Concept Learning and Searching Over Networks
Using Java Agents for Meta-learning

THE JAM PROJECT

Application: FRAUD AND INTRUSION DETECTION
IN FINANCIAL INFORMATION SYSTEMS

CMAD IV

Salvatore J. Stolfo
Department of Computer Science
Columbia University
Electronic Commerce on the WEB provides New Challenges

- More data and services are available everyday on the WEB
- We seek a new way to search and LEARN FROM very large and remote databases
- Electronic Commerce provides new opportunities for Electronic FRAUD
- We seek a new way to LEARN about FRAUD on the WEB
- Proposal: Build an IMMUNOLOGICAL Capability for the WEB to DETECT FRAUD
- Learn SELF (Good Transactions) from NON-SELF (Bad Transactions)
A New Information Extraction Paradigm

- Empower the User with *Data Mining* Tools to Learn Knowledge from Data
- Agent Proxies that Learn Knowledge over Remote Data
- Agent Proxies that Learn Collective Knowledge over Remote Agents
- Agent Proxies Use Learned Knowledge to Search Other Data
Terminology

- Data Mining: Scalable Machine Learning Applied to Very Large Databases
- Learning Agent: A Machine Learning program launched to and applied at a remote source of data
- Classifier Agent: A derived program learned over some remote site of data, labels or tags data with class labels
- Meta-Learning Agent: A Machine Learning program that Learns how to combine a number of remote classifier agents, the result is a single classifier agent
Meta-learning: An Algorithm-independent Technique for Scalable and Accurate Inductive Learning

Salvatore J. Stolfo
Department of Computer Science
Columbia University and
Philip Chan
Florida Institute of Technology
Learn and Integrate Classifiers

- Large datasets are partitioned into subsets
- Distributed databases are inherently partitioned
- Collective knowledge is harvested from individual knowledge sources

How to integrate the classifiers?
Integrating Classifiers

• Integrating the *concept descriptions languages* (a logical cross-bar switch)?
  – different representations: probabilities, hyperplanes, logical expressions
  – difficult if not impossible to accurately map all representations into one standard

• Integrating the behavior of classifiers (their predictions)?
  – algorithm/representation-independent
  – existing and new algorithms can be plugged in with ease
  – voting and statistical techniques abound
  – meta-learning:
    * arbitration: conflicting predictions are resolved by a learned arbiter
    * combining/coalescing: learn a function over classifiers’ predictions
SHARING REMOTE CLASSIFIERS

SHARING KNOWLEDGE WITHOUT SHARING DATA
Meta-learning: Arbiters and Combiners

- The *arbiter* Resolves conflicting predictions (disagreements)

- The *combiner* makes a final prediction based on the base predictions
Hierarchical Meta-learning in Agent Infrastructures

- **Arbiter tree**

```
A_{14}  
/  
A_{12} A_{34}  
/  /  
C_1 C_2 C_3 C_4  
/  /  /  
T_1 T_2 T_3 T_4
```

Arbiters
Classifiers
Training data subsets

- **Combiner tree**

```
C_{14}  
/  
C_{12} C_{34}  
/  /  
C_1 C_2 C_3 C_4  
/  /  /  
T_1 T_2 T_3 T_4
```

Combiners
Classifiers
Training data subsets
Evaluation Studies

- Many issues exist and are addressed by various experiments
- Main focus is on prediction accuracy
  - disjoint training and test sets
  - 10-fold cross validation
  - 2 to 64 data subsets
  - global classifier (whole dataset or 1 data subset)
- “Off-the-shelf” learning algorithms
  - ID3 (Quinlan 86)
  - CART (Breiman et al. 84)
  - BAYES (Clark & Niblett 87)
  - WPEBLS (Cost & Salzberg 93)
- “Off-the-shelf” learning tasks
  - DNA splice junctions (3,190) (Towell et al. 90)
  - Protein coding regions (21,625) (Craven & Shavlik 93)
  - Protein secondary structures (20,000) (Qian & Sejnowski 88)
Subsets and Sampling

- How do the # of subsets and subset size affect accuracy?
- Is random sampling of a subset sufficient?

- Subsets can’t be too small to generate reasonable classifiers
- Random sampling is not sufficient; combining is necessary
Arbiter Trees

- Is hierarchical meta-learning necessary?
- How do the order of the arbiter trees and training set size limit affect the accuracy?

