XGBOOST: A SCALABLE TREE BOOSTING SYSTEM

(T. CHEN, C. GUESTRIN, 2016)

NATALLIE BAIKEVICH

HARDWARE ACCELERATION FOR DATA PROCESSING SEMINAR
ETH ZÜRICH
MOTIVATION

- Effective statistical models
- Scalable system
- Successful real-world applications

XGBoost

eXtreme Gradient Boosting
BIAS-VARIANCE TRADEOFF

Random Forest

Variance ↓

Boosting

Bias ↓

Voting
A BIT OF HISTORY

AdaBoost, 1996
Random Forests, 1999
Gradient Boosting Machine, 2001
A BIT OF HISTORY

AdaBoost, 1996
Random Forests, 1999
Gradient Boosting Machine, 2001
Various improvements in tree boosting
XGBoost package
A BIT OF HISTORY

AdaBoost, 1996
Random Forests, 1999
Gradient Boosting Machine, 2001
Various improvements in tree boosting
XGBoost package

1st Kaggle success: Higgs Boson Challenge

17/29 winning solutions in 2015
WHY DOES XGBOOST WIN "EVERY" MACHINE LEARNING COMPETITION?

- (MASTER THESIS, D. NIELSEN, 2016)

- Maksims Volkovs, Guangwei Yu and Tomi Poutanen, 1st place of the 2017 ACM RecSys challenge. Link to paper.
- Vlad Sandulescu, Mihai Chiru, 1st place of the KDD Cup 2016 competition. Link to the arxiv paper.
- Marios Michailidis, Mathias Müller and HJ van Veen, 1st place of the Dato Truely Native? competition. Link to the Kaggle interview.
- Vlad Mironov, Alexander Guschin, 1st place of the CERN LHCb experiment Flavour of Physics competition. Link to the Kaggle interview.
- Josef Slavicek, 3rd place of the CERN LHCb experiment Flavour of Physics competition. Link to the Kaggle interview.
- Mario Filho, Josef Feigl, Lucas, Gilberto, 1st place of the Caterpillar Tube Pricing competition. Link to the Kaggle interview.
- Qingchen Wang, 1st place of the Liberty Mutual Property Inspection. Link to the Kaggle interview.
- Chenglong Chen, 1st place of the Crowdflower Search Results Relevance. Link to the winning solution.
- Alexandre Barachant ("Cat") and Rafał Cycoń ("Dog"), 1st place of the Grasp-and-Lift EEG Detection. Link to the Kaggle interview.
- Halla Yang, 2nd place of the Recruit Coupon Purchase Prediction Challenge. Link to the Kaggle interview.
- Owen Zhang, 1st place of the Avito Context Ad Clicks competition. Link to the Kaggle interview.
- Keiichi Kuroyanagi, 2nd place of the Airbnb New User Bookings. Link to the Kaggle interview.
- Marios Michailidis, Mathias Müller and Ning Situ, 1st place Homesite Quote Conversion. Link to the Kaggle interview.

Source: https://github.com/dmlc/xgboost/tree/master/demo#machine-learning-challenge-winning-solutions
TREE ENSEMBLE

**tree1**

- **age < 15**
  - Y
    - **is male?**
      - Y
        - +2
      - N
        - +0.1
    - N
      - -1

**tree2**

- **Use Computer Daily**
  - Y
    - +0.9
  - N
    - -0.9

\[
f(\text{boy}) = 2 + 0.9 = 2.9 \\
f(\text{old man}) = -1 - 0.9 = -1.9
\]
**REGULARIZED LEARNING OBJECTIVE**

\[
L = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k)
\]

\[
\hat{y}_i = \sum_{k=1}^{K} f_k(x_i)
\]

\[
\Omega(f) = \gamma T + \frac{1}{2} \lambda \|w\|^2
\]

SCORE CALCULATION

1st order gradient

2nd order gradient

Statistics for each leaf

Score

The smaller the score is, the better the structure is.

Instance index  gradient statistics

1  
2  
3  
4  
5  

age < 15

is male?

Y  N

Y  N

I_1 = \{1\}  
I_2 = \{4\}  
I_3 = \{2, 3, 5\}  
G_1 = g_1  
G_2 = g_4  
G_3 = g_2 + g_3 + g_5  
H_1 = h_1  
H_4 = h_4  
H_3 = h_2 + h_3 + h_5  

Obj = - \sum_j \frac{G_j^2}{H_j + \lambda} + 3\gamma
ALGORITHM FEATURES

✓ Regularized objective
✓ Shrinkage and column subsampling
✓ Split finding: exact & approximate, global & local
✓ Weighted quantile sketch
✓ Sparsity-awareness
SYSTEM DESIGN: BLOCK STRUCTURE

Blocks can be
✓ Distributed across machines
✓ Stored on disk in out-of-core setting

Max depth

\[ O(Kd \|x\|_0 \log n) \]

\[ \rightarrow \]

\[ O(Kd \|x\|_0 + \|x\|_0 \log B) \]

Sorted structure → linear scan

# trees

# non-missing entries
SYSTEM DESIGN: CACHE-AWARE ACCESS

- Improved split finding
- Non-continuous memory access
- Allocate internal buffer
- Prefetch gradient statistics

Datasets: Larger vs Smaller

(b) Higgs 10M

(d) Higgs 1M
SYSTEM DESIGN: BLOCK STRUCTURE

Prefetch in independent thread

Compression by columns (CSC):

Decompression vs Disk Reading

Block sharding: Use multiple disks

Too large blocks, cache misses

Too small, inefficient parallelization
EVALUATION

AWS c3.8xlarge machine: 32 virtual cores, 2x320GB SSD, 60 GB RAM

32 m3.2xlarge machines, each: 8 virtual cores, 2x80GB SSD, 30GB RAM
<table>
<thead>
<tr>
<th>Dataset</th>
<th>n</th>
<th>m</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allstate</td>
<td>10M</td>
<td>4227</td>
<td>Insurance claim classification</td>
</tr>
<tr>
<td>Higgs Boson</td>
<td>10M</td>
<td>28</td>
<td>Event classification</td>
</tr>
<tr>
<td>Yahoo LTRC</td>
<td>473K</td>
<td>700</td>
<td>Learning to rank</td>
</tr>
<tr>
<td>Criteo</td>
<td>1.7B</td>
<td>67</td>
<td>Click through rate prediction</td>
</tr>
<tr>
<td>WHAT'S NEXT?</td>
<td>Tuning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>--------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XGBoost</td>
<td>Hyperparameter optimization</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scalability</td>
<td>Parallel Processing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weighted quantiles</td>
<td>GPU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sparsity-awareness</td>
<td>FPGA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cache-awareness</td>
<td>Model Extensions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data compression</td>
<td>DART (+ Dropouts)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>LinXGBoost</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>More Applications</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
QUICK OVERVIEW

+ Nicely structured paper, easily comprehensible
+ Real framework, widely used for many ML problems
+ Combination of improvements both on model and implementation sides to achieve scalability
+ Reference point for further research in tree boosting,

- The concepts are not that novel themselves
- Does not explain why some of the models are not compared in all experiments
- Is the compression efficient for dense datasets?
- What if there’s a lot of columns rather than rows (e.g. medical data)?
THANK YOU!