WebMate : A Personal Agent for Browsing and Searching

Liren Chen and Katia Sycara
Carnegie Mellon University
발표자 : 김인영

Introduction

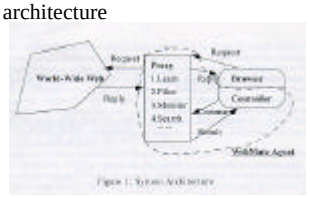
- WebMate : an agent helping users to effectively browse and search the web
  - Art
    - multiple TF-IDF vectors
    - Trigger Pair model for keyword refinement
    - Relevance feedback during the search
  - using these techniques
    - provides effective browsing and searching
    - sends to users personal newspaper

Features

- Searching enhancement
  - Parallel search, searching keywords refinement, relevant feedback
- Browsing assistant
  - Learning, Recommending, Alias, Monitoring bookmarks, More like, Sending page to friends, Prefetching hiperlinks
- Offline browsing
  - downloading following pages, getting references and printing it out
- Filtering HTTP header, cookie, block animation to speed up
- Checking html page : error finding, dead links,
- Dynamically setting up : search engines, dictionary
- Programming in java

WebMate architecture

- Learning user interests
  - a personalized newspaper
- Helping the user refine search
- system architecture



- HTTP proxy(monitoring) + applet controller(interface)

Learning profile to compile personal newspaper

- Profile Representation and Learning Algorithm
  - filtering task : judge whether relevant or not based on the user profile
  - multiple user interests : single user profile, ask explicitly
    - WebMate learns the categories automatically
    - updates the profile incrementally and continuously
      - other systems do not like this.
    - Using TF-IDF with multiple vectors representation

$$IDF(w) = \log \frac{|D|}{}$$

$$d^{(i)} \quad TF(w, d) \cdot IDF(w)$$

- Multi TF-IDF vector learning
  - N : domains of interest (category 개수) - predefined number
  - initial profile set V, |V|=0
  - M : preset number of elements of a vector
  - for each positive example (“I like it”)
    - 1. Preprocess :
      - Parse HTML page, deleting the stop words, stemming the plural noun to single form, giving more weights to title word
    - 2. Extract the TF-IDF vector for this document V_t
    - 3. IF |V| < N (|V| is the number of vectors in the profile set V),
    - then V_t
    - 4. Otherwise, calculate the cosine similarity between every two TF-IDF vectors including the vectors in the profile set V and the new document vector V_t.Assume the profile set V is $V_1, V_2, \ldots, V_N$

$$Sim(V_j, V_k) = \frac{V_j \bullet V_k}{|V_j| \times |V_k|} \quad j, k \in \{1, 2, \ldots, n, t\}$$

- 5. Combine the two vectors V_l and V_m with the greatest similarity

$$V_l = V_l + V_m \quad l, m) = \arg\max_{l,m} Sim(V_l, V_l) \quad x, y \in \{1, 2, \ldots, n\}$$

- 6. Sort the weights in the new vector V_k in decreasing order and keep the highest M elements
- this algorithm is run whenever a user marks a document as "I like it", thus the user profile is incrementally updated

- Compiling personal newspaper
  - automatically spide a list of URLs that the user wants monitored
    - parse the html page
    - extract the links of each headline
    - fetches those pages
    - constructs the TF-IDF vector for each of those pages
    - calculate the similarity with the profile
    - If the similarity is greater than some threshold, then recommend

- If the user does not provide any URLs that he would like to be the information sources
  - WebMate constructs a query using current profile
- Experiments

Date	Accuracy in top 10	Accuracy in top 20	Accuracy in whole
Sep.16	70%	60%	17/58=31%
Sep.17	40%	35%	11/42=26%
Sep.18	50%	35%	9/33=27%
Sep.19	60%	65%	18/76=24%
Sep.20	50%	40%	9/29=31%
Sep.22	40%	40%	12/40=30%
Sep.23	80%	50%	18/78=23%
Sep.24	60%	50%	10/18=56%
Average	62%	49%	30.4%

Table 1: Experiment Results

Search refinement by keywords expansion and relevance feedback

- **Trigger Pairs model** to extract relevant words
  - single keywords are ambiguous
    - "stock" has more than 10 definition in the WordNet
    - pruning : manual query expansion, semi-manual, automatic
      - manual expansion : user may not be able to provide the best refinement words
      - "Best" - most frequently co-occur with the word in its intended meaning
    - "trigger pair" (S, T)
      - If a word S is significantly correlated with another word T, then (S, T) is considered a "trigger pair", with S being the trigger and T the triggered word

