The Learning Problem

In order to study the learning problem in a mathematical framework, we assume the existence of an *unknown* distribution $\mathbf{P}_{\mathsf{XY}}$ over an *input space* $\mathcal{X}$ (e.g. $\mathbb{R}^n$) and an *output space* $\mathcal{Y}$ (e.g. $\{0,1\}$). We are only given a *sample* $\boldsymbol{z} = ((x_1, y_1), \ldots, (x_m, y_m)) \in (\mathcal{X} \times \mathcal{Y})^m = \mathcal{Z}^m$ which, is assumed to be drawn *iid* (independent identically distributed) from $\mathbf{P}_{\mathsf{XY}}$; we define $\mathbf{P}_{\mathsf{Z}} := \mathbf{P}_{\mathsf{XY}}$. (In this article, random variables are always written in sans-serif, e.g. X.)

In an attempt to discover the unknown relation $\mathbf{P}_{\mathsf{Y}|\mathsf{X}=x}$ between inputs and outputs, a *learning algorithm* $\mathcal{A}$ chooses a deterministic *hypothesis* $h\colon \mathcal{X} \to \mathcal{Y}$

solely based on a given training sample $\boldsymbol{z} \in \mathcal{Z}^m$. Formally,

$$\mathcal{A}\colon \bigcup_{i=1}^{\infty} \mathcal{Z}^i \to \mathcal{H}\,,$$

where $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ is the *hypothesis space* used by the algorithm. (Recall $\mathcal{Y}^{\mathcal{X}}$ denotes the set of maps from $\mathcal{X}$ to $\mathcal{Y}$.) Some of the bounds take account of more information regarding $\mathcal{A}$ than just $\mathcal{H}$.

The performance of the learning algorithm is judged according to a *loss function* $l\colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ which measures the cost of the prediction $\hat{y}$ if y is the correct output. The choice of the loss function is a key part of the formal specification of the learning problem. The *learning problem* is to find an hypothesis $h\colon \mathcal{X} \to \mathcal{Y}$ such the *expected risk*, $R[h] := \mathbf{E}_{\mathsf{XY}}[l(h(\mathsf{X}), \mathsf{Y})]$, is minimized.

Pattern recognition In this case, $|\mathcal{Y}| < \infty$. Typically one is interested in the misclassification error $\mathbf{P}_{\mathsf{XY}}(h(\mathsf{X}) \neq \mathsf{Y})$. This can be modeled by the *zero-one loss*, $l_{0-1}(\hat{y}, y) := \mathbb{I}_{\hat{y} \neq y}$. (Here $\mathbb{I}$ denotes the indicator function.) More complex loss functions are obtained by using a cost matrix $\mathbf{C} \in \mathbb{R}^{|\mathcal{Y}| \times |\mathcal{Y}|}$.

Function learning Here, $\mathcal{Y} = \mathbb{R}$. The classical regression scenario utilizes squared loss, $l_2(\hat{y}, y) := (\hat{y} - y)^2$. Other loss functions are the ℓ_1 loss function, $l_1(\hat{y}, y) := |\hat{y} - y|$, and the ϵ–insensitive loss, $l_\epsilon(\hat{y}, y) := \max\{|\hat{y} - y|, \epsilon\} - \epsilon$.

If we knew $\mathbf{P}_{\mathsf{Z}}$, the solution of the learning problem would be straightforward:

$$h_{\text{opt}}(x) := \operatorname*{argmin}_{y \in \mathcal{Y}} \mathbf{E}_{\mathsf{Y}|\mathsf{X}=x}[l(y, \mathsf{Y})] \,. \tag{1}$$

The fact that h_{opt} can not be identified only on the basis of the training sample $\boldsymbol{z}$ is the motivation for studying *theoretical bounds* on the generalization error of learning algorithms. These bounds are only valid for most random draws of the training sample. Formally, they read as follows:

$$\mathbf{P}_{\mathsf{Z}^m}\left(R[\mathcal{A}(\mathbf{Z})] \leq \varepsilon_{\mathcal{A}}(\mathbf{Z}, \ldots, \delta)\right) \geq 1 - \delta \,. \tag{2}$$

In the analysis of such bounds it is convenient to think of the loss function induced function class

$$\mathcal{L}_{\mathcal{H}} := \{(x, y) \mapsto l(h(x), y) \mid h \in \mathcal{H}\} \,.$$

For simplicity we will mostly consider the pattern recognition case and the zero-one loss; the reasoning in the function learning case is conceptually similar.

