CS 277, Data Mining

Web Data Analysis

Padhraic Smyth
Department of Computer Science
Bren School of Information and Computer Sciences
University of California, Irvine
Web Mining

Web = a potentially enormous “data set” for data mining

Multiple aspects of “Web mining”

1. Web Content
   e.g., categorizing Web pages based on their text content
2. Web Connectivity/Link Analysis
   e.g., characterizing distributions on path lengths between pages
   e.g., determining importance of pages from graph structure
3. Web Usage
   e.g., understanding user behavior from Web and search logs
4. Web Advertising
   e.g., algorithms for optimizing which ads to show which users

All are interconnected/interdependent
   – E.g., Google (and most search engines) use both content and connectivity
## Different Aspects of User Data on the Web

<table>
<thead>
<tr>
<th>Type</th>
<th>Content</th>
<th>How Collected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navigation data</td>
<td>URLs, time-stamps, etc</td>
<td>Web logs</td>
</tr>
<tr>
<td>Search query data</td>
<td>Text strings</td>
<td>Web logs</td>
</tr>
<tr>
<td>Transaction data</td>
<td>Item purchased, credit card, home address, etc</td>
<td>Server-side database</td>
</tr>
<tr>
<td>Registration data</td>
<td>Name, address, demographics</td>
<td>Server-side database</td>
</tr>
<tr>
<td>Cookie files</td>
<td>Information to link user to session</td>
<td>File “dropped” by Web server on client machine</td>
</tr>
<tr>
<td>User-generated text</td>
<td>Blogs, tweets, product reviews, customer emails</td>
<td>Server-side database or Web crawling</td>
</tr>
<tr>
<td>Social network data</td>
<td>Who is connected to who</td>
<td>Social networking sites</td>
</tr>
<tr>
<td>Location data</td>
<td>Location (x,y) at time t</td>
<td>IP address or GPS</td>
</tr>
</tbody>
</table>
Eye-Tracking: The Golden Triangle for Search
from Hotchkiss, Alston, Edwards, 2005; EnquiroResearch
Who has this Data?

- Every Web site has server side navigation and query data
- Search engines, social network sites
  - Google, Microsoft, Yahoo!, Facebook
- Content Publishers
  - Newspapers, magazines, portals
- Retailers
  - Amazon, eBay, Target, Experian
- Data aggregators/advertising networks
  - Doubleclick, BlueKai
- Web analytics
  - Neilsen, ComScore
The Company Perspective: Learning about the User

• Companies would like to integrate all of this data
  – create a rich and dynamic picture of each user
  – Analogy: think of the local village store owner and his/her regular customers

• Challenges:
  – Accessing the data:
    • Retailer generally only sees their own Website data
    • Google, Facebook, etc
  – Legal restrictions
    • Privacy laws prohibit widespread sharing of data
  – Technical challenges
    • Is “J. Smith” registered at Target the same as “John Smith” who has a Facebook account, or jsmith@gmail.com?
    • Is John Smith, 24 Main Street, Maitland likely to be the same person as John Smith with an IP address in Maitland?
Predicted Increase in Internet Share of Advertising

Total Annual Expenditure on Advertising worldwide = $500 billion

Source: ZenithOptimedia

From Techcrunch.com, Sept 30, 2013
Trends in Spending on Internet Advertising

Internet adspend by type 2012-2015 (US$bn)

- **2012**: Total display 33.7, Classified 11.0, Paid search 40.7
- **2013**: Total display 40.2, Classified 11.5, Paid search 46.9
- **2014**: Total display 48.2, Classified 12.0, Paid search 53.6
- **2015**: Total display 57.6, Classified 12.6, Paid search 61.4

*Source: ZenithOptimedia*

From Techcrunch.com, Sept 30, 2013
Link Analysis and the PageRank Algorithm
The Web Graph

• Graph $G = (V, E)$
  – $V =$ set of all Web pages, let $n = |V|$  
  – $E =$ set of all hyperlinks

• Number of nodes?
  – Difficult to estimate, $> 10$ billion?  
  – Crawling the Web is highly non-trivial

• Number of edges?
  Graph is sparse, i.e.,
  mean number of outlinks per page is a small constant and not $O(n)$
The Web Graph

• The Web graph is inherently dynamic
  – nodes and edges are continually appearing and disappearing

