PEST

Term-Propagation over Structured Data using Eigenvector Computation

Fabian Kneißl

Final Diploma Report
Advisors: Klara Weiand, Dr. Tim Furche
Supervisor: Prof. Dr. François Bry

Institute for Informatics
University of Munich

21st October 2010
Outline

Introduction

Summary of Approach

PEST Deciphered
Example Graph
Algorithm
Implementation

Evaluations

Simpsons Wiki
Simpsons User Study
Delicious
PESTP

Conclusion
Outline

Introduction

Summary of Approach
  PEST Deciphered
  Example Graph
  Algorithm
  Implementation

Evaluations
  Simpsons Wiki
  Simpsons User Study
  Delicious
  PESTP

Conclusion
Motivation

Some relevant pages are not included in Top 100 search results

- Search for *Bart* on simpsons.wikia.com
  ⇒ *Homer* is not in Top 100
- Search on Google: the same

Why?

- Only content / page itself matters for determining a match
- PageRank only modifies the ranking
- PageRank is query-independent
What is the problem?

Area of interest
Search in structured data (wikis, social networks, linked data)

Problems

• Search for a keyword on structured data only yields documents directly containing it

• Documents linked to by several relevant documents might be (more) relevant but are not regarded

• Problems become even more relevant the richer a wiki is structured
What do we want to achieve?

Goal: Fuzzy matching

- Include results that are relevant but were not regarded before
- Produce a better search result ranking
- Applicability for all kinds of keyword search problems
Outline

Introduction

Summary of Approach

- PEST Deciphered
- Example Graph
- Algorithm
- Implementation

Evaluations

- Simpsons Wiki
- Simpsons User Study
- Delicious
- PESTP

Conclusion
PEST Deciphered

- Term-Propagation
- using Eigenvector Computation
- over STructured Data
Simple Example of a Structured Data Graph

Doc. 1
(Java, 0.8), (RDF, 0.2)

Doc. 2
(Lucene, 1.0)

Tag 1
(Search, 1.0)

Tag 2
(Introduction, 1.0)
Algorithm - How It Works

1. Weighted propagation graph
   - Convert datasource into graph structure
   - Assign appropriate edge weights

2. Normalization of the adjacency matrix (term-independent)
   - Compute adjacency matrix $H$
   - Normalize each column: divide by # outgoing edges

3. Compute PEST matrix $P_\tau = (1 - \alpha)H + L_\tau$
   - Compute leap matrix $L_\tau$ (term-dependent)
   - $L_\tau$ consists of informed leap and random leap

4. Eigenvector computation
   - Apply power method to $P_\tau$
   - Resulting eigenvector $p_\tau$ gives the new term weights for the vertices in the content graph
Implementation

- Java
- Lucene (vector space model search engine)
- Data structures for the dataset:
  - DOT language input format
  - MySQL as backend
- Commandline application + Graphical user interface
- Parameters as commandline arguments and in a custom settings file
- Available at http://www.pms.ifi.lmu.de/pest
### PEST - Term-Propagation using Eigenvector Computation on Structured Data

