Easy Learning Theory for Highly Scalable Algorithms

Claudio Gentile
DICOM
Universita’ dell’Insubria, Italy
claudio.gentile@uninsubria.it
August, 2005
Content of this tutorial/1

Part 1
Intro to on-line learning problems, methods, relative loss bounds:

- On-line learning setting, examples
- Learning with expert advice (Bayes voting), relative loss bounds
- On-line learning linear-threshold functions
- Extensions: tracking the target, label-efficiency
- Learning regression functions

focus on BINARY classification
Content of this tutorial/2

Part 2
Intro to statistical Pattern Recognition problem, martingales, on-line to batch conversions:

- The statistical Pattern Recognition problem
- Types of error bounds (data/algorithim - independent/dependent)
- Reduction on-line pointwise \( \rightarrow \) i.i.d.
  - Expectation analysis
  - Data-dependent analysis
- Final comments and conclusions
### (Worst-case) on-Line Learning

<table>
<thead>
<tr>
<th></th>
<th>$E_1$</th>
<th>$E_2$</th>
<th>$E_3$</th>
<th>...</th>
<th>$E_n$</th>
<th>pred.</th>
<th>true lab.</th>
<th>loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day 1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>...</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Day 2</td>
<td>1</td>
<td>-1</td>
<td>1</td>
<td>...</td>
<td>-1</td>
<td>1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>Day 3</td>
<td>-1</td>
<td>1</td>
<td>1</td>
<td>...</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Day $t$</td>
<td>$z_{t,1}$</td>
<td>$z_{t,2}$</td>
<td>$z_{t,3}$</td>
<td>...</td>
<td>$z_{t,n}$</td>
<td>$\hat{Y}_t$</td>
<td>$y_t$</td>
<td>$\frac{1}{2}</td>
</tr>
</tbody>
</table>

**On-line protocol**

For $t = 1, ..., T$ do:

- Get vector $z_t \in \{-1, 1\}^n$
- Predict $\hat{Y}_t \in \{-1, 1\}$
- Get label $y_t \in \{-1, 1\}$
- Incur loss $\frac{1}{2}|y_t - \hat{Y}_t| \in \{0, 1\}$
Halving Algorithm [BF72]

- Predicts with majority
- If mistake is made then number of consistent Experts is (at least) halved
A run of the Halving Algorithm (HA)

<table>
<thead>
<tr>
<th>$E_1$</th>
<th>$E_2$</th>
<th>$E_3$</th>
<th>$E_4$</th>
<th>$E_5$</th>
<th>$E_6$</th>
<th>$E_7$</th>
<th>$E_8$</th>
<th>majority</th>
<th>true label</th>
<th>loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

consistent

\forall \text{ sequence with } k \text{ consistent experts (out of } n) \nHA \text{ makes } m \leq \log_2(n/k) \text{ mistakes: } n/2^m \geq k
Learning with expert advice/1

What if no expert $E_i$ is consistent?

Sequence of examples $S = (z_1, y_1), \ldots, (z_T, y_T)$

- $L_A(S)$ be total loss of alg. $A$ on sequence $S$
- $L_i(S)$ be total loss of $i$-th expert $E_i$ on $S$

Want bounds of the form:

$$\forall S : \quad L_A(S) \leq a \min_i L_i(S) + b \log(n)$$

where $a, b$ are constants

Bounds loss of algorithm relative to loss of best expert
Learning with expert advice/2

Can’t wipe out experts!
Keep one weight per expert

The Weighted Majority Algorithm [L89a,LW94]

- Predicts with larger side
- Weights of wrong experts are slashed by a factor $\beta \in [0, 1)$
Learning with expert advice/3
Number of mistakes of the WM algorithm/1

\[ L_{i,t} = \# \text{ of mistakes of } E_i \text{ before time step } t \]
\[ w_{t,i} = \beta^{L_{i,t}} \text{ weight of } E_i \text{ at beginning of time step } t \]
\[ W_t = \sum_{i=1}^{n} w_{i,t} \text{ total weight at time step } t \]

Minority \leq \frac{1}{2} W_t \quad \text{Majority} \geq \frac{1}{2} W_t

If no mistake then minority multiplied by \( \beta \): \( W_{t+1} \leq 1 \ W_t \)

If mistake then majority multiplied by \( \beta \):

\[ W_{t+1} \leq 1 \ \text{Minority} + \beta \ \text{Majority} \leq \frac{1 + \beta}{2} \ W_t \]
Learning with expert advice/3
Number of mistakes of the WM algorithm/2

Hence: \[ W_{T+1} \leq \left( \frac{1 + \beta}{2} \right)^{L_{WMA}(S)} W_1 \]

total final weight

\[ W_{T+1} = \sum_{j=1}^{n} w_{T+1,j} = \sum_{j=1}^{n} \beta^{L_j(S)} \geq \beta^{L_i(S)} \]

We got: \[ \left( \frac{1 + \beta}{2} \right)^{L_{WMA}(S)} \underbrace{W_1}_{n} \geq \beta^{L_i(S)} \]

Solving for \( L_{WMA}(S) \): \[ L_{WMA}(S) \leq \frac{\ln 1/\beta}{\ln 2} L_i(S) + \frac{1}{\ln 2} \ln n \]

\[ L_{WMA}(S) \leq \underbrace{2.63 \min_i L_i(S)}_{a} + \underbrace{2.63 \ln n}_{b} \]

\( \beta = 1/e \)
Learning with expert advice/4
Slightly more general: absolute loss/1