- Lower order trees are more accurate
- Doubling the arbiter training set size maintains accuracy
Combiner Trees

- How does the combiner trees fare?
- Class-attribute-combiner strategy

- Statistically significant and consistent improvement in the PCR dataset beyond the original accuracy
Summary of Meta-learning Results

- Random sampling is not sufficient
- Existing voting and statistical combining techniques are not sufficient
- “One-level” meta-learning outperforms the voting and statistical techniques
- Hierarchical meta-learning can sustain high accuracy
- Meta-level training set size needs only to be twice the subset size
- Proportional distribution of classes in the data subsets is beneficial
- Lower-order trees are more accurate than higher-order trees
- Combiner trees can boost accuracy beyond the global classifier’s
- Data replication does not improve accuracy
An Illustration: Distributed DNA Sequence Databases

SITES 1 and 2:

<table>
<thead>
<tr>
<th>DNA sequence #</th>
<th>Nucleotide sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>...CCAGCTGCATCACAGGAGGCCCAGCCGAGGAGGTCCTGTTCCAGGGCCTTCGAGCAGGTCTCG...</td>
</tr>
<tr>
<td>2</td>
<td>...GAGAGAGAGAAGCAGAAATAATCTTGATTGCTTCCCTCAGCCAGTGTCTTACCATTGCA...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DNA sequence #</th>
<th>Nucleotide sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>...ACAGGCTTTCCACGCCTCCAGCGAGGATGTACTGATTCCAGGCCCTCGGAGCCAGTCTCG...</td>
</tr>
<tr>
<td>2</td>
<td>...TAGCCGAGCAGAAGGATAAGTCTTGATGATGCTTACCAAGGTCTAATGCTTTCCATACCT...</td>
</tr>
</tbody>
</table>
Sample SPLICE JUNCTION sequences at SITE 3

<table>
<thead>
<tr>
<th>Junction</th>
<th>P=0</th>
<th>P=20</th>
<th>P=28</th>
<th>P=83</th>
<th>P=26</th>
<th>P=1</th>
<th>P=2</th>
<th>P=36</th>
<th>P=80</th>
<th>P=80</th>
</tr>
</thead>
<tbody>
<tr>
<td>intron-exon [IE]</td>
<td>C</td>
<td>T</td>
<td>...TAATAACATTCTTAT</td>
<td>A</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>...ATCCATTCAATGAAT</td>
<td>A</td>
<td>T</td>
</tr>
<tr>
<td>exon-intron [EI]</td>
<td>G</td>
<td>A</td>
<td>...GCCCGTCATAAATC</td>
<td>T</td>
<td>G</td>
<td>G</td>
<td>T</td>
<td>...GAGACTCATGCCCAGG</td>
<td>T</td>
<td>C</td>
</tr>
<tr>
<td>neither [N]</td>
<td>T</td>
<td>A</td>
<td>...CTATCCACAGACGAT</td>
<td>A</td>
<td>G</td>
<td>G</td>
<td>A</td>
<td>...TGCCCCTCTGGGCA</td>
<td>A</td>
<td>A</td>
</tr>
</tbody>
</table>
A (logic-based) rule equivalent of the first branch at the top of the ID3 Decision tree is:

“If ($X.p_{-1} = A$) and ($X.p_2 = A$) then the center doesn’t have a junction, i.e. $X.Junction = N$.”

A rule equivalent to the second branch is:

“If ($X.p_{-1} = A$) and ($X.p_2 = C$) then the center doesn’t have a junction, i.e. $X.Junction = N$.”
Sample Sequences To Be Extracted

Classifier Agent Sent to SITE 1:

*Select $X_*$ From DNA-Sequence Where $C_{IP3-1}(X,p_{-30}...X,p_{30}) = EI$. 

