An introduction to Web Mining
part II

Ricardo Baeza-Yates, Aristides Gionis
Yahoo! Research
Barcelona, Spain & Santiago, Chile

ECML/PKDD 2008 Antwerp
Agenda

- Statistical methods: the size of the web
- Content mining
- Link analysis for spam detection
• Issues
  – The web is really infinite
    • Dynamic content, e.g., calendar
    • Soft 404: www.yahoo.com/anything is a valid page
  – Static web contains syntactic duplication, mostly due to mirroring (~20-30%)
  – Some servers are seldom connected
• Who cares?
  – Media, and consequently the user
  – Engine design
  – Engine crawl policy. Impact on recall
What can we attempt to measure?

- The relative size of search engines
  - The notion of a page being indexed is *still* reasonably well defined.
  - Already there are problems
    - Document extension: e.g. Google indexes pages not yet crawled by indexing anchor-text.
    - Document restriction: Some engines restrict what is indexed (first $n$ words, only relevant words, etc.)

- The coverage of a search engine relative to another particular crawling process
• [Bharat & Broder 98]
• Main idea:
• $\Pr[A \& B \mid A] = \frac{s(A \& B)}{s(A)}$
• $\Pr[A \& B \mid B] = \frac{s(A \& B)}{s(B)}$
• Thus:
  
  $$\frac{s(A)}{s(B)} = \frac{\Pr[A \& B \mid B]}{\Pr[A \& B \mid A]}$$
• Need
  
  – **Sampling** a random page from the index of a SE
  – **Checking** if a page exists at the index of a SE
Sampling and checking pages

- Both tasks by using the public interface SEs
- **Sampling:**
  - Construct a large lexicon
  - Use the lexicon to fire random queries
  - Sample a page from the results
  - (introduces query and ranking biases)
- **Checking:**
  - Construct a *strong* query from the most k most distinctive terms of the page
  - (in order to deal with aliases, mirror pages, etc.)
Refinement of the B&B technique
[Gulli & Signorini, 2005]

- Total web = 11.5 B
- Union of major search engines = 9.5 B
- Common web = 2.7 B (Much higher correlation than before)
Random-walk sampling

- [Bar-Yossef and Gurevich, WWW 2006]
- Define a graph on documents and queries:
  - Edge \((d,q)\) indicates that document \(d\) is a result of a query \(q\)
- Random walk gives biased samples
- Bias depends on the degree of docs and queries
- Use Monte Carlo methods to unbias the samples and obtain uniform samples
- Paper shows how to obtain estimates of the degrees and weights needed for the unbiasing
Bias towards long documents

Deciles of documents ordered by size

Percent of documents from sample

Pool Based
Random Walk
Bharat-Broder
Relative size of major search engines

- [Bar-Yossef and Gurevich, 2006]

Google = 1
Yahoo! = 1.28
MSN Search = 0.73
Content mining

- Duplicate and near-duplicate document detection
- Content-based spam detection
Duplicate/Near-Duplicate Detection

- Duplication: Exact match with fingerprints
- Near-Duplication: Approximate match
  - Overview
    - Compute syntactic similarity with an edit-distance measure
    - Use similarity threshold to detect near-duplicates
      - E.g., Similarity > 80% => Documents are “near duplicates”
      - Not transitive though sometimes used transitively
Computing Similarity

• Features:
  – Segments of a document (natural or artificial breakpoints) [Brin95]
  – Shingles (Word N-Grams) [Brin95, Brod98]
    “a rose is a rose is a rose” =>
    \[
    \begin{align*}
    \text{a}_\text{rose}_\text{is}_\text{a} \\
    \text{rose}_\text{is}_\text{a}_\text{rose} \\
    \text{is}_\text{a}_\text{rose}_\text{is}
    \end{align*}
    \]
    are all added in the bag of word representation

• Similarity Measure
  – TFIDF [Shiv95]
  – Set intersection [Brod98]
    (Specifically, \( \frac{\text{Size}_\text{of}_\text{Intersection}}{\text{Size}_\text{of}_\text{Union}} \))
Jaccard coefficient

- Consider documents $a$ and $b$
- Are represented by bag of words $A$ and $B$, resp.
- Then:

$$J(a,b) = \frac{|A \text{ intersect } B|}{|A \text{ union } B|}$$
Shingles + Jaccard coefficient

• Computing exact Jaccard coefficient between all pairs of documents is expensive (quadratic)
• Approximate similarities using a cleverly chosen subset of shingles from each (a sketch)
• Idea based on hashing

