Optimizing Search Engines using Clickthrough Data

Thorsten Joachims
Presented by Botty Dimanov
Overview

1. new algorithm for ranking
2. a way to personalize search engine queries

• Data
• Method
• Experiments
Data Collection

- clickthrough
  - $(q, r, c)$ where $q \in \text{query}$, $r \in (N \times \text{links})$, $c \equiv \text{clickedLinks} \subseteq \text{domain}(r)$
- easy acquisition
- information contained
  - relative relevance
Data Collection

• clickthrough
  • \((q, r, c) - q \in \text{query}, r \in \text{ranking}, c = \text{linksClicked}(q, r)\)

• easy acquisition

• information contained
  • relative relevance
  • partial relative relevance
1. Kernel Machines
   http://svm.first.gmd.de/
2. Support Vector Machine
   http://jbolivar.freeservers.com/
3. SVM-Light Support Vector Machine
   http://ais.gmd.de/~thorsten/svm.light/
4. An Introduction to Support Vector Machines
   http://www.support-vector.net/
5. Support Vector Machine and Kernel Methods References
   http://svm.research.bell-labs.com/SVMrefs.html
6. Archives of SUPPORT-VECTOR-MACHINES@JISCMAIL.AC.UK
   http://www.jiscmail.ac.uk/lists/SUPPORT-VECTOR-MACHINES.html
7. Lucent Technologies: SVM demo applet
   http://svm.research.bell-labs.com/SVT/SVMsvt.html
8. Royal Holloway Support Vector Machine
   http://svm.dcs.rhbnc.ac.uk/
   http://www.support-vector.net/software.html
10. Lagrangian Support Vector Machine Home Page
    http://www.cs.wisc.edu/dmi/lsvm
\[ \text{link}_3 <_{r^*} \text{link}_2 \quad \text{link}_7 <_{r^*} \text{link}_2 \]
\[ \text{link}_7 <_{r^*} \text{link}_4 \]
\[ \text{link}_7 <_{r^*} \text{link}_5 \]
\[ \text{link}_7 <_{r^*} \text{link}_6 \]

\[ \text{link}_i <_{r^*} \text{link}_j \]

for all pairs \(1 \leq j < i\), with \(i \in C\) and \(j \notin C\).
unsuitable format for machine learning algorithms

\[ link_3 \preceq_r link_2 \quad link_7 \preceq_r link_2 \]
\[ link_7 \preceq_r link_4 \]
\[ link_7 \preceq_r link_5 \]
\[ link_7 \preceq_r link_6 \]
\[ link_i \preceq_r link_j \]

for all pairs \( 1 \leq j < i \), with \( i \in C \) and \( j \notin C \).
Learn Ranking

• How good is a ranking?
  • Kendall’s $\tau \equiv \tau(r_a, r_b) = \frac{P - Q}{P + Q} = \frac{1 - 2Q}{\binom{m}{2}}$
Learn Ranking

• How good is a ranking?
  • Kendall’s $\tau \equiv \tau(r_a, r_b) = \frac{P - Q}{P + Q} = \frac{1 - 2Q}{\binom{m}{2}}$
Learn Ranking

• How good is a ranking?
  • Kendall’s $\tau \equiv \tau(r_a, r_b) = \frac{P - Q}{P + Q} = \frac{1 - 2Q}{\binom{m}{2}}$
Learn Ranking

• How good is a ranking?
  • Kendall’s $\tau \equiv \tau(r_a, r_b) = \frac{P - Q}{P + Q} = \frac{1 - 2Q}{\binom{m}{2}}$
  • higher $\tau$ means higher similarity
  • appropriate measure for IR
  • learn retrieval function

$$\max \tau(f)$$

$$\tau(f) = \int \tau(r_{f(q)}, r^*)dPr(q, r^*)$$

$$\tau(f) = \frac{1}{n} \sum_{(q,r^*)} (r_{f(q)}, r^*)$$

$$Q = |y - f(x)| \sim \tau(r_{f(q)}, r^*)$$
Retrieval Function

\[(d_i, d_j) \in f_w(q) \iff \bar{w} \Phi(q, d_i) > \bar{w} \Phi(q, d_j).\]
Optimization problem

\[
\max \tau(f) = \frac{1}{n} \sum_{(q, r^*) \in \text{train}} (r_{f(q)}, r^*) = \min \#Q
\]

Kendall's $\tau \equiv \tau(r_a, r_b) = \frac{p - q}{p + q} = \frac{1 - 2q}{r_2}$

\[
\forall (d_i, d_j) \in r_i^*: \quad w\Phi(q_1, d_i) > w\Phi(q_1, d_j)
\]

... 

\[
\forall (d_i, d_j) \in r_n^*: \quad w\Phi(q_n, d_i) > w\Phi(q_n, d_j)
\]
Optimization Problem 1. (Ranking SVM)

\[
\text{minimize: } V(\bar{w}, \xi) = \frac{1}{2} \bar{w} \cdot \bar{w} + C \sum \xi_{i,j,k} \tag{12}
\]

subject to:

\[
\forall (d_i, d_j) \in r_1^*: \bar{w} \Phi(q_1, d_i) \geq \bar{w} \Phi(q_1, d_j) + 1 - \xi_{i,j,1}
\]

\[
\vdots
\]

\[
\forall (d_i, d_j) \in r_n^*: \bar{w} \Phi(q_n, d_i) \geq \bar{w} \Phi(q_n, d_j) + 1 - \xi_{i,j,n}
\]

\[
\forall i \forall j \forall k : \xi_{i,j,k} \geq 0 \tag{14}
\]

\[
\bar{w} (\Phi(q_k, d_i) - \Phi(q_k, d_j)) \geq 1 - \xi_{i,j,k},
\]
\[(d_i, d_j) \in f_{\bar{w}^*}(q) \quad \iff \quad \bar{w}^* \Phi(q, d_i) > \bar{w}^* \Phi(q, d_j)\]

\[\iff \sum \alpha_k^* \Phi(q_k, d_l) \Phi(q, d_i) > \sum \alpha_k^* \Phi(q_k, d_l) \Phi(q, d_j)\]

\[rsu(q, d_i) = \bar{w}^* \Phi(q, d_i) = \sum \alpha_k^* \Phi(q_k, d_l) \Phi(q, d_j)\]
Experiments