Alg. $A$ keeps exponential weights
and predicts with the weighted average \cite{KW99}

$$w_{t,i} = \beta^{L_{i,t}}, \quad L_{i,t} = \sum_{j<t} \frac{1}{2}|z_{i,j} - y_j|$$

$$v_{t,i} = \frac{w_{t,i}}{\sum_{i=1}^{n} w_{t,i}} \quad \text{normalized weights}$$

$$\hat{y}_t = v_{t}^T z_t,$$

where $z_{t,i} \in [-1, +1]$

Absolute loss of $A$: $L_A(S) = \sum_{t=1}^{T} \frac{1}{2}|\hat{y}_t - y_t|$
Learning with expert advice/4
Slightly more general: absolute loss/2

Relative loss bound

∀ sequences $S = (z_1, y_1), \ldots, (z_T, y_T)$, $z_t \in [-1, 1]^n$, $y_t \in [-1, 1]$,

$$L_A(S) \leq \min_i \left(1 + \eta\right) L_i(S) + \frac{1 + \eta}{\eta} \ln(n), \quad \eta = \log 1/\beta$$

Notice $a > 1$ (constant $\eta$)

Regret bounds ($a = 1$) need time-changing $\beta$ \[ACBG02\]
Learning with expert advice/5

- Weighted Majority is just a Bayes voting scheme
- Easy to combine good experts (algorithms) so that prediction alg. is almost as good as best expert
- Bounds are logarithmic in \# of experts

So far:
Learning relative to best expert/component

From now on:
Learning relative to best (thresholded) linear combination of experts/components
## A more general setting

<table>
<thead>
<tr>
<th>Instance</th>
<th>Prediction of alg $A$</th>
<th>Label</th>
<th>Loss of alg $A$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$\hat{y}_1$</td>
<td>$y_1$</td>
<td>$L(y_1, \hat{y}_1)$</td>
</tr>
<tr>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
</tr>
<tr>
<td>$x_t$</td>
<td>$\hat{y}_t$</td>
<td>$y_t$</td>
<td>$L(y_t, \hat{y}_t)$</td>
</tr>
<tr>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
</tr>
<tr>
<td>$x_T$</td>
<td>$\hat{y}_T$</td>
<td>$y_T$</td>
<td>$L(y_T, \hat{y}_T)$</td>
</tr>
</tbody>
</table>

**Total Loss**

$\frac{1}{n} \sum_{i=1}^{T} L(y_i, \hat{y}_i)$

Sequence of examples $S = (x_1, y_1), ..., (x_T, y_T) \in \mathbb{R}^n \times \{-1, 1\}$

**Comparison class** $\{u\}$

Relative loss $L_A(S) - \inf_{\{u\}} Loss_u (S)$

**Goal:** Bound relative loss for arbitrary sequence $S$
Learning linear-threshold functions

Another run of the Halving Algorithm

Sequence of examples $S = (x_1, y_1), \ldots, (x_T, y_T) \in \mathbb{R}^2 \times \{-1, 1\}$

$S$ is lin. separated by $u \in \mathbb{R}^2 : \|u\|_2 = 1$ with margin

$0 < \gamma \leq y_t u^T x_t \ \forall t \quad R = \max_t \|x_t\|_2$

Experts:
$n$ (large) linear−threshold functions evenly spread over unit circle
Expert $i$ predicts $z_{it} = \text{sgn}(u_i^T x_t)$

Feed experts with $x_1$ and get expert prediction vector $z_1$
Learning linear-threshold functions/1

Another run of the Halving Algorithm/2

\[ m_{HA} \leq \log_2(n/k) = O(\log(R/\gamma)) \text{ for large } n \]
Learning linear-threshold functions/1
Another run of the Halving Algorithm/3

[HG02,GBNT04,...]

For $d$-dim vectors:

$$m_{HA} \leq \log_2 1/\mu(\text{consistent}(S))$$

$$= O(d \log(R/\gamma)),$$

$$R = \max_t ||x_t||_2$$

Proof: $y_t u^\top x_t \geq \gamma$ and $||u - u'||_2 < \gamma/R$

$\implies y_t (u')^\top x_t > 0$

$\implies \exists$ ball $B$ of radius $\gamma/2R$: $B \subseteq \text{consistent}(S)$,

$\mu(B) = (\gamma/2R)^{d-1}\mu$ (surface of $d$-dim unit sphere)

Drawbacks: Linear dependence on dimension $d$

Looks too time-consuming (linear dependence on $d$?)
Learning linear-threshold functions/2
The (first-order) Perceptron algorithm \footnote{Ro62, ...}

Keep weight vector $\mathbf{w}_t \in \mathbb{R}^n$

In trial $t$:

- Get instance $\mathbf{x}_t \in \mathbb{R}^n$
- Predict with $\hat{y}_t = \text{SGN}(\mathbf{w}_t^\top \mathbf{x}_t) \in \{-1, 1\}$
- Get label $y_t \in \{-1, 1\}$
- **If mistake** $(y_t \mathbf{w}_t^\top \mathbf{x}_t \leq 0)$ **then** update $\mathbf{w}_{t+1} := \mathbf{w}_t + y_t \mathbf{x}_t$
Learning linear-threshold functions/3

Perceptron convergence theorem/1 \[\text{[Bl62,No62,...]}\]

Arbitrary sequence \(S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T) \in \mathbb{R}^n \times \{-1, 1\}\)

\[
\# \text{ of mistakes} \leq \inf_{\gamma > 0, \|u\|_2 = 1} \left( D_\gamma(u; S') + \frac{\sqrt{\sum_{t \in M} \|x_t\|_2^2}}{\gamma} \right) \]