Consistency of Learning Algorithms

Consistency is a property of a learning algorithm that guarantees that in the limit of an infinite amount of data the learning algorithm will achieve the minimum possible expected risk. The definition is relative to a fixed hypothesis space

$\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ and requires

$$\forall \epsilon > 0 \colon \quad \lim_{m \to \infty} \mathbf{P}_{\mathbf{Z}^m} \left(R[\mathcal{A}(\mathbf{Z})] - \inf_{h \in \mathcal{H}} R[h] > \epsilon \right) = 0 \,. \tag{3}$$

For the results stated below [15], a more complex notion of *nontrivial consistency* is needed. In particular, this notion requires that (3) holds even if $\mathcal{H}$ is replaced by $\mathcal{H}_c := \{h \in \mathcal{H} \mid R[h] \geq c\}$ for all $c \in \mathbb{R}$. Note that in this case $\inf_{h \in \mathcal{H}_c} R[h] = c$. It is known that for the class of *empirical risk minimization (ERM) algorithms*

$$\mathcal{A}^{\mathcal{H}}_{\mathrm{ERM}}(\boldsymbol{z}) := \operatorname*{argmin}_{h \in \mathcal{H}} \underbrace{\frac{1}{m} \sum_{i=1}^{m} l(h(x_i), y_i)}_{\widehat{R}[h, \boldsymbol{z}] \quad \text{(the empirical risk)}}$$

consistency is equivalent to uniform one-sided convergence of empirical risks to expected risk; that is,

$$\forall \epsilon > 0 \colon \quad \lim_{m \to \infty} \mathbf{P}_{\mathbf{Z}^m} \left(\sup_{h \in \mathcal{H}} \left(R[h] - \widehat{R}[h, \mathbf{Z}] \right) > \epsilon \right) = 0 \,. \tag{4}$$

A slightly stronger condition than (4), namely uniform two-sided convergence, is equivalent to

$$\forall \epsilon > 0 \colon \quad \lim_{m \to \infty} \frac{\ln(\mathbf{E}_{\mathbf{Z}^m}[\mathcal{N}(\epsilon, \mathcal{L}_{\mathcal{H}}, \mathbf{Z})])}{m} = 0 \tag{5}$$

where $\mathcal{N}(\epsilon, \mathcal{L}_{\mathcal{H}}, \boldsymbol{z})$ is the *covering number* of $\mathcal{L}_{\mathcal{H}}$ on the sample $\boldsymbol{z}$ at scale ϵ. This is the smallest number of functions $\hat{g} \colon \mathcal{Z} \to \mathbb{R}$ such that for every induced loss

function $g \in \mathcal{L}_{\mathcal{H}}$ there exists a function $\hat{g}$ with

$$\frac{1}{m}\sum_{i=1}^{m} |g(z_i) - \hat{g}(z_i)| \leq \epsilon \,.$$

In the case of the zero-one loss, l_{0-1}, the covering number $\mathcal{N}(\frac{1}{m}, \mathcal{L}_{\mathcal{H}}, \boldsymbol{z})$ equals the number of different error patterns $(g(z_1), \ldots, g(z_m)) \in \{0,1\}^m$ incurred by induced loss functions $g \in \mathcal{L}_{\mathcal{H}}$.

This *characterization* result (that consistency of $\mathcal{A}^{\mathcal{H}}_{\mathrm{ERM}}$ is "almost" equivalent to (5)) is the justification for the central place that covering numbers play in statistical learning theory. It is important to note that the results are *only* for $\mathcal{A}^{\mathcal{H}}_{\mathrm{ERM}}$. It is still an open problem to characterize consistency for algorithms other than $\mathcal{A}^{\mathcal{H}}_{\mathrm{ERM}}$ and thus it is not known what their "right" technical parameters are.

Theoretical Bounds for Learning Algorithms

The starting point of all the analysis presented here is the observation that for a *fixed* hypothesis $h\colon \mathcal{X} \to \mathcal{Y}$ (and induced loss function g, $g((x,y)) := l(h(x),y)$) we know that

$$\begin{aligned} \mathbf{P}_{\mathsf{Z}^m}\left(R[h] - \widehat{R}[h,\mathbf{Z}] > \epsilon\right) &= \mathbf{P}_{\mathsf{Z}^m}\left(\mathbf{E}_{\mathsf{Z}}\left[g(\mathsf{Z})\right] - \frac{1}{m}\sum_{i=1}^{m} g(\mathsf{Z}_i) > \epsilon\right) \\ &< \exp(-c \cdot m\epsilon^{\beta}) \end{aligned} \tag{6}$$

where c is some constant and $\beta \in [1,2]$, if the loss is bounded or has bounded moments. This is due to well known results in large deviation theory (see [5, Chapter 1]).

The second tool is the *union bound* which states that for events A and B,

$$\mathsf{P}(A \cup B) = \mathsf{P}(A) + \mathsf{P}(B) - \mathsf{P}(A \cap B) \leq \mathsf{P}(A) + \mathsf{P}(B)\,.$$

As a consequence, if we consider a hypothesis space of finite size, say n, then the chance that for at least one of the hypotheses the expected risk is larger than the empirical risk by more than ϵ is of order $n \cdot \exp(-m\epsilon^\beta)$. The general application of this simple inequality for learning theory is that given n *high-probability bounds* $\Upsilon_i \colon \mathcal{Z}^m \times \cdots \times [0,1] \to \{\text{false}, \text{true}\}$ such that

$$\forall i \in \{1, \ldots, n\} \colon \forall \delta \in [0,1] \colon \quad \mathsf{P}_{\mathsf{Z}^m}\left(\Upsilon_i(\mathbf{Z}, \ldots, \delta)\right) \geq 1 - \delta\,, \tag{7}$$

then

$$\forall \delta \in [0,1] \colon \quad \mathsf{P}_{\mathsf{Z}^m}\left(\Upsilon_1\left(\mathbf{Z}, \ldots, \frac{\delta}{n}\right) \wedge \cdots \wedge \Upsilon_n\left(\mathbf{Z}, \ldots, \frac{\delta}{n}\right)\right) \geq 1 - \delta\,.$$

There are two conceptual simplifications that aid the study of the generalization performance of learning algorithms:

Algorithm independence Motivated by (4), consider the uniform convergence

and bound this probability. This automatically gives a bound which holds for all hypotheses, including the one learned with a given learning algorithm. Although this is a very crude step, it has largely dominated statistical learning theory for the last 30 years; the whole analysis is independent of the learning algorithm used except via $\mathcal{H}$.

