Real-World applications of Boosting

Yoav Freund
UCSD
Practical Advantages of AdaBoost

• fast
• simple and easy to program
• no parameters to tune (except $T$)
• flexible — can combine with any learning algorithm
• no prior knowledge needed about weak learner
• provably effective, provided can consistently find rough rules of thumb
  → shift in mind set — goal now is merely to find classifiers barely better than random guessing
• versatile
  • can use with data that is textual, numeric, discrete, etc.
  • has been extended to learning problems well beyond binary classification
Caveats

- performance of AdaBoost depends on data and weak learner
- consistent with theory, AdaBoost can fail if
  - weak classifiers too complex
    → overfitting
  - weak classifiers too weak ($\gamma_t \to 0$ too quickly)
    → underfitting
    → low margins → overfitting
- empirically, AdaBoost seems especially susceptible to uniform noise
UCI Experiments

• tested AdaBoost on UCI benchmarks
• used:
  • C4.5 (Quinlan’s decision tree algorithm)
  • “decision stumps”: very simple rules of thumb that test on single attributes

- eye color = brown ?
  - yes: predict +1
  - no: predict -1

- height > 5 feet ?
  - yes: predict -1
  - no: predict +1
UCI Results

boosting Stumps

boosting C4.5
Boosting Stumps
(for text classification)

• “AT&T, How may I help you?”
• Classify voice requests
• Voice -> text -> category
• Fourteen categories

  Area code, AT&T service, billing credit, calling card, collect, competitor, dial assistance, directory, how to dial, person to person, rate, third party, time charge, time
Examples

- Yes I’d like to place a collect call long distance please
- Operator I need to make a call but I need to bill it to my office
- Yes I’d like to place a call on my master card please
- I just called a number in Sioux city and I musta rang the wrong number because I got the wrong party and I would like to have that taken off my bill

- collect
- third party
- calling card
- billing credit
Weak rules generated by “boostexter”

1. collect
2. card
3. my home

Category

Word occurs

Calling card
Collect call
Third party

Word does not occur
Results

- 7844 training examples
  - hand transcribed
- 1000 test examples
  - hand / machine transcribed
- Accuracy with 20% rejected
  - Machine transcribed: 75%
  - Hand transcribed: 90%
Viola and Jones face detector
Face Detection / Viola and Jones

- Struggle to get paper accepted
- Live Demo - detect faces of people in audience.
- Now standard feature in many cameras.
Face Detection as a Filtering process

50,000 Locations/Scales

Most Negative
Classifier is Learned from Labeled Data

- 5000 faces, $10^8$ non faces
- Faces are normalized
  - Scale, translation
  - Rotation remains…
Image Features

“Rectangle filters”

Similar to Haar wavelets
Papageorgiou, et al.

\[ h_t(x_i) = \begin{cases} 
1 & \text{if } f_t(x_i) > \theta_t \\
0 & \text{otherwise}
\end{cases} \]

Very fast to compute using “integral image”.

\[ 60,000 \times 100 = 6,000,000 \]

Unique Features

Combined using adaboost
• A classifier with 200 rectangle features was learned using AdaBoost
• 95% correct detection on test set with 1 in 14084 false positives.
• To be competitive, needs ~6,000 features
• But that makes detector prohibitively slow.
• Learning is always slow, but done only once.
Employing a cascade to minimize average feature computation time

The accurate detector combines 6000 simple features using Adaboost.

In most boxes, only 8-9 features are calculated.
Co-Training
Using confidence to avoid labeling

Levin, Viola, Freund 2003
Image 1
Image 1 - diff from time average
Image 2 - diff from time average
Co-training

Blum and Mitchell 98
Grey-scale detection score

Subtract-average detection score

Cars

Non cars
Co-Training Results

Raw Image detector

Before co-training

Difference Image detector

After co-training
Alternating Decision Trees

With Llew Mason
Decision Trees

X > 3

Y > 5

no

no

yes

yes

-1

-1

+1

+1

-1

-1

3

5
Decision tree as a sum

\[ \text{sign} \]

- \( X > 3 \)
  - no: -0.1
  - yes: +0.1

- \( Y > 5 \)
  - no: -0.3
  - yes: +0.2

\[ X \]
\[ Y \]
An alternating decision tree

- Y < 1
  - no
  - yes
    - 0.0
    - +0.7
- X > 3
  - no
  - yes
    - -0.1
    - +0.1
- Y > 5
  - no
  - yes
    - -0.3
    - +0.2

