On Boosting Sparse Parities

Lev Reyzin

UIC

@ ISAIM 2014
Boosting converts a “weak learner” to a “strong learner”

- **weak learner**: achieves error $\frac{1}{2} - \gamma$ on any distribution
- **strong (PAC) learner**: achieves arbitrarily small error on any distribution

useful in theory:
- show a class is PAC learnable by showing it’s weakly learnable.
- many interesting explanations why it works

useful in practice:
- weak learners are easy to design – can be “boosted”
Algorithm 1 AdaBoost [Freund and Schapire, 1997]

Given: \((x_1, y_1), \ldots, (x_m, y_m)\),
where \(x_i \in X, y_i \in Y = \{-1, +1\}\).

Initialize \(D_1(i) = 1/m\).

for \(t = 1, \ldots, T\) do

Train base learner using distribution \(D_t\).

Get base classifier \(h_t : X \rightarrow \{-1, +1\}\).

Let \(\gamma_t = \sum_i D_t(i)y_ish_t(x_i)\).

Choose:
\[
\alpha_t = \frac{1}{2} \ln \frac{1 + \gamma_t}{1 - \gamma_t}.
\]

Update:
\[
D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_ty_ih_t(x_i))}{Z_t},
\]

where \(Z_t\) is a normalization factor (chosen so that \(D_{t+1}\)
will be a distribution).

end for

Output the final classifier:
\[
H(x) = \text{sign} \left( \sum_{t=1}^{T} \alpha_th_t(x) \right).
\]
Algorithm 1 AdaBoost [Freund and Schapire, 1997]

Given: \((x_1, y_1), \ldots, (x_m, y_m)\),
where \(x_i \in X, y_i \in Y = \{-1, +1\}\).

Initialize \(D_1(i) = 1/m\).

for \(t = 1, \ldots, T\) do

Train base learner using distribution \(D_t\).

Get base classifier \(h_t : X \rightarrow \{-1, +1\}\).

Let \(\gamma_t = \sum_i D_t(i) y_i h_t(x_i)\).

Choose:
\[
\alpha_t = \frac{1}{2} \ln \frac{1 + \gamma_t}{1 - \gamma_t}.
\]

Update:
\[
D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t},
\]
where \(Z_t\) is a normalization factor (chosen so that \(D_{t+1}\)
will be a distribution).

end for

Output the final classifier:
\[
H(x) = \text{sign} \left( \sum_{t=1}^{T} \alpha_t h_t(x) \right).
\]
Weak Learners Used in Practice

decision stumps:
  – predict using 1 (best) feature

decision trees:
  – grow a tree, prune, etc.
  – (here we’ll use CART trees)
Boosting Theory

• weak learning = strong learning
• proving PAC bounds
• margin bounds
• dynamics of AdaBoost
• ...

• almost nothing on weak learning
What do we want in a weak learner?

we define 7 desirable properties: more intuitive than formal

1. diversity
2. coverage
3. simplicity
4. error
5. evaluability
6. richness
7. optimizability
Diversity

want many hypotheses that disagree on a large fraction of examples

boosting reweights examples so that previous weak learner has error $\frac{1}{2}$. The next selected weak learner to have high disagreement.

Coverage

Have the entire space “covered”.

E.g. look at all features.
Simplicity
Want $|H|$ as small as possible
  Occam’s razor bounds.

Error
Not too big and not too small!
  Too large – weak learning guarantee violated.
  Too small – can’t boost!

Evaluability
Hypotheses need to be efficiently evaluable
  Otherwise taking final vote will be intractable.
**Richness**

A linear combination of hypothesis from the weak learner’s class must be able to represent a large class of functions.

To have a higher chance of approximating the target.

**Optimizability**

Finding an approximate ERM over the weak learners should be tractable.

Otherwise, finding a hypothesis with sufficiently small error will be too difficult.
Trees and Stumps

• Decision Stumps
  diversity, coverage, simplicity, error, evaluability, richness, optimizability

• Decision Trees
  diversity, coverage, simplicity, error, evaluability, richness, optimizability
Parity Functions

parity functions: \( \chi_S(x) = (-1)^{x \cdot S} \)
- \( S \) gives relevant attributes of \( x \) (both in \( \{0,1\}^n \))
- \( ||S||_1 \) is the degree of \( S \) (stumps are degree 1 parities)

Fact from discrete Fourier analysis: any Boolean fn can be written as a linear combination of parities.

\[ f(x) = \sum_{S \in \{0,1\}^n} \hat{f}_S \chi_S(x) \]
- the \( \chi_S \) are called “characters” in Fourier analysis
- the \( \hat{f}_S \) are the Fourier coefficients
Parities as Weak Learners

**Main Idea**: use parities as weak learners

- Note: others, eg [Jackson ’97], have combined parities/boosting for theoretical results.
- Also, using *all* parities won’t work
  - for many reasons
- So, we propose using d-parities (for constant d)
  - If $||S||_1 < d$, we call S a d-parity
- d-parities can represent “low degree” functions, which capture e.g. linear functions.
Trees, Stumps, and Sparse Parities

• Decision Stumps
  diversity, coverage, simplicity, error, evaluability, richness, optimizability

• Decision Trees
  diversity, coverage, simplicity, error, evaluability, richness, optimizability

• Sparse Parities (e.g. degree 2 or 3 or 4.)
  diversity, coverage, simplicity, error, evaluability, richness, optimizability(?)
Optimizing Over r-Parities

• A brute force approach takes $m(n^d)$ time for m examples on n features.

• Recent advances [Grigorescu-R-Vempala ’11] and [Valiant ’12] reduce this to $\sim m(n^{d/2})$.
  – E.g. 3- or 4-parities now become tractable
### How Well Do Sparse Parities Work?

<table>
<thead>
<tr>
<th></th>
<th>decision stumps</th>
<th>3-parities</th>
<th>CART-16 trees</th>
</tr>
</thead>
<tbody>
<tr>
<td>oct17</td>
<td>1.09</td>
<td>0.62</td>
<td>1.11</td>
</tr>
<tr>
<td>ocr49</td>
<td>6.08</td>
<td>2.77</td>
<td>2.16</td>
</tr>
<tr>
<td>splice</td>
<td>7.37</td>
<td>4.99</td>
<td>3.18</td>
</tr>
<tr>
<td>census</td>
<td><strong>18.50</strong></td>
<td>22.00</td>
<td>22.00</td>
</tr>
<tr>
<td>cancer</td>
<td>4.45</td>
<td>3.64</td>
<td><strong>3.19</strong></td>
</tr>
<tr>
<td>ecoli</td>
<td>7.06</td>
<td><strong>5.87</strong></td>
<td>8.98</td>
</tr>
<tr>
<td>heart</td>
<td><strong>22.24</strong></td>
<td>22.94</td>
<td>24.06</td>
</tr>
</tbody>
</table>

Table 3: Error rates of decision stumps, 3-parities, and CART-16 trees used as weak learners for AdaBoost run for 250 rounds, averaged over 20 trials.
Summary

• Proposed seriously considering the problem of weak learner design.

• Gave some informal properties of a good weak learner.

• Showed that sparse parities somewhat satisfied these properties.

• Experiments indicate these are competitive with some of the best weak learners used in practice.
Open Problems

Formalize the theory.

Extend to multiclass prediction.

Find better weak learners!