<table>
<thead>
<tr>
<th>Pos</th>
<th>Score</th>
<th>Title</th>
<th>Pos orig</th>
<th>Changed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.234954</td>
<td>Bart Simpson</td>
<td>2</td>
<td>+1</td>
</tr>
<tr>
<td>2</td>
<td>0.418051</td>
<td>List of guest stars</td>
<td>95</td>
<td>+93</td>
</tr>
<tr>
<td>3</td>
<td>0.380922</td>
<td>Homer Simpson</td>
<td>590</td>
<td>+587</td>
</tr>
<tr>
<td>4</td>
<td>0.167855</td>
<td>List of one-time characters</td>
<td>29</td>
<td>+25</td>
</tr>
<tr>
<td>5</td>
<td>0.132021</td>
<td>List of Simpsons releases by date</td>
<td>45</td>
<td>+40</td>
</tr>
<tr>
<td>6</td>
<td>0.119343</td>
<td>Lisa Simpson</td>
<td>186</td>
<td>+180</td>
</tr>
<tr>
<td>7</td>
<td>0.114355</td>
<td>Bart Gets an F</td>
<td>3</td>
<td>-5</td>
</tr>
<tr>
<td>8</td>
<td>0.108994</td>
<td>Bart Simpson (comic book series)</td>
<td>18</td>
<td>+9</td>
</tr>
<tr>
<td>9</td>
<td>0.091129</td>
<td>Made-up words</td>
<td>4</td>
<td>-6</td>
</tr>
<tr>
<td>10</td>
<td>0.090575</td>
<td>Bart the General</td>
<td>1</td>
<td>-10</td>
</tr>
<tr>
<td>11</td>
<td>0.069909</td>
<td>List of Bart Episodes in The Simpsons</td>
<td>1112</td>
<td>+1100</td>
</tr>
<tr>
<td>12</td>
<td>0.079820</td>
<td>Springfield</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0.072093</td>
<td>Non-English versions</td>
<td>241</td>
<td>+228</td>
</tr>
<tr>
<td>14</td>
<td>0.069164</td>
<td>Marge Simpson</td>
<td>1116</td>
<td>+1102</td>
</tr>
<tr>
<td>15</td>
<td>0.056729</td>
<td>The Bart Book</td>
<td>5</td>
<td>-10</td>
</tr>
<tr>
<td>16</td>
<td>0.056239</td>
<td>Character Gallery</td>
<td>402</td>
<td>+387</td>
</tr>
<tr>
<td>17</td>
<td>0.054049</td>
<td>Springfield’s State</td>
<td>77</td>
<td>+60</td>
</tr>
<tr>
<td>18</td>
<td>0.049866</td>
<td>Charles Montgomery Burns</td>
<td>1113</td>
<td>+1095</td>
</tr>
<tr>
<td>19</td>
<td>0.047510</td>
<td>Bart Sells His Soul</td>
<td>15</td>
<td>-4</td>
</tr>
<tr>
<td>20</td>
<td>0.047324</td>
<td>Bart Gets Hit by a Car/Full Synopsis</td>
<td>6</td>
<td>-14</td>
</tr>
<tr>
<td>21</td>
<td>0.046229</td>
<td>Bart vs. Thanksgiving</td>
<td>9</td>
<td>-12</td>
</tr>
<tr>
<td>22</td>
<td>0.045939</td>
<td>Bart vs. Lisa vs. the Third Grade</td>
<td>8</td>
<td>-14</td>
</tr>
<tr>
<td>23</td>
<td>0.044260</td>
<td>Radio Bart</td>
<td>14</td>
<td>-9</td>
</tr>
<tr>
<td>24</td>
<td>0.040790</td>
<td>Simpsons Comics in the US</td>
<td>54</td>
<td>+30</td>
</tr>
<tr>
<td>25</td>
<td>0.043442</td>
<td>Bart the Genius</td>
<td>33</td>
<td>+8</td>
</tr>
<tr>
<td>26</td>
<td>0.043505</td>
<td>Bart’s Girlfriend</td>
<td>10</td>
<td>-15</td>
</tr>
<tr>
<td>27</td>
<td>0.042529</td>
<td>Maggie Simpson</td>
<td>442</td>
<td>+415</td>
</tr>
<tr>
<td>28</td>
<td>0.042941</td>
<td>Bart Star</td>
<td>21</td>
<td>-7</td>
</tr>
<tr>
<td>29</td>
<td>0.040859</td>
<td>List of Simpson Episodes by Production Code</td>
<td>311</td>
<td>+282</td>
</tr>
<tr>
<td>30</td>
<td>0.040857</td>
<td>Bart vs. Australia</td>
<td>24</td>
<td>-6</td>
</tr>
<tr>
<td>31</td>
<td>0.040589</td>
<td>Bart Gets an F/Quotes</td>
<td>11</td>
<td>-20</td>
</tr>
<tr>
<td>32</td>
<td>0.039971</td>
<td>Bart to the Future</td>
<td>16</td>
<td>-15</td>
</tr>
<tr>
<td>33</td>
<td>0.039600</td>
<td>Barting Over</td>
<td>13</td>
<td>-20</td>
</tr>
<tr>
<td>34</td>
<td>0.036503</td>
<td>Character Guide</td>
<td>42</td>
<td>+7</td>
</tr>
<tr>
<td>35</td>
<td>0.036860</td>
<td>The Simpsons: Hit and Run</td>
<td>12</td>
<td>-24</td>
</tr>
<tr>
<td>36</td>
<td>0.037740</td>
<td>Bart the General/Quotes</td>
<td>17</td>
<td>-20</td>
</tr>
</tbody>
</table>

Ranking successfully calculated
Outline

Introduction

Summary of Approach
  PEST Deciphered
  Example Graph
  Algorithm
  Implementation

