"Divide and Conquer"
Machine Learning
to
Exploit
Big Data
Knowledge Discovery

Una-May O’Reilly
The Alfa Group

MIT ICT Conference
April 24, 2013
Lots of Data Everywhere

Knowledge Mining Opportunities
Help
Agenda

For each of SCALE, FlexGP and EC-Star

• **System Layer:**
  – Resource and Task Management/Mapping
  – Task = execute a ML algorithm on a distributed system

• **ML Algorithm Layer**
  – Algorithm scaling:
  – Divide and Conquer data strategy
    » Factor
    » Filter,
    » Fuse Look at all the data everywhere
  – Experiments related to divide and conquer with the data

Endfor

• **Compare and contrast**
• **Going Forward**
SCALE Introduction
SCALE

Divide and Conquer
SCALE Server-Client Architecture

Resource and System Management Layer
SCALE Results

- 320 hours
- ~60 nodes
- 32,256 tasks
- 1200 classifiers

<table>
<thead>
<tr>
<th>Classifier Type</th>
<th>#Classifiers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discrete Bayesian Network</td>
<td>355</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>455</td>
</tr>
<tr>
<td>Naive Bayes</td>
<td>390</td>
</tr>
</tbody>
</table>
SCALE

Divide and Conquer
Scaling Up to 1000s: from SCALE to FlexGP

**SCALE:**
- Every learner has to know IP of task handler
- Task handler is a bottleneck and central point of failure
- No communication to accelerate learning
- Inelastic

**FlexGP**
- No central task handler or point of failure
- Learners gossip to learn each others IP
- Elastic
Scaling up to 1000’s: from SCALE to FlexGP

**SCALE**
- Modest # of features
- Data must fit into RAM
- Explicit algorithm tasks
- Small scale, serial algorithms
- No learner communication

**FlexGP**
- Factor
  - 100s of features-
- Big Data
- Statistically directs algorithm islands
- Large scale, distributed algorithm
  - Local algorithms coordinated
- Learners share and integrate progress
FlexGP

Introduction
FlexGP

Goal: Model $y = f(x_1, x_2, \ldots, x_n)$
FlexGP: launch and IP discovery

N, k

128.21.32.237

Launch
FlexGP: launch and IP discovery

- Launch

128.21.32.237
\(\mathcal{N}, k\) [IP-list]

128.21.32.238

128.21.32.239
\(\mathcal{N}, k\) [IP-list]
FlexGP: launch and IP discovery
FlexGP: launch and IP discovery

FlexGP Launch

π, D

Launch

Gossip

128.21.32.238
128.21.32.239
128.21.32.113
128.21.32.506
128.21.32.230
128.21.32.734

N, k [IP-list]
N, k [IP-list]
N, k [IP-list]
N, k [IP-list]
FlexGP: launch and IP discovery

FlexGP Launch
FlexGP: launch and IP discovery
FlexGP: launch and IP discovery

FlexGP Launch
FlexGP: launch and IP discovery

![Diagram showing FlexGP launch and IP discovery]

**Launch**  
**Gossip**  
$L(\pi, D)$
FlexGP: launch and IP discovery

128.21.32.123
128.21.31.512
128.21.31.542
128.21.31.6
128.21.31.332
128.21.31.832
128.21.41.832

Launch
Gossip

FlexGP Launch
FlexGP: launch and IP discovery

FlexGP Launch

FlexGP: launch and IP discovery
FlexGP: launch and IP discovery
Launch complete!

... and ready to expand or contract
.... and gossiping intermittently
Statistically Parameterized Factoring

\( \Pi \): Probability of feature, objective function, operator

\( \hat{\mathcal{D}} \): factoring of the data

FlexGP Launch
Divide and Conquer

- **Factor:**
  - Random subsets using statistical distribution
    - Demonstrate independent learners -> weak learners
  - Features

- **Filter:**
  - uncorrelated and accurate models

- **Fuse**
  - How far can we pare down the dataset?
    - Effectiveness of divide and conquer
      - Show results and conclude whether it is sensitive to dataset?
  - harvest "best to date" results from the system as it continues to work away
FlexGP Filter and Fusion

$\pi, D$

$F_{\pi, D}$

$\pi, D$

$F_{\pi, D}$

$\pi, D$

$F_{\pi, D}$

$\pi, D$

$\pi, D$

$F_{\pi, D}$

$\pi, D$

$F_{\pi, D}$

$x_1 \sin(x_5) + x_2 \sqrt{x}$

$\cos(x_4)/\sin(x_2) + \sqrt{x_3 - x_4}$

$\ln(x_1)/\exp(x_3) + x_5 x_3 + \frac{x_2}{x_3}$

$\frac{\cos(x_4)}{\tan(x_2) + x_2} + \sqrt{x_3}$

Filter to select diverse models

$x_1 \sin(x_5) + x_2 \sqrt{x}$

$\frac{x_1}{\exp(x_2)} + x_5 x_3 + \frac{x_2}{x_3}$

$\frac{\cos(x_4)}{\tan(x_2) + x_2} + \sqrt{x_3}$

Fusion to derive an ensemble prediction

FlexGP Overview
Divide and Conquer

- **Factor:** Random subsets using statistical distribution
  - Demonstrate independent learners -> weak learners