Data independence If the training sample is entering the bound only via the empirical risk we call the analysis *sample independent* as we are unable to exploit the serendipity of the training sample to obtain a better bound.

Algorithm Independent Bounds

Algorithm independent analysis has historically been the most common. Below we examine the VC framework, data-dependent structural risk minimization and the PAC-Bayesian framework.

The VC Framework

The VC (Vapnik-Chervonenkis) framework was established in 1971 and studies $\mathcal{A}_{\mathrm{ERM}}^{\mathcal{H}}$ via uniform convergence (see [15, 1] for more details). The bounds are sample independent in the sense used above. The only extra tool required is the *basic lemma*. This result makes precise the idea that whenever it is likely that two empirical risks measured on a training and a *ghost sample* (another sample of the same size drawn independently) are close to each other than it must also

be likely that the empirical risk on a training sample is close to the expected risk. A result of this is a generalization bound in terms of $\mathbf{E}_{\mathsf{Z}^{2m}}\left[\mathcal{N}\left(\frac{1}{2m}, \mathcal{L}_{\mathcal{H}}, \mathbf{Z}\right)\right]$ where the $2m$ is a consequence of the basic lemma. However this is still not really useful since computing $\mathbf{E}_{\mathsf{Z}^{2m}}\left[\mathcal{N}\left(\frac{1}{2m}, \mathcal{L}_{\mathcal{H}}, \mathbf{Z}\right)\right]$ requires knowledge of the distribution $\mathbf{P}_{\mathsf{Z}}$. For l_{0-1} loss use is made of the inequalities

$$\mathbf{E}_{\mathsf{Z}^{2m}}\left[\mathcal{N}\left(\frac{1}{2m}, \mathcal{L}_{\mathcal{H}}, \mathbf{Z}\right)\right] \leq \sup_{\boldsymbol{z}\in\mathcal{Z}^{2m}} \mathcal{N}\left(\frac{1}{2m}, \mathcal{L}_{\mathcal{H}}, \boldsymbol{z}\right) \leq \left(\frac{2em}{d_{\mathcal{H}}}\right)^{d_{\mathcal{H}}},$$

where $d_{\mathcal{H}}$ is known as the *VC dimension* of $\mathcal{H}$:

$$d_{\mathcal{H}} := \max\left\{m \in \mathbb{N} \;\middle|\; \sup_{\boldsymbol{z}\in\mathcal{Z}^{m}} \mathcal{N}\left(\frac{1}{m}, \mathcal{L}_{\mathcal{H}}, \boldsymbol{z}\right) = 2^{m}\right\}.$$

The generalization bound for the zero-one loss l_{0-1} then reads as follows: *With probability at least* $1-\delta$ *over the random draw of the training sample* $\boldsymbol{z} \in \mathcal{Z}^m$, *for all hypotheses* $h \in \mathcal{H}$, $R[h] \leq \varepsilon_{\mathrm{VC}}(\boldsymbol{z}, d_{\mathcal{H}}, \delta)$ *where*

$$\varepsilon_{\mathrm{VC}}(\boldsymbol{z}, d_{\mathcal{H}}, \delta) := \widehat{R}[h, \boldsymbol{z}] + \sqrt{\frac{8}{m}\left(\underbrace{d_{\mathcal{H}} \ln\left(\frac{2em}{d_{\mathcal{H}}}\right)}_{\text{effective complexity}} + \ln\left(\frac{4}{\delta}\right)\right)}. \tag{8}$$

The key term in this bound is labeled the *effective complexity* and in this case is essentially determined by the VC-dimension $d_{\mathcal{H}}$. Note that for general loss functions $l\colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ similar results are obtained by studying the family

$\{(\hat{y}, y) \mapsto \mathbb{I}_{l(\hat{y},y)>\theta} \mid \theta \in \mathbb{R}\}$ of zero-one loss functions.

There are many results bounding the VC dimension for specific hypothesis spaces (see VAPNIK CHERVONENKIS DIMENSION OF NEURAL NETWORKS and PAC LEARNING AND NEURAL NETWORKS). Since the result in (8) is uniform, it automatically provides a bound on the generalization error of any algorithm that chooses its hypotheses from some fixed hypothesis space $\mathcal{H}$.

