USING CLASSIFIER CASCADES FOR SCALABLE E-MAIL CLASSIFICATION

Jay Pujara
Hal Daumé III
Lise Getoor

jay@cs.umd.edu
me@hal3.name
getoor@cs.umd.edu

9/1/2011
Building a scalable e-mail system

- **Goal:** Maintain system throughput across conditions
- **Varying conditions**
  - Load varies
  - Resource availability varies
  - Task varies
- **Challenge:** Build a system that can adapt its operation to the conditions at hand
Problem structure informs scalable solution

Feature Structure
- IP
- Mail From
- Subject
- Body

Derived features

Class Structure
- Ham
- Spam
  - Business
  - Social Network
    - Personal
    - Newsgroup

Cost
- $
- $$$

Granularity
Important facets of problem

- **Structure in input**
  - Features may have an order or systemic dependency
  - Acquisition costs vary: cheap or expensive features

- **Structure in output**
  - Labels naturally have a hierarchy from coarse-to-fine
  - Different levels of hierarchy have different sensitivities to cost

- Exploit structure during classification

- Minimize costs, minimize error
Two overarching questions

- When should we acquire features to classify a message?
- How does this acquisition policy change across different classification tasks?
- Classifier Cascades can answer both questions!
Introducing Classifier Cascades

• Series of classifiers: \( f_1, f_2, f_3 \ldots f_n \)
Introducing Classifier Cascades

• Series of classifiers: $f_1, f_2, f_3 \ldots f_n$
• Each classifier operates on different, increasingly expensive sets of features ($\phi$) with costs $c_1, c_2, c_3 \ldots c_n$
Introducing Classifier Cascades

• Series of classifiers: \( f_1, f_2, f_3 \ldots f_n \)

• Each classifier operates on different, increasingly expensive sets of features \( \phi \) with costs \( c_1, c_2, c_3 \ldots c_n \)

• Classifier outputs a value in \([-1,1]\), the margin or confidence of decision
Introducing Classifier Cascades

- Series of classifiers: $f_1, f_2, f_3 \ldots f_n$
- Each classifier operates on different, increasingly expensive sets of features ($\phi$) with costs $c_1, c_2, c_3 \ldots c_n$
- Classifier outputs a value in $[-1,1]$, the margin or confidence of decision
- $\gamma$ parameters control the relationship of classifiers
Optimizing Classifier Cascades

- Loss function: $L(y, F(x))$ – errors in classification

- Minimize loss function, incorporating cost
  - Cost-constraint with budget (load-sensitive):
    $$\min \sum_{(x,y) \in D} L(y, F(x)) \quad \text{s.t.} \quad C(x) < B$$
  - Cost Sensitive loss function (granular):
    $$\min \sum_{(x,y) \in D} L(y, F(x)) + \lambda C(x)$$

- Use grid-search to find optimal $\gamma$ parameters
Features have costs & dependencies

IP is known at socket connect time, is 4 bytes in size
Features have costs & dependencies

The Mail From is one of the first commands of an SMTP conversation. From addresses have a known format, but higher diversity.
The subject, one of the mail headers, occurs after a number of network exchanges. Since the subject is user-generated, it is very diverse and often lacks a defined format.
Load-Sensitive Problem Setting

- Train IP, MailFrom, and Subject classifiers
- For a given budget, $B$, choose $\gamma_1, \gamma_2$ that minimize error within $B$
- Constraint: $C(x) < B$
Load-Sensitive Challenges

- Overfitting model when choosing \( \gamma_1, \gamma_2 \)
- Train-time costs underestimated versus test-time performance
- Use a regularization constant \( \Delta \)
  - Sensitive to cost variance (\( \sigma \))
  - Accounts for variability
- Revised constraint: \( C(x) + \Delta \sigma < B \)
Granular Classification
E-mail Challenges: Spam Detection

- Most mail is spam
- Billions of classifications
- Must be incredibly fast
E-mail Challenges: Categorizing Mail

