WordNet-based User Profiles for Semantic Personalization

G. Semeraro, M. Degemmis, P. Lops, and I. Palmisano

Dipartimento di Informatica
Università di Bari
Via E. Orabona, 4 - 70125 Bari - Italia
{semeraro,degemmis,lops,palmisano}@di.uniba.it

Abstract. Algorithms designed to support users in retrieving relevant information base their relevance computations on user profiles in which representations of the users' interests are maintained. A crucial issue is that users want to retrieve information on the basis of *conceptual content*, but words provide unreliable evidence about the content of documents. This paper explores a possible solution for this kind of problems: the adoption of supervised machine learning techniques to induce *semantic user profiles* from text documents.

1 Introduction

Current search services take a "one fits all" approach, which takes little account of the user's individual needs and preferences. Recent developments at the intersection of information retrieval, information filtering, machine learning, user modeling and natural language processing offer novel solutions for *personalized information access*. Most of this work focuses on the use of machine learning algorithms for the automated induction of a structured model of a user's interests, the *user profile*, from labeled text documents. The keyword approach to searching suffers from problems of POLYSEMY, the presence of multiple meanings for one word, and SYNONYMY, that stands for multiple words having the same meaning. The result is that, due to synonymy, relevant information can be missed if the profile does not contain the exact keywords occurring in the documents and, due to polysemy, wrong documents could be deemed as relevant. These problems call for alternative methods able to learn *semantic* profiles that capture key concepts representing users' interests from relevant documents. Semantic profiles will contain references to concepts defined in lexicons or, in a further step, ontologies. This paper shows how the content-based algorithms for learning user profiles can be extended using WordNet [1] as a reference lexicon in substituting word forms with word meanings into profiles. The paper is organized as follows: after introducing the task of learning user profiles as a text categorization problem in Section 2, in Section 3 we present the relevance feedback approach we adopted to accomplish this task. In Section 4 a strategy to represent documents and user profiles using WordNet synsets is proposed. Section 5 describes the experimental evaluation of semantic user profiles, while some conclusions are drawn in Section 6.

2 Learning User Profiles as a Text Categorization Problem

The content-based paradigm for information filtering (IF) is analog to the relevance feedback in information retrieval literature [2], which adapts the query vector by iteratively absorbing users relevance judgments on newly returned documents. In the IF paradigm, the tuned query vector is a profile model, specifying both keywords and their informative power. A new item relevance is measured by computing a similarity measure between the query vector and the items feature vector. Machine Learning (ML) techniques are used to generate a predictive model that, when given a new information item, will predict whether the new item is likely to be of interest, based on information previously labeled by the user. The ML techniques generally used are those that are well-suited for text categorization (TC) [3]. TC is the task of assigning a Boolean value to each pair $\langle d_j, c_i \rangle \in D \times C$, where D is a domain of documents and $C = \{c_1, \ldots, c_n\}$ is a set of predefined categories. A value of *True* assigned to $\langle d_j, c_i \rangle$ indicates a decision to assign c_i to d_j, while a value of *False* indicates the opposite decision. The task is to approximate the unknown target function $\Phi : D \times C \longrightarrow \{True, False\}$, that describes how documents should be classified, by means of a function $\Phi' : D \times C \longrightarrow \{True, False\}$ called the classifier or the *model* such that Φ and Φ' "coincide as much as possible". In the ML approach to TC, an inductive process automatically builds a text classifier by learning, from a set of *training documents* - documents labeled with the categories they belongs to - the features of the categories. We consider the problem of learning user profiles as a binary TC task: each document has to be classified as interesting or not with respect to the user preferences. Therefore, the set of categories is restricted to c_+, that represents the positive class (user-likes), and c_- the negative one (user-dislikes). We present a relevance feedback method able to learn profiles for content-based filtering. The accuracy of the keyword-based profiles inferred by this method will be compared with advanced semantic user profiles obtained by the same method using an indexing procedure based on WordNet.

2.1 Documents Representation

The representation that dominates the TC literature is known as *bag of words* (BOW). In this approach each feature corresponds to a single word found in the training set. In our application scenario, items to be suggested to users are movies. Each movie is represented by a set of *slots*, where each slot is a textual field corresponding to a specific feature of the movie: *title*, *cast*, *director*, *summary* and *keywords*. The text in each slot is represented using the BOW model taking into account the occurrences of words in the original text. Thus, each instance is represented by five BOWs, one for each slot. This strategy considers separately the occurrences of a word in the slots in which it appears. The idea behind this approach is that by considering the number of occurrences separately in each slot could supply a more effective way to catch the discriminatory power of a word in a document.