Data-Dependent Structural Risk Minimization

An application of the union bound allows the combination of several VC bounds for different hypothesis spaces $\mathcal{H}_1 \subseteq \mathcal{H}_2 \subseteq \cdots \subseteq \mathcal{H}_k \subseteq \mathcal{Y}^{\mathcal{X}}$. This is the idea underlying *structural risk minimization* (SRM): Using the combination of VC bounds, an SRM algorithm aims to minimize the bound directly. It is thus applicable to regularized risk minimization learning algorithms. The bound, however, requires that the series of hypothesis spaces must be defined *independent* of the training sample. Hence, we cannot directly use the training sample to control the effective complexity (only implicitly via the resulting training error).

We can relax this assumption by introducing an ordering among the hypotheses to be covered for a given sample $\boldsymbol{z} \in \mathcal{Z}^m$. Such a function, $L \colon \cup_{i=1}^{\infty} \mathcal{Z}^i \times \mathcal{H} \to \mathbb{R}$, is called a *luckiness* (see [14]). For each luckiness function it is required that a value measured on the training sample allows one to bound the covering number on the training *and* ghost sample of hypotheses which increase the luckiness. This property is called *probable smoothness* w.r.t. a function $\omega \colon \mathbb{R} \times \mathbb{N} \times [0, 1] \to \mathbb{N}$.

The main result (which is data dependent in the sense used above) for the zero-one loss l_{0-1} reads as follows: *For all luckiness functions L which are probably smooth w.r.t. ω, with probability at least $1-\delta$ over the random draw of the training sample $\boldsymbol{z} \in \mathcal{Z}^m$, for all hypotheses $h \in \mathcal{H}$ such that $\widehat{R}[h, \boldsymbol{z}] = 0$, $R[h] \leq \varepsilon_{\mathrm{DSRM}}(\boldsymbol{z}, h, \omega, L, \delta)$ where*

$$\varepsilon_{\mathrm{DSRM}}(\boldsymbol{z}, h, \omega, L, \delta) := \frac{2}{m}\left(\underbrace{\log_2\left(\omega\left(L(h, \boldsymbol{z}), m, \frac{\delta}{2m}\right)\right)}_{\text{effective complexity}} + \log_2\left(\frac{2m}{\delta}\right)\right) . \quad (9)$$

The result can also be stated for non-zero training error and general loss functions [8]. Each probably smooth luckiness function defines a data-dependent structuring $\mathcal{H}_1(\boldsymbol{z}) \subseteq \mathcal{H}_2(\boldsymbol{z}) \subseteq \cdots \subseteq \mathcal{H}_m(\boldsymbol{z}) \subseteq \mathcal{H}$ of the hypothesis space $\mathcal{H}$ by

$$\mathcal{H}_i(\boldsymbol{z}) := \left\{ h \in \mathcal{H} \;\middle|\; \omega\left(L(h, \boldsymbol{z}), m, \frac{\delta}{2m}\right) \leq 2^i \right\} .$$

The choice of the luckiness function is not unique; it is best compared to the choice of a prior in a Bayesian analysis (see Bayesian Methods for Supervised Neural Networks).

PAC-Bayesian Framework

The PAC-Bayesian framework [11] studies only Bayesian learning algorithms. The main ideas are very similar to the luckiness framework. One of the motiva-

tions is to capture an important feature of Bayesian confidence intervals — their width depends on the sample itself and not just its size.

A direct application of the union bound with factors different from $\frac{1}{n}$ leads to the following result: *For all measures* $\mathbf{P}_{\mathsf{H}}$ *and* $\mathbf{P}_{\mathsf{Z}}$*, with probability at least* $1-\delta$ *over the random draw of the training sample* $\boldsymbol{z} \in \mathcal{Z}^m$*, for all hypotheses* $h \in \mathcal{H}$ *such that* $\mathbf{P}_{\mathsf{H}}(h) > 0$*,* $R[h] \leq \varepsilon_{\mathrm{PB}}(\boldsymbol{z}, h, \mathbf{P}_{\mathsf{H}}, \delta)$ *where*

$$\varepsilon_{\mathrm{PB}}(\boldsymbol{z}, h, \mathbf{P}_{\mathsf{H}}, \delta) := \widehat{R}[h, \boldsymbol{z}] + \sqrt{\frac{1}{2m}\left(\underbrace{\ln\left(\frac{1}{\mathbf{P}_{\mathsf{H}}(h)}\right)}_{\text{eff. complexity}} + \ln\left(\frac{1}{\delta}\right)\right)}\,.$$

If the likelihood function $\mathbf{P}_{\mathsf{Z}|\mathsf{H}=h}((x,y))$ equals $\mathbb{I}_{h(x)=y}$ then the bound maximizer is given by the *maximum a posteriori* estimator $h_{\mathrm{MAP}} := \operatorname{argmax}_{h\in\mathcal{H}} \mathbf{P}_{\mathsf{H}|\mathsf{Z}^m=\boldsymbol{z}}(h)$.