\(M\) is set of mistaken trials \(t\),

\(D_\gamma(u; S) = \sum_{t \in M} \max\{0, 1 - y_t u^\top x_t / \gamma\}\)

\[
\max\{0, 1 - y u^\top x / \gamma\}
\]
Learning linear-threshold functions/3
Perceptron convergence theorem/2

When $S$ is separated by $\mathbf{u} : ||\mathbf{u}||_2 = 1$ with margin
$\gamma \leq y_t \mathbf{u}^\top \mathbf{x}_t \ \forall t$
gets
$\# \text{ of mistakes} \leq \frac{R^2}{\gamma^2}$,

$||\mathbf{x}_t|| \leq R$

Pointwise bound:
Depends on radius $R$ and margin $\gamma$
Learning linear-threshold functions/4
The second-order Perceptron algorithm [CBCG05]

Keep weight vector $w_t \in \mathbb{R}^n$ and matrix $S_t$

In trial $t$:

- Get instance $x_t \in \mathbb{R}^n$
- Predict with $\hat{y}_t = \text{SGN}(w_t^T(aI + S_t)^{-1}x_t) \in \{-1, 1\}$
- Get label $y_t \in \{-1, 1\}$

- **If** mistake **then** update
  - $w_{t+1} := w_t + y_t \hat{x}_t$
  - $S_{t+1} = S_t + \hat{x}_t\hat{x}_t^\top$,
    \[
    \hat{x}_t = x_t / \|x_t\| \quad \text{positive parameter}
    \]

Turns to first-order when $a \rightarrow \infty$
Learning linear-threshold functions \(\mathbf{5}\) 
Second-order convergence theorem \([\text{Ge04}]\)

When \(S = (\hat{x}_1, y_1), \ldots, (\hat{x}_T, y_T) \in \mathbb{R}^n \times \{-1, 1\}\) is separated by \(\mathbf{u}\) with margin \(\gamma \leq y_t \mathbf{u}^\top \hat{x}_t, \|\hat{x}_t\| \leq 1 \forall t\) gets

\[
\# \text{ of mistakes} \leq \frac{a + \sum_{i=1}^{n} \ln(1 + \frac{\lambda_i}{a})}{\gamma}
\]

More complicated bound in the nonseparable case

Pointwise bound:
Depends on eigenstructure \(\{\lambda_i\}\) of Gram matrix \([\hat{x}_j^\top \hat{x}_k]_{j,k \in \mathcal{M}}\) and linearly on inverse margin \(\gamma\)
Learning linear-threshold functions/6

Kernel Perceptron [FS98,...]

Keep pool of ”support vectors” $\mathcal{M}_t$

In trial $t$:

- Get instance $\mathbf{x}_t \in \mathbb{R}^n$
- Predict with $\hat{y}_t = \text{SGN}(\sum_{i \in \mathcal{M}_t} y_i K(\mathbf{x}_i, \mathbf{x}_t)) \in \{-1, 1\}$
- Get label $y_t \in \{-1, 1\}$
- If mistake then update $\mathcal{M}_{t+1} := \mathcal{M}_t \cup \{t\}$
Learning linear-threshold functions/7
Kernel Perceptron convergence theorem/1

Arbitrary sequence $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T) \in \mathbb{R}^n \times \{-1, 1\}$

$$\# \text{ of mist.} \leq \inf_{\gamma > 0, f \in H_K, \|f\| = 1} \left( D_\gamma(f; S) + \frac{\sqrt{\sum_{t \in \mathcal{M}} K(\mathbf{x}_t, \mathbf{x}_t)}}{\gamma} \right)$$

$H_K = \{ f(\cdot) = \sum_{t=1}^T \alpha_t K(\mathbf{x}_t, \cdot) : \alpha_t \in \mathbb{R} \}$,

$\mathcal{M}$ is set of mistaken trials $t$,

$D_\gamma(f; S) = \sum_{t \in \mathcal{M}} \max\{0, 1 - y_t f(\mathbf{x}_t)/\gamma\}$

Separable case:

$\# \text{ of mistakes} \leq R^2/\gamma^2, \quad K(\mathbf{x}_t, \mathbf{x}_t) \leq R^2$
Learning linear-threshold functions / 8
Kernel Second-order Perceptron

Keep pool of ”support vectors” $\mathcal{M}_t$

In trial $t$:

- Get instance $\boldsymbol{x}_t \in \mathbb{R}^n$
- Predict with $\hat{y}_t = \text{SGN} \left( \begin{pmatrix} y_t \\ \hat{y}_t \end{pmatrix}^\top \left( a I + \underbrace{\hat{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)_{i,j\in\mathcal{M}_t}}_{\text{current Gram matrix}} \right)^{-1} \begin{pmatrix} \hat{y}_t \\ \mathbf{v}_t \end{pmatrix} \right) \in \{-1, 1\}$

- Get label $y_t \in \{-1, 1\}$

- **If mistake** then update $\mathcal{M}_{t+1} := \mathcal{M}_t \cup \{t\}$
Learning linear-threshold functions/9
Kernel Second-order convergence theorem

When \( S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T) \in \mathbb{R}^n \times \{-1, 1\} \)
is separated by \( f(\cdot) = \sum_{t=1}^{T} \alpha_t \hat{K}(\mathbf{x}_t, \cdot), \alpha_t \in \mathbb{R}, \)
with margin \( \gamma \leq y_t f(\mathbf{x}_t) \forall t \)
gets
\[
\# \text{ of mist.} \leq \frac{a + \sum_i \ln(1 + \frac{\lambda_i}{a})}{\gamma},
\]

