DEVELOPMENT OF A GENERIC RECOMMENDER SYSTEM

Dan Munteanu, Severin Bumbaru

"Dunărea de Jos" University of Galatz
Faculty of Computer Science
Department of Computers and Applied Informatics
111 Domneascã Street, 6200-Galatz, Romania
Phone/Fax: (+40) 36 460182; (+40) 36 461353
E-mail: dan.munteanu@ugal.ro severin.bumbaru@ugal.o

Abstract: this paper presents a recommender system for textual documents taken from web (given as bookmarks). The system uses for classification a combination of content, event and collaborative filters and for recommendation a modified Pearson-r algorithm. It uses implicit and explicit feedback for evaluating documents.

Keywords: information systems, information analysis, algorithms, machine learning, agents

1. INTRODUCTION. INFORMATION MANAGEMENT ON WEB

A popular method to keep useful information from Internet is represented by using bookmark manager from web browser. These systems have some drawbacks:
- lack of immediate portability;
- lack of visibility from different locations;
- difficult management.

It's reasonable to suppose that if anyone adds an URL to a bookmark manager, save a document from Internet, print a document is because he is interested in the information contained by that document.

Recommender systems make a recommendation for a specific object by using evaluations for that object made by other users with similar interests. Examples of such systems are Firefly (www.firefly.com) for music and MovieCritic (www.moviecritic.com) for movies. These systems ignore any information that can be extracted from the content.

This paper tries to present a recommender system that combine content filtering, collaborative filtering and agent technology. Every user has a personal agent which helps him to classify the information found on Internet and the information he had on his personal computer and also helps at recommending the documents to other users with similar interests. The agent suggests a classification of a document and extracts ratings for every document by analyzing user's actions (accept, reject, and modify agent's suggestion).

2. RECOMMENDER SYSTEMS

Recommender systems were introduced as a computer-based intelligent technique to deal with the problem of information and product overload.

The two basic entities which appear in any Recommender System are the user and the item. A user is a person who utilizes the Recommender System providing his opinion about various items and receives recommendations about new items from the system.

The input of a Recommender System depends on the type of the employed filtering algorithm. Generally, the input belongs to one of the following categories:
1. Ratings (also called votes), which express the opinion of users on items. Ratings are normally

This paper was recommended for publication by Cornelia NOVAC

provided by the user and follow a specified numerical scale (example: 1-bad to 5-excellent). A common rating scheme is the binary rating scheme, which allows only ratings of either 0 or 1. Ratings can also be gathered implicitly from the web logs, hyperlink visits, browsing habits or other types of information access patterns.

2. Demographic data, which refer to information such as the age, the gender and the education of the users. This kind of data is usually difficult to obtain. It is normally collected explicitly from the user.
3. Content data, which are based on a textual analysis of documents related to the items rated by the user. The features extracted by this analysis are used as input to the filtering algorithm in order to infer a user profile.

The goal of Recommender Systems is to generate suggestions about new items or to predict the utility of a specific item for a particular user.

Relevance can be defined for a particular user and in the context of a particular subject.

Documents and user profiles are represented using keywords vectors for comparing and learning. For a specific user, processing a lot of relevant documents correctly classified and irrelevant documents from a domain can lead to identify the relevant terms for that domain.

This system has two major components: one for classification and the other for recommendation. For classification it will use a text classification algorithm based on Rocchio's algorithm (Salton and Buckley, 1990). The difference is that the keywords used for representing the domain can be added and modified. The classifier uses relevance feedback (Douglas and Jinmook, 1998) when a document is added to the database by using implicit evaluation of the document.

For updating the classifiers (that are used in the process of classification) the system uses the information gain measure to select the most informative keywords. The keywords will be words and roots of the words that are obtained using the Porter's stemming algorithm (Porter, 1980). A text classifier contains a number of keywords (128) that are manually selected (28) and the rest are extracted from the well classified documents.

The recommendation process uses a modified *Pearson-r* algorithm (Breese, 1998), computing the correlation between users and modifying by adding the correlation between categories. The Pearson correlation coefficient was first defined in the context of the GroupLens project (Resnick *et al*., 1994) as the basis for the weights.

The goal of the system is to assist the user in the process of classifying web documents and to automatically recommend them to other user with similar interest.

