How to control the generalization ability of learning processes?

Target: An inductive principle for minimizing the risk functional using a small sample of training samples.

The sample size \( l \) is considered to be small if the ratio \( \frac{l}{h} \) (where \( h = \text{VC dimension} \)) is small, say \( \frac{l}{h} < 20 \).

Recall our bounds for the generalization ability of learning machines with sets of totally bounded non-negative functions:

\[
R(x_0) \leq \text{Remp}(x_0) + \frac{BE}{2} \left( 1 + \sqrt{1 + \frac{4 \text{Remp}(x_0)}{BE}} \right)
\]

(1)
bounds for the generalization ability of learning machines with sets of unbounded functions

\[ R(\varepsilon) \leq \frac{\text{Rem}_p(\varepsilon)}{(1 - a(p) \cdot \sqrt{\varepsilon})} \tag{2} \]

where
\[ a(p) = \sqrt[4]{\frac{1}{2} \left( \frac{p-1}{p-2} \right)^{p-1}} \]

Here \( \varepsilon = 2 \frac{\ln N - \ln n}{n} \) if the set of functions \( \Omega(z, \alpha_i) \) \( i=1 \ldots N \) contains \( N \) elements

and \( \varepsilon = 4 \frac{h(\ln \frac{2l}{h} + 1) - \ln \frac{n}{4}}{n} \) if the set of functions contains an infinite number of elements with finite VC dimension \( h \)

Each bound is valid with probability \( \geq 1 - \eta \).

We've been working under ERM principle: if \( \frac{L}{n} \to 0 \) then \( \varepsilon \) is small \( \Rightarrow \) consistency.
But in algorithmic practice we may face \( \frac{d}{n} \) small Structural Risk Minimization Principle (SRM principle) \( \text{(Vapnik-Chervonenkis 1974)} \) "admissible structure"

Let the set \( S \) of functions \( \Omega(z, \alpha) \), \( \alpha \in \Lambda \) be provided with a structure consisting of nested subsets of functions \( S_k = \{ \Omega(z, \alpha), \alpha \in \Lambda_k \} \) such that \( S_1 \subset S_2 \subset \ldots \subset S_n \subset \ldots \)

where the elements of the structure satisfy the following two properties:

(i). The VC dimension \( h_k \) of each set \( S_k \) of functions is finite, and \( h_1 \leq h_2 \leq \ldots \leq h_n \leq \ldots \n
(ii). Any element \( S_k \) of the structure contains either a set of totally bounded functions \( 0 \leq \Omega(z, \alpha) \leq B_k \), \( \alpha \in \Lambda_k \)

or a set of functions satisfying the inequality

\[
\sup_{\alpha \in \Lambda_k} \left( \frac{\int \Omega^p(z, \alpha) \, dF(z)}{\int \Omega(z, \alpha) \, dF(z)} \right)^{\frac{1}{p}} \leq T_k \quad , \quad p > 2
\]

for some pair \((p, T_k)\) \( \text{(3)} \)
Given a set of observations $\varepsilon_1, \ldots, \varepsilon_n$,

SRM principle chooses $(\hat{\omega}, \lambda^K)$

that minimizes the empirical risk in the
subset $S_k$ for which the guaranteed risk
is minimal.

"tradeoff between the quality of the approximator
of the given data and the complexity of the
approximating function".

As subset index $n \uparrow$ the minima of $R_{\text{emp}}(\lambda^K) \downarrow$

But the confidence interval (second summand in (1))
or multiplier in (2)
increase $\uparrow$

![Diagram showing empirical risk, bound on the risk, confidence interval, and subset index $n$.]
Analysis of convergence based on SRM principle

\[ S^* = \bigcup_{k=1}^{\infty} S_k \]

\( S^* \) is everywhere dense in \( S = \{ Q(z, \alpha), \alpha \in \Lambda \} \)

w.r.t. metric

\[ p(\mathcal{Q}(z, \alpha_1), \mathcal{Q}(z, \alpha_2)) = \int |\mathcal{Q}(z, \alpha_1) - \mathcal{Q}(z, \alpha_2)| \, dF(z) \]

Given any \( l \), we consider \( n = n(l) \) corresponding to \( S_n \), in which we minimize the empirical risk.

What is the asymptotic law of \( n(l) \)?