2.2 Related Works

Content-based systems have been used successfully in various domains.

Syskill & Webert [4] is an agent that learns a user's interests saved as a user profile used to identify interesting Web pages. The learning process is conducted by using algorithms like Bayesian classifiers, a nearest neighbor algorithm and a decision tree learner. Mooney and Roy [5] adopt a naïve Bayes text classifier in their *LIBRA* system, that makes content-based book recommendations exploiting the product descriptions obtained from the Web pages of the Amazon store. SiteIF [6] is a personal agent that exploits a sense-based representation to build a model of the user's interests as a semantic network whose nodes represent senses (not just words) of the documents requested by the user. Several methods have been proposed for integrating lexical information to training documents for text categorization. A study by Rodriguez et al. [7] used WordNet to enhance neural network learning algorithms. This approach only made use of synonymy and involved a manual word sense disambiguation step, whereas our approach uses synonymy and hypernymy and is completely automatic. Scott and Matwin [8] propose to expand each word in the training set with all the synonyms extracted from WordNet for it, including those available for each sense in order to avoid a word sense disambiguation process. This approach has shown a decrease of effectiveness in the classifier obtained, mostly due to the word ambiguity problem. Some researches have also applied WordNet to information retrieval tasks. In [9], it is proposed a retrieval strategy that adapts a classical vector space based system using synsets as indexing space instead of word forms.

3 A Relevance Feedback Method for User Profiling

In the Rocchio algorithm [10], documents are represented with the vector space representation and the major heuristic component is the TFIDF (Term Frequency/Inverse Document Frequency) word weighting scheme [2]:

$$\text{TFIDF}(t_k, d_j) = \underbrace{\text{TF}(t_k, d_j)}_{\text{TF}} \cdot \underbrace{log\frac{N}{n_i}}_{\text{IDF}} \tag{1}$$

where N is the total number of documents in the training set and n_i is the number of documents in which the term t_k appears. $TF(t_k, d_j)$ is a function that computes the frequency of the token t_k in the document d_j. Learning is achieved by combining document vectors of positive and negative examples into a prototype vector $\overrightarrow{c}$ for each class in the set of classes C. The method computes a classifier $\overrightarrow{c_i} = \langle \omega_{1i}, \ldots, \omega_{|T|i} \rangle$ for category c_i (T is the *vocabulary*, that is the set of distinct terms in the training set) by means of the formula:

$$\omega_{ki} = \beta \cdot \sum_{\{d_j \in POS_i\}} \frac{\omega_{kj}}{|POS_i|} - \gamma \cdot \sum_{\{d_j \in NEG_i\}} \frac{\omega_{kj}}{|NEG_i|} \tag{2}$$

where ω_{kj} is the $TFIDF$ weight of the term t_k in document d_j, POS_i and NEG_i are the set of positive and negative examples in the training set for the specific class, β and γ are control parameters that allow setting the relative importance of *all* positive and negative examples. To assign a class $\widetilde{c}$ to a document d_j, the similarity between each prototype vector $\overrightarrow{c_i}$ and the document vector $\overrightarrow{d_j}$ is computed and $\widetilde{c}$ will be the c_i with the highest value of similarity. We propose a method that manages documents represented using different slots. If m is the index of the slot, a movie is represented by the concatenation of five BOWs:

$$d_j = \langle w^m_{1j}, \ldots, w^m_{|T_m|j} \rangle$$

where $|T_m|$ is the cardinality of the vocabulary for the slot s_m and w^m_{kj} is the weight of the term t_k in slot s_m of the document d_j, computed as:

$$\text{TFIDF}(t_k, d_j, s_m) = \text{TF}(t_k, d_j, s_m) \cdot log \frac{N}{n_{km}} \tag{3}$$

$\text{TF}(t_k, d_j, s_m)$ is the frequency of term t_k in the document d_j in the slot s_m; the inverse document frequency of the term t_k in the slot s_m is computed as the logarithm of the ratio between the total number of documents N and the number of documents containing the term t_k in the slot s_m.