• Research on general properties of the Web graph
  – What is the distribution of the number of in-links and out-links?
  – What is the distribution of number of pages per site?
    • Typically power-laws for many of these distributions
  – How far apart are 2 randomly selected pages on the Web?
    • What is the “average distance” between 2 random pages?
  – And so on...
An Aside: Social Networks

- Social networks = graphs
  - \( V \) = set of “actors” (e.g., students in a class)
  - \( E \) = set of interactions (e.g., collaborations)
  - Typically small graphs, e.g., \( n = |V| = 10 \) or 50

- Long history of social network analysis (e.g. at UCI)

- Quantitative data analysis techniques that can automatically extract “structure” or information from graphs
  - E.g., who is the most important “actor” in a network?
  - E.g., are there clusters in the network?

- Comprehensive reference:
Node Importance in Networks

• General idea is that some nodes are more important than others in terms of the structure of the graph
  – “importance” is also referred to as “centrality” in the social network literature

• In a directed graph, “in-degree” may be a useful indicator of importance
  – e.g., for a citation network among authors (or papers)
    • in-degree is the number of citations => “importance”
  – However, “in-degree” is too simple in practice in that it implicitly assumes that all edges are of equal importance
  – In the next few slides we will develop a more powerful recursive approach
Other Notions of Node Importance or Centrality

Betweenness Centrality:
The fraction of shortest-paths (between all pairs of nodes in a network) that a node is part of

In this simple example, Node 4 would clearly have a much higher betweenness centrality score compared to any of the other nodes.

Time complexity is an issue: $O(n^3)$ time in general for all-pairs shortest paths
This can be reduced to $O(n |E|)$ in sparse graphs (Brandes’ method).
Recursive Notions of Node Importance in Directed Graphs

• \( w_{ij} = \text{weight of link from node } i \text{ to node } j \)

  – assume \( \sum_j w_{ij} = 1 \) and weights are non-negative

  – e.g., default choice: \( w_{ij} = 1/\text{outdegree}(i) \)

    • more outlinks => less importance attached to each

• Define \( r_j = \text{importance of node } j \) in a directed graph \( (n = \text{number of nodes}) \)

\[
  r_j = \sum_i w_{ij} r_i \quad \text{i,j } = 1,\ldots,n
\]

• Importance of a node is a weighted sum of the importance of nodes that point to it

  – Makes intuitive sense

  – Leads to a set of recursive linear equations
Simple Example
Simple Example
### Simple Example

Weight matrix \( W \)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0.5</td>
<td>0</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Each row in \( W \) represents the set of outgoing weights and sums to 1.
Matrix-Vector form

- Recall $r_j$ = importance of node $j$

$$r_j = \sum_i w_{ij} r_i \quad i,j = 1,\ldots,n$$

e.g., $r_2 = 1 r_1 + 0 r_2 + 0.5 r_3 + 0.5 r_4$

= dot product of $r$ vector with column 2 of $W$

Let $r = n \times 1$ vector of importance values for the $n$ nodes
Let $W = n \times n$ matrix of link weights

=> we can rewrite the importance equations as

$$r = W^T r$$
Eigenvector Formulation

Reformulate our problem in matrix-vector terms

Solve the importance equations for unknown \( r \), with known \( W \)

\[
r = W^T r
\]

We recognize this as a standard eigenvalue problem, i.e.,

\[
A \ r = \lambda \ r \quad \text{(where } A = W^T)\]

with \( \lambda = \) an eigenvalue with value 1  (assuming there is such an eigenvalue)
and \( r = \) the eigenvector corresponding to \( \lambda = 1 \)
Eigenvector Formulation

Need to solve for $r$ in the following equation:

$$(W^T - \lambda \ I) \ r = 0$$

Note: $W$ is a stochastic matrix, i.e., rows are non-negative and sum to 1

Results from linear algebra tell us that:

(a) Since $W$ is a stochastic matrix, $W$ and $W^T$ have the same eigenvectors/eigenvalues

(b) The largest of these eigenvalues $\lambda$ is always 1

(c) the vector $r$ corresponds to the eigenvector corresponding to the largest eigenvector of $W$ (or $W^T$)
Solution for the Simple Example