\( \lambda_i \) is \( i \)-th eigenvalue of (normalized) kernel Gram matrix
\( [\hat{K}(\mathbf{x}_i, \mathbf{x}_j)]_{i,j} \in \mathcal{M}, \)
\( \mathcal{M} \) is set of mistaken trials
Learning linear-threshold functions/10
Second-order Perceptron: computational aspects

Time per trial

Primal formulation
compute \((aI + S_t)^{-1}\) based on \((aI + S_{t-1})^{-1}\)
\(O(n^2)\) extra time per trial

Dual formulation
compute \( \left( aI + [\hat{G}(\mathbf{x}_i, \mathbf{x}_j)]_{i,j \in \mathcal{M}_t} \right)^{-1} \)
based on \( \left( aI + [\hat{G}(\mathbf{x}_i, \mathbf{x}_j)]_{i,j \in \mathcal{M}_{t-1}} \right)^{-1} \)
\(O(\mathcal{M}_t^2)\) extra inner products (kernel evaluations) per trial

# of mistakes so far
Learning linear-threshold functions/11
Additive algorithms

An additive algorithm (e.g. first/second-order Perceptron):

• Relies on linear algebra
• Is rotation invariant (depends on data via angles)
• Can be easily kernelized ($\mathbf{x}_i^\top \mathbf{x}_j \rightarrow K(\mathbf{x}_i, \mathbf{x}_j)$)
• Has no bias for axes-parallel directions (no feature selection)
Learning linear-threshold functions/12
Nonadditive algorithms

- No linear algebra
- No rotation invariance
- Harder to kernelize
- Bias for sparse solutions (built-in feature selection)

Example: $p$-norm algorithms
Learning linear-threshold functions/13

$p$-norm algs [GLS01, GL99, Ge03]

Keep weight vector $\mathbf{w}_t \in \mathbb{R}^n$

In trial $t$: \[ f(\cdot) = \nabla \frac{1}{2} \| \cdot \|_p^2, \ p \geq 2 \]

- Get instance $\mathbf{x}_t \in \mathbb{R}^n$
- Predict $\hat{y}_t = \text{SGN}(f(\mathbf{w}_t)^\top \mathbf{x}_t) \in \{-1, 1\}$
- Get label $y_t \in \{-1, 1\}$
- If mistake then update $\mathbf{w}_{t+1} := \mathbf{w}_t + y_t \mathbf{x}_t$

Notice:
- $p = 2$ gets (first-order) Perceptron
- $p = O(\ln n)$ gets Weighted Majority/Winnow [L88, LW94]
- $2 < p < O(\ln n)$ interpolates between the two extremes
Learning linear-threshold functions/14

\( p \)-norm Perceptron convergence theorem/1

[GLS01, GL99, Ge03]

Arbitrary sequence \( S = (x_1, y_1), \ldots, (x_T, y_T) \in \mathbb{R}^n \times \{-1, 1\} \)

\[
\# \text{ mistakes} \leq \inf_{\gamma > 0, ||u||_q = 1} \left( D_\gamma(u; S) + \frac{\sqrt{(p - 1) \sum_{t \in M} ||x_t||_p^2}}{\gamma} \right)
\]

\( M \) is set of mistaken trials \( t \),

\[
D_\gamma(u; S) = \sum_{t \in M} \max\{0, 1 - y_t u^\top x_t / \gamma\}
\]
Learning linear-threshold functions/14

$p$-norm Perceptron convergence theorem/2

When $S$ is separated by $\mathbf{u} : ||\mathbf{u}||_q = 1$ with margin

$$\gamma \leq y_t \mathbf{u}^T \mathbf{x}_t \forall t \ (1/p + 1/q = 1)$$

dual norms

gets

$$\# \ of \ mistakes \leq (p - 1) \frac{R^2}{\gamma^2}$$

$$||\mathbf{x}_t|| \leq R$$

Pointwise bound:

Depends on $p$-norm radius $R$

and $(q$-norm$)$ margin $\gamma$
Learning linear-threshold functions/15

\[ p\text{-norm algorithms with kernels/1 (wild slide)} \quad [\text{Ge04}] \]

\[
\begin{align*}
1 \\
x_1 \\
x_2 \\
& \quad \cdots \\
x_n \\
x_1 \ x_2 \\
& \quad \cdots \\
x_1 \ x_2 \ \cdots \ x_n \\
\end{align*}
\]

\[
K(x, y) = \Phi(x) \Phi(y) = \prod_{i=1}^{n} (1 + x_i y_i) \quad \text{(Simple poly kernel)}
\]
Learning linear-threshold functions/15

$p$-norm algorithms with kernels/2 (wild slide)

$p$-norm hypothesis: \( w = \sum_{i \in M} y_i \Phi(x_i) \)

\[ p\text{-norm margin: } = f(w)^\top \Phi(x) \]

\[ f(w) = w^{p-1} \]

\[ = \left( \sum_{i \in M} y_i \Phi(x_i)^\top \right)^{p-1} \Phi(x) \]

\[ \text{expand!} \]

Then expand polynomial and use \( \Phi(x)\Phi(y) = \Phi(xy) \)
Learning linear-threshold functions/15

$p$-norm algorithms with kernels/3 (wild slide)