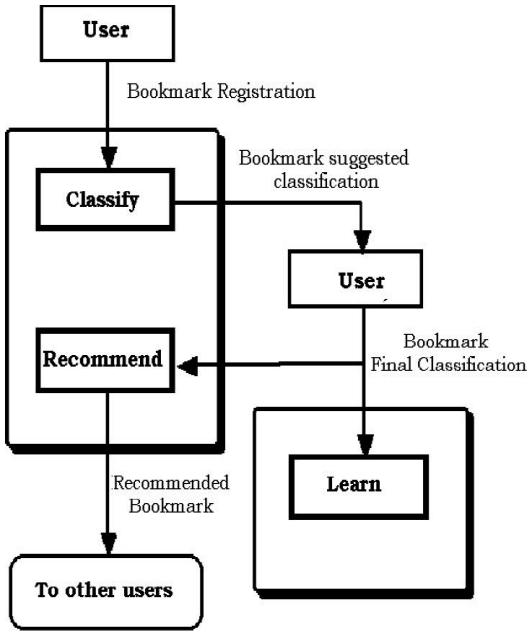


Fig. 1. System Architecture.

The system contains a database with bookmarks and references to local documents for each user and an agent that monitors the user's actions. When a document is registered, the agent suggests a classification in a category by analyzing the content of the document and user's profiles. The user can confirm the suggestion or choose another category which he considers to be better. In the meantime the agent checks to see if there are new bookmarks and recommends them to other users.

The system has a number of *n* categories to classify a document. From here the term category is considered to be similar with class, topic. In the same way document will represent web page, web document and bookmark.

In the registration process the user has to select the areas of interest. With this information an initial profile is build for every category. The agent modifies the classifier of a category when a number of *k* documents have been correctly classified in it.

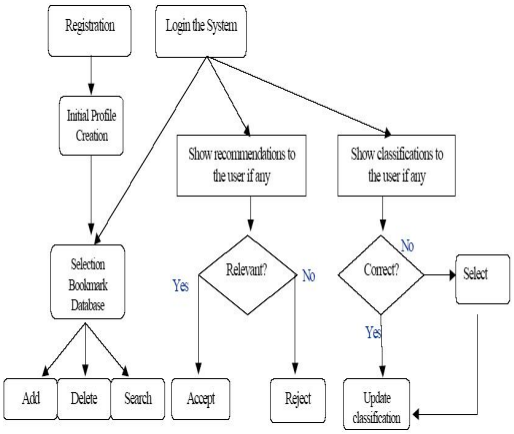


Fig. 2. System workflow

2.1 Classification process

Text classification is the automatic categorization of texts into topical categories. In this case, by using relevance feedback the topical categories are modified.

It is assumed (Sebastiani, 1999) that the categories are just symbolic labels, and no additional knowledge (of a procedural or declarative nature) of their meaning is available.

It is also assumed that no exogenous knowledge (i.e. data provided for classification purposes by an external source) is available; therefore, classification must be accomplished on the basis of endogenous knowledge only (i.e. knowledge extracted from the documents). In particular, this means that metadata such as e.g. publication date, document type, publication source, etc. is not assumed to be available.

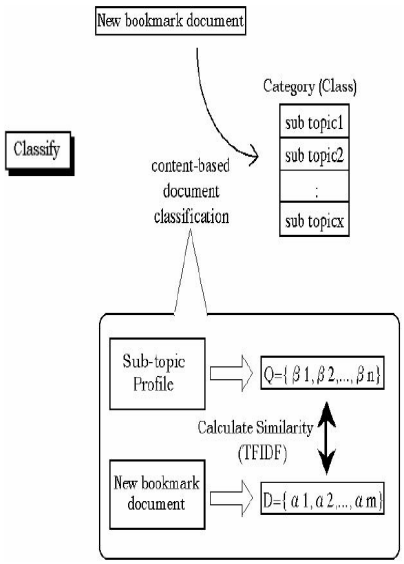


Fig. 3. Content text-based classification

2.2 The construction of text classifiers

A document belonging to user u_i is represented as a list of the most informative keywords from that document.