Evaluations
  Simpsons Wiki
  Simpsons User Study
  Delicious
  PESTP

Conclusion
## Ranking the Simpsons

### Top 10 ranking for a search for 'Bart’

<table>
<thead>
<tr>
<th>PEST score</th>
<th>Page title</th>
<th>Wik</th>
<th>Google</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bart Simpson</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Homer Simpson</td>
<td>980</td>
<td>36</td>
</tr>
<tr>
<td>3</td>
<td>Lisa Simpson</td>
<td>281</td>
<td>181</td>
</tr>
<tr>
<td>4</td>
<td>Bart the Genius</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>Marge Simpson</td>
<td>1321</td>
<td>548</td>
</tr>
<tr>
<td>6</td>
<td>Bart Gets an F</td>
<td>19</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>Bart’s Bike</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>8</td>
<td>Maggie Simpson</td>
<td>497</td>
<td>678</td>
</tr>
<tr>
<td>9</td>
<td>Bart the General</td>
<td>11</td>
<td>20</td>
</tr>
<tr>
<td>10</td>
<td>List of Bart Episodes in The Simpsons</td>
<td>3</td>
<td>216</td>
</tr>
</tbody>
</table>

Ranks of the PEST algorithm compared to an enhanced tf-idf ranking ("Wik", similar to Wikipedia’s search) and the Google rank.
Simpsons User Study

Setup

• 20 keyword-queries
• Comparison of PEST with enhanced tf-idf
• Users decide which ranking they like better
• Small-size user study (11 participants)

Results

• Across all queries and users: PEST 67% of the time preferred
• 14 queries: PEST better with acceptance of 63 – 100% of all users
• 6 queries: weak preference for or against PEST
Simpsons User Study II - Per Query

0% 25% 50% 75% 100%

Q1 Q2 Q3 Q4 Q5 Q6 Q7 Q8 Q9 Q10 Q11 Q12 Q13 Q14 Q15 Q16 Q17 Q18 Q19 Q20
Percentage of users who preferred the \texttt{PEST}-enhanced ranking for

- less than half of the queries ("Preference for Wik rank")
- 50-70\% of the queries ("Preference for \texttt{PEST}")
- and 75\% and more of the queries ("Strong Preference for \texttt{PEST}")

⇒ Paper submitted to Information Systems Journal
Delicious as Social Network

Sample graphs

Delicious

- Social network for sharing URLs
- URLs can be tagged by users
- 8 different kinds of connecting URLs were tested
Delicious Evaluation Results

Evaluation results

- Employing web-graph links improves the search result
- “Social connections” often do not lead to a better result
- Properties for datasets suitable for PEST were gained
  - Sensible identification of data items, links and terms
  - Connectivity between data items
  - Choice and manipulation of terms
PEST for Personalization (PESTP)

- Collaboration with Frederico Durao, Aalborg University (KiWi)
- Technique
  - User preferences: propagation of terms from movies to the user, based on his edited/commented/tagged movies
  - Movie search: terms get propagated between linked movies and from a user to movies
- Evaluation on a movie database
  - Improves precision and recall over PEST and tf-idf
  - Decreased runtime performance
- Paper submitted to DASFAA 2011 (Database Systems for Advanced Applications)
Outline

Introduction

Summary of Approach
  PEST Deciphered
  Example Graph
  Algorithm
  Implementation

Evaluations
  Simpsons Wiki
  Simpsons User Study
  Delicious
  PESTP

Conclusion
Conclusion

• PEST shows an impressive improvement of search results compared to tf-idf as well as Google
• Evaluation on several real-world datasets
• Guidance to what kinds of data are specifically suited for PEST
• Already the straightforward implementation yields good performance; possible improvements described in thesis

Improvements for the future

• Tuning of edge weights
• Include ontologies into the dataset
• High performance implementation
Any Questions?

Thank you for your attention!
Backup
Generation of the **PEST** Matrix

- Transposed adjacency matrix:

<table>
<thead>
<tr>
<th></th>
<th>Doc 1</th>
<th>Doc 2</th>
<th>Tag 1</th>
<th>Tag 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doc 1</td>
<td>0</td>
<td>0.1</td>
<td>0.8</td>
<td>0</td>
</tr>
<tr>
<td>Doc 2</td>
<td>0.2</td>
<td>0</td>
<td>0</td>
<td>0.8</td>
</tr>
<tr>
<td>Tag 1</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>0.1</td>
</tr>
<tr>
<td>Tag 2</td>
<td>0</td>
<td>0.5</td>
<td>0.2</td>
<td>0</td>
</tr>
</tbody>
</table>

- Normalized adjacency matrix $H$:

<table>
<thead>
<tr>
<th></th>
<th>Doc 1</th>
<th>Doc 2</th>
<th>Tag 1</th>
<th>Tag 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doc 1</td>
<td>0</td>
<td>0.05</td>
<td>0.40</td>
<td>0</td>
</tr>
<tr>
<td>Doc 2</td>
<td>0.10</td>
<td>0</td>
<td>0</td>
<td>0.40</td>
</tr>
<tr>
<td>Tag 1</td>
<td>0.25</td>
<td>0</td>
<td>0</td>
<td>0.05</td>
</tr>
<tr>
<td>Tag 2</td>
<td>0</td>
<td>0.25</td>
<td>0.10</td>
<td>0</td>
</tr>
</tbody>
</table>