Solving for the eigenvector of $W$ we get
$r = [0.2, 0.4, 0.133, 0.2667]$

Results are quite intuitive, e.g., 2 is “most important”
PageRank Algorithm: Applying this idea to the Web

1. Crawl the Web to get nodes (pages) and links (hyperlinks) [highly non-trivial problem!]
2. Weights from each page = 1/(# of outlinks)
3. Solve for the eigenvector $r$ (for $\lambda = 1$) of the weight matrix

Computational Problem:
- Solving an eigenvector equation scales as $O(n^3)$
- For the entire Web graph $n > 10$ billion (!!)
- So direct solution is not feasible

Can use the power method (iterative)

$$r^{(k+1)} = W^T r^{(k)}$$ for $k=1,2,.....$
Power Method for solving for \( \mathbf{r} \)

\[ \mathbf{r}^{(k+1)} = \mathbf{W}^T \mathbf{r}^{(k)} \]

Define a suitable starting vector \( \mathbf{r}^{(1)} \)
  e.g., all entries \( 1/n \), or all entries = indegree(node)/|E|, etc

Each iteration is matrix-vector multiplication \( \Rightarrow O(n^2) \)
  - problematic?
    - no: since \( \mathbf{W} \) is highly sparse each iteration is effectively \( O(n) \)
      - (Web pages have limited outdegree)

For sparse \( \mathbf{W} \), the iterations typically converge quite quickly:
  - rate of convergence depends on the “spectral gap”
    - \( \Rightarrow \) how quickly does error\((k) = (\lambda_2/\lambda_1)^k \) go to 0 as a function of \( k \) ?
    - \( \Rightarrow \) if \( |\lambda_2| \) is close to 1 (= \( \lambda_1 \)) then convergence is slow

- empirically: Web graph with 300 million pages
  - \( \Rightarrow \) 50 iterations to convergence (Brin and Page, 1998)
Illustration of Power Iteration Convergence for $W$ Example

Values of components of $r$ at each iteration

Number of Power Iterations
Basic Principles of Markov Chains

Discrete-time finite-state first-order Markov chain, K states

Transition matrix $A = K \times K$ matrix

- Entry $a_{ij} = P(\text{state}_t = j | \text{state}_{t-1} = i)$, $i, j = 1, \ldots, K$

- Rows sum to 1 (since $\sum_j P(\text{state}_t = j | \text{state}_{t-1} = i) = 1$)

- Note that $P(\text{state} | ..)$ only depends on $\text{state}_{t-1}$

$P_0 = \text{initial state probability} = P(\text{state}_0 = i)$, $i = 1, \ldots, K$
Simple Example of a Markov Chain

K = 3 states

Transition matrix A:

\[
\begin{pmatrix}
0.8 & 0.2 & 0.0 \\
0.0 & 0.9 & 0.1 \\
0.2 & 0.2 & 0.6 \\
\end{pmatrix}
\]

Initial state vector \(P_0\) : \([1/3 \ 1/3 \ 1/3]\)
Steady-State (Equilibrium) Distribution for a Markov Chain

Irreducibility:
- A Markov chain is irreducible if there is a directed path from any node to any other node

Steady-state distribution $\pi$ for an irreducible Markov chain*:
$\pi_i = \text{probability that in the long run, chain is in state } i$

From Markov chain theory, the steady state $\pi$’s are the solutions to $\pi = A^t \pi$

Note that this is exactly the same as our earlier recursive equations for node importance in a graph!

*Note: technically, for a meaningful solution to exist for $\pi$, $A$ must be both irreducible and aperiodic
Markov Chain Interpretation of PageRank

- $W$ is a stochastic matrix (rows sum to 1) by definition
  - can interpret $W$ as defining the transition probabilities in a Markov chain
  - $w_{ij} = \text{probability of transitioning from node } i \text{ to node } j$

- Markov chain interpretation:
  
  \[ r = W^T r \]

  -> these are the solutions of the steady-state probabilities for a Markov chain

  page importance $\Leftrightarrow$ steady-state Markov probabilities $\Leftrightarrow$ eigenvector
The Random Surfer Interpretation