Using a tool known as the *quantifier reversal lemma* it is possible to study the *Gibbs classification strategy* which uses a randomly drawn hypothesis for each new data point to be classified:

$$\mathcal{A}^{H}_{\mathrm{Gibbs}}(x) := h(x)\,, \qquad h \sim \mathbf{P}_{\mathsf{H}|\mathsf{H}\in H}\,.$$

The quantifier reversal lemma is a high-probability equivalent of the union bound: *Given* n high-probability bounds Υ_i *(see (7)) and any distribution* $\mathbf{P}_{\mathsf{I}}$ *over the*

numbers $\{1, \ldots, n\}$,

$$\forall \alpha \in [0,1] \colon \forall \delta \in [0,1] \colon \quad \mathbf{P}_{\mathsf{Z}^m} \left(\mathbf{P}_{\mathsf{I}} (\Upsilon_{\mathsf{I}} (\mathbf{Z}, \ldots, \alpha\delta) \geq 1 - \alpha) \geq 1 - \delta \right).$$

The proof is very simple and makes use of Markov's inequality. Noticing that for all loss functions $l \colon \mathcal{Y} \times \mathcal{Y} \to [0,1]$,

$$R[\mathcal{A}^H_{\text{Gibbs}}] = \mathbf{E}_{\mathsf{H}|\mathsf{H}\in H}[R[\mathsf{H}]] \leq c \cdot \mathbf{P}_{\mathsf{H}|\mathsf{H}\in H}(R[\mathsf{H}] \leq c) + 1 \cdot \mathbf{P}_{\mathsf{H}|\mathsf{H}\in H}(R[\mathsf{H}] > c)$$

it is possible to prove the following result: *Given a prior measure* $\mathbf{P}_{\mathsf{H}}$*, with probability at least* $1 - \delta$ *over the random draw of the training sample* $\boldsymbol{z} \in \mathcal{Z}^m$*, for all subsets* $H \subseteq \mathcal{H}$*, the generalization error of the Gibbs classification strategy* $\mathcal{A}^H_{\text{Gibbs}}$ *satisfies*

$$R\left[\mathcal{A}^H_{\text{Gibbs}}(\boldsymbol{z})\right] \leq \mathbf{E}_{\mathsf{H}|\mathsf{H}\in H}\left[\widehat{R}[\mathsf{H}, \boldsymbol{z}]\right] + \sqrt{\frac{1}{2m}\left(\underbrace{\ln\left(\frac{1}{\mathbf{P}_{\mathsf{H}}(H)}\right)}_{\text{effective complexity}} + \ln\left(\frac{m^2}{\delta}\right)\right)} + \frac{1}{m}.$$

The effective complexity scales inversely with $\mathbf{P}_{\mathsf{H}}(H)$ which in the case of the likelihood function $\mathbf{P}_{\mathsf{Z}|\mathsf{H}=h}((x,y)) = \mathbb{I}_{h(x)=y}$ and the Bayesian posterior $\mathbf{P}_{\mathsf{H}|\mathsf{Z}^m=\boldsymbol{z}}$ equals the *evidence* $\mathbf{E}_{\mathsf{H}}[\mathbf{P}_{\mathsf{Z}^m|\mathsf{H}=h}(\boldsymbol{z})]$ (see Bayesian Methods for Supervised Neural Networks). The complexity term is minimized if we choose H such that $\mathbf{P}_{\mathsf{H}}(H) = 1$. However, for a small overall bound value it is also required

that the expected empirical risk $\mathbf{E}_{\mathsf{H}|\mathsf{H}\in H}[\widehat{R}[\mathsf{H}, \boldsymbol{z}]]$ is small. It is worth mentioning that the results are still algorithm independent since they not only hold for the Bayesian posterior but for all hypotheses $h \in \mathcal{H}$ and all subsets $H \subseteq \mathcal{H}$.

Algorithm Dependent Bounds

We now summarize three distinct but related approaches to the analysis of learning algorithms that utilize particular properties of the algorithm apart from the space $\mathcal{H}$ it draws its hypotheses from.

The Compression Framework

The compression framework [6] is based on the idea that a good learning algorithm is able to reconstruct its hypothesis using only a small fraction of the training sample $\boldsymbol{z}$. It is assumed that the learning algorithm can be written as

$$\mathcal{A}(\boldsymbol{z}) := \mathcal{R}(\boldsymbol{z}_{\mathcal{C}(\boldsymbol{z})}) \tag{10}$$

where $\mathcal{C}\colon \cup_{m=1}^{\infty} \mathcal{Z}^m \to \mathcal{I}$ maps the training sample to indices $\mathbf{i} \in \mathcal{I}$, $\mathcal{I} = \{(i_1, \ldots, i_n) \mid n \in \mathbb{N}, i_1 \neq \cdots \neq i_n\}$, $\boldsymbol{z}_{\mathbf{i}} := (z_{i_1}, \ldots, z_{i_n})$, and $\mathcal{R}\colon \cup_{m=1}^{\infty} \mathcal{Z}^m \to \mathcal{Y}^{\mathcal{X}}$ computes the final hypothesis using only the subsample indexed by $\mathcal{C}(\boldsymbol{z})$. A typical example of such an algorithm is the perceptron learning algorithm (see Perceptrons, Adalines, and Backpropagation) which can reconstruct its hypothesis using only the training patterns on which it needed to update the

weight vector.