Sign

Y

- 0.2
- 0.1
+0.1
+0.0
+0.7
-0.1
-0.3
+0.2

X

+1
-1
+1
-1
+1
Example: Medical Diagnostics

• **Cleve** dataset from UC Irvine database.
• Heart disease diagnostics (+1=healthy, -1=sick)
• 13 features from tests (real valued and discrete).
• 303 instances.
## Cross-validated accuracy

<table>
<thead>
<tr>
<th>Learning algorithm</th>
<th>Number of splits</th>
<th>Average test error</th>
<th>Test error variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADtree</td>
<td>6</td>
<td>17.0%</td>
<td>0.6%</td>
</tr>
<tr>
<td>C5.0</td>
<td>27</td>
<td>27.2%</td>
<td>0.5%</td>
</tr>
<tr>
<td>C5.0 + boosting</td>
<td>446</td>
<td>20.2%</td>
<td>0.5%</td>
</tr>
<tr>
<td>Boost Stumps</td>
<td>16</td>
<td>16.5%</td>
<td>0.8%</td>
</tr>
</tbody>
</table>
Adtree for Cleveland heart-disease diagnostics problem

1: thal = normal
   y: 0.541
   n: -0.626

2: number-vessels-colored = 0
   y: 0.425
   n: -0.731

3: chest-pain-type is asymptomatic
   n: 0.441
   y: -0.536

4: oldpeak < 2.45
   y: 0.138
   n: -1.495

5: cholesterol < 240.5
   y: 0.508
   n: -0.444

6: sex = female
   y: 1.057
   n: -0.167
Call Detail analysis (AT&T)

• Distinguish business/residence customers
• Using statistics from call-detail records
• Label unknown for ~30%
Massive datasets

- 260M calls / day
- 230M telephone numbers
- **Hancock**: software for computing statistical signatures (today we might have used Hadoop)
- 100K randomly selected training examples,
- ~10K is enough
- Training takes about 2 hours.
- Generated classifier has to be both **accurate** and **efficient**
Alternating tree for “buizocity”
Alternating Tree (Detail)

Positive predictions ⇔ Residences
Negative predictions ⇔ Businesses

WKD_IN_tod_18-0 > 14.5

weekday < 1.345

WKD_OUT_tod_18-0 > 13.5

Total < 0.1745
Precision/recall graphs

Score

Accuracy
JBoost
Installation

- Go to jboost.sourceforge.net
- Download and unzip jboost-x.x (current latest 2.3)
- Move jboost-x.x directory to a good place in your directory structure
- open a terminal and cd to the jboost-x.x directory.
Required software packages

• Needed packages:
  • java (version 1.6 works) - Base language
  • python (version 2.7.2 works) - Scripting Language
  • jboost (Latest version is 2.3)
  • GraphViz - node-edge graph visualization (2.28 works)
  • gnuplot - X-Y graph visualization (4.2 works)
  • Cygwin - a unix-like shell for Windows.
$ scripts/checkVersions.sh

---------- java
java version "1.6.0_33"
Java(TM) SE Runtime Environment (build 1.6.0_33-b03-424-10M3720)
Java HotSpot(TM) 64-Bit Server VM (build 20.8-b03-424, mixed mode)

---------- python
Python 2.7.2

---------- gnuplot
gnuplot 4.2 patchlevel 5

---------- graphviz
dot - graphviz version 2.28.0 (20110509.1545)
Quick Start

• After installation and checking versions perform:
  • source setPath.sh
  • scripts/runScripts.sh
The Seville project

- Pedestrian Alert System
- Camera mounted on front of car.
- Funded by Renault
- Collaboration with Yotam Abramson (Then at Ecole Des Mines, Paris).
Pedestrian detection - typical segment
The training process

1500 pedestrians
Collected 6 Hrs of video  ->  540,000 frames
170,000 boxes per frame

20 seconds for marking a box around a pedestrian.

3 seconds for deciding if box is pedestrian or not.