• Recall that for the Web model, we set $w_{ij} = 1/outdegree(i)$

• Thus, in using $W$ for computing importance of Web pages, this is equivalent to a model where:
  – We have a random surfer who surfs the Web for an infinitely long time
  – At each page the surfer randomly selects an outlink to the next page
  – “importance” of a page = fraction of visits the surfer makes to that page
  – this is intuitive: pages that have better connectivity will be visited more often
Potential Problem with “Sinks” in the Web Graph

Page 1 is a “sink” (no outlink) (also known as an absorbing state)

Pages 3 and 4 are also “sinks” (no outlink from the system)

Markov chain theory tells us that no steady-state solution exists - depending on where you start you will end up at 1 or \{3, 4\}

Markov chain is “reducible”
Making the Web Graph Irreducible

- One simple solution to our problem is to modify the Markov chain:
  - With probability $\alpha$ the random surfer jumps to any random page in the system (with probability of $1/n$, conditioned on such a jump)
  - With probability $1-\alpha$ the random surfer selects an outlink (randomly from the set of available outlinks)

- The resulting transition graph is fully connected $\Rightarrow$ Markov system is irreducible $\Rightarrow$ steady-state solutions exist

- Typically $\alpha$ is chosen to be between 0.1 and 0.2 in practice

- But now the graph is dense - so we lose the nice sparsity for computation!
  However, power iterations can be written as:
  $$r^{(k+1)} = (1-\alpha) W^T r^{(k)} + (\alpha/n) 1^T$$
  - Complexity is still $O(n)$ per iteration for sparse $W$
The PageRank Algorithm


• PageRank = the method on the previous slide, applied to the entire Web graph
  – Crawl the Web
    • Store both connectivity and content
  – Calculate (off-line) the “pagerank” \( r \) for each Web page using the power iteration method

• How can this be used to answer Web queries:
  – Terms in the search query are used to limit the set of pages of possible interest
  – Pages are then ordered for the user via precomputed pageranks
  – The Google search engine combines \( r \) with text-based measures
  – This was the first demonstration that link information could be used for content-based search on the Web
Link Manipulation

Query = french military victories

Did you mean: french military defeats

No standard web pages containing all your search terms were found.

Your search - french military victories - did not match any documents.

Suggestions:
- Make sure all words are spelled correctly.
- Try different keywords.
- Try more general keywords.
- Try fewer keywords.

Also, you can try Google Answers for expert help with your search.
Conclusions

- **PageRank algorithm was the first algorithm for link-based search**
  - Many extensions and improvements since then
  - Same idea used in social networks for determining importance

- **Real-world search involves many other aspects besides PageRank**
  - E.g., use of logistic regression for ranking
    - Learns how to predict relevance of page (represented by bag of words) relative to a query, using historical click data
    - See paper by Joachims on class Web page
Web Analytics
# Web Logs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>The date that the activity occurred</td>
</tr>
<tr>
<td>Time</td>
<td>The time that the activity occurred</td>
</tr>
<tr>
<td>Client IP address</td>
<td>The IP address of the client that accessed your server</td>
</tr>
<tr>
<td>User Name</td>
<td>The name of the authenticated user who accessed your server</td>
</tr>
<tr>
<td>Server Port</td>
<td>The port number the client is connected to</td>
</tr>
<tr>
<td>Method</td>
<td>The action the client was trying to perform</td>
</tr>
<tr>
<td>URI Stem</td>
<td>The resource accessed</td>
</tr>
<tr>
<td>URI Query</td>
<td>The query, if any, the client was trying to perform</td>
</tr>
<tr>
<td>Protocol Status</td>
<td>The status of the action, in HTTP or FTP terms</td>
</tr>
<tr>
<td>Bytes Sent</td>
<td>The number of bytes sent by the server</td>
</tr>
<tr>
<td>Bytes Received</td>
<td>The number of bytes received by the server</td>
</tr>
<tr>
<td>Time Taken</td>
<td>The duration of time, in milliseconds, that the action consumed</td>
</tr>
<tr>
<td>Protocol Version</td>
<td>The protocol (HTTP, FTP) version used by the client</td>
</tr>
<tr>
<td>Host</td>
<td>Display the content of the host header</td>
</tr>
<tr>
<td>User Agent</td>
<td>The browser used on the client</td>
</tr>
<tr>
<td>Cookie</td>
<td>The content of the cookie sent or received, if any</td>
</tr>
<tr>
<td>Referrer</td>
<td>The previous site visited by the user (including search query if any)</td>
</tr>
</tbody>
</table>
## Web Logs

