Extraction of User Preferences from a Few Positive Documents

Byeong Man Kim, Qing Li
Dept. of Computer Sciences
Kumoh National Institute of Technology
Kumi, kyungpook, 730-701,South Korea
(Bmkim, liqing)@se.Kumoh.ac.kr

Jong-Wan Kim
School of Computer & Information
Taegu University
Kyungsan-City, Kyungpook, South Korea
jwkim@biho.taegu.ac.kr

Abstract

In this work, we propose a new method for extracting user preferences from a few documents that might interest users. For this end, we first extract candidate terms and choose a number of terms called initial representative keywords (IRKs) from them through fuzzy inference. Then, by expanding IRKs and reweighting them using term co-occurrence similarity, the final representative keywords are extracted. Performance of our approach is heavily influenced by effectiveness of selection method for IRKs so we choose fuzzy inference because it is more effective in handling the uncertainty inherent in selecting representative keywords of documents. The problem addressed in this paper can be viewed as the one of finding a representative vector of documents in the linear text classification literature. So, to show the usefulness of our approach, we compare it with two famous methods - Rocchio and Widrow-Hoff - on the Reuters-21578 collection. The results show that our approach outperforms the other approaches.

1 Introduction

Agent technology is able to provide increasingly more services for individuals, groups, and organizations. Agents, which have been developed for Internet, have addressed many tasks such as information finding, filtering and presentation, contract negotiation, and electronic commerce (Soltysiak and Crabtree, 2000). Most of them rely on the knowledge of the user. The inclusion of user information becomes a key area.

A user model that represents some aspects of a user's information needs or preferences can be useful in any information system design, and in the case of information filtering (Kim et al., 2000). User models can be constructed by hand, or learned automatically based on feedback provided by the users. Some systems require users to explicitly specify their profiles, often as a set of keywords or categories. But it is difficult for a user to exactly and correctly specify their information needs. The machine learning techniques offer the potential to automatic construction and continuous refinement of user model.

The research systems adopting the machine learning techniques have been applied feedback techniques that explicitly provide relevance judgments on documents. Studies have shown that such explicit feedback from the user is clearly useful (Goldberg, 1992; Yan and Garcia-Molina, 1995), but, in practice, many users are unwilling to provide relevance judgments on documents (Pazzani, M., Billsus, 1997; Baeza-Yates and Ribeiro-Neto, 1999) . Users may have problems to decide about some documents. An alternative is to use implicit feedback where document relevance is inferred from user's behavior, which has received increased attention in recent years (Nichols, 1997; Konstan et al., 1997; Kim, 2000)

This paper focuses upon the extraction of user preferences from a few documents that might interest a user. It does not consider how to provide relevance judgment on documents, i.e. it assumes

This work was supported by grant No. 2000-1-51200-008-2 from the Korea Science & Engineering Foundation

that relevant documents are given explicitly or implicitly. Our approach is based on the vector space model (Baeza-Yates and Ribeiro-Neto, 1999), where text-based documents are represented as vectors of term weights. So, the problem addressed in this paper is how to extract representative keywords from documents provided by a user and what weights should be assigned to these keywords. We present a new technique to solve this problem.

The proposed method is composed of two parts, one is to select initial representative keywords (IRKs) and the other is to automatically expand and reweight IRKs. For the first part, we can consider feature selection methods (Yang and Pedersen, 1997) that focus on performance improvement and dimensionality reduction of document classifiers for a huge amount of documents covering various categories. However, since this kind of methods select features using information of other categories and negative document sets as well as positive ones, it is impossible to apply these to the target problem in this paper that extract feature keywords from only few positive documents in the same category. As alternatives, we can consider the Rocchio algorithm and Widrow-Hoff algorithm used as a training algorithm for linear text classifier since these algorithms can extract keywords and assign weights to them effectively with only positive document sets. However, here, a new technique that adopts fuzzy inference to extract or generate IRKs from a few example documents (the set of documents judged relevant by the users) is suggested since the existing algorithms did not show good results as we expected.