Given a user u and a set of rated movies in a specific category of interest (for example, *Comedy*), the goal is to learn a profile able to recognize movies liked by the user in that category. The learning process consists in inducing one prototype vector for *each slot*: these five vectors will represent the user profile. Each prototype vector of the profile could contribute in a different way to the calculation of the similarity between the vectors representing a movie and the vectors representing the user profile. Another key issue of our algorithm is that it learns two different profiles $\overrightarrow{p_i} = \langle \omega^m_{1i}, \ldots, \omega^m_{|T_m|i} \rangle$, for a user u and a category c_i by taking into account the ratings given by the user on documents in that category. The rating $r_{u,j}$ on the document d_j is a discrete judgment ranging from 1 to 6. It is used to compute the coordinates of the vectors in both the positive and the negative user profile:

$$\omega^m_{ki} = \sum_{\{d_j \in POS_i\}} \frac{\omega^m_{kj} \cdot r'_{u,j}}{|POS_i|} \quad (4) \qquad \omega^m_{ki} = \sum_{\{d_j \in NEG_i\}} \frac{\omega^m_{kj} \cdot r'_{u,j}}{|NEG_i|} \quad (5)$$

where $r'_{u,j}$ is the normalized value of $r_{u,j}$ ranging between 0 and 1 (respectively corresponding to $r_{u,j} = 1$ and 6), $POS_i = \{d_j \in T_r | r_{u,j} > 3\}$, $NEG_i = \{d_j \in T_r | r_{u,j} \leq 3\}$, and ω^m_{kj} is the weight of the term t_k in the document t_j in the slot s_m computed as in equation (3) where the *idf* factor is computed over POS_i or NEG_i depending on the fact that the term t_k is in the slot s_m of a movie rated as positive or negative (if the term is present in both positive and negative movies two different values for it will be computed). Computing two different idf values for a term led us to consider the rarity of a term in positive and negative movies, in an attempt to catch the informative power of a term in recognizing

interesting movies. Equations (4) and (5) differ from the classical formula in the fact that the parameters β and γ are substituted by the ratings $r'_{u,j}$ that allow to give a different weight to each document in the training set. As regards the computation of the similarity between a profile $\overrightarrow{p_i}$ and a movie $\overrightarrow{d_j}$, the idea is to compute five partial similarity values between each pair of corresponding vectors in $\overrightarrow{p_i}$ and $\overrightarrow{d_j}$. A weighted average of the five values is computed:

$$sim(\overrightarrow{d_j}, \overrightarrow{p_j}) = \sum_{s=1}^{5} sim(\overrightarrow{d_j^s}, \overrightarrow{p_j^s}) \cdot \alpha_s \tag{6}$$

where α_s reflects the importance of a slot in classifying a movie. In our experiments, we used $\alpha_1 = 0.1$ (title), $\alpha_2 = 0.15$ (director), $\alpha_3 = 0.15$ (cast), $\alpha_4 = 0.25$ (summary) and $\alpha_5(keywords) = 0.35$. The values α_s were decided according to experiments not reported in the paper due to space limitations. We considered different values for each α_s and repeated the experiments reported in section 5 using the selected values. The values reported here are those that gave the best predictive accuracy of the profiles. Since the user profile is composed by both the positive and the negative profiles, we compute two similarity values, one for each profile. The document d_j is considered as interesting only if the similarity value of the positive profile is higher than the similarity of the negative one.

4 Semantic User Profiles

We propose a novel document representation used to build *semantic user profiles* taking into account the senses of the words in the training documents. The task of disambiguation consists in determining which of the senses of an ambiguous word is invoked in a particular use of the word [11]. As for sense repository, we have adopted WordNet (version 1.7.1) [1], a large lexical database for English in which nouns, verbs, adjectives and adverbs are organized into *synsets* (*syn*onym *set*s), each representing one underlying lexical concept. Synsets are linked by different semantic relations (IS-A, PART-OF, etc...) and organized in hierarchies. The main advantage of a synset-based document representation is that synonym words belonging to the same synset can contribute to the user profile definition by referring to the same concept. Moreover, the use of a WSD procedure reduces classification errors due to ambiguous words, and consequently allows a better precision in the user model construction. We have addressed the WSD problem by proposing an algorithm based on semantic similarity between WordNet synsets. The idea behind the algorithm is that semantic similarity between synsets is inversely proportional to the semantic distance between synsets in the WordNet IS-A hierarchy [1]. The path length similarity between synsets is used by the WSD procedure to associate the appropriate synset to a polysemous word, as reported in Algorithm 1. Each document in the collection is mapped into a list of WordNet synsets following these steps:

1. each monosemous word w in a slot of a document d is mapped into the corresponding WordNet synset;

2. for each couple of words $\langle noun, noun \rangle$ or $\langle adjective, noun \rangle$ (for instance, "white house"), a search in WordNet is made in order to verify if at least one synset exists for the bigram $\langle w_1, w_2 \rangle$. In the positive case, Algorithm 1 is applied on the bigram, otherwise it is applied separately on w_1 and w_2, using all words in the slot as the context C of w;
3. each polysemous unigram w is disambiguated by algorithm 1, using all words in the slot as the context C of w.

Algorithm 1 The WordNet-based WSD algorithm

procedure WSD(w, d) ▷ find the appropriate synset of a polysemous word w in the document d; w may be also a bigram
  $C \leftarrow \{w_1, ..., w_n\}$ ▷ C is the context of w and it is defined as the window of all words that surround w with a fixed radius. For example, $C = \{w_1, w_2, w_3, w_4\}$ is a window with radius=2, if the sequence of words $\{w_1, w_2, w, w_3, w_4\}$ appears in d
  $S \leftarrow \{s_1, ...s_k\}$ ▷ S is the set of all candidate synsets for w
  $s \leftarrow null$ ▷ s is the synset to be returned
  $score \leftarrow 0$ ▷ $score$ is a similarity score assigned to s
  $T \leftarrow \emptyset$ ▷ T is the set of all candidate synsets for all words in C
  for $j \leftarrow 1, n$ **do**
    if $POS(w_j) = POS(w)$ **then** ▷ $POS(x)$ is the part-of-speech of x
      $S_j \leftarrow \{s_{j1}, ...s_{jm}\}$ ▷ S_j is the set of m possible senses for w_j
      $T \leftarrow T \cup S_j$
    end if
  end for
  for $i \leftarrow 1, k$ **do**
    for all $s_h \in T$ **do**
      $score_{ih} \leftarrow$ SinSim(s_i, s_h) ▷ computing similarity scores between s_i and every synset $s_h \in T$
      if $score_{ih} \geq score$ **then**
        $score \leftarrow score_{ih}$
        $s \leftarrow s_i$ ▷ s is the synset $s_i \in S$ with the highest similarity score with the synsets in T
      end if
    end for
  end for
  return s
end procedure

function SinSim(a, b) ▷ The similarity of the synsets a and b
  $N_p \leftarrow$the number of nodes in path p from a to b
  $D \leftarrow$maximum depth of the taxonomy ▷ In WordNet 1.7.1 $D = 16$
  $r \leftarrow -log(N_p/2D)$
  return r
end function

Algorithm 1 has been used to represent documents belonging to the EachMovie dataset according to the new model, that we call "bag-of-synsets" (BOS): the final representation of a document consists of a list of WordNet synsets recognized from the words in the document. Each slot of a document is processed separately and the occurrences of the synsets (instead of words) are computed. For example, if the words "artificial" and "intelligence" occur in the same slot of a document, in the corresponding BOW we count one occurrence for each word; in the BOS, we count only one occurrence of the synset "{05766061} $\langle noun.cognition \rangle$ ARTIFICIAL INTELLIGENCE, AI – (THE BRANCH OF COMPUTER SCIENCE THAT DEAL WITH WRITING COMPUTER PROGRAMS THAT CAN SOLVE PROBLEMS CREATIVELY)". A clear advantage of this representation regards synonyms. For example, if the words "processor" and "CPU" appear in the same slot of document, in the corresponding BOW we count *one* occurrence for each word, even if they refer to the same concept; in the BOS, we count *two* occurrences of the synset "{02888449} $\langle noun.artifact \rangle$ CENTRAL PROCESSING UNIT, CPU, C.P.U., CENTRAL PROCESSOR, PROCESSOR, MAINFRAME ". The final goal of our investigation is to compare the results of word-based and synset-based user profiles, then we do not modify the structure of the profiles and the learning mechanisms proposed in section 3. The difference with respect to word-based profiles is that synset unique identifiers are used instead of words.