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>The date that the activity occurred</td>
</tr>
<tr>
<td>Time</td>
<td>The time that the activity occurred</td>
</tr>
<tr>
<td>Client IP address</td>
<td>The IP address of the client that accessed your server</td>
</tr>
<tr>
<td>User Name</td>
<td>The name of the authenticated user who accessed your server</td>
</tr>
<tr>
<td>Server Port</td>
<td>The port number the client is connected to</td>
</tr>
<tr>
<td>Method</td>
<td>The action the client was trying to perform</td>
</tr>
<tr>
<td>URI Stem</td>
<td>The resource accessed</td>
</tr>
<tr>
<td>URI Query</td>
<td>The query, if any, the client was trying to perform</td>
</tr>
<tr>
<td>Protocol Status</td>
<td>The status of the action, in HTTP or FTP terms</td>
</tr>
<tr>
<td>Bytes Sent</td>
<td>The number of bytes sent by the server</td>
</tr>
<tr>
<td>Bytes Received</td>
<td>The number of bytes received by the server</td>
</tr>
<tr>
<td>Time Taken</td>
<td>The duration of time, in milliseconds, that the action consumed</td>
</tr>
<tr>
<td>Protocol Version</td>
<td>The protocol (HTTP, FTP) version used by the client</td>
</tr>
<tr>
<td>Host</td>
<td>Display the content of the host header</td>
</tr>
<tr>
<td>User Agent</td>
<td>The browser used on the client</td>
</tr>
<tr>
<td>Cookie</td>
<td>The content of the cookie sent or received, if any</td>
</tr>
<tr>
<td>Referrer</td>
<td>The previous site visited by the user (including search query if any).</td>
</tr>
</tbody>
</table>
Web Log Analytics

• Use Web logs to generate statistics on....
  – Which pages are visited
  – Where users came from (referrer page)
  – Geographic distribution of visitors (from IP addresses)
  – How many pages were clicked
  – Actions taken by visitors (queries, purchases, etc)
  – How long people stayed on pages/site (difficult to do accurately)

• Software tools for reports and interactive dashboards
  – E.g., Google Analytics (free)

• Useful for understanding user behavior on a particular site

• Reference:
  – Web Analytics 2.0, Avinash Kaushik, Wiley, 2010
This dashboard is for Atlantis Aquariums, a fictional manufacturer of low maintenance aquariums sold through a national chain of pet stores. Atlantis doesn’t sell direct, but has a consumer-focused website for the purpose of driving offline sales of filters (their most profitable product).

The educational-focused website features many high quality educational videos. There are calls to action throughout the site for visitors’ to sign up for the opt-in email list which is used to remind aquarium owners when to replace their filters.

Atlantis has a small online marketing budget and relies heavily on organic search to drive traffic to the website, but they also have a few small PPC campaigns and some banner ads. Atlantis has recently added their videos to YouTube and is beginning to track referrals from social sites such as Facebook, MySpace and Twitter.
Descriptive Summary Statistics

• Histograms, scatter plots, time-series plots
  – Very important!
  – Helps to understand the big picture
  – Provides “marginal” context for any model-building
    • models aggregate behavior, not individuals
  – Challenging for Web log data

• Examples
  – Session lengths (e.g., power laws)
  – Click rates as a function of time, content
$L = \text{number of page requests in a single session from visitors to ICS Web site over 1 week (robots removed)}$
A time-series plot of ICS Website data

Number of page requests per hour as a function of time from page requests in the www.ics.uci.edu Web server logs
Identifying individual users from Web server logs

- Useful to associate specific page requests to specific individual users

- IP address most frequently used

- Disadvantages
  - One IP address can belong to several users
  - Dynamic allocation of IP address

- Better to use cookies (or login ID if available)
  - Information in the cookie can be accessed by the Web server to identify an individual user over time
  - Actions by the same user during different sessions can be linked together

- Another option is to enforce user registration
  - High reliability
  - But can discourage potential visitors
  - Large portals (such as Yahoo!) have high fraction of logged-in users
Sessionizing

- **Time oriented (robust)**
  - e.g., by gaps between requests
    - not more than 20 minutes between successive requests
    - this is a heuristic – but is a standard “rule” used in practice