• Offline – SVM learn a retrieval function $f$. $\max$ Kendel’s $\tau$
• Online - $\max$ Kendel’s $\tau$ improves retrieval quality

• “Striver” - combination method
Combined Results:

1. Kernel Machines
   http://svm.first.gmd.de/
2. Support Vector Machine
   http://jbolivar.freeservers.com/
3. SVM-Light Support Vector Machine
   http://ais.gmd.de/~thorsten/svm_light/
4. An Introduction to Support Vector Machines
   http://www.support – vector.net/
5. Support Vector Machine and Kernel Methods References
   http://svm.research.bell-labs.com/SVMRefs.html
6. Archives of SUPPORT-VECTOR-MACHINES@JISCMAIL.AC.UK
   http://www.jiscmail.ac.uk/lists/SUPPORT-VECTOR–MACHINES.html
7. Lucent Technologies: SVM demo applet
   http://svm.research.bell-labs.com/STV/SVMsvt.html
8. Royal Holloway Support Vector Machine
   http://www.cs.wisc.edu/dmi/lsvm
   http://www.support – vector.net/software.html
10. Lagrangian Support Vector Machine Home Page
    http://www.cs.wisc.edu/dmi/lsvm
rank\textsubscript{X}: 100 minus rank in $X \in \{\text{Google, MSN-Search, Altavista, Hotbot, Excite}\}$ divided by 100 (minimum 0)

top1\textsubscript{X}: ranked #1 in $X \in \{\text{Google, MSN-Search, Altavista, Hotbot, Excite}\}$ (binary $\{0, 1\}$)

top10\textsubscript{X}: ranked in top 10 in $X \in \{\text{Google, MSN-Search, Altavista, Hotbot, Excite}\}$ (binary $\{0, 1\}$)

top50\textsubscript{X}: ranked in top 50 in $X \in \{\text{Google, MSN-Search, Altavista, Hotbot, Excite}\}$ (binary $\{0, 1\}$)

top1\text{count}\textsubscript{X}: ranked #1 in $X$ of the 5 search engines

top10\text{count}\textsubscript{X}: ranked in top 10 in $X$ of the 5 search engines

top50\text{count}\textsubscript{X}: ranked in top 50 in $X$ of the 5 search engines

2. Query/Content Match (3 features total):

\texttt{query.url.cosine}: cosine between URL-words and query (range [0, 1])

\texttt{query.abstract.cosine}: cosine between title-words and query (range [0, 1])

\texttt{domain.name.in.query}: query contains domain-name from URL (binary $\{0, 1\}$)

3. Popularity-Attributes ($\sim 20,000$ features total):

\texttt{url.length}: length of URL in characters divided by 30

\texttt{country.X}: country code $X$ of URL (binary attribute $\{0, 1\}$ for each country code)

\texttt{domain.X}: domain $X$ of URL (binary attribute $\{0, 1\}$ for each domain name)

\texttt{abstract.contains.home}: word "home" appears in URL or title (binary attribute $\{0, 1\}$)

\texttt{url.contains.tilde}: URL contains "\~" (binary attribute $\{0, 1\}$)

\texttt{url.X}: URL $X$ as an atom (binary attribute $\{0, 1\}$)
Offline

- Combination Google + MSN Search
- 112 queries
- ranked $d_i <_r$ random $d_j$
Online

- training - 216 queries
- evaluation - Combination (Learned + random SE)

<table>
<thead>
<tr>
<th>Comparison</th>
<th>more clicks on learned</th>
<th>less clicks on learned</th>
<th>tie (with clicks)</th>
<th>no clicks</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learned vs. Google</td>
<td>29</td>
<td>13</td>
<td>27</td>
<td>19</td>
<td>88</td>
</tr>
<tr>
<td>Learned vs. MSNSearch</td>
<td>18</td>
<td>4</td>
<td>7</td>
<td>11</td>
<td>40</td>
</tr>
<tr>
<td>Learned vs. Toprank</td>
<td>21</td>
<td>9</td>
<td>11</td>
<td>11</td>
<td>52</td>
</tr>
<tr>
<td>weight</td>
<td>feature</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.60</td>
<td>query_abstract.cosine</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.48</td>
<td>top10.google</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.24</td>
<td>query_url.cosine</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.24</td>
<td>top1count.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.24</td>
<td>top10.msnsearch</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.22</td>
<td>host.citeseer</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.21</td>
<td>domain_nec</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.19</td>
<td>top10count.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.17</td>
<td>top1.google</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.17</td>
<td>country.de</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.16</td>
<td>abstract.contains.home</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.16</td>
<td>top1.hotbot</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.14</td>
<td>domain.name.in.query</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.13</td>
<td>domain.tu-bs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.15</td>
<td>country.fi</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.16</td>
<td>top50count.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.17</td>
<td>url_length</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.32</td>
<td>top10count.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.38</td>
<td>top1count.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Features with largest and smallest weights as learned from the training data in the online experiment.
Conclusion

• Ranking SVM can successfully learn an improved retrieval function
• function automatically adapts
• Good results - taking the best of all search engines