For the second part, we can choose one of query term expansion and term weight modification methods based on vector model (Xu and Croft , 1996; Mitra et al.,1998; Baeza-Yates and Ribeiro-Neto, 1999). Instead, we take a new approach where the term co-occurrence similarity is introduced as a measure of similarity between the distributions within the feedbacked documents of a given term and the initial query. With this similarity and the document frequency in feedbacked documents, the weight of the term in the new query was calculated.

In the next section, Rocchio and Widrow-Hoff algorithms are reviewed. Section 3 presents a method for user's preference extraction. The experiments to test the proposed method will be outlined in Section 4. Finally, conclusion is followed.

2 Background

To extract a user's preference from example documents is the same problem as finding their representative vector in linear text classifiers. A variety of algorithms for training linear classifiers have been suggested. Among them, here, we only review two widely used algorithms, Rocchio algorithm and Widrow-Hoff algorithm, for comparing with our method.

The Rocchio algorithm (David et al., 1996) is a batch algorithm. So, it produces a new weight vector w from an existing weight vector w_{old} by analyzing the entire set of training data at once. The $j'th$ component of w is :

$$w_j = \alpha w_{old,j} + \beta \frac{\sum_{i \in C} x_{i,j}}{n_c} - \gamma \frac{\sum_{i \notin C} x_{i,j}}{n - n_c} \quad (1)$$

where, $x_{i,j}$ means $j'th$ component of $i'th$ document vector x_i and n is the number of training documents. C is the set of positive training documents, and n_c is the number of positive training documents. The parameter α, β and γ control the relative impact of the original weight vector, the positive examples, and the negative examples, respectively. However, in our experiments, α = 0, β =1, and γ = 0 because only positive examples are given in our application. Neither original weight vector nor negative examples is given.

The Widrow-Hoff algorithm (David et al., 1996) is an online algorithm where one training example is presented at a time. It updates its current weight vector based on the example and then discards the example, retaining only the new weight vector. A new weight vector w_{i+1} is computed from an old weight vector w_i and a training document x_i with class label y_i. The class label y_i is 1 if a training document x_i is in the set of positive or relevant training documents, otherwise 0. In our application, y_i is always 1 because we deal with only positive examples. The initial weight vector w_1 is typically set to zero vector, w_1 = (0, ... 0).

$$w_{i+1,j} = w_{i,j} - 2\eta(w_i \bullet x_i - y_i)x_{i,j} \quad (2)$$

where, η is the learning rate which controls how quickly the weight vector w is allowed to change and $w_i \bullet x_i$ is the cosine value of the two vectors.

3 Extraction of user preferences

User preferences are extracted from a few example documents through two steps: a) the first step generates a set of keywords called IRKs (Initial Representative Keywords) which corresponds to the initial user query in the relevance feedback techniques of IR and b) these IRKs are expanded and reweighted by a relevance feedback technique.

It is very important to select IRKs reflecting user's preferences well from example or training documents (set of documents judged relevant by the user) because we have to calculate term co-occurrences similarity between these IRKs and candidate terms within each example document. Three factors of a term (term frequency, document frequency within positive examples, and IDF) are used to calculate the importance of a specific term. Since these factors essentially have inexact and uncertain characteristics, we combine them by fuzzy inference instead of a simple equation.

The IRKs are selected based on the selection criteria that each example document has at least one or more IRKs. After selecting the IRKs, we perform term modification process based on the term co-occurrence similarity between these IRKs and candidate terms. The Rocchio and Widrow-Hoff algorithms do not consider the term co-occurrence relationship within training documents. But, we regard the term co-occurrence relationship as the key factor to calculate the importance of terms under the assumption that the IRKs reflect user's preferences well.

3.1 Calculation of the Representativeness of Terms through Fuzzy Inference

The given positive examples are transformed into the set of candidate terms through eliminating stopwords and stemming by Porter's algorithm. The TF, DF, and IDF of each term are calculated based on this set and used as inputs of fuzzy inference. From now on, we will explain these three input variables. The TF (Term Frequency) is the term frequency of a specific term not in a document but in a set of documents, which is calculated by dividing total occurrences of the term in a set of documents by the number of documents in the set containing the term. It needs to be normalized for being used in fuzzy inference. The following shows the normalized term frequency (NTF).

$$NTF_i = \frac{\frac{TF_i}{DF_i}}{\max_j \left[\frac{TF_j}{DF_j} \right]} \tag{3}$$

where, TF_i is the frequency of term t_i in the example documents, DF_i is the number of documents having term t_i in the example document, $Max_j[x_j]$ means the maximum value of variable x_j.

