Decay-based Ranking for Social Application Content

Wolfgang Nejdl, Claudia Niederèe, George Papadakis

Maximilian Peters
Introduction

Social web, wikis, blogs

UGC, a lot, effortless sharing

Information overload, hard to find relevant information, heterogenous information space
Problem

- information overload
- heterogeneous information space of social apps
  - formats
  - not interconnected
- classical IR methods are not adequate
- people don‘t find relevant information easily
Idea

- present resources to users by guessing...
- guessing their likelihood to be used in the future...
- while considering their recent usage in the past
Example

<table>
<thead>
<tr>
<th>t0</th>
<th>t1</th>
<th>t2</th>
<th>...</th>
<th>tN</th>
<th>tN+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>8</td>
<td>...</td>
<td>22</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>19</td>
<td>...</td>
<td>38</td>
<td>190</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>9</td>
<td>...</td>
<td>157</td>
<td>28</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>16</td>
<td>...</td>
<td>3</td>
<td>17</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>78</td>
<td>...</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
### Example

<table>
<thead>
<tr>
<th></th>
<th>t0</th>
<th>t1</th>
<th>t2</th>
<th>...</th>
<th>tN</th>
<th>tN+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>0</td>
<td>0</td>
<td>19</td>
<td></td>
<td>38</td>
<td>190</td>
</tr>
<tr>
<td>C</td>
<td>4</td>
<td>1</td>
<td>9</td>
<td></td>
<td>157</td>
<td>28</td>
</tr>
<tr>
<td>D</td>
<td>8</td>
<td>3</td>
<td>16</td>
<td></td>
<td>3</td>
<td>17</td>
</tr>
<tr>
<td>A</td>
<td>1</td>
<td>3</td>
<td>8</td>
<td></td>
<td>22</td>
<td>3</td>
</tr>
<tr>
<td>E</td>
<td>5</td>
<td>0</td>
<td>78</td>
<td></td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Idea

rank information items after n-th transaction batch so that their average ranking position after n+1-th transaction batch is minimized
Approach

• assign a value to each information item

• take into account
  • recency of usage
  • degree of usage

• rank them according to this value
4 Parameters

1. size of transaction batch
   #transactions to trigger update of information space

2. size of sliding window
   #transactions considered in calculating an information item‘s value

3. time decay function

4. accesses/editings ratio
Time Decay Function

- calculate value based on usage history

\[ v(i, \bar{u}, n) = \sum d(i, \bar{u}_k, n) \]

- several time decay functions, distinguishable in rate of decay
  - steep rate of decay: recency
  - slow rate of decay: degree of usage
    e.g. exponential, polynomial, logarithmic
4 Experiments

- evaluate the 4 parameters

- datasets
  - L3S internal wiki, small information space
  - cms in OKKAM, big information space

<table>
<thead>
<tr>
<th></th>
<th>L3S wiki</th>
<th>OKKAM cms</th>
</tr>
</thead>
<tbody>
<tr>
<td>#days</td>
<td>367</td>
<td>539</td>
</tr>
<tr>
<td>#transactions</td>
<td>33.808</td>
<td>237.118</td>
</tr>
<tr>
<td>#accesses</td>
<td>28.848</td>
<td>224.402</td>
</tr>
<tr>
<td>#editings</td>
<td>4.960</td>
<td>12.716</td>
</tr>
<tr>
<td>#pages</td>
<td>646</td>
<td>2.097</td>
</tr>
</tbody>
</table>
Experiment #1

- effect of **transaction batch size** on performance

- result: update value of all information items with every new transaction to gain highest performance
Experiment #2

- effect of **sliding window size** on performance

- result: best performance for \( w \) between 13,000 and 14,000
**Experiment #3**

- **Effect of time decay function on performance**

- **LOG performs worst**: slow decay, only degree of usage
- **EXP = LRU**: steep decay, only recency
- **PLN performs best**: balance in recency & degree of usage
Experiment #3

- effect of **time decay function** on performance

observation:
in L3S wiki PLN performs 10% better than LRU,
in OKKAM cms it performs only 3% better than LRU
  - how come?

- my experiment:
  - compare PLN over LRU in 200 **not subsequent** usages
  - to PLN over LRU in 20x10 **subsequent** usages

- result
  - not subsequent PLN/LRU > subsequent PLN/LRU
  - L3S wiki has less subsequent transactions than OKKAM cms
Experiment #4

- effect of **a/e ratio** on performance

- scenario: if one user updates a resource in a social app, what would you expect others to do?

- others want to keep up to date, so they access it

- what would you expect to be more important for ranking an information item?
Experiment #4

- effect of **a/e ratio** on performance

- result contrast assumption: editings are not more important that accesses; though editings are followed by a lot of updates on edited resource
Conclusion

- novel method of information valuation remedying information overload in context of social applications
  - ranking of content by likelihood of being used in future
  - balance between recency and degree of usage
  - applicable to several environments due to parameters

- outperforms methods normally employed

- though not yet perfect: avg rank of 10 for small, 20-30 for big information space
Related Work

• information valuation is a branch of „Information Lifecycle Management“ (ILM)

• most related approaches quantify in a binary scale - intensively and slightly used resources
  • this approach ranks on a floating scale

• data streams use time decay models as well
  • data stream methods focus on highest stream approximation through summarization
  • this approach focuses on accuracy in ranking
Future Work

• propagate value of an information resource to its neighbouring ones to improve overall ranking

• purge redundant information accumulated over time in a wiki

• accelerate calculations in process of rapidly updating information resources’ values

• adaption to work with queries as well