**Theorem 1.** The SRM method provides approximations \( \mathcal{Q}(z, x_{n(l)}) \) for which the sequence of risks \( R(x_{n(l)}) \xrightarrow{l \to \infty} R(x_0) = \inf_{\alpha \in \Lambda} \int |\mathcal{Q}(z, \alpha)| \, dF(z) \)

s.t. the rate of convergence is \( V(l) \)

\[ V^{-1}(l) \left| R(x_{n(l)}) - R(x_0) \right| \xrightarrow{\| \to \infty} C \]

\[ V(l) = e_{n(l)} + T_n(l) \sqrt{\frac{\ln l}{n(l)}} \]
If the law \( n = n(x) \) is such that

\[
\lim_{l \to \infty} \frac{T_n(l) \ln(x)}{l} = 0
\]

where

(i). \( T_n = B_n \) if one considers a structure with totally bounded functions \( \mathcal{Q}(2, x) \leq B_n \) in \( S_n \)

(ii). \( T_n = T_n \) if one considers a structure with elements satisfying (3)

and \( T_n(l) \) is the rate of function approximation

\[
T_n = \inf_{x \in \mathcal{A}_n} \int \mathcal{Q}(2, x) dF(z) - \inf_{x \in \mathcal{A}_n} \int \mathcal{Q}(2, x) dF(z)
\]

The above theorem reduces the problem of best rate of convergence to the rate \( r_n \) of approximation for chosen structure.

Estimate \( r_n \) for different structures on sets of functions

\( r_n \) known \( \Rightarrow \) Find \( n = n(x) \) a-priori

Classical function approximation theory
Example of applying Theorem 1

\( \Omega(2, x) \) \& \( Y \) satisfy (3) for \( p > 2 \) and \( T_k < T^* < \infty \)

Consider \( S_1 < \ldots < S_n c \ldots \) with \( n = h_n \)

and \( r_n = \left( \frac{1}{n} \right)^c \)

Then \( V(l) = \left( \frac{1}{n(l)} \right)^c + \frac{T_n(l)}{\sqrt{\frac{n(l) \ln l}{l} \sqrt{rac{1}{l}}} \}

maximal of \( V(l) \) achieved if \( n(l) = \left[ \frac{l}{\ln l} \right]^{2c+1} \)

and \( [V(l)]_{\max} = \left( \frac{\ln l}{l} \right)^c \frac{c}{2c+1} \)

One can \( a \)-pri\( ri \) (before the learning process begins)

find the law \( n = n(l) \) which provides the best (asymptotic) rate of convergence, and one can \( a \)-pri\( ri \) estimate the value of asymptotic rate of convergence

Rate depends on the construction of the admissible structure \( (h_n, T_n) \), \( n = 1, 2, \ldots \)

and rate of approximation \( T_n \), \( n = 1, 2, \ldots \)
\[ V(t) = r_n(t) + T_n(t) \sqrt{\frac{\ln(n)}{n}} \]

**Deterministic Component**

of the learning process

usually only has an

asymptotic bound

**Stochastic behavior**

of the learning process

usually determined

by non-asymptotic bounds

of the risk


---

**Theorem 2** Suppose the set of functions \( f(x) \)

is such that for some \( d \geq 0 \) and \( \hat{f}(w) \) Fourier transform

\[
\int |w|^d |\hat{f}(w)| dw = C_d(f) < \infty \quad d \geq 0
\]

Then the rate of approximation of the desired function by the best function of the elements of the structure is bounded by \( O\left( \frac{1}{\sqrt{n}} \right) \) if one of the following holds

1. \( d = 0 \) Elements \( S_n \) of the structure contains the functions

\[
f(x, \xi, w, v) = \sum_{i=1}^{n} \lambda_i \sin \left[ (x, \xi) + u_i \right]
\]

(Jones, 1992)
(2) \( d = 1 \). Elements \( S_n \) of the structure contains the functions

\[
f(x, \alpha, w, v) = \sum_{i=1}^{n} \alpha_i \cdot S[(x, w_i) + v_i]
\]

where \( S(u) = \text{sigmoid function} \) (Breiman, 1993) e.g. \( S(u) = \tanh u \)

\[
S(u) = \frac{e^u - e^{-u}}{e^u + e^{-u}}
\]

(3) \( d = 2 \) Elements \( S_n \) of the structure contains the functions

\[
f(x, \alpha, w, v) = \sum_{i=1}^{n} \alpha_i \left[ (x, w_i) + v_i \right]_+ \text{, } (\text{if } u = \max u_i)
\]

(Breiman, 1993)

In general, the rate of convergence of approximation of functions with \( S \) continuous derivatives is given by \( O(n^{-\frac{5}{N}}) \).

\( S_n \) — follow (2) in Theorem 2

\( N \) — dimensionality of input space
That is why "regularization" is called "smoothing".