Example: $p = 4$, $f(w) = w^3$

$$w = y_1 \Phi(x_1) + y_2 \Phi(x_2)$$

follow pattern $(a + b)^3 = a^3 + 3a^2b + 3ab^2 + b^3$

$$f(w) =$$

$$y_1^3 \Phi^3(x_1) + 3y_1^2 y_2 \Phi^2(x_1) \Phi(x_2) + 3y_1 y_2^2 \Phi(x_1) \Phi^2(x_2) +$$

$$y_2^3 \Phi^3(x_2) =$$

$$y_1 \Phi(x_1^3) + 3y_2 \Phi(x_1^2) \Phi(x_2) + 3y_1 \Phi(x_1) \Phi(x_2^2) + y_2 \Phi(x_2^3) =$$

$$y_1 \Phi(x_1^3) + 3y_2 \Phi(x_1^2 x_2) + 3y_1 \Phi(x_1 x_2^2) + y_2 \Phi(x_2^3)$$

Then $p$-norm margin $f(w)^\top \Phi(x) =$

$$y_1 K(x_1^3, x) + 3y_2 K(x_1^2 x_2, x) + 3y_1 K(x_1 x_2^2, x) + y_2 K(x_2^3, x)$$
Batch algorithm run on-line

\[ S = (x_1, y_1), \ldots, (x_T, y_T) \in \mathbb{R}^n \times \{-1, +1\} \]

\( A \) is generic batch classification alg.

In trial \( t \):

- train \( A \) on prefix \( S_{t-1} = (x_1, y_1), \ldots, (x_{t-1}, y_{t-1}) \)

- Get \( h_t(x) = A_{S_{t-1}}(x) \)

- Mistake if \( h_t(x_t) \neq y_t \)

Can count \# mistakes on sequence \( S \)
Tracking the target:
Motivation and relative loss \[\text{[LW94,HW98,AW98]}\]

Previous methods yield good results when single target (expert/vector) is good on whole sequence \(S\)

Non-stationary tasks:
good targets change with time

Given \(S = (x_1, y_1), \ldots, (x_T, y_T)\), want to bound

\[
L_A(S) - a \min_{u_1, u_2, \ldots, u_T} \left( L_0(u_0, u_1, \ldots, u_T) + \sum_{t=1}^{T} \text{dist}(u_t, u_{t+1}) \right)
\]

loss of "shifting" target
Tracking the target: Examples

- Experts framework:
  \( u_t \) are the \( n \) unit vectors, \( x_t \in [-1, +1]^n \)

\[
\text{Loss}_{u_0, u_1, \ldots, u_T}(S) = \sum_{t=1}^{T} \frac{1}{2} |u_t^\top x_t - y_t| \quad \text{(mistakes)}
\]

\[
dist(u_t, u_{t+1}) = ||u_{t+1} - u_t|| = \begin{cases} 
1 & \text{if } u_{t+1} \neq u_t \\
0 & \text{otherw.}
\end{cases} \quad \text{(shifts)}
\]

- Linear-threshold functions:
  \( u_t, x_t \in \mathbb{R}^n \)

\[
\text{Loss}_{u_0, u_1, \ldots, u_T}(S) = \sum_{t=1}^{T} \max\{0, 1 - y_t u_t^\top x_t / \gamma\}
\]

\[
dist(u_t, u_{t+1}) = ||u_{t+1} - u_t||
\]
Tracking the target: Algorithms/1

- Experts framework:
Basic expert algs. (exponential update) prevent 
exterts to recover when they start performing better

Modification to Weighted Majority (fixed share alg.):

• $w'_{t,i} = \beta^{L_{i,t+1}}$

• Each expert sends fraction $\frac{\alpha}{n-1}$, $\alpha > 0$, 
of total weight to other $n - 1$ experts $\rightarrow w_{t+1}$

• $v_{t+1,i} = w_{t+1,i} / \sum_{j=1}^{n} w_{t+1,j}$

• $\hat{y}_{t+1} = v_{t+1}^T x_{t+1}$, $x_{t+1,i} \in [-1, +1]$
Tracking the target: Algorithms/2

- Linear-threshold functions:

Same story ...

Modification to linear-threshold algs. (projection-based)

Simplest example:

- \( \mathbf{w}_t' = \mathbf{w}_t + \eta y_t x_t \)

- *projects* \( \mathbf{w}_t' \) onto
  - convex set (e.g. unit ball)
  \( \rightarrow \mathbf{w}_{t+1} \)

- \( \hat{y}_{t+1} = \text{SGN}(\mathbf{w}_{t+1}^\top x_{t+1}) \)
Tracking the target: Bounds

- Experts framework:

\[ A \text{ is share update alg. with } \alpha = \frac{k}{T-1} \]

\[ L_A(S) \leq \min_{P(k)} L_{P(k)}(S) + b \left( k \log(T/K) + k \log n \right) \]

- Linear-threshold functions:

Simple case: compare to sequence of linear separators

\[ u_1, \ldots, u_T : 0 < \gamma \leq y_s u_t^T x_s \quad s, t = 1, \ldots, T \]

\[ A \text{ is projection-based Perceptron with } \eta = \frac{\gamma}{2R^2}, \quad \|x_t\| \leq R \]

\[ \text{# of mistakes} \leq \frac{4R^2}{\gamma^2} \left( \frac{1}{2} + \sum_{t=1}^{T} \|u_{t+1} - u_t\| \right) \]

Can be generalized in several ways
Label-efficient prediction [ACL90, FSST97, CGZ04...]