5 Experimental Sessions

The goal of the experiments was to evaluate if synset-based profiles had a better performance than word-based profiles. The documents in the EachMovie dataset have been disambiguated using Algorithm 1, obtaining a reduction of the number of features ($172,296$ words vs. $107,990$ synsets, the reduction is roughly 38%). This result is mainly due to the fact that, thanks to the WSD algorithm, bigrams are represented using only one synset and synonym words are represented by the same synset.

5.1 The EachMovie Dataset

The experimental work has been carried out on a collection of $1,628$ textual descriptions of movies rated by $72,916$ real users, the EachMovie dataset[1]. The movies are rated on a 6-point scale mapped linearly to the interval [0,1]. The content information for each movie was collected from the Internet Movie Database[2] using a crawler. Appropriate preprocessing operations[3] have been applied to obtain the BOW from the original movie descriptions. Movies are categorized into different genres. For each genre or category, a set of 100 users was randomly selected among users that rated n items, $30 \leq n \leq 100$ in that movie category (only for genre 'animation', the number of users that rated n movies was 33,

[1] http://www.research.compaq.com/SRC/
[2] IMDb, http://www.imdb.com
[3] stopwords elimination and stemming.

due to the low number of movies if that genre). In this way, for each category, a dataset of at least 3000 triples (user,movie,rating) was obtained (at least 990 for 'animation'). Table 1 summarizes the data used for the experiments. The number of movies rated as positive and negative for each genre is balanced in datasets 2, 5, 7, 8 (60-65 % positive, 35-40% negative), while is slightly unbalanced in datasets 1, 9, 10 (70-75 % positive, 25-30% negative), and is strongly unbalanced in datasets 3, 4, 6 (over 75% positive).

Table 1. 10 'Genre' datasets obtained from the original EachMovie dataset.

Id Genre	Genre	Number of Movies rated	% POS	% NEG
1	Action	4,474	72.05	27.95
2	Animation	1,103	56.67	43.33
3	Art_Foreign	4,246	76.21	23.79
4	Classic	5,026	91.73	8.27
5	Comedy	4,714	63.46	36.54
6	Drama	4,880	76.24	23.76
7	Family	3,808	63.71	36.29
8	Horror	3,631	59.89	40.11
9	Romance	3,707	72.97	27.03
10	Thriller	3,709	71.94	28.06
		39,298	71.84	28.16

5.2 Experimental Setup and Results

Classification effectiveness is evaluated by the classical Information Retrieval measures *precision* and *recall*, adapted to the case of text categorization [2]. Also used is *F-measure*, a combination of precision and recall. We adopted the Normalized Distance-based Performance Measure (NDPM) [12] to measure the distance between the ranking imposed on items by the user ratings and the ranking predicted by the Rocchio method, that ranks items according to the similarity to the profile of the class *likes*. Values range from 0 (agreement) to 1 (disagreement). In all the experiments, a movie description d_i is considered as *relevant* by a user if the rating is greater or equal than 3, while the Rocchio method considers an item as relevant if the similarity score for the class *likes* is higher than the one for the class *dislikes*. We executed one experiment for each user in the dataset: the ratings of each specific user and the content of the rated movies have been used for learning the user profile and measuring its predictive accuracy, using the aforementioned measures. Each experiment consisted in:

1. selecting ratings of the user and the content of the movies rated by that user;
2. splitting the selected data into a training set Tr and a test set Ts;
3. using Tr for learning the corresponding user profile;
4. evaluating the predictive accuracy of the induced profile on Ts, using the aforementioned measures.

Table 2. Comparison between the BOW and the BOS approach.

Id Genre	Precision		Recall		F1		NDPM	
	BOW	BOS	BOW	BOS	BOW	BOS	BOW	BOS
1	0.72	0.75	0.82	0.86	0.75	0.79	0.46	0.44
2	0.65	0.64	0.66	0.66	0.64	0.63	0.34	0.38
3	0.77	0.85	0.79	0.86	0.77	0.84	0.46	0.48
4	0.92	0.94	0.94	0.96	0.93	0.94	0.45	0.43
5	0.66	0.69	0.72	0.75	0.67	0.70	0.44	0.46
6	0.78	0.79	0.84	0.87	0.80	0.81	0.45	0.45
7	0.68	0.74	0.75	0.84	0.69	0.77	0.41	0.40
8	0.64	0.69	0.74	0.82	0.67	0.73	0.42	0.44
9	0.73	0.76	0.79	0.81	0.74	0.77	0.48	0.48
10	0.74	0.75	0.85	0.84	0.77	0.78	0.45	0.44
Mean	0.73	0.76	0.78	0.83	0.74	0.78	0.44	0.44