The mathematical tool needed to study this class of learning algorithms is again the union bound:

$$\begin{aligned}\mathbf{P}_{\mathbf{Z}^m}\left(R[\mathcal{A}(\mathbf{Z})] - \widehat{R}[\mathcal{A}(\mathbf{Z}), \mathbf{Z}] > \epsilon\right) &\leq \mathbf{P}_{\mathbf{Z}^m}\left(\exists \mathbf{i} \in \mathcal{I}\colon R[\mathcal{R}(\mathbf{Z}_\mathbf{i})] - \widehat{R}[\mathcal{R}(\mathbf{Z}_\mathbf{i}), \mathbf{Z}] > \epsilon\right) \\ &\leq \sum_{\mathbf{i}\in\mathcal{I}} \mathbf{P}_{\mathbf{Z}^m}\left(R[\mathcal{R}(\mathbf{Z}_\mathbf{i})] - \widehat{R}[\mathcal{R}(\mathbf{Z}_\mathbf{i}), \mathbf{Z}] > \epsilon\right).\end{aligned}$$

Interestingly, for any index vector $\mathbf{i}$ the sample $\boldsymbol{z} \setminus \boldsymbol{z}_\mathbf{i}$ is an iid test sample on which the fixed hypothesis $\mathcal{R}(\boldsymbol{z}_\mathbf{i})$ is assumed to have a difference in empirical and expected risk of more than ϵ. Using (6) — which holds independent of $\mathbf{i}$ — and the fact that there are no more than $\binom{m}{d} \leq (\frac{em}{d})^d$, $d = |\mathbf{i}|$, many different index sets for a training sample $\boldsymbol{z}$ of size m, leads to the main result of the compression framework: *For the zero-one loss l_{0-1} and any learning algorithm that can be written as (10), with probability at least $1-\delta$ over the random draw of the training sample $\boldsymbol{z} \in \mathcal{Z}^m$, $R[\mathcal{A}(\boldsymbol{z})] \leq \varepsilon_{\mathrm{cr}}(\boldsymbol{z}, |\mathcal{C}(\boldsymbol{z})|, \delta)$ where for $d = |\mathcal{C}(\boldsymbol{z})|$*

$$\varepsilon_{\mathrm{cr}}(\boldsymbol{z}, d, \delta) := \frac{m}{m-d} \cdot \widehat{R}[\mathcal{A}(\boldsymbol{z}), \boldsymbol{z}] + \sqrt{\frac{1}{2m-d}\left(\underbrace{d \ln\left(\frac{em}{d}\right)}_{\text{eff. complexity}} + \ln\left(\frac{m^2}{\delta}\right)\right)}. \quad (11)$$

A similar result can be stated for general loss functions. Note that this bound is data-dependent since $|\mathcal{C}(\boldsymbol{z})|$ depends both on the learning algorithm $\mathcal{A}$ and the training sample $\boldsymbol{z}$.

The compression framework has its roots in the theory of on-line learning [10]. An *on-line learning algorithm* proceeds in trials. In each trial, the algorithm is presented with a training sample $x_i \in \boldsymbol{x}$ and makes a prediction $\hat{y} \in \mathcal{Y}$. It then receives the desired output $y_i \in \boldsymbol{y}$ and incurs a mistake whenever $\hat{y} \neq y_i$. The performance measure of an on-line learning algorithm is the number of mistakes it incurs on a training sample $\boldsymbol{z}$. If the on-line algorithm is *mistake driven*, that is, it only updates the hypothesis whenever a mistake is incurred, then any mistake bound is also an upper bound on $|\mathcal{C}(\boldsymbol{z})|$. This scheme allows the determination of generalization error bounds for on-line learning algorithms applied in batch mode (see for example [3]).

The Algorithmic Stability Framework

In the algorithmic stability framework [2], it is assumed that any additional training example has a limited influence on the function learned insofar as the prediction on any possible test point is concerned. Such algorithms are called *uniformly stable* and have the property that for all $i \in \{1, \ldots, m\}$:

$$\forall \boldsymbol{z} \in \mathcal{Z}^m \colon \forall (x, y) \in \mathcal{Z} \colon \quad |l(\mathcal{A}(\boldsymbol{z})(x), y) - l(\mathcal{A}(\boldsymbol{z}_{\setminus i})(x), y)| \leq \beta(m)\,,$$

where $\boldsymbol{z}_{\setminus i} := (z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_m)$. The $\beta(\cdot)$–stability of learning algorithms can be determined if the loss function is *Lipschitz continuous* with (Lipschitz) constant C_l: the difference $|l(\hat{y}, \cdot) - l(\tilde{y}, \cdot)|$ is bounded from above by $C_l \cdot |\hat{y} - \tilde{y}|$.

The ℓ_1 loss l_1 and the ϵ–insensitive loss l_ϵ are both Lipschitz continuous with the constant $C_l = 1$.

Given a Lipschitz continuous loss function l and a reproducing kernel Hilbert space $\mathcal{H}$ with kernel $k\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, the class of regularized risk minimization learning algorithms

$$\mathcal{A}^{\mathcal{H},\lambda}_{\mathrm{RRM}} := \operatorname*{argmin}_{h \in \mathcal{H}} \left(\widehat{R}[h, \boldsymbol{z}] + \lambda \|h\|^2 \right)$$

are $\beta(\cdot)$–stable with $\beta(m) \leq C_l \sup_{x \in \mathcal{X}} k(x,x)/2\lambda m$. Intuitively, the larger $\lambda > 0$ the smaller the influence of the empirical term $\widehat{R}[h, \boldsymbol{z}]$ and hence the more stable the learning algorithm (see also Generalization and Regularization in Nonlinear Learning Systems).