- **Navigation oriented (good for short sessions and when timestamps unreliable)**
  - Referrer is previous page in session, or
  - Referrer is undefined but request within 10 secs, or
  - Link from previous to current page in web site
Client-side data

• Advantages of collecting data at the client side:
  – Direct recording of page requests (eliminates ‘masking’ due to caching)
  – Recording of all browser-related actions by a user (including visits to multiple websites)
  – More-reliable identification of individual users (e.g. by login ID for multiple users on a single computer)

• Preferred mode of data collection for studies of navigation behavior on the Web

• Companies like ComScore and Nielsen use client-side software to track home computer users
Example of client-side data from Alexa, each “dot” is a URL request in a browser
Summary of Issues in Web Log Analytics

- Traffic may be dominated by “robots”
  - E.g., search engine Web crawlers

- Users can be identified via
  - IP address (noisy)
  - Cookie (noisy)
  - Login (ideal) – can be linked to other data

- Ramifications
  - Data is very noisy
  - Data on each user is often quite error-prone

- Nonetheless,
  - Say 30% of user models are good
  - So 30% of users see relevant ads (and the other 70% the usual default ads)
  - Can be millions of dollars better than default ads for all 100%
CASE STUDY:
CLUSTERS OF MARKOV CHAINS FOR MODELING USER NAVIGATION ON A WEBSITE
Markov Models for Modeling User Navigation

• General approach is to use a finite-state Markov chain
  – Each state can be a specific Web page or a category of Web pages
  – If only interested in the order of visits (and not in time), each new request can be modeled as a transition of states

• Issues
  – Self-transition
  – Time-independence
Modeling Web Page Requests with Markov chain mixtures

- MSNBC Web logs (circa 2000)
  - Order of 2 million individual users per day
  - different session lengths per individual
  - difficult visualization and clustering problem

- WebCanvas
  - uses mixtures of Markov chains to cluster individuals based on their observed sequences
  - software tool: EM mixture modeling + visualization

Next few slides are based on material in:

From Web logs to sequences

128.195.36.195, -, 3/22/00, 10:35:11, W3SVC, SRVR1, 128.200.39.181, 781, 363, 875, 200, 0, GET, /top.html, -,
128.195.36.195, -, 3/22/00, 10:35:16, W3SVC, SRVR1, 128.200.39.181, 5288, 524, 414, 200, 0, POST, /spt/main.html, -,
128.195.36.101, -, 3/22/00, 16:18:50, W3SVC, SRVR1, 128.200.39.181, 60, 425, 72, 304, 0, GET, /top.html, -,
128.200.39.17, -, 3/22/00, 20:54:37, W3SVC, SRVR1, 128.200.39.181, 140, 199, 875, 200, 0, GET, /top.html, -,
128.200.39.17, -, 3/22/00, 20:56:33, W3SVC, SRVR1, 128.200.39.181, 0, 262, 72, 304, 0, GET, /top.html, -,

<table>
<thead>
<tr>
<th>User 1</th>
<th>2</th>
<th>3</th>
<th>2</th>
<th>2</th>
<th>3</th>
<th>3</th>
<th>3</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>3</th>
<th>1</th>
<th>1</th>
<th>3</th>
<th>3</th>
<th>3</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>User 2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User 3</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User 4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User 5</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Graphical Model for Markov Chains
Multiple Users...One Common Markov Chain
Multiple Users...One Chain per User
One Chain per Cluster of Users

Cadez, Meek, Heckerman, Smyth, 2003
Likelihood of a Sequence in a Markov Chain Model

\[ P(\text{sequence}) = P_0(\text{first state}) \times \prod_{ij} (a_{ij})^{n_{ij}} \]

Maximum likelihood estimation:
- given the \( n_{ij} \)'s, find the \( a_{ij} \)'s that maximize \( P(\text{sequence}) \)

Number of transitions from i to j
Transition probability from i to j
Probability Estimation in Markov Chains

\[ p(\text{state } j \mid \text{state } i) = a_{ij} = \frac{n_{ij}}{n_i} \]

\[ a_{ij} = \frac{n_{ij} + \alpha_{ij}}{n_i + \sum_m \alpha_{im}} \]

where \( \frac{\alpha_{ij}}{\sum_m \alpha_{im}} \) is the prior probability of transitioning from \( i \) to \( j \)

and where \( \sum_m \alpha_{im} \) is the strength (equivalent sample size) of the prior

