WebWatcher

What is the WebWatcher?
Browsing the WWW is like visiting a museum
Accompanies the user as he or she browses the Web like a museum tour guide
User can communicate with the system and give feedback
Over time, learn to acquire greater expertise for previous visiting parts and interest types

Outline
- What is the WebWatcher?
- Structure of WebWatcher
- Operation of WebWatcher
- Learning in WebWatcher
- Experiment
- Related Work
- Summary and Future Research

Difference from search engine
- Search engine
  - Require specific words as keywords
  - Match keywords in the target Web page
  - Documents are not designed as hypertext
- WebWatcher
  - Learn that “machine learning” matches a hyperlink such as “neural networks”
  - Self-improve

What is the WebWatcher?
- Browsing the WWW is like visiting a museum
- Accompanies the user as he or she browses the Web like a museum tour guide
- User can communicate with the system and give feedback
- Over time, learn to acquire greater expertise for previous visiting parts and interest types

Structure of WebWatcher
- WebWatcher is implemented as a server
- Acts much like a proxy
- Return a page to the user by three modifications
  - Command list is added to the top
  - A original URL is replaced by a new URL
  - Highlight the most promising links

WebWatcher 학습을 하는 방법에 대해 나열하시오.
Continued..

Operation of WebWatcher

- Invoked by clicking on the hyperlink “The WebWatcher Tour Guide”
- Leads us to a page of our current interest
- Accompanies us from the additions
  - WebWatcher Commands: communication method
  - Highlighted hyperlinks: suggestion directions
- To end the tour, two options in the command list
  - Exit: Goal reached
  - Exit: Goal not found

Learning in WebWatcher

- LinkQuality
  - The probability that a user will select Link given the current Page and Interest
  - \[ \text{LinkQuality} : \text{Page} \times \text{Interest} \times \text{Link} \rightarrow [0,1] \]
- Three approaches to learning this target function
  - Learning from Previous Tours
  - Reinforce learning
  - Combined method

Learning from Previous Tours

- Annotating each hyperlink with the interest on previous tours.
- Compare current user’s interest with descriptions of hyperlinks
- Interests and hyperlink description are represented by very high-dimensional feature vectors
- Elements of a vector are calculated using the TFIDF heuristic
- Similarity is calculated as the cosine between vectors

Continued..
Keyword Vectoring
- Removes the suffix of the words
- Filters all commonly-used words
- Each word is weighted by TFIDF measure
- Term Frequency Inverse Document Frequency
  \[ V_i = \mathbb{1}_i \times \text{Freq}(\text{Word}_i) \times \log(n) \times \log(DocFreq(word_i)) \]
- Collection of documents is the set of all weighted keyword vectors

Learning from Hypertext Structure
- Augments a given hyperlink using words encountered in pages downstream of it
- Reinforcement Learning
  - Learns control strategies that select optimal actions in certain settings
  - The objective is to find paths through the Web which maximize the amount of relevant information

Continued...

Discounted sum of future rewards
\[ Q(s, a) = \sum_{t=0}^{\infty} \gamma^t R(s, a, s', a') + \gamma^t \max_{a'} Q(s', a') \]

Reinforcement Learning and Hypertext
- Interest "intelligent"
  - \( R_{\text{intelligent}}(s) \): the TFIDF value of "intelligent" for page \( s \)
  - \( Q_{\text{intelligent}}(s, a) \): the sum of discounted TFIDF values of "intelligent" over the optimal tour beginning with \( a \)
- WebWatcher uses a separate reward function \( R_w(s) \) and learns a distinct \( Q_w(s, a) \) for every word \( w \)
- Problem
  - WebWatcher cannot expect that users will always stick to pages it has already seen

Experiment

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>31.3%</td>
</tr>
<tr>
<td>Popularity</td>
<td>41.9%</td>
</tr>
<tr>
<td>Match</td>
<td>40.5%</td>
</tr>
<tr>
<td>Annotate</td>
<td>42.2%</td>
</tr>
<tr>
<td>RL</td>
<td>44.6%</td>
</tr>
<tr>
<td>Combine</td>
<td>48.9%</td>
</tr>
</tbody>
</table>

Random - suggests hyperlinks at random from current page
Popularity: followed most frequently in the past
Match: TFIDF cosine similarity between underlined text and user’s interest
Annotate: Learning from Previous Tours
RL: Reinforcement learning
Combine: combines above all using logistic regression
## Experiment

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>22.4%</td>
</tr>
<tr>
<td>Annotate</td>
<td>42.9%</td>
</tr>
<tr>
<td>Human</td>
<td>47.5%</td>
</tr>
</tbody>
</table>

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Human performance on the Front-Door page</td>
</tr>
</tbody>
</table>

## Related Work

- **Letizia**
- **Syskill and Webert**
- Manually constructed index page
- User can rate hyperlinks off page
- System use the ratings to learn
- **Lira**
- Works in an offline setting

## Conclusion and Future Research

- WebWatcher is a Self-improving tour guide agents
- Provide helpful advice to many users
- But there are some topics for future research
  - Personalized WebWatcher
  - Combining user-specific and Web locale-specific learning
  - Richer dialogs with users
  - New machine learning algorithms
  - Intelligent distributed hyperlinks