In order to exploit the $\beta(\cdot)$–stability of a learning algorithm, a result from the theory of large deviations of functions of random variables known as *McDiarmid's inequality* [5] is used. This inequality asserts that the probability of a deviation of ϵ between the value of a function f of m iid variables and the expected value of that function decays as $\exp(-\epsilon^2/mc^2)$ where c is the maximal deviation of the functions value when exchanging one variable. In this sense, McDiarmid's inequality is a generalization of (6) for non-pointwise loss functions. Considering the deviation between the expected risk and the empirical risk of the function learned by $\mathcal{A}$ as a function of m iid random variables leads to the following result: *For any $\beta(\cdot)$–stable learning algorithm $\mathcal{A}$ and a bounded loss function*

$l\colon \mathcal{Y} \times \mathcal{Y} \to [0,1]$, *with probability at least* $1-\delta$ *over the random draw of the training sample* $\boldsymbol{z} \in \mathcal{Z}^m$, $R[\mathcal{A}(\boldsymbol{z})] \leq \varepsilon_{\mathrm{AS}}(\boldsymbol{z},\beta,\delta)$, *where*

$$\varepsilon_{\mathrm{AS}}(\boldsymbol{z},\beta,\delta) := \widehat{R}[\mathcal{A}(\boldsymbol{z}),\boldsymbol{z}] + 2\beta(m) + \sqrt{\frac{2\left(4\beta(m)\cdot m+1\right)^2 \ln\left(\frac{1}{\delta}\right)}{m}}\,. \tag{12}$$

There are three interesting observations to make:

1. In order for the result to be non-trivial, it is required that $\beta(m)$ decays faster than $1/m$. This readily tells us the range of λ values to consider for $\mathcal{A}^{\mathcal{H},\lambda}_{\mathrm{RRM}}$.

2. The result as stated in (12) is not directly applicable to the zero-one loss l_{0-1} as the difference in the latter cannot decay at a rate of $1/m$ but is fixed to the values $\{0,1\}$. Noticing that in practice we often use thresholded real-valued functions $h(\cdot) = \mathrm{sign}(f(\cdot))$ for classification, it is possible to overcome this limitation by bounding the zero-one loss function from above. In particular, if $\mathcal{Y} = \{-1,+1\}$ then

$$l_{\mathrm{margin}}(f(x),y) := \min(\max(0,1-yf(x)),1) \geq l_{0-1}(f(x),y) := \mathbb{I}_{yf(x)\leq 0}\,,$$

that is, any upper bound on the expected risk $\mathbf{E}_{\mathsf{XY}}[l_{\mathrm{margin}}(f(\mathsf{X}),\mathsf{Y})]$ is by definition an upper bound on $R[h]$ for the zero-one loss l_{0-1} and the associated binary classification function h.

3. The result is data-independent as the stability $\beta(m)$ needs to be known

before the training samples arrives. Recent developments in this area aim to overcome this problem by the notion of a stability measured on the given training sample.

The Algorithmic Luckiness Framework

Finally, we present a recently developed algorithm-dependent framework [8] which builds on ideas of the data-dependent structural risk minimization framework. The key observation is that the basic lemma is not only true when considering the maximum deviation between the expected and empirical risk but is also true for the deviation between the expected and empirical risk of the *one* function learned using a fixed learning algorithm $\mathcal{A}$. As a consequence, for any double sample $\boldsymbol{z}\boldsymbol{z}' \in \mathcal{Z}^{2m}$ (training sample $\boldsymbol{z}$ and ghost sample $\boldsymbol{z}'$) one only needs to consider the set $H \subseteq \mathcal{Y}^{\mathcal{X}}$ of functions which can be learned by a fixed learning algorithm $\mathcal{A}$ from any subsample of size m. If the learning algorithm under consideration is permutation-invariant then this set cannot be larger than $|H| \leq 2^{2m}$ regardless of the loss function considered.

The notion of *luckiness* changes in that it now maps a given learning algorithm $\mathcal{A}$ and a given training sample $\boldsymbol{z}$ to a real value which effectively measures the extent to which the given data aligns with an encoded prior belief. In accordance with the data-dependent structural risk minimization framework, it is required that the measured value of the luckiness on a random training sample $\boldsymbol{z}$ can be used to upper bound the number of subsets of a double sample which will lead to

an increase in the luckiness value. This rather technical condition is known as ω–smallness and is best compared to the probable smoothness of luckiness functions earlier. Using the union bound together with the refined basic lemma leads to the following generalization error bound for all loss function $l\colon \mathcal{Y} \times \mathcal{Y} \to [0,1]$:

For all algorithmic luckiness functions L which are ω–small, with probability at least $1-\delta$ over the random draw of the training sample $\boldsymbol{z} \in \mathcal{Z}^m$, $R[\mathcal{A}(\boldsymbol{z})] \leq \varepsilon_{\mathrm{AL}}(\boldsymbol{z}, \mathcal{A}, \omega, L, \delta)$

$$\varepsilon_{\mathrm{AL}}(\boldsymbol{z}, \mathcal{A}, \omega, L, \delta) := \widehat{R}[\mathcal{A}(\boldsymbol{z}), \boldsymbol{z}] + \sqrt{\frac{8}{m}\left(\underbrace{\log_2\left(\omega\left(L(\mathcal{A}, \boldsymbol{z}), \frac{\delta}{2m}\right)\right)}_{\text{effective complexity}} + \log_2\left(\frac{2m}{\delta}\right)\right)}\,.$$

The main difference to (9) is in the definition of the luckiness function. In contrast to (9), we can now exploit properties of the learning algorithm in the definition of the ω–smallness. As an easy example, consider the luckiness function $L_0(\mathcal{A}, \boldsymbol{z}) := -|\mathcal{C}(\boldsymbol{z})|$ for algorithms of the form (10). Then, given a value $d = -L_0(\mathcal{A}, \boldsymbol{z})$ of the luckiness function on any training sample, there cannot be more than $\binom{2m}{d}$ distinct subsets of the training and ghost sample which shows that $\omega(L_0, m, \delta) = \binom{2m}{-L_0}$ is a valid ω function. Note that this example removes the factor $\frac{m}{m-d}$ in front of the empirical term in (11) at the cost $2m$ rather than m in the complexity term $d \ln(\frac{2em}{d})$.

Discussion

Our presentation of the theory of learning and generalization is non-standard since we aimed to present many, seemingly different approaches. For standard presentations with more details the interested reader is referred to [4, 15, 1, 7, 13]. A fairly comprehensive overview is given in [9]. In this chapter, we were assuming that the genuine interest is in bounds on the generalization error (see (2)). It is worth mentioning that another way to quantify generalization behavior of learning algorithms is in terms of bounds on the leave-one-out error (for further details, the interested reader is referred to [4]).

Although we would like to use theoretical bounds directly for model selection and model validation, it currently seems that the potential value of these results is to provide insight into the design of learning algorithms. For example, the question of consistency says that covering numbers are the "right" quantities to look at for ERM algorithms.

However, for other algorithms the situation is less clear, though now there are several variants on classical VC analysis methods using the same formal learning problem setup. The various bounds we presented ($\varepsilon_{\mathrm{VC}}(\boldsymbol{z}, d_{\mathcal{H}}, \delta)$, $\varepsilon_{\mathrm{DSRM}}(\boldsymbol{z}, h, \omega, L, \delta)$, $\varepsilon_{\mathrm{PB}}(\boldsymbol{z}, h, \mathbf{P}_{\mathsf{H}}, \delta)$, $\varepsilon_{\mathrm{cr}}(\boldsymbol{z}, |\mathcal{C}(\boldsymbol{z})|, \delta)$, $\varepsilon_{\mathrm{AS}}(\boldsymbol{z}, \beta, \delta)$, $\varepsilon_{\mathrm{AL}}(\boldsymbol{z}, \mathcal{A}, \omega, L, \delta)$) were in terms of a range of parameters; we still do not really know what the "right" ones are. Recent work [12] has shown the power of alternate geometric approaches to develop certain classes of generalization bounds. We expect that these and other approaches

will lead to deeper understanding of the generalization ability of learning machines.

References

* [1] M. Anthony and P. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999.

[2] O. Bousquet and A. Elisseeff. Algorithmic stability and generalization performance. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 196–202. MIT Press, 2001.

[3] N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth. How to use expert advice. *Journal of the ACM*, 44(3):427–485, 1997.

* [4] L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Number 31 in Applications of Mathematics. Springer, New York, 1996.

[5] L. Devroye and G. Lugosi. *Combinatorial Methods in Density Estimation*. Springer, 2001.

[6] S. Floyd and M. Warmuth. Sample compression, learnability, and the Vapnik Chervonenkis dimension. *Machine Learning*, 27:1–36, 1995.

* [7] R. Herbrich. *Learning Kernel Classifiers: Theory and Algorithms*. MIT Press, 2002.

[8] R. Herbrich and R. C. Williamson. Algorithmic luckiness. In *Advances in Neural Information Processing Systems 14*. MIT Press, 2002. In press.

* [9] S. Kulkarni, G. Lugosi, and S. Venkatesh. Learning pattern classification — a survey. *IEEE Transactions on Information Theory*, 44:2178–2206, 1998.

[10] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

[11] D. A. McAllester. Some PAC Bayesian theorems. In *Proceedings of the Annual Conference on Computational Learning Theory*, pages 230–234, Madison, Wisconsin, 1998. ACM Press.

[12] S. Mendelson. Geometric methods in the analysis of Glivenko-Cantelli classes. In D. Helmbold and B. Williamson, editors, *14th Annual Conference on Computational Learning Theory COLT, Proceedings*, pages 256–272, 2001.

* [13] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[14] J. Shawe-Taylor, P. L. Bartlett, R. C. Williamson, and M. Anthony. Structural risk minimization over data-dependent hierarchies. *IEEE Transactions on Information Theory*, 44(5):1926–1940, 1998.

* [15] V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.