Positive examples for user u_i and class c_j are the documents explicitly registered and accepted by the user u_i in class c_j. Negative examples are deleted or misclassified bookmarks, or rejected recommendations which are classified in category c_j.

Notations: $C_{i,j}^{+}$ – documents classified as positive examples for user u_i and class c_j; $C_{i,j}^{-}$ – documents classified as negative examples for user u_i and class c_j.

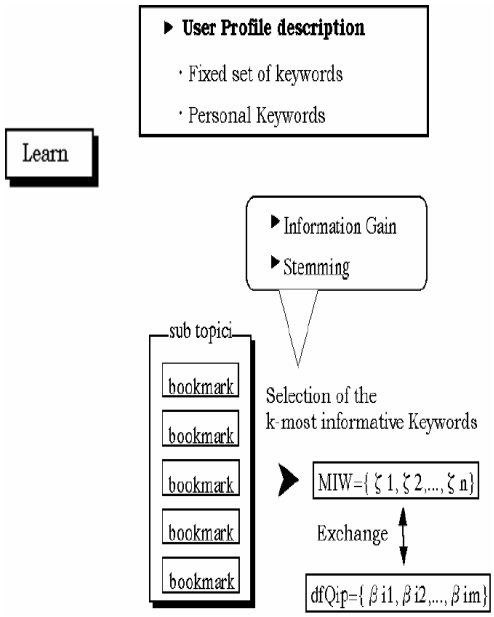


Fig. 4. Learning classifiers

For each user it is build a classifier *Q* for each category as a list of keywords (the system will contain *m* x *n* classifiers – where *m* is the numbers of users and *n* is the numbers of categories). The scope is to apply the similarity measure:

$$(1) \quad sim(Q,D) = \sum_{\tau \in Q} w(\tau,Q)w(\tau,D)$$

To compute the term weight the TF-IDF (Term Frequency – Inverse Document Frequency) algorithm it is used (Tokunaga and Iwayama 1994).

$$(2)\quad w(\tau,d)=\frac{tf_{\tau,d}\left[\log_2\left(\frac{N}{df_\tau}\right)+1\right]}{\sqrt{\sum_i\left(tf_{\tau_i,d}\left[\log_2\left(\frac{N}{df_{\tau_i}}\right)+1\right]\right)^2}}$$

Where $tf_{\tau,d}$ is term frequency of τ in document *d*,

$$(3)\quad idf_\tau = \log_2(N/dt_\tau)+1$$

is the inverse document frequency, *N* is the total number of documents, dt_τ is the number of appearances of term τ in collection. For each user a separate collection will be kept.

2.3 Relevance Feedback

Let

$$(4)\quad P=\{\tau_1,\tau_2,...,\tau_r\}$$

be the set of terms used for updating the classifiers.

The classifier and the document are represented by numeric vectors that contain the frequency of each term from P in them.

$$(5)\quad Tf(Q)=<tf_{\tau_1,Q},f_{\tau_1,Q},...,f_{\tau_r,Q}>$$

and

$$(6)\quad Tf(D)=<tf_{\tau_1,D},f_{\tau_1,D},...,f_{\tau_r,D}>$$

Relevance feedback can be described mathematically:

$$(7)\quad Tf(Q^{i+1})=Tf(Q^i)+\alpha Tf(D)$$

In which $\alpha=1$, if $D\in C_{i,j}^+$ and $\alpha=-1$, if $D\in C_{i,j}^-$

Then compute Q^{i+1} using values from $Tf(Q^{i+1})$.

The problem is that the dimension of vectors *Q* and *D* cannot be changed.

An algorithm will be used, to build the classifier *Q* step by step.

Notations:

$$(8)\quad P_{i,j}^{(t)+}=\{\tau_1,\tau_2,...,\tau_r\}$$

is a set of unique terms, relevant for class c_i until the time moment *t*; $P_{i,j}^{(t)-}$ a subset of $P_{i,j}^{(t)+}$ in which every element is found in the set of negative examples for class c_i.