Labels are scarce and/or expensive

Learner decides in on-line fashion which labels are needed:
The higher the margin the less likely label $y_t$ is requested
Label-efficient prediction: Algorithms

Simplest example: Label-efficient Perceptron

Keep weight vector \( \mathbf{w}_t \in \mathbb{R}^n \) and margin \( r_t \in \mathbb{R} \)

In trial \( t \):

- Get instance \( \mathbf{x}_t \in \mathbb{R}^n \), set \( r_t = \mathbf{w}_t^\top \mathbf{x}_t \)
- Predict with \( \hat{y}_t = \text{sgn}(r_t) \in \{-1, 1\} \)
- Draw Bernoulli random variable \( Z_t \in \{0, 1\} \) of bias \( \frac{b}{b+|r_t|} \)

- **If** \( Z_t = 1 \) **then**
  - Ask for label \( y_t \in \{-1, 1\} \)
  - if mistake \( (y_tr_t \leq 0) \) then update \( \mathbf{w}_{t+1} = \mathbf{w}_t + y_t \mathbf{x}_t \)

**Else** \( (Z_t = 0) \) \( \mathbf{w}_{t+1} = \mathbf{w}_t \)
Label-efficient prediction: Bounds

Simplest result:
When $S$ is separated by $u : \|u\|_2 = 1$ with margin $\gamma \leq y_t u^\top x_t \forall t$ and $\|x_t\| \leq R$ gets

$$\text{Expected \ # \ of \ labels} = \sum_{t=1}^{T} \mathbb{E} \left[ \frac{b}{b + |r_t|} \right] \leq T,$$

and, for $b = R^2/2$,

$$\text{Expected \ # \ of \ mistakes} \leq \frac{R^2}{\gamma^2}$$

Can be generalized in several ways
Label-efficient prediction: Empirical evidence/1
Label-efficient prediction: Empirical evidence/2

(c)

(d)
End of Part 1
Generalization bounds/1

Given

- class $\mathcal{H}$ of $\pm 1$ functions
- i.i.d. sequence $S = (X_1, Y_1), \ldots, (X_T, Y_T)$ over $\mathbb{R}^n \times \{-1, 1\}$, want to compute hypothesis $\hat{H} = \hat{H}_S$ with small risk

$$\text{risk}(\hat{H}) = \mathbb{E}_{X, Y}[\text{loss}(Y, \hat{H}(X))]$$:

$$\mathbb{P} \left( \text{risk}(\hat{H}) \leq \inf_{h \in \mathcal{H}} \text{risk}(h) + \epsilon \right) \geq 1 - \delta$$

0–1 loss in our case
Generalization bounds/2: VC Uniform conv. [VC71]

Key quantity is **empirical risk**

\[
\text{risk}_{\text{emp}}(h) = \frac{1}{T} \sum_{t=1}^{T} \text{loss}(Y_t, h(X_t))
\]

VC-bound:

\[
\mathbb{P} \left( \sup_{h \in \mathcal{H}} |\text{risk}_{\text{emp}}(h) - \text{risk}(h)| \geq c \sqrt{\frac{d + \ln 1/\delta}{T}} \right) \leq \delta
\]

\[\Rightarrow \quad \hat{H} = \arg\inf_{h \in \mathcal{H}} \text{risk}_{\text{emp}}(h) \quad \text{is s.t.} \quad \mathbb{P} \left( \text{risk}(\hat{H}) \leq \inf_{h \in \mathcal{H}} \text{risk}(h) + 2c \sqrt{\frac{d + \ln 2/\delta}{T}} \right) \geq 1 - \delta \]
Generalization bounds/3: Data-dep. uniform conv./1

[Ba98, BLM00, WSTSS99, BM02, ...]

\[
\sqrt{\frac{d + \ln 2/\delta}{T}} \rightarrow C_T(S) + \sqrt{\frac{\ln 1/\delta}{T}}
\]

\(C_T(S) = C_T(S, \mathcal{H})\)

is sample statistic:

\[
\begin{align*}
\text{empirical VC-entropy} & \quad \text{[BLM00, WSTSS99]} \\
\text{Rademacher complexity} & \quad \text{[BM02]} \\
\text{Maximum discrepancy} & \quad \text{[BLM00]} \\
\text{...}
\end{align*}
\]

Stronger than VC since \(C_T(S) \approx \mathbb{E}[C_T(S)] \ll \sqrt{d/T}\)
Generalization bounds/3: Data-dep. uniform conv./2

Others (e.g., margin-based bounds for linear-threshold functions) [AKLL02,KP02,LSM01,SFBL98, ...]

\[
\mathbb{P}\left( \forall h \in \mathcal{H} : \text{risk}(h) \leq \text{risk}_{\text{emp}}(h) + C_T(h, S) + c \sqrt{\frac{\ln 1/\delta}{T}} \right) \geq 1 - \delta
\]

Leave algorithmic problem of computing \( h \in \mathcal{H} \) optimizing trade-off

\[
\text{risk}_{\text{emp}}(h) \text{ vs } C_T(h, S)
\]
Digression: martingales/1

Coin-tossing game:
$X_1, X_2, ..., X_t$ are ±1 i.i.d. variables with $\mathbb{P}(X_t = +1) = 1/2$

Gambler’s strategy:
$L_t = L_t(X_1, X_2, ..., X_t)$ is gain (or loss) at time $t$

$L_1, L_2, ..., L_t$ are no longer independent

Game is fair if

$$\mathbb{E}[L_{t+1} \mid X_1, ..., X_t] = 0 \quad \text{(w.p.1) \ \forall t}$$

- Sequence $L_1, L_2, ..., $ is martingale difference sequence (w.r.t. $X_1, X_2, ...,$)

- Partial sums $S_t = L_1 + L_2 + ... + L_t$ is martingale (w.r.t. $X_1, X_2, ...,$)
Digression: martingales/2