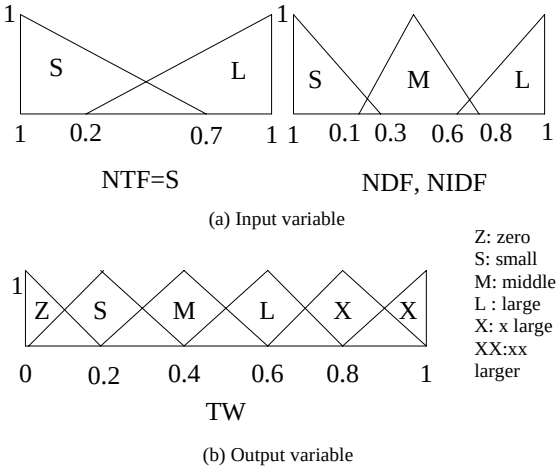
The DF (Document Frequency) represents the frequency of documents having a specific term within the example documents. The normalized document frequency, NDF, is defined in equation (4), where DF_i is the number of documents having term t_i in the example documents.

$$NDF_i = \frac{DF_i}{\max_j DF_j} \tag{4}$$

The IDF (Inverse Document Frequency) represents the inverse document frequency of a specific term over an entire document collection not example documents. The normalized inverse document frequency, NIDF, is defined as follows:

$$NIDF_i = \frac{IDF_i}{\max_j IDF_j}, \quad IDF_i = \log \frac{N}{n_i} \tag{5}$$

where, N is the total number of documents and n_i is the number of documents containing term t_i.



(a) Input variable

(b) Output variable

Figure 1. Fuzzy input/output variables

Figure 1 shows the membership functions of the input/output variables - 3 inputs (NTF, NDF, NIDF) and 1 output (TW) - used in our method. As you can see in Figure 1(a), NTF variable has { S(Small), L(Large) }, and NDF and NIDF variables have { S(Small), M(Middle), L(Large) } as linguistic labels (or terms). The fuzzy output variable, TW (Term Weight) which represents the importance of a term, has six linguistic labels as shown in Figure 1(b).

The 18 fuzzy rules are involved to infer the term weight (TW). The rules are constructed based on the intuition that the important or representative terms may occur across many positive example documents but not in general documents, i.e., their NDF and NIDF are very high. As shown in Table 1, the TW of a term is Z in most cases regardless of its NDF and NTF if its NIDF is S, because such term may occur frequently in any document and thus its NDF and NTF can be high. When NDF of a term is high and its NIDF is also high, the term is considered as a representative keyword and then the output value is between X and XX. The other rules were set similarly.

Table 1. Fuzzy inference rules

NIDF / NDF	S	M	L
S	Z	Z	S
M	Z	M	L
L	S	L	X

NTF = S

NIDF / NDF	S	M	L
S	Z	S	M
M	Z	L	X
L	S	X	XX

NTF = L

We can get the term weight TW through the following procedure. But, the output is in the form of fuzzy set and thus has to be converted to the crisp value. In this paper, the center of gravity method is used to defuzzify the output (Lee, 1990).

- Apply the NTF, NDF, and NIDF fuzzy values to the antecedent portions of 18 fuzzy rules.
- Find the minimum value among the membership degrees of three input fuzzy values.
- Classify every 18 membership degree into 6 groups according to the fuzzy output variable TW.
- Calculate the maximum output value for each group and then generate 6 output values.

3.2 Selection of Initial Representative Keywords

After calculation of the term weights of candidate terms through fuzzy inference, some candidate terms are selected as IRKs based on their weights with the constraint that each example document should contain at least one or more IRKs. The algorithm for selection of IRKs is given in Figure 2. Let us consider the following example to understand our selection procedure.

i) An example document set, DS, is composed of documents d1, d2, d3, d4, d5, and d6. Each document contains the following terms:
d1 = {a, b, f}, d2 = {a, c, d}, d3 = {d, e, f},
d4 = {d, f}, d5 = {b, c, e}, d6 = {e, f}

ii) A candidate term set, TS, is composed of {(a, 0.9), (b, 0.8), (c, 0.7), (d, 0.6), (e, 0.5), (f, 0.4)}, where (t_i, TW_i) represents that TW_i is the term weight of term t_i.