The algorithm for constructing Q^+ is the following

At the moment *t*+1

if ($P_{i,j}^{(t+1)+}=P_{i,j}^{(t)+}$)

then

$Tf(Q_{i,j}^{(t+1)+})=Tf(Q_{i,j}^{(t)+})+Tf(D_{i,j}^{(t)+})$

modify $Q_{i,j}^{(t+1)+}$ using

$Tf\ (Q_{i,j}^{(t+1)+})$

else

for each $\tau\in P_{i,j}^{(t+1)+}-P_{i,j}^{(t)+}$

compute $w(\tau,d)$ for the most recent *n* documents $\in C_{i,j}^+$ and modify these values in $Q_{i,j}^{(t+1)+}$

Where *n* is the number of documents used for updating.

The algorithm it is applied in the same way for the negative classifier Q^-.

The similarity between class c_j and document *D* is:

$$(9)\quad sim_i^j(c_j,D)=(Q_{i,j}^{(t)+}\cdot D_{i,j}^{(t)})-(Q_{i,j}^{(t)-}\cdot D_{i,j}^{(t)})$$

This equation tells that a document is similar to a class if it is similar with the positive classifier and is not similar with the negative classifier. The category with the highest score for similarity will be chosen.

2.4 Selection of terms for updating the classifier

Information Gain method is used to select the most informative terms from the documents collection.

$$(10)\quad E_{i,j}(\tau,S)=I(S)-[P(\tau=prezent)I(S_{\tau=prezent})+P(\tau=absent)I(S_{w=absent})]$$

$$(11)\quad I_{i,j}(S)=\sum_{c\in\{C_{i,j}^+,C_{i,j}^-\}}-P(S_c)\log_2(p(S_c))$$

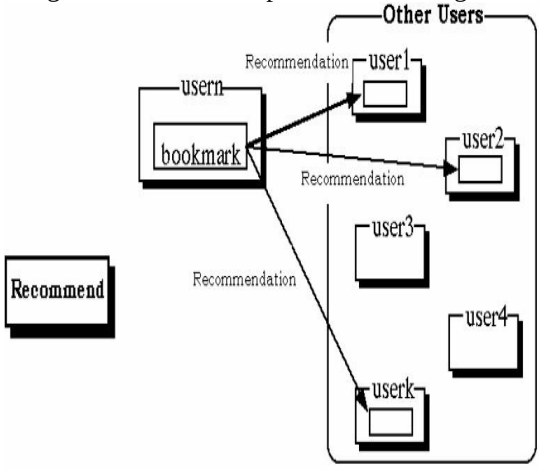
Where $P(\tau = prezent)$ is the probability that τ to be present in a document $S_{\tau = prezent}$ is the set of documents that contains at least one appearance of τ and S_c are the documents that belong to the class *c*.

The agent finds first *k* most informative terms from the set S of the last *n* classified documents. Pazzani in the Syskill & Webbert project (Pazzani *et al*., 1994) have proposed $k = 128$ and $n = 3$. The classifier contains 128 terms, from which 28 are fixed. For the rest of the terms the next method is used to add/delete terms in the positive classifier:

1. stemming algorithm is applied to extract stems(roots) of words.
2. the terms that are in the classifier and not in the list of the most informative words are replaced
3. the weights of added/replaced terms are updated (using *n* processed documents)

2.5 Recommender process

The agent constructs user-category matrix which will be used in the process of recommendation. The user-category matrix (M_{mxn}, *m* number of users and *n* number of categories) is constructed automatically counting for each user when a document is classified correctly in a class. This matrix is initialized with the categories chosen in the process of user registration.



	user1	user2	...	userk
sub topic1	7	7	...	3
sub topic2	4	3	...	2
sub topic3	1	3	...	3
.	.	.	.	.
sub topicx	2	5	...	7

Fig. 5. Recommendation process

The selection of the users who will receive the recommendations the correlation between user *n* and the users from 1 to *k* must be computed. This way for every category it is computed the number of correctly classified documents. Using these values the correlation between users can be obtained.

	user 1	user 2	...	user k
category 1	7	7		3
category 2	4	3		2
category 3	1	3		3
...				
category x	2	5		7

Fig. 6. User-category matrix

User-category matrix is used to compute the correlation between user u_x and the rest of the users using *Pearson-r* algorithm and the users with the highest correlation are selected for recommendation.