Laws of large numbers
(empirical average concentrates around mean)
extend to martingales

<table>
<thead>
<tr>
<th>Independent variables</th>
<th>Dependent variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Zero-mean) independent r.v.</td>
<td>Martingale diff. sequence</td>
</tr>
<tr>
<td>Sum of (zero-mean) indep. r.v.</td>
<td>Martingale</td>
</tr>
</tbody>
</table>

(Hoeffding-Azuma) If $L_1 + L_2 + ... + L_T$ is martingale difference sequence with bounded $L_t$

$$\frac{L_1 + L_2 + ... + L_T}{T} \approx 0 \quad \text{(with high probability)}$$
On-line pointwise bounds

\[ S = (x_1, y_1), \ldots, (x_T, y_T) \]

Pointwise bounds so far:

Total \# mistakes \( A(S) \) \( \leq \) some function \( S \)

\( n, R, \gamma \) (Halving) \quad R, \gamma \) (1st Perc) \quad \lambda_i, \gamma \) (2nd Perc) \quad R, \gamma \) (dual) \quad \ldots \quad (p-\text{norm})
On-line pointwise $\rightarrow$ i.i.d. data-dependent/1

Sweep through sequence of examples $S$ just once!

Get sequence of hypotheses

$H_0, H_1, H_2, ..., H_T: H_t = H_t((x_1, y_1), ..., (x_t, y_t))$

**Goal:** Extract one with small risk

Early ref: [L] (separate test set)
On-line pointwise → i.i.d. data-dependent/2

Which one?
1. Last one: $H_T$ (back to uniform convergence ...)
2. Average one: $\overline{H} = \frac{1}{T} \sum_{t=0}^{T} H_t \in [0, 1]$ (convex upper bound on 0-1 loss)
3. Best penalized one:

$$\text{risk}_{\text{emp}}(H_t, t + 1) = \frac{1}{T - t} \sum_{i=t+1}^{T} \text{loss}(Y_i, H_t(X_i))$$

$$\hat{H} = \arg\min_{t=0...T-1} \left( \text{risk}_{\text{emp}}(H_t, t + 1) + \sqrt{\frac{1}{T - t} \ln \frac{T}{\delta}} \right)$$

penalty
On-line pointwise $\rightarrow$ i.i.d. data-dependent/3
Proof technique/1

$$H_0 \leftarrow (X_0, Y_0)$$

$$H_1 \leftarrow (X_1, Y_1) \quad H_0 = H_1((X_1, Y_1))$$

$$H_2 \leftarrow (X_2, Y_2) \quad H_1 = H_2((X_1, Y_1), (X_2, Y_2))$$

... ... ... ...

Build martingale kind of process

$$L_t = L_t((X_1, Y_1), ..., (X_t, Y_t))$$

$$= \text{loss}(Y_t, H_{t-1}(X_t)) - \text{risk}(H_{t-1})$$

$$\mathbb{E}[\text{loss}(Y_t, H_{t-1}(X_t)) \mid (X_1, Y_1), ..., (X_{t-1}, Y_{t-1})]$$
On-line pointwise → i.i.d. data-dependent/3
Proof technique/2

From the very definition of \( L_t \)
\( L_1, L_2, ..., L_T \) is bounded (\(|L_t| \leq 1\))
martingale difference sequence
w.r.t. \((X_1, Y_1), ..., (X_T, Y_T)\)

Hoeffding-Azuma:

\[
\frac{1}{T} \sum_{t=1}^{T} \left[ \text{loss}(Y_t, H_{t-1}(X_t)) - \text{risk}(H_{t-1}) \right] \approx 0
\]
On-line pointwise $\rightarrow$ i.i.d. data-dependent/3
Proof technique/3

$$\frac{1}{T} \sum_{t=1}^{T} \text{loss}(Y_t, H_{t-1}(X_t)) \approx \frac{1}{T} \sum_{t=1}^{T} \text{risk}(H_{t-1})$$

\# of mistakes

$\{\geq \text{risk}(\overline{H}) \}$

(*) (Hoeffding-Azuma) \hspace{1cm} [DGL96]

(**) bounded and convex (Jensen)

(***) general bounded (Chernoff-Hoeffding) \hspace{1cm} [DGL96]
On-line pointwise $\rightarrow$ i.i.d. data-dependent/4

Simplest bounds

Convex: $\mathbb{P} \left( \text{risk}(\overline{H}) \leq M_T + L \sqrt{\frac{2}{T} \ln \frac{2}{\delta}} \right) \geq 1 - \delta$

bound on range of convex loss

More general: $\mathbb{P} \left( \text{risk}(\hat{H}) \leq M_T + 6 \sqrt{\frac{1}{T} \ln \frac{T}{\delta}} \right) \geq 1 - \delta$
On-line pointwise $\rightarrow$ i.i.d. data-dependent /5
Some applications: plug and play /1

Recall bound on Halving Algorithm for separable case:

$$M_T \leq \frac{1}{T} O(d \log(R/\gamma))$$

Just plug back into

$$\mathbb{P} \left( \text{risk}(\hat{H}) \leq M_T + 6 \sqrt{\frac{1}{T} \ln \frac{T}{\delta}} \right) \geq 1 - \delta$$

Gets

$$\mathbb{P} \left( \text{risk}(\hat{H}) \leq \frac{1}{T} O(n \log(R/\gamma)) + 6 \sqrt{\frac{1}{T} \ln \frac{T}{\delta}} \right) \geq 1 - \delta$$