<table>
<thead>
<tr>
<th>$C_{IP3-1}$</th>
<th>Meta-classifier</th>
<th>$p_{-30}$</th>
<th>$p_{-29}...p_{-3}$</th>
<th>$p_{-2}$</th>
<th>$p_{-1}$</th>
<th>$p_1$</th>
<th>$p_2$</th>
<th>$p_3...p_{29}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>EI</td>
<td>EI</td>
<td>A</td>
<td>CCAGAGGATCTATACCTCTTCGATAC</td>
<td>A</td>
<td>G</td>
<td>G</td>
<td>T</td>
<td>AAAGTTAACATAGATGAGATCTG</td>
</tr>
<tr>
<td>EI</td>
<td>EI</td>
<td>T</td>
<td>GCCAGCTACGGGCGAGGCGCTCGGAG</td>
<td>A</td>
<td>G</td>
<td>G</td>
<td>T</td>
<td>GAGGACCTGGATTCCTCGCTGCAAGT</td>
</tr>
<tr>
<td>N</td>
<td>EI</td>
<td>G</td>
<td>GAGCTGCGACACGAGGAGGAGGACAT</td>
<td>G</td>
<td>A</td>
<td>G</td>
<td>T</td>
<td>AAGTAGGCCCGCGCTGAGGGGCTGCTG</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>A</td>
<td>TCTACTTGTAACTCATAATTTTCTCTGT</td>
<td>G</td>
<td>C</td>
<td>T</td>
<td>A</td>
<td>GATAACAAATTAAGAAAACCAAAACA</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>A</td>
<td>GGCTGCTATCAGAGGTGGGGCTGG</td>
<td>T</td>
<td>G</td>
<td>T</td>
<td>G</td>
<td>GCTGCTGCTCTGGGTACAAAGTACAT</td>
</tr>
</tbody>
</table>
A Sample Meta-Classifier Learned From 4 Base Classifiers

c-id3-1 = EI: EI

c-id3-1 = IE:
  | p-3 = A: N
  | p-3 = C: IE
  | p-3 = G: N
  | p-3 = T: IE

c-id3-1 = N:
  | p1 = A: N
  | p1 = C: N
  | p1 = G:
    |     | p5 = A: N
    |     | p5 = C: N
    |     | p5 = G:
    |     |     | p2 = A: N
    |     |     | p2 = C: N
    |     |     | p2 = G: N
    |     |     | p2 = T: EI
    |     | p5 = T: N
  | p1 = T: N
A Host Meta-Learning Environment

- Partitioning and Distributing data,
- Invoking Different Meta-Learning Strategies In Parallel,
- Pairing Classifiers to Reduce Intermediate Training Sets for Meta-Learning,
- Filtering and Communication of Training and Testing Data Between Processors, and,
- Instrumentation to Gather Statistics Used in Formulating or Designing Specific Meta-Learning Architectures.

- LAUNCHING OF ENCAPSULATED LEARNING AND META-LEARNING AGENTS OVER NETWORKS
Future Research: The JAM PROJECT

• Specialized representations (new attributes/predicates) and algorithms for meta-learning

• New meta-learning strategies and training-set composition rules

• Agent computing: collaboration with FSTC in field-testing learning agents on the Internet:
  • Acquisition of TRANSACTION DATABASES with FRAUD LABELS
    • Demonstration of Remote Learning and Meta-Learning Agents
    • Exchange of Learned Classifiers
    • Installation of Learned Classifiers as SENTRIES to warn of FRAUD
JAM Prototype: One coordinator, multiple data sites

- Coordinator
  - Dispatches agents to different data sites
  - Multithreaded for concurrent service
  - Simple error recovery from data sites crashes
- Data Site
  - Accepts and executes agents
  - Agent Independent
- Agent: the ID3 machine learning algorithm
- Platform Independent (Java)
- Simple Graphical User Interface
Data Schema and Stats for (Fraud) Transaction Data Sets

- Number of Attributes: $30 + \Delta$ (all numeric)
  - Many fields are categorical (i.e. numbers represent a few discrete categories)
  - Developed over years to capture important information
- Size: Fixed 137 bytes per transaction
- Type of Information:
  - A (jumbled) account number (no real identifiers)
  - Scores produced by a COTS authorization/detection system
  - Date/Time of transaction
  - Past payment information of the transactor
  - Amount of transaction
  - Geographic information: where the transaction was initiated, the location of the merchant and transactor
  - Codes for validity and manner of entry of the transaction
  - An industry standard code for the type of merchant
  - A code for other recent “non-monetary” transaction types by transactor
  - The age of the account and the card
  - Other card/account information
  - Confidential/Proprietary Fields (other potential indicators)
  - Fraud Label (0/1)
- .5MM records by each Bank:
  - sampling 50,000 per month
  - Span 11/95 - 10/96
DETAILS of the JAM Project

VISIT with your favorite Browser:

- http://www.fstc.org - and click on Fraud Page
- http://www.cs.columbia.edu/~sal

SUPPORTED BY:

- NYSSTF Polytechnic Univeristy CATT
- NSF CISE KMCS and DBES Programs
- DARPA ITO Intrusion Dection Program