How to choose “hard” negative examples?
### Summary of Active Training

Only examples whose normalized score is in this range are hand-labeled

<table>
<thead>
<tr>
<th>Step</th>
<th>total candidates</th>
<th>$\mu^-$</th>
<th>$\mu^+$</th>
<th>presented</th>
<th>labeled</th>
<th>human labor</th>
<th>positive</th>
<th>negative</th>
<th>training time</th>
<th>Weak rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>510 K</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>16</td>
<td>3m</td>
<td>6</td>
<td>10</td>
<td>2s</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>680 K</td>
<td>0</td>
<td>1</td>
<td>364</td>
<td>403</td>
<td>3m</td>
<td>36</td>
<td>374</td>
<td>6s</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>3,400 K</td>
<td>0.6</td>
<td>1</td>
<td>153</td>
<td>156</td>
<td>4m</td>
<td>46</td>
<td>520</td>
<td>22s</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>66,470 K</td>
<td>0.4</td>
<td>1</td>
<td>805</td>
<td>852</td>
<td>10m</td>
<td>86</td>
<td>1332</td>
<td>1m30s</td>
<td>30</td>
</tr>
<tr>
<td>5</td>
<td>37,910 K</td>
<td>0.1</td>
<td>0.8</td>
<td>1350</td>
<td>1439</td>
<td>10m</td>
<td>182</td>
<td>2675</td>
<td>8m</td>
<td>59</td>
</tr>
<tr>
<td>6</td>
<td>116,960 K</td>
<td>0</td>
<td>0.6</td>
<td>5150</td>
<td>5364</td>
<td>1h30m</td>
<td>417</td>
<td>7804</td>
<td>1h10m</td>
<td>270</td>
</tr>
<tr>
<td>7</td>
<td>24,140 K</td>
<td>-0.02</td>
<td>0.5</td>
<td>1320</td>
<td>863</td>
<td>3h</td>
<td>848</td>
<td>8236</td>
<td>7h30m</td>
<td>893</td>
</tr>
<tr>
<td>8</td>
<td>189,550 K</td>
<td>-0.02</td>
<td>0.5</td>
<td>8690</td>
<td>8707</td>
<td>3h</td>
<td>1178</td>
<td>16613</td>
<td>17h</td>
<td>1500</td>
</tr>
<tr>
<td>9</td>
<td>209,610 K</td>
<td>-0.02</td>
<td>0.5</td>
<td>2933</td>
<td>2933</td>
<td>3h</td>
<td>1486</td>
<td>19238</td>
<td>30h</td>
<td>2034</td>
</tr>
<tr>
<td>10</td>
<td>274,210 K</td>
<td>-0.02</td>
<td>0.5</td>
<td>3861</td>
<td>3861</td>
<td>4h</td>
<td>2046</td>
<td>22533</td>
<td>30h</td>
<td>3150</td>
</tr>
</tbody>
</table>
Easy examples

Positive

Negative
Harder examples

Positive

Negative
very hard examples

Iteration

Positive

Negative
And the figure in the gown is ...
Detection Accuracy
Current best results
Genome-Wide Association Studies
Genetic Disorders

• The influence of heredity on disease.
• Mendalian Diseases: Influenced by a single gene:
  • Sickle-cell Anemia - two copies of a single recessive gene.
  • One copy increases resistance to Malaria.
• Non Mendalian diseases are influenced by many genes.
GWAS, the idea

- According to longitudinal studies many common diseases have a significant heritable component.
  - High Blood Pressure, Diabetes, Cron Disease, Otism ...
- Can we find which genes are the culprits?
- Genome Wide Association Studies: sequence ~500,000 DNA locations (SNPs) on patients (and controls)
- Use statistical methods to find associations (correlations) between DNA location and disease.
GWAS, current status

- Several large datasets (5,000 - 10,000) published (but getting access is not trivial)
- Association studies find a few SNPs with statistically significant correlation. But,
- The percentage of variance explained is usually low (1% - 5%)
  - Especially glaring for universal traits such as height.
Machine learning to the rescue!

- Instead of finding correlations between disease and single SNPs, learn a function that maps the SNP vector to the disease.
- Find the set of SNPs on which the function depends.
- Good idea, people did it using SVM, random forests, ...
- Good test set performance
  - **BUT:** the geneticists are not convinced.
- Predictability does not imply causality.
- What is the p-value?
Boost-Remove

- We have 500,000 features (SNPs)
- Run Boosting for $k$ (50) iterations.
- Remove the SNPs used.
- Consider all of $n \times k$ SNPs
Why is it hard to interpret?

• **Linkage Disequilibrium:** dependencies between SNPs:
  • Location Linkage: recombination rate depends on distance between SNPs.
  • Population Stratification: groups of related people (ethnicities)
  • Selection: Fitness depends on combination of SNP states.
  • Different mutation rates, selective mating ...

• Result: many non-causal correlations.

• Which correlations are causal?
Results on two datasets

WT consortium: 2000 cases, 3000 controls

GC consortium: 4061 cases and 2571 controls
Measuring closeness of location
Location Consistency

Mann-Whitney U test yields $p=10^{-30}$
related SNPs

Tree structure of ADT hints at relations btwn SNPs
The protein crystallization problem

• ~1,000,000 protein sequences extracted from DNA.

• ~10,000 have known 3D structure.

• Best method: X-ray crystallography.

• Requires protein crystals (coherent lattice).