Similar to [HG02]
On-line pointwise $\rightarrow$ i.i.d. data-dependent/5
Some applications: plug and play/2
Recall bound on Kernel Perceptron:

$$M_T \leq \inf_{\gamma > 0, \ f \in H_K, \|f\| = 1} \frac{1}{T} \left( D_\gamma(f; S) + \frac{\sqrt{\sum_{t \in \mathcal{M}} K(x_t, x_t)}}{\gamma} \right)$$

Separable case:

$$M_T \leq \frac{1}{T} \max_{t \in \mathcal{M}} K(x_t, x_t)$$

Plug back into

$$\mathbb{P} \left( \text{risk}(\hat{H}) \leq M_T + 6 \sqrt{\frac{1}{T} \ln \frac{T}{\delta}} \right) \geq 1 - \delta$$

Similar to [BM02] for SVM
On-line pointwise $\rightarrow$ i.i.d. data-dependent/5
Some applications: plug and play/3

Recall bound on Kernel Second-order Perceptron
(separable case)

$$ M_T \leq \frac{1}{T} \frac{a + \sum_i \ln (1 + \frac{\lambda_i}{a})}{\gamma}, $$

Plug into

$$ \mathbb{P} \left( \text{risk}(\hat{H}) \leq M_T + 6 \sqrt{\frac{1}{T} \ln \frac{T}{\delta}} \right) \geq 1 - \delta $$

Similar to [WSTSS99] for SVM
On-line pointwise $\rightarrow$ i.i.d. data-dependent/5

Some applications: plug and play/4

Recall bound on Perceptron for shifting targets

$$M_T \leq \frac{1}{T} \frac{4 R^2}{\gamma^2} \left( \frac{1}{2} + \sum_{t=1}^{T} \|\mathbf{u}_{t+1} - \mathbf{u}_t\| \right)$$

Again, combine with

$$\mathbb{P} \left( \text{risk} (\hat{H}) \leq M_T + 6 \sqrt{\frac{1}{T} \ln \frac{T}{\delta}} \right) \geq 1 - \delta$$
On-line pointwise → i.i.d. data-dependent/5
Some applications: plug and play/5

Recall bounds on label-efficient Perceptron:

$$
\mathbb{E}(\ast)[M_T] \leq \frac{1}{T} \frac{R^2}{\gamma^2}
$$

$$
\mathbb{E}(\ast)[\text{\# of labels}] = \sum_{t=1}^{T} \mathbb{E}(\ast) \left[ \frac{b}{b + |r_t|} \right]
$$

(*) = internal randomization

Get with prob > 1 − δ:

$$
\mathbb{E}(\ast)[\text{risk(\hat{H})}] \leq \frac{1}{T} \frac{R^2}{\gamma^2} + 6 \sqrt{\frac{1}{T} \ln \frac{T}{\delta}}
$$

and
On-line pointwise $\rightarrow$ i.i.d. data-dependent
Some applications: plug and play

A is your favourite batch classification alg.
Run it in on-line fashion and count $\#$ of mistakes

Still get

$$\Pr \left( \text{risk}(\hat{H}) \leq M_T + 6 \sqrt{\frac{1}{T} \ln \frac{T}{\delta}} \right) \geq 1 - \delta$$

No direct mention of, e.g., “complexity of function classes”

Try it yourself with other specific algs.
On-line pointwise $\rightarrow$ i.i.d. data-dependent/6

Remarks

These bounds:

- are algorithm-specific (NO uniform convergence arguments, closer in spirit to algorithmic stability/luckiness) [BE02,HeWi02,...]

- proven by simple large deviation on martingales

- refer to efficient algs (on-line, one sweep)

- are tight (I believe ...)

- Are widely applicable (in principle)
On-line pointwise $\rightarrow$ i.i.d. data-dependent/7

Refinements/1

$$\mathbb{P}\left( \text{risk}(\hat{H}) \leq \min_{t=0\ldots T-1} \left( M_{t,T} + 6 \sqrt{\frac{1}{T-t} \ln \frac{T}{\delta}} \right) \right) \geq 1 - \delta,$$

where $M_{t,T} = \frac{1}{T-t} \sum_{i=t+1}^{T} \text{loss}(Y_i, H_{i-1}(X_i))$ (loss on suffix)

Basically “$\min_{t=0\ldots T-1}$” replaces “$t = 0$”
On-line pointwise $\rightarrow$ i.i.d. data-dependent/7

Refinements/2

$$\mathbb{P}\left(\text{risk}(\hat{H}) \leq M_T + O\left(\frac{1}{T} \ln \frac{T}{\delta} + \sqrt{\frac{M_T}{T} \ln \frac{T}{\delta}}\right)\right) \geq 1 - \delta,$$

$$\hat{H} = \arg\min_{t=0\ldots T-1} \left(\text{risk}_{\text{emp}} + \frac{1}{T-t} \ln \frac{T}{\delta} + \sqrt{\frac{\text{risk}_{\text{emp}}}{T-t} \ln \frac{T}{\delta}}\right)$$

$$\text{risk}_{\text{emp}} = \text{risk}_{\text{emp}}(H_t, t+1) = \frac{1}{T-t} \sum_{i=t+1}^{T} \text{loss}(Y_i, H_t(X_i))$$

(Uses Bernstein-type inequalities for martingales) [F75, DZ01]

Can be combined with “Refinement/1”
Conclusions

- Pointwise bounds for on-line algorithms directly turn to (tight) data-dependent i.i.d. bounds
- Easy plug and play
- Resulting algs. are still as efficient as on-line (one epoch over training sequence)
- Simple proofs, algorithm-specific, no uniform convergence
- Can be immediately extended to regression frameworks