• Also known as locality-sensitive hashing (LSH)
  • A family of hash functions for which items that are similar have higher probability of colliding
Shingles + Jaccard coefficient

- Estimate size_of_intersection / size_of_union based on a short sketch ([Broder 97, Broder 98])
  - Create a “sketch vector” (e.g., of size 200) for each document
  - Documents which share more than $t$ (say 80%) corresponding vector elements are similar
  - For doc D, sketch$[i]$ is computed as follows:
    - Let $f$ map all shingles in the universe to $0..2^m$ (e.g., $f =$ fingerprinting)
    - Let $\pi_i$ be a specific random permutation on $0..2^m$
    - Pick MIN $\pi_i(f(s))$ over all shingles $s$ in D
Computing Sketch[i] for Doc1

Document 1

Start with 64 bit shingles

Permute on the number line with $\pi_i$

Pick the min value
Test if Doc1.Sketch[i] = Doc2.Sketch[i]

Are these equal?

Test for 200 random permutations: $\pi_1, \pi_2, \ldots, \pi_{200}$
A = B iff the shingle with the MIN value in the union of Doc1 and Doc2 is common to both (I.e., lies in the intersection)

This happens with probability:

\[
\frac{\text{Size_of_intersection}}{\text{Size_of_union}}
\]
Mirror detection

- Mirroring is systematic replication of web pages across hosts.
  - Single largest cause of duplication on the web
- Host1/α and Host2/β are mirrors iff
  For all (or most) paths p such that when
  http://Host1/ α / p exists
  http://Host2/ β / p exists as well
  with identical (or near identical) content, and vice versa.
- E.g.,
  - Structural Classification of Proteins
    - http://scop.mrc-lmb.cam.ac.uk/scop
    - http://scop.berkeley.edu/
    - http://pdb.weizmann.ac.il/scop
    - http://scop.protres.ru/
### Featured Items

<table>
<thead>
<tr>
<th>Title</th>
<th>Status</th>
<th>Bids</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td><del>Flow Blue Cake Plate With Pedestal</del>Gorgeous!!!</td>
<td>A</td>
<td>5</td>
<td>$50.00</td>
</tr>
<tr>
<td><del>Flow Blue Taureen With Soup Spoon</del>Gorgeous~ All Porcelain~*</td>
<td>R</td>
<td>3</td>
<td>$55.00</td>
</tr>
<tr>
<td>Vintage Swiss Silver Case Pocket Watch by Remontoir</td>
<td>R</td>
<td>1</td>
<td>$30.00</td>
</tr>
<tr>
<td>One Nina &amp; Three Rara Kyyu Paintings</td>
<td>A</td>
<td>-</td>
<td>$20.00</td>
</tr>
<tr>
<td>0b2150502 / GORGEOUS HANDICRAFT TEAKWOOD ELEPHANT NCS152</td>
<td>A</td>
<td>-</td>
<td>$75.98</td>
</tr>
<tr>
<td>0b2151102 / BEAUTIFUL HAND MADE TEAKWOOD ELEPHANT NCS152</td>
<td>A</td>
<td>-</td>
<td>$75.98</td>
</tr>
</tbody>
</table>

- Current Bid: $50.00  
  - Auction Ends 8/18/01 11:00 PM
- Current Bid: $55.00  
  - Auction Ends 8/18/01 10:40 PM
- Current Bid: $30.00  
  - Auction Ends 8/18/01 1:00 AM
- Current Bid: $20.00  
  - Auction Ends 8/17/01 11:00 PM
- Current Bid: $75.98  
  - Auction Ends 8/18/01 1:00 AM
Motivation of near-duplicate detection

• Why detect mirrors?
  – Smart crawling
    • Fetch from the fastest or freshest server
    • Avoid duplication
  – Better connectivity analysis
    • Combine inlinks
    • Avoid double counting outlinks
  – Redundancy in result listings
    • “If that fails you can try: <mirror>/samepath”
  – Proxy caching
Study genealogy of the Web

- [Baeza-Yates et al., 2008]
- New pages copy content from existing pages
- Web genealogy study:
  - How textual content of source pages (parents) are reused to compose part of new Web pages (children)
  - Not near-duplicates, as similarities of short passages are also identified
- How can search engines benefit?
  - By associating more relevance to a parent page?
  - By trying to decrease the bias?
Web genealogy

parents

sterile

coexistent

inter-site relation (w/o mirrors)

orphan

intra-site relation

parents

sterile

coexistent

inter-site relation (w/o mirrors)

orphan
Pagerank for each component

Average Pagerank

collection 2003
collection 2004
collection 2005

intra OrP intra OLP intra ChP intra Ste inter OrP inter OLP inter ChP inter Ste
Content-based spam detection