- E-mail does more, tasks such as:
  - Extract receipts, tracking info
  - Thread conversations
  - Filter into mailing lists
  - Inline social network response

- Computationally intensive processing
- Each task applies to one class
Coarse task is constrained by feature cost

Feature Structure

1. IP
2. Mail From
3. Subject
4. Body

Class Structure

1. Ham
   - Business
   - Personal
   - Newsgroup
2. Spam
   - Social Network

Derived features

Cost

Granularity

\( \lambda_c \)
Fine task is constrained by misclassification cost
Granular Classification Problem Setting

• Two separate models for different tasks, with different classifiers and cascade parameters
• Choose $\gamma_1, \gamma_2$ for each cascade to balance accuracy and cost with different tradeoffs $\lambda$
Experimental Results
Experimental Setup: Overview

- Two tasks: load-sensitive & granular classification
- Two datasets: Yahoo! Mail corpus and TREC-2007
  - Load-sensitive uses both datasets, granular uses only Yahoo!
- Results are L1O, 10-fold CV with **bold** values significant (p<.05)
- Cascade stages use MEGAM MaxEnt classifier
Experimental Setup: Yahoo! Data

- Data from 1227 Yahoo! Mail messages from 8/2010
- Feature costs calculated from network + storage cost
Experimental Setup: TREC data

- Data from TREC-2007 Public Spam Corpus, 47,194 messages
- Use same feature cost estimates

<table>
<thead>
<tr>
<th>Class</th>
<th>Messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spam</td>
<td>39,055</td>
</tr>
<tr>
<td>Ham</td>
<td>8,139</td>
</tr>
</tbody>
</table>
Results: Load-Sensitive Classification

Regularization prevents cost excesses

Classification Cost vs. Classification Budget

Y!Mail Dataset

<table>
<thead>
<tr>
<th>Δ</th>
<th>Y!Mail</th>
<th>TREC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.115</td>
<td>0.059</td>
</tr>
<tr>
<td>.25</td>
<td>0.020</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Average excess cost
Results: Load-Sensitive Classification

Significant error reduction

Classification Error across methods in different datasets

Dataset: Yahoo! Mail, TREC-2007

- Naive
- ACC, Δ=0
- ACC, Δ=.25
- ACC, Δ=.5
## Results: Granular Classification

<table>
<thead>
<tr>
<th>Feature Set</th>
<th>Feature Cost</th>
<th>Misclass Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coarse</td>
<td>Fine</td>
</tr>
<tr>
<td>Fixed: IP</td>
<td>.168</td>
<td>.139</td>
</tr>
<tr>
<td>ACC: $\lambda_c=1.5$, $\lambda_f=1$</td>
<td>.187</td>
<td>.140</td>
</tr>
<tr>
<td>Fixed: IP+MailFrom</td>
<td>.490</td>
<td>.128</td>
</tr>
<tr>
<td>ACC: $\lambda_c=.1$, $\lambda_f=.075$</td>
<td>.431</td>
<td>.111</td>
</tr>
<tr>
<td>Fixed: IP+MailFrom+Subject</td>
<td>1.00</td>
<td>.106</td>
</tr>
<tr>
<td>ACC: $\lambda_c=.02$, $\lambda_f=.02$</td>
<td>.691</td>
<td>.108</td>
</tr>
</tbody>
</table>

- Compare fixed feature acquisition policies to adaptive classifiers
- Significant gains in performance or cost (or both) depending on tradeoff
Dynamics of choosing $\lambda_c$ and $\lambda_f$
Different approaches, same tradeoff

![Tradeoff Between Classification Error and Cost in Granular Classification](image)

![Tradeoff Between Classification Error and Cost in Load Sensitive Classification](image)
Conclusion

- Problem of scalable e-mail classification
- Introduce two settings
  - Load-sensitive Classification: known budget
  - Granular Classification: task sensitivity
- Use classifier cascades to achieve tradeoff between cost and accuracy
- Demonstrate results superior to baseline

Questions?