• Crystallizing proteins: a black art with very small yield.
The post-doc method

- Assign protein to post-doc.
- If post-doc crystallizes protein: s/he publishes a paper - can advance to next stage of academic career.
- This is currently the most cost effective method.
“high throughput” method

• Use robots to create hundreds of droplets of solutions of protein and salts in different concentrations.

• Take image of each droplet.

• **Identify droplets that contain micro-crystals.**

• Harvest micro-crystals, X-ray, analysis ....
Problems with high-throughput

- Yield is very low and varies from protein to protein. Most droplets create “percpitants” rather than crystals.
- Detecting and harvesting the micro-crystals requires human expertise.
- The backlog of images to be analyzed is ~ two weeks long. By which time, the crystal often dissolves back into the solution...
Detecting micro-cristals
Detecting micro-crystals
Detecting micro-crystals
Detecting micro-crystals
Detecting micro-crystals
C-Elegans image analysis for high-throughput screening

- microscopic worm is a very popular model organism in biology.
- Used in drug development. Potential for high throughput screening - testing thousands of compounds.
- Worms are bred in pleasant medium of agar. (Pleasant for worms not for image analysis.)
- Worms are imaged under normal light and fluorescent light.
- Collaboration with Anne Carpenter (Broad institute) and Annie Lee Connery (MGH, Ruvkun Lab and Ausubel Lab).
Results

• Four 96-well plates
• Known Phenotype in each well.
• Half of the wells used for training, half for testing (phenotype is hidden).
• 2 Experimentalists – post-docs that are running the experiments.

<table>
<thead>
<tr>
<th>Type</th>
<th>Plate number</th>
<th>Experimentalist 1 Error (%)</th>
<th>Experimentalist 2 Error (%)</th>
<th>Automated method’s error - without abstention (%)</th>
<th>Automated method’s error – with abstension (%)</th>
<th>Examples on which we abstain (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>wild type vs lNR</td>
<td>1</td>
<td>1.6</td>
<td>0.0</td>
<td>0.4</td>
<td>0.0</td>
<td>5.4</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.0</td>
<td>0.0</td>
<td>0.1</td>
<td>0.0</td>
<td>2.1</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.0</td>
<td>0.0</td>
<td>1.4</td>
<td>0.0</td>
<td>8.3</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.3</td>
</tr>
<tr>
<td>wild type vs hNR</td>
<td>1</td>
<td>1.1</td>
<td>8.9</td>
<td>4.8</td>
<td>0.4</td>
<td>29.0</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.0</td>
<td>2.6</td>
<td>6.6</td>
<td>1.2</td>
<td>29.5</td>
</tr>
</tbody>
</table>
The image processing work-flow

- Worm Detector
- Fluorescent Detector

Phenotype Classifier
- INR
- wild type
- hNR
Basic blocks for worms

• For learning, use simple yet characteristic block.
• For worms, we use worm segments.
• A worm segment is represented by the center line.
• When properly identified, worm segments would give us the direction and size.
Aim of learning

• Classify correct segments from incorrect ones.
• Correct segments are perpendicular to the median line with ends on the worm boundary.
• Any other segment is negative.
User input

- User draws the outline of worms and the median line.
- We find the segments perpendicular to the median line that end at the worm boundaries.
- These segments are treated as positive.
- Random segments are used as negative.
Features for Classification

- Properties of different regions are used as features.
- Typically, green regions would be lighter for worms, blue will be darker and have texture, red would have edges.
- Many filters are applied to the image.
- Filter responses within the boxes are used as features.
Feature finding
Input bright-field
Filtered Images: Laplacian of Gaussian (l)
Filtered Images:
Laplacian of Gaussian (II)
Worm Detection: initial training set
Worm Detection - 2 feedback iterations
Iteration 1
Iteration 2

ROC

Histogram
Iteration 10
Iteration 20
Iteration 100
scores after retraining
Online Boosting and Tracking
Online Boosting

- Large data stream.
- Distribution of data changes over time.
- Partition stream into batches
  - Re-weight examples in batch using current strong learner.
  - Learn a new weak learner.
  - Remove oldest weak learner.

[Oza & Russel 2001]
Tracking using online boosting

- **Detect**: Find tile that best fits
  1. Appearance model of tracked object.
  2. Constraints on movement.

- **Label**: Use detected tile as positive, far tiles as negative.

- **Learn**: Update model using online boosting.

[Grabner, Grabner & Bischof 2006]
Tracking David

[Stalder & Grabner 2009]
Tracking under Partial Occlusion

[Stalder & Grabner 2009]
Tricking the online tracker

[Stalder & Grabner 2009]
Define bounding box and double click inside.

TLD: Track, Learn, Detect