- Machine-learning approach --- training
Content-based spam detection

- Machine-learning approach --- prediction
The dataset

- Label “spam” nodes on the host level
  - agrees with existing granularity of Web spam
- Based on a crawl of .uk domain from May 2006
- 77.9 million pages
- 3 billion links
- 11,400 hosts
The dataset

- 20+ volunteers tagged a subset of host
- Labels are “spam”, “normal”, “borderline”
- Hosts such as .gov.uk are considered “normal”
- In total 2,725 hosts were labelled by at least two judges
- hosts in which both judges agreed, and “borderline” removed
- Dataset available at http://www.yr-bcn.es/webspam/
Content-based features

• Number of words in the page
• Number of words in the title
• Average word length
• Fraction of anchor text
• Fraction of visible text

See also [Ntoulas et al., 06]
Content-based features
Entropy related

• Let \( T = \{ (w_1, p_1), \ldots, (w_k, p_k) \} \) the set of trigrams in a page, where trigram \( w_i \) has frequency \( p_i \).

• Features:
  ✓ Entropy of trigrams: \( H = -\sum_i p_i \log(p_i) \)
  ✓ Independent trigram likelihood: \(- (1/k) \sum_i \log(p_i)\)
  ✓ Also, compression rate, as measured by bzip
Content-based features related to popular keywords

- $F$ set of most frequent terms in the collection
- $Q$ set of most frequent terms in a query log
- $P$ set of terms in a page
- Features:
  - Corpus “precision” $| P \cap F | / | P |$
  - Corpus “recall” $| P \cap F | / | F |$
  - Query “precision” $| P \cap Q | / | P |$
  - Query “recall” $| P \cap Q | / | Q |$
Content-based features

number of words in home page

number of words in page --- home

An introduction to Web Mining, ECML/PKDD 2008, Antwerp
Content-based features compression rate

compression rate --- home

An introduction to Web Mining, ECML/PKDD 2008, Antwerp
The classifier

• C4.5 decision tree with bagging and cost weighting for class imbalance

• With content-based features achieves:
  – True positive rate: 64.9%
  – False positive rate: 3.7%
  – F-Measure: 0.683
• Link-based spam detection
Link-based spam detection

- Link farms used by spammers to raise popularity of spam pages
- Link farms and other spam strategies leave traces on the structure of the web graph
- Dependencies between neighbouring nodes of the web graph are created
- Naturally, spammers try to remove traces and dependencies
Link farms

- Single-level link farms can be detected by searching for nodes sharing their out-links
- In practice more sophisticated techniques are used
Link-based features
Degree related

• in-degree
• out-degree
• edge reciprocity
  – number of reciprocal links
• assortativity
  – degree over average degree of neighbors
Link-based features
PageRank related

• PageRank
• indegree/PageRank
• outdegree/PageRank
• ...
• Truncated PageRank [Becchetti et al., 2006]
  – A variant of PageRank that diminishes the influence of a page the PageRank score of its neighbors
• TrustRank [Gyongyi et al., 2004]
  – As PageRank but with teleportation at Open Directory pages
Link-based features
Supporters

- Let \( x \) and \( y \) be two nodes in the graph.
- Say that \( y \) is a \( d \)-supporter of \( x \), if the shortest path from \( y \) to \( x \) has length at most \( d \).
- Let \( N_d(x) \) be the set of the \( d \)-supporters of \( x \).
- Define bottleneck number of \( x \), up to distance \( d \) as

\[
b_d(x) = \min_{j \leq d} \frac{N_j(x)}{N_{j-1}(x)}
\]

- minimum rate of growth of the neighbors of \( x \) up to a certain distance.
Link-based features
Supporters

An introduction to Web Mining, ECML/PKDD 2008, Antwerp
Link-based features
Supporters

- How to compute the supporters?
- Utilize *neighborhood function*
  \[
  N(h) = \left| \{ (u,v) \mid d(u,v) \leq h \} \right| = \bigtriangleup_u N(u,h)
  \]
- and ANF algorithm [Palmer et al., 2002]
- Probabilistic counting using Flajolet-Martin sketches or other data-stream technology
- Can be done with a few passes and exchange of sketches, instead of executing BFS from each node
Link-based features - In-degree
Link-based features - Assortativity
Link-based features - Supporters
The classifier: Combining features