- **Filter:**
  - uncorrelated and accurate models

- **Fuse**
  - Which fusion algorithm is best?
  - Is fusion better than best?
  - How far can we pare down the dataset?
    - Effectiveness of divide and conquer
      - Show results and conclude whether it is sensitive to dataset?
  - Harvest "best to date" results from the system as it continues to work away
FlexGP Fusion: Better than Best?

Figure 1: The quartile distribution of $\text{PG}_{\text{MSE}}$ of models used for fusion in each experiment. The circles represent the best $\text{PG}_{\text{MSE}}$ from fusion. Left Results for NOx experiments; KDE was the best fusion method. Right Results for MSD experiments; ARM was the best fusion method.
FlexGP: Best So Far Fusion

$\mathbf{P}_{\text{mse}}$ v.s. Time for NOx1

![Graph showing $\mathbf{P}_{\text{mse}}$ v.s. Time for NOx1 with different lines for BAM, AVE, MAD, ARM, and KDE.]
ECStar

- Goal: compute very cost effectively on *VAST* number of nodes
  - Runs on thousand to 10’Ks 100K’s million nodes
  - Vast requires cost effective -> volunteer
- Domain: learn from time series
  - Finance, medical signals domain
- Solution is strategy or classifier expressed as rule sets
From FlexGP to ECStar

- Clear separation between system layer vs algorithm layer
- The volunteer compute nodes of EC-Star change that picture
  - They have unpredictable availability – when they start and stop
  - A client’s host can fail
  - Host imposes
    » Small memory footprint,
    » need to save and migrate state
    » client-to-client communication ban
    » Design decisions negotiating responsibilities between VCN and dedicated servers
    » Result is a distributed algorithm with divide and conquer strategy for data handling that is
      - tightly integrated with the resource layer design
EC-Star Divide and Conquer
Resources Federation
Under to Over Sampling

EC-Star Divide and Conquer
Blood Pressure Problem

Experimental Results
Impact of Partially Evaluating Models

Experimental Results
# System Layer Comparison

<table>
<thead>
<tr>
<th></th>
<th>Scale</th>
<th>FlexGP</th>
<th>EC-Star</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ML domain</strong></td>
<td>Classification</td>
<td>Regression Classification</td>
<td>Rule Learning</td>
</tr>
<tr>
<td><strong>Resource Scale</strong></td>
<td>10's to 100</td>
<td>100’s to 1000</td>
<td>10^3 to 10^6</td>
</tr>
<tr>
<td><strong>Resource Type</strong></td>
<td>Cloud</td>
<td>Cloud</td>
<td>Volunteer and Dedicated</td>
</tr>
<tr>
<td><strong>Fusion</strong></td>
<td>External</td>
<td>External</td>
<td>Integrated</td>
</tr>
<tr>
<td><strong>Local Algorithm</strong></td>
<td>Different</td>
<td>Same</td>
<td>Same</td>
</tr>
<tr>
<td><strong>Server:Client ratio</strong></td>
<td>1: many</td>
<td>Decentralized</td>
<td>many:many</td>
</tr>
<tr>
<td></td>
<td>SCALE</td>
<td>FlexGP</td>
<td>EC-Star</td>
</tr>
<tr>
<td>-------</td>
<td>---------</td>
<td>---------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>Factor</td>
<td>Subsets</td>
<td>Subsets</td>
<td>Under to oversampling</td>
</tr>
<tr>
<td>Filter</td>
<td>Correlation Accuracy</td>
<td>Correlation Accuracy</td>
<td>Layered competition</td>
</tr>
<tr>
<td>Fuse</td>
<td>Voting</td>
<td>Non-parametric output space approaches</td>
<td>Migration and ancestral properties</td>
</tr>
</tbody>
</table>
Automation

• "In the end, the biggest bottleneck is not data or CPU cycles, but human cycles."

Looking Forward
ML requires a lot of Human Effort

- Domain Knowledge Analysis and Transfer
- Problem Definition
- Data Preconditioning
- Feature Identification
- Feature Extraction
- Algorithm Selection
- Algorithm Customization
- Parameter Selection
- Training and Test Data Selection
- Results Evaluation
- Solution Deployment

Looking Forward
Thanks for your attention!

Thanks to...

- **ALFA group members**
  - Large team of students
  - Postdoc: Dr. Erik Hemberg
  - Research Scientist: Dr. Kalyan Veeramachaneni

- **Our collaborators and sponsors**