Could set \( \alpha_{ij} = \alpha_j \) (same priors for transitioning out of each state) or
\( \alpha_{ij} = \alpha \) (same priors for transitioning in and out of each state)
Probability Estimation with Multiple Sequences

T sequences
Assume sequences are independent
\( n_{ijt} = \text{number of times going from } i \text{ to } j \text{ in sequence } t, \; t = 1, \ldots, T \)

\[
p(state \; j \mid state \; i) = a_{ij} = \frac{\sum_t n_{ijt}}{\sum_t n_{it}}
\]

With smoothing:
\[
a_{ij} = \frac{(\sum_t n_{ijt} + \alpha_{ij})}{(\sum_t n_{it} + \sum_m \alpha_{im})}
\]
Clusters of Markov Chains

• Assume our data is generated by K different Markov chains
  – Each chain has its own parameters $A, P_0$
  – This is a mixture of Markov chains

• T sequences – but we don’t know which sequence came from which cluster

• Chicken-and-egg problem
  – If we knew which cluster each sequence belonged to, we could group sequences by cluster, and estimation of cluster parameters is easy
  – If we knew the parameters for each Markov chain, we could figure out (by Bayes rule) the most likely cluster for each sequence
Clusters of Probabilistic State Machines

Cluster 1

Cluster 2

Motivation:
approximate the heterogeneity of Web surfing behavior
EM Algorithm for Markov Clusters

EM = expectation-maximization

E-step
– For each sequence, and given current parameter estimates for each cluster, estimate $p(\text{cluster } k \mid \text{sequence}), k = 1, \ldots, K$

M-step
– Given $p(\text{cluster } k \mid \text{sequence})$, estimate $A_k$ and $P_{k0}$ for each cluster

Algorithm:
• Start with an initial random guess at parameters or $p(\text{cluster } k \mid \text{sequence})$
• Iteration = pair of EM steps
• Halt iterations when parameters or $p(\text{cluster } k \mid \text{sequence})$ are not changing

Guaranteed to converge to a local maximum of the likelihood (under general conditions)
E Step of EM Algorithm

T sequences

\[ P(\text{cluster } k \mid \text{sequence } t) \]

proportional to \( P(\text{sequence } t \mid \text{cluster } k) P(\text{cluster } k) \)

\[ P(\text{sequence } t \mid \text{cluster } k) \]

\[ = p_{kt} = P_{k0} \text{(first state)} \times \prod_{ij} (a_{kij})^{n_{ijt}} \]

Compute these “membership” probabilities for each sequence and each cluster k.
Yields a T x K matrix of membership probabilities.
M Step of EM Algorithm

For each transition probability parameter in each cluster $k=1,..K$

$$a_{kij} = \frac{\left(\sum_t n_{ijt} p_{kt}\right)}{\left(\sum_t n_{it} p_{kt}\right)}$$

- Transition probability $i->j$ for cluster $k$
- Transitions from $i$ to $j$ in sequence $t$ are “fractionally weighted” by $p_{kt}$, probability that sequence $t$ came from cluster $k$
- Number of times in state $i$ in sequence $t$ are “fractionally weighted” by $p_{kt}$, probability that sequence $t$ came from cluster $k$

Compute $P_{k0}$ (first state) in a similar fashion

Also estimate $P(cluster\ k) = \frac{1}{T} \sum_t p_{kt}$
Time Complexity

• E-Step
  – T sequences, average length L
  – K clusters
  – Compute T x K matrix
    • Each entry takes $O(L)$ time to compute
  – $O(TKL)$ overall

• M-step
  – For each of $M^2$ transition probabilities
    • Sum over each sequence $T$
  – $O(TM^2)$
  – other parameters take less time
Experimental Methodology

• Model Training:
  – fit 2 types of models
    • mixtures of histograms (multinomials)
    • mixtures of finite state machines
  – Train on a full day’s worth of MSNBC Web data