- C4.5 decision tree with bagging and cost weighting for class imbalance

<table>
<thead>
<tr>
<th>features</th>
<th>Content</th>
<th>Link</th>
<th>Both</th>
</tr>
</thead>
<tbody>
<tr>
<td>True positive rate:</td>
<td>64.9%</td>
<td>79.4%</td>
<td>78.7%</td>
</tr>
<tr>
<td>False positive rate:</td>
<td>3.7%</td>
<td>9.0%</td>
<td>5.7%</td>
</tr>
<tr>
<td>F-Measure:</td>
<td>0.683</td>
<td>0.659</td>
<td><strong>0.723</strong></td>
</tr>
</tbody>
</table>
Dependencies among spam nodes

An introduction to Web Mining, ECML/PKDD 2008, Antwerp
Dependencies among spam nodes

- Spam nodes in out-links
- Spam nodes from in-links
Exploiting dependencies

- Use a dataset with labeled nodes
- Extract content-based and link-based features
- Learn a classifier for predicting spam nodes independently
- Exploit the graph topology to improve classification
  - Clustering
  - Propagation
  - Stacked learning
Let $G=(V,E,w)$ be the host graph
Cluster $G$ into $m$ disjoint clusters $C_1,\ldots,C_m$
Compute $p(C_i)$, the fraction of nodes classified as spam in cluster $C_i$
  - if $p(C_i) > t_u$ label all as spam
  - if $p(C_i) < t_l$ label all as non-spam
A small improvement:

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>Clustering</th>
</tr>
</thead>
<tbody>
<tr>
<td>True positive rate:</td>
<td>78.7%</td>
<td>76.9%</td>
</tr>
<tr>
<td>False positive rate:</td>
<td>5.7%</td>
<td>5.0%</td>
</tr>
<tr>
<td>F-Measure:</td>
<td>0.723</td>
<td>0.728</td>
</tr>
</tbody>
</table>
Exploiting dependencies
Propagation

- Perform a random walk on the graph
- With probability $a$ follow a link
- With prob $1-a$ jump to a random node labeled spam
- Relabel as spam every node whose stationary distribution component is higher than a threshold

- Improvement:

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>Propagation</th>
</tr>
</thead>
<tbody>
<tr>
<td>True positive rate:</td>
<td>78.7%</td>
<td>75.0%</td>
</tr>
<tr>
<td>False positive rate:</td>
<td>5.7%</td>
<td>4.3%</td>
</tr>
<tr>
<td>F-Measure:</td>
<td>0.723</td>
<td>0.733</td>
</tr>
</tbody>
</table>
Exploiting dependencies
Stacked learning

• Meta-learning scheme [Cohen and Kou, 2006]
• Derive initial predictions
• Generate an additional attribute for each object by combining predictions on neighbors in the graph
• Append additional attribute in the data and retrain
• Let $p(h)$ be the prediction of a classification algorithm for $h$
• Let $N(h)$ be the set of pages related to $h$
• Compute:

$$f(h) = \sum_{g \in N(h)} \frac{p(g)}{|N(h)|}$$

• Add $f(h)$ as an extra feature for instance $h$ and retrain
Exploiting dependencies
Stacked learning

• First pass:

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>in</th>
<th>out</th>
<th>both</th>
</tr>
</thead>
<tbody>
<tr>
<td>True positive rate:</td>
<td>78.7%</td>
<td>84.4%</td>
<td>78.3%</td>
<td>85.2%</td>
</tr>
<tr>
<td>False positive rate:</td>
<td>5.7%</td>
<td>6.7%</td>
<td>4.8%</td>
<td>6.1%</td>
</tr>
<tr>
<td>F-Measure:</td>
<td>0.723</td>
<td>0.733</td>
<td>0.742</td>
<td>0.750</td>
</tr>
</tbody>
</table>

• Second pass:

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>1st pass</th>
<th>2nd pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>True positive rate:</td>
<td>78.7%</td>
<td>85.2%</td>
<td>88.2%</td>
</tr>
<tr>
<td>False positive rate:</td>
<td>5.7%</td>
<td>6.1%</td>
<td>6.3%</td>
</tr>
<tr>
<td>F-Measure:</td>
<td>0.723</td>
<td>0.750</td>
<td>0.763</td>
</tr>
</tbody>
</table>
Overall summary

• Open problems and challenges:
  – Modeling web graph and other web data
  – Model evolution
  – Data cleaning and anonymization
  – Improve IR relevance
  – Manage and integrate highly heterogeneous information: content, links, social links, tags, feedback, usage logs, wisdom of crowd, etc.
  – Design improved web applications
  – Battle adversarial attempts and collusions
Special thanks

- Carlos Castillo
- Alessandro Tiberi
- Barbara Poblete
- Alvaro Pereira
Thank you!

WWW2008 Beijing