- In the trigger Pairs Model (S, T) is different from (T, S), so the Trigger Pairs Model is different from the method of using co-occurrence of two words that is generally used in other keywords expansion
- Mutual information (MI) : considers the words order

$$MI(s,t) = P(s,t)\log\frac{P(s,t)}{P(s)P(t)}$$

- Broadcast News Corpus
  - set the maximum distance between S and T : 500
  - Trigger Pairs : sorted in decreasing order
    - car ← {motor, auto, model, maker, vehicle, ford, buick, honda, inventory, assembly, chevrolet, sale, nissan, incentif, pontiac, planet, toyota, dealer, chrysler}
    - music ← {musical, symphony, orchestra, composer, song, concert, tune, concerto, sound, musician, classical, album, violin, violinist, jazz, audience, conductor, play, audio, rock, cello, perform, dance}
- Wall Street Journal Corpus

- Trigger Pairs : domain specific

- **Keywords Expansion**
  - Trigger Pair method can provide several candidate refinement keywords

1. $S_1 = \{s_{11}, s_{12}, \ldots, s_{1l}\} \rightarrow A_1$, S_1 is the triggers set to A_1. $s_{11}, s_{12}, \ldots, s_{1l}$ are sorted in decreasing order of the mutual information.
   $S_2 = \{s_{21}, s_{22}, \ldots, s_{2l}\} \rightarrow A_2$, S_2 is the triggers set to A_2
   ...
   $S_m = \{s_{m1}, s_{m2}, \ldots, s_{ml}\} \rightarrow A_m$, S_m is the triggers set to A_m
2. $S = S \cup (S_1 \cap S_2, S_1 \cap S_3, \ldots, S_1 \cap S_2 \cap S_3, \ldots, S_1 \cap S_2 \cap \ldots \cap S_n)$, and $(S_1, S_2, \ldots, S_n)$ is one of the combinations of n sets out of m. The words in the S are sorted in decreasing order of mutual information.
3. If $|S| \geq N$, let the top N words in the S be the refinement words and stop.
4. otherwise, let $n = n - 1$, goto 2.

- K1 = *charge* and S1={federal, investigation, attorney, plead, indict, ……}
- K2 = fee and S2={pay, dollar, million, bank, service, tax, raise, ……}
- K={K1, K2} ={charge, fee} and $S = S_1 \cup S_2$ ={million, pay, dollar, tax, service, federal, client, …..}
  - so triggers, such as million, pay, dollar, tax, service, help confine and disambiguate the meaning of the word "charge"

Examples : keyword "stock"

{stock share}

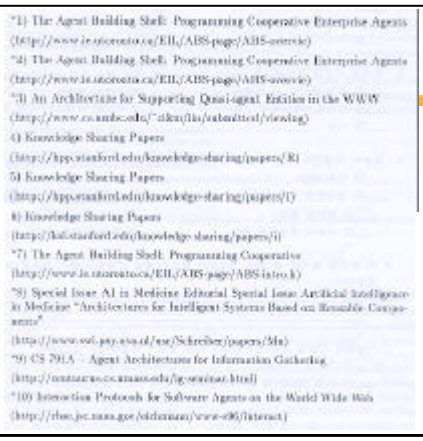
Relevance feedback

- Central problem in relevance feedback :
  - searching "features" (words, phrases) from relevant documents
  - calculating weights for these features in the context of a new query
- given a relevant page
  - first, looks for the keywords (assume K_i is one of the keywords)
  - for each keyword *K(i)* , extracts the chunk

    $W_{-5}W_{-4}W_{-3}W_{-2}W_{-1}K_iW_1W_2W_3W_4W_5$
  - then, a bag of chunks are collected and passed to the processes of deleting the stop words and calculating the frequency
  - top several frequent words are used to expand the current search keywords

For example

- suppose a user gives text as a relevant feedback to the search keywords "intelligent agent"
  - URL : "http://www.cs.cmu.edu/~softagents"
  - using our method
    - {software structure reusable architecture technology} used to expand the search "intelligent agent"




Keywords:
Background
Search
Clear
Cancel
Controller
Alias:
URL:
OK
Quit
words refinement !
Advanced search and keywords refinement
Alias
Client
Dictionary
Thesaurus
Music
Profile
Reference
Records
Setup recorder time
Immediately
Done