• Model Evaluation:
  – “one-step-ahead” prediction on unseen test data
    • Test sequences from a different day of Web logs
  – compute $\log P(\text{user’s next click | previous clicks, model})$
    – Using equation on the previous slide
  – $\log P$ score:
    • Rewarded if next click was given high $P$ by the model
    • Punished if next click was given low $P$ by the model
  – negative average of $\log P$ scores ~ “predictive entropy”
    • Has a natural interpretation
    • Lower bounded by 0 bits (perfect prediction)
    • Upper bounded by $\log M$ bits, where $M$ is the number of categories
Predictive Entropy Out-of-Sample

Negative log-likelihood [bits/token]

Number of mixture components [K]

Mixtures of Multinomials

Mixtures of SFSMs
Timing Results

![Graph showing timing results with lines for different data points: N=70,000, N=110,000, N=150,000. The x-axis represents the number of mixture components [K], and the y-axis represents time [sec].]
WebCanvas

- Software tool for Web log visualization
  - uses Markov mixtures to cluster data for display
  - extensively used within Microsoft
  - also applied to non-Web data (e.g., how users navigate in Word, etc)
  - Algorithm and visualization are in SQLServer (the “sequence mining” tool)

- Model-based visualization
  - random sample of actual sequences
  - interactive tiled windows displayed for visualization
  - more effective than
    - planar graphs
    - traffic-flow movie in Microsoft Site Server v3.0
Microsoft Sequence Clustering Algorithm

The Microsoft Sequence Clustering algorithm is a sequence analysis algorithm provided by Microsoft SQL Server Analysis Services. You can use this algorithm to explore data that contains events that can be linked by following paths, or sequences. The algorithm finds the most common sequences by grouping, or clustering, sequences that are similar. The following are some examples of sequences:

- Data that describes the click paths that are created when users navigate or browse a Web site.
- Data that describes the order in which a customer adds items to a shopping cart at an online retailer.

This algorithm is similar in many ways to the Microsoft Clustering algorithm. However, instead of finding clusters of cases that contain similar attributes, the Microsoft Sequence Clustering algorithm finds clusters of cases that contain similar paths in a sequence.

**Example**

The Adventure Works Cycles Web site collects information about what pages site users visit, and about the order in which the pages are visited. Because the company provides online ordering, customers must log in to the site. This provides the company with click information for each customer profile. By using the Microsoft Sequence Clustering algorithm on this data, the company can find groups, or clusters, of customers who have similar patterns or sequences of clicks. The company can then see these clusters to analyze how users move through the Web site, to identify which pages are most closely related to the sale of a particular product, and to predict which pages are most likely to be visited next.

**How the Algorithm Works**

The Microsoft Sequence Clustering algorithm is a hybrid algorithm that combines clustering techniques with Markov chain analysis to identify clusters and their sequences. One of the hallmarks of the Microsoft Sequence Clustering algorithm is that it uses sequence data. This data typically represents a series of events or transitions between states in a dataset, such as a series of product purchases or Web clicks for a particular user. The algorithm examines all transition probabilities and measures the differences, or dissimilarities, between all the possible sequences in the dataset to determine which sequences are the best to use as inputs for clustering. After the algorithm has created the list of candidate sequences, it uses the sequence information as an input for the DM method of clustering.

For a detailed description of the implementation, see Microsoft Sequence Clustering Algorithm Technical Reference.

**Data Required for Sequence Clustering Models**

When you prepare data for use in training a sequence clustering model, you should understand the requirements for the particular algorithm, including how much data is needed, and how the data is used.

The required for a sequence clustering model are as follows:

- **A single key column**
- **A sequence column**
- **Optional nonsequence attributes**

For example, in the example cited earlier of the Adventure Works Cycles Web site, a sequence clustering model might include order information as the key column, demographics about the specific customer for each order as the nonsequence attributes, and a nested table containing the sequence in which the customer viewed the Web site as the sequence attributes.
Insights from WebCanvas for MSNBC data

- From msnbc.com site administrators....
  - significant heterogeneity of behavior
  - relatively focused activity of many users
    - typically only 1 or 2 categories of pages
  - many individuals not entering via main page
  - detected problems with the weather page
  - missing transitions (e.g., tech <=> business)
Possible Extensions of this Approach

• Adding time-dependence
  – adding time-between clicks, time of day effects

• Uncategorized Web pages
  – coupling page content with sequence models

• Modeling “switching” behaviors
  – allowing users to switch between behaviors
  – Could use a topic-style model: users = mixtures of behaviors