$$corel\ (u_x, u_r) = \frac{\sum_{i=1}^{n} (u_{x,i} - \bar{u}_{x,i})(u_{r,i} - \bar{u}_{r,i})}{\sqrt{\sum_{i=1}^{n} (u_{x,i} - \bar{u}_{x,i})^2 \sum_{i=1}^{n} (u_{r,i} - \bar{u}_{r,i})^2}} \quad (12)$$

in which

$$\bar{u}_{j,i} = \frac{\sum_{i=1}^{n} u_{j,i}}{n}; j \in \{x, r\} \quad (13)$$

The problem with the above equation is that does not take into account the relation between categories. The agent may recommend a document to some users just because they are correlated with the initial user not because they are interested in the subject of the document and this is not a good recommendation. That's why the agent will increase the weight of the correlation between users interested in categories correlated with the class of the document. It is calculated the similarity between two classes for the user u_i, at the moment:

$$rel_i^t(c_m, c_n) = \frac{2 \times \left| P_{i,m}^{(t)+} \cap P_{i,n}^{(t)+} \right|}{\left| P_{i,m}^{(t)+} + P_{i,n}^{(t)+} \right|} \quad (14)$$

in which $|A \cap B|$ is the number of common terms and $|A|$ is the number of terms from A.

Given class c_j of the document, class similarity vector is:

$$(15)\ \vec{R}_j = < rel_i^t(c_j, c_1), rel_i^t(c_j, c_2), ..., rel_i^t(c_j, c_n) >$$

where *n* is the number of classes.

This vector is multiplied by the user-category matrix and the result is a weighted user-category matrix.

$$(16)\quad WM = \vec{R}_j \times M$$

Using this new matrix it is computed the weight between user u_x (which recommends) and other users u_i (which may receive recommendation) as the correlation between them.

$$(17)\quad weight(u_x, u_i) = correl(u_x, u_i)$$

To decide to whom to recommend a threshold value of 0.5 is used.

The agent also checks if the document isn't already in the database so the multiple recommendation of the same document to be avoided.

3. CONCLUSIONS AND FUTURE WORK

This paper has presented a recommender system for textual documents from web (given as bookmarks) and for local documents. The system uses for classification a combination of content, event and collaborative filters and for recommendation a modified Pearson-r algorithm.

In the future the algorithm shall be modified for updating categories (the user to be able to create a new category which belongs to him and the category to be represented by a variable number or terms – not fixed number like it is now).

This system is efficient only if there are many users that use the system and if each user bring a lot of documents. Another thing that it should be useful is to make the personal agent to find new documents (by searching the web – in this case it will become a web searching agent) and to act in the name of its user (to accept/reject recommendation, to classify documents and so on).

4. REFERENCES

Blum, A. (1996). *On-line Algorithms in Machine Learning (a survey)*. In: *Dagstuhl workshop on On-Line algorithms*.

Breese, J., D. Heckerman and C. Kadie, (1998). *Empirical Analysis of Predictive Algorithms for Collaborative Filtering*. In: *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*.

Douglas W. O. and K. Jinmook (1998). *Implicit Feedback for Recommender Systems*. Digital Library Research Group, College of Library and Information Services, University of Maryland

Pazzani M., J. Muramatsu and D. Billsus (1996). *Syskill & Webert: Identifying interesting websites*. In: *Proceedings of the American National Conference on Artificial Intelligence (AAAI'96), Portland, OR.*

Porter, M.F. (1980). *An Algorithm For Suffix Stripping In*. In: *Program 14 (3),* pp. 130-137.

Resnick, P., N. Iacovou, M. Sushak, P. Bergstrom and J. Riedl (1994). *GroupLens: An Open Architecture for Collaborative Filtering of Netnews*. In the *Proceedings of the CSCW 1994 conference*.

Salton, G. and C. Buckley (1990). *Improving retrieval performance by relevance feedback*. In *Journal of the American Society for Information Science* Vol. 41, pp. 288-297

Sebastiani, F. (1998) *A Tutorial on Automated Text Categorisation, Istituto di Elaborazione dell'Informazione*, Consiglio Nazionale delle Ricerche1

Tokunaga, T. and M. Iwayama (1994). *Text categorization based on weighted inverse document frequency*. In *Technical Report 94-TR0001*, Department of Computer Science, Tokyo Institute of Technology