The methodology adopted for obtaining *Tr* and *Ts* was the 10-fold cross validation [13]. The results of the comparison between the profiles obtained from documents represented using the two indexing approaches, namely BOW and BOS, are reported in Table 2. We can notice a slight improvement in precision (+3%). Going in more detail, the BOS model outperforms the BOW model on datasets 3 (+8%), 7 (+6%), 8 (+5%). This could be an indication that the improved results are independent from the distribution of positive and negative examples in the datasets: the number of movies rated as positive and negative is balanced in datasets 8, while is strongly unbalanced in datasets 3 and 7. Similar results have been observed as regards recall and F-measure (+4%). Only on dataset 2 we have not observed any improvement. This is probably due both to the low number of rated movies and to the specific features of the movies (in most cases, stories) that makes difficult the disambiguation. NDPM has not been improved, but it remains acceptable. This measure was adopted in order to compare the ranking imposed by the user ratings and the similarity score for the class c_+ (likes): further investigations will be carried out in order to define a better ranking score for computing NDPM, that takes into account the negative part of the profile as well. A Wilcoxon signed ranked test ($p < 0.05$) has been performed in order to validate the results. We considered each experiment as a single trial for the test. The test confirmed that there is a statistically significant difference in favor of the BOS model with respect to the BOS model as regards precision, recall and F-measure.

6 Conclusions and Future Work

We have presented a system that exploits a relevance feedback learning method to induce semantic user profiles from documents represented using WordNet synsets. Our hypothesis that substituting words with WordNet synsets in the indexing phase produces a more accurate document representation that could

be successfully used by learning algorithms to infer more accurate user profiles. This hypothesis is confirmed by the experimental results, since, as expected, a synset-based classification allows to prefer documents with high degree of semantic coherence, which is not guaranteed in case of a word-based classification. As a future work, we will evaluate the effectiveness of the WSD algorithm, by comparing its performance to state-of-the-art systems.

Acknowledgments

This research was partially funded by the European Commission under the 6th Framework Programme IST Integrated Project VIKEF - Virtual Information and Knowledge Environment Framework (Contract no. 507173, Priority 2.3.1.7 Semantic-based Knowledge Systems - `http://www.vikef. net`).

References

1. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press (1998)
2. Salton, G., McGill, M.: Introduction to Modern Information Retrieval. McGraw-Hill, New York (1983)
3. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys **34** (2002)
4. Pazzani, M., Billsus, D.: Learning and revising user profiles: The identification of interesting web sites. Machine Learning **27** (1997) 313–331
5. Mooney, R.J., Roy, L.: Content-based book recommending using learning for text categorization. In: Proceedings of the 5th ACM Conference on Digital Libraries, San Antonio, US, ACM Press, New York, US (2000) 195–204
6. Magnini, B., Strapparava, C.: Improving user modelling with content-based techniques. In: Proc. of 8th International Conference on User Modeling, Springer Verlag (2001) 74–83
7. Rodriguez, M.d.B., Gomez-Hidalgo, J.M., Diaz-Agudo, B.: Using wordnet to complement training information in text categorization. In: 2nd Int. Conf. on Recent Advances in NLP. (1997) 150–157
8. Scott, S., Matwin, S.: Text classification using wordnet hypernyms. In: COLING-ACL Workshop on usage of WordNet for in NLP Systems. (1998) 45–51
9. Gonzalo, J., Verdejo, F., Chugur, I., J., C.: Indexing with wordnet synsets can improve text retrieval. In: COLING-ACL Workshop on usage of WordNet for in NLP Systems. (1998)
10. Rocchio, J.: Relevance feedback information retrieval. In Salton, G., ed.: The SMART retrieval system - experiments in automated document processing, Prentice-Hall, Englewood Cliffs, NJ (1971) 313–323
11. Manning, C.D., Schutze, H.: Foundations of Statistical Natural Language Processing. The MIT Press, Cambridge, US (1984)
12. Yao, Y.Y.: Measuring retrieval effectiveness based on user preference of documents. Journal of the American Society for Information Science **46** (1995) 133–145
13. Mitchell, T.: Machine Learning. McGraw-Hill, New York (1997)