Examples of structures of Neural Nets

1. Structure given by the architecture of the neural network

2. Structure given by learning procedure
   Assign "weights" to a given structured NN
   \[ S_p = \{ f(x, w) : \| w \|_1 < C_p \} \]
   \[ C_1 < C_2 < \ldots < C_p \]
   \[ \text{minimize} \quad E(w, V_p) = \frac{1}{e} \sum_{i=1}^{e} L(y_i, f(x_i, w)) + \gamma \| w \|_2 \]
   "weight-decay" procedure
3. Structure given by preprocessing

NN with fixed architecture

Input representation \( \mathbf{x} \rightarrow z = K(\mathbf{x}, \beta) \)

\( \beta \) - controls the degree of degeneracy introduced by this transformation (e.g., width of a smoothing kernel)

Structure \( S = \{ f(K(\mathbf{x}, \beta), w), w \in W \} \).

\( \beta > C_P \quad C_1 > C_2 > ... > C_P \)

Implement SRM principle: \( \{ \)

1. VC dim of \( S_k \)
2. Minimizer of empirical risk within \( S_k \)

Instead of global approximation one can also consider local approximation: kernel method

“Local risk functional” \( \mathcal{R}(d, \beta; x_0) = \int L(y, f(x, \mathbf{x})) \frac{k(x, x_0, \beta)}{k(x_0, \beta)} dF(x, \mathbf{y}) \)

\( \mathcal{A} \leq d \leq \mathcal{B} \quad d \in \mathcal{V} \quad 0 \leq k \leq 1 \quad \beta \in (0, \infty) \)

\( \mathcal{R}(d, \beta; x_0) \leq \frac{1}{d} \sum_{i=1}^{d} L(y_i, f(x_i, \mathbf{x}))K(x_i, x_0, \beta) + (\beta - A)E(\mathcal{L}, \eta) \)

\( \left( \frac{1}{d} \sum_{i=1}^{d} K(x_i; x_0, \beta) - E(\mathcal{L}, \eta) \right) + \)
\[ E(l, h) = \sqrt{l h \left( \ln \frac{2l}{h} + 1 \right) - \ln \left( \frac{h}{2} \right)} \]

- \( h \Sigma \): VC dimension of \( L(y, f(x, \alpha)) K(x, x_0, \beta) \)
- \( h_\beta \): VC dimension of \( K(x, x_0, \beta) \)

(Vapnik - Botton 1993)

Minimum Description Length (MDL) Principle

Algorithm Complexity (Kolmogorov 1965) of an object:
- Length of the shortest binary computer program that describes this object.

Kolmogorov proved that the value of the algorithmic complexity, up to an additive constant, does not depend on the type of computer.

What is a random object?

Randomness = High algorithm complexity

i.e. the string which describes the object cannot be compressed significantly
MDL Principle

Training set \((W_i, X_i), \ldots, (W_e, X_e)\)

Code book \(C_b\): \(N \ll 2^d\) different tables

\[ T_i = i = 1, 2, \ldots, N \]

Table \(T_i\): \(\star \rightarrow w\) function

From table \(T_i\) we get

\[ x_1, \ldots, x_e \xrightarrow{T_i} w_1^*, \ldots, w_e^* \]

Decoding: Perfect table \(T_0\) to that minimize

Hamming distance \(w_1^*, \ldots, w_e^*\) and \(w_1, \ldots, w_e\)

Shannon: Minimal number of bits that describe

any one of the \(N\) tables = \(\lceil \log_2 N \rceil \)

Coefficient of Compression

\[ K(T_0) = \frac{\lceil \log_2 N \rceil + \lceil \log_2 C \rceil + \lceil \log_2 d \rceil + \Delta_d}{d} \]

\(d\) — minimal errors in Hamming distance

\[ K(T) \text{ small} \rightarrow \text{String is not random and it depends on input } x \]
The compression coefficient $K(T)$ determines the probability of test error in classification (decoding) vectors $x$ by the table $T$.

**Theorem** If, on a given structure of code-books one compresses by a factor $K(T)$ the description of string $(w_1, x_1), \ldots, (w_e, x_e)$ using a table $T$, then with probability $\geq 1 - \eta$ one can assert that the probability of committing an error by the table $T$ is bounded by $R(T) < 2(K(T) \ln 2 - \frac{\ln \eta}{e})$.

**Advantage** Only need to know $K(T)$

Do not need:
(i). # of examples
(ii). Structure of code book
(iii). Which code book
(iv). # of Table in codebook
(v). Training Error made using this table

**Disadvantage** "Smart" code-book.