- PEST matrix $P_{Java}$:

<table>
<thead>
<tr>
<th></th>
<th>Doc 1</th>
<th>Doc 2</th>
<th>Tag 1</th>
<th>Tag 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doc 1</td>
<td>0.57</td>
<td>0.66</td>
<td>0.80</td>
<td>0.50</td>
</tr>
<tr>
<td>Doc 2</td>
<td>0.13</td>
<td>0.04</td>
<td>0.04</td>
<td>0.38</td>
</tr>
<tr>
<td>Tag 1</td>
<td>0.26</td>
<td>0.04</td>
<td>0.04</td>
<td>0.08</td>
</tr>
<tr>
<td>Tag 2</td>
<td>0.04</td>
<td>0.26</td>
<td>0.12</td>
<td>0.04</td>
</tr>
</tbody>
</table>
Computation of the PEST Matrix

\[ \mathbf{P}_\tau = (1 - \alpha) \cdot \mathbf{H} + \mathbf{L}_\tau \]

\[ \mathbf{L}_\tau = \left( P(\text{leap}|j) \cdot ((1 - \rho) \cdot l_{\tau}^{\text{inf}}(i, j) + \rho \cdot l_{\tau}^{\text{rnd}}(i, j)) \right)_{i,j} \]

\[ P(\text{leap}|j) = \alpha + (1 - \alpha) \cdot (1 - \sum_i \mathbf{H}_{i,j}) \]

\[ l_{\tau}^{\text{inf}}(i, j) = \frac{w_t(i, \tau)}{\sum_k w_t(k, \tau)} \]

\[ l_{\tau}^{\text{rnd}}(i, j) = \frac{1}{|V_d \cup V_t|} \]
Eigenvector Computation

- Resulting eigenvector after applying power method:

\[
p_{Java} = \begin{pmatrix} 0.62 \\ 0.12 \\ 0.12 \\ 0.08 \end{pmatrix}, \text{ vs. } \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}
\]

- Explanation: 
  *Java* term weight of Doc. 1 is propagated to the other nodes

⇒ **Fuzzy Matching** mission accomplished
Runtime Performance in the Simpsons Wiki

In absolute terms:
Initialization of term-independent adjacency matrix: 16 sec
Total computation for one term: 8 sec

Time spent in different steps of the algorithm

- Initialization of Adjacency Matrix: 63%
- Cloning of Adjacency Matrix: 12%
- Normalization of Adjacency Matrix: 2%
- Computation of Starting Vector: 7%
- Computation of PEST Matrix: 9%
- Eigenvector Computation: 6%
- Normalization of Search Query: 1%
Simpsons User Study - Query List

Characters
bart, homer, krusty, lisa, marge, milhouse, montgomery, ned, nelson, skinner

Places
brewery, lake, school, springfield, tavern

Other
beer, nuclear, pistol, retirement, skateboarding
### PESTP Ranking

<table>
<thead>
<tr>
<th>PEST</th>
<th>Movie title (gender)</th>
<th>Luc</th>
<th>PCH</th>
<th>Mov</th>
<th>PCH</th>
<th>PEST</th>
<th>PCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scary Movie 2 (comedy)</td>
<td>17</td>
<td>+16</td>
<td>13</td>
<td>+12</td>
<td>38</td>
<td>+37</td>
</tr>
<tr>
<td>5</td>
<td>Gremlins (comedy)</td>
<td>19</td>
<td>+14</td>
<td>15</td>
<td>+10</td>
<td>22</td>
<td>+17</td>
</tr>
<tr>
<td>25</td>
<td>Friday the 13th (horror)</td>
<td>22</td>
<td>-03</td>
<td>18</td>
<td>-07</td>
<td>20</td>
<td>-05</td>
</tr>
<tr>
<td>56</td>
<td>The Dark Knight (horror)</td>
<td>15</td>
<td>-41</td>
<td>24</td>
<td>-32</td>
<td>16</td>
<td>-41</td>
</tr>
<tr>
<td>67</td>
<td>Freddy vs. Jason (horror)</td>
<td>11</td>
<td>-56</td>
<td>40</td>
<td>-27</td>
<td>9</td>
<td>-54</td>
</tr>
</tbody>
</table>

Five movies from the PESTP ranking for query “Scary” issued by user with ID 4654. The ranking approaches are followed by the PESTP change. (PCH)