If we apply the algorithm in Figure 2 to this example, then temporary variables in line 2, 3 and 4 are initialized. The statement block from line 5 to line 14 is executed repeatedly until at least one or more IRKs are extracted from every example document in DS. Let us assume that the documents in the example document set are processed in sequence. After the first loop of the statement block from line 5 to line 14 is executed, the output value of ITS contains only term “a”. There is no change in ITS after the second loop of the block because term “a” has already been included in ITS. After d3, the third loop of the block, is processed, a term “d” is newly added to ITS. So, there is {a, d} in ITS. After d4, d5, and d6 are sequentially processed, none, term “b”, and term “e” are added to ITS, respectively. Therefore the algorithm return ITS having a set of terms {a, b, d, e}. We can find the algorithm in Figure 2 works well according to our constraint.

```
Input: DS (Example Documents Set)
       TS (Candidate Terms Set)
 1] Procedure get_ITS(DS, TS)
 2] ITS: Initial Representative Terms Set, initialized to empty.
 3] TS': Temporary Terms Set, initialized to TS.
 4] d, t: Document and Term element respectively.
 5] Repeat
 6]   Select a document element as d from DS.
 7]   Repeat
 8]     Select the highest element as t in TS'
          according to the weight.
```

```
9]    If t appears in d and not member in ITS
        Then Add t to ITS.
10]   Remove t from TS.
11]  Until t appears in d.
12]  Remove d from DS.
13]  Assign TS to TS'.
14] Until DS is empty.
15] Return ITS.
```

Figure 2. The algorithm for selection of initial representative terms

3.3 Automatic Expansion and Reweighting of IRKs

After the IRKs are selected, additional terms are selected to be expanded in the order of their weights calculated by the method in Section 3.1. Let us assume that 5 terms are used to represent a user's preference and the number of IRKs is 3. Then, 2 terms with highest weights except IRKs are selected additionally. The IRKs and these terms constitute the final representative keywords (FRKs) and are reweighted by considering the co-occurrence similarity with IRKs. For this end, the relevance degrees of the FRKs in every document are calculated with the equation (6). Each positive example document represents user's preferable content. In other words, each document tends to contain general or specific or partial contents. We regard the IRKs as the essential terms of the given positive examples. So, the possibility that the related terms, e.g., synonym, collocated terms and so on, occurred together with these IRKs in the same document set increases.

$$RD_{ik} = 1 - \log_p \sqrt{\frac{\sum_{j=1}^{n} (kf_{jk} - tf_{ik})^2}{n} + 1} \qquad (6)$$

where, RD_{ik} is the relevance degree between IRKs and candidate term t_i in document d_k, kf_{jk} is the frequency of initial representative keyword j in document d_k, tf_{ik} is the frequency of candidate term t_i in document d_k, n is the number of IRKs, p is a control parameter. In our experiments, p is set to 10. The RD_{ik} is treated as 0 if it has negative value.

For example, let K be a set of IRKs consisting of *k1*, *k2* and *k3* terms and their frequencies in document *d1* be 4, 3, and 1, respectively. Also, let the frequency of term t_1 be 2. Then, its relevance degree is calculated as follows:

$$RD_{11} = 1 - \log_{10} \sqrt{\frac{2^2 + 1^2 + (-1)^2}{3} + 1} = 0.762$$

As shown in the above equation, RD_{ik} is inversely proportional to the sum of term frequency difference between initial representative term and candidate term. So, the higher is the value of *Rd*, the more similar the co-occurrence is, that is, the equation reflects the co-occurrence similarity between initial representative terms and a candidate term appropriately. After calculating the relevance degree of a candidate term, the weight of the term in the set of example documents is determined by the following equation:

$$w_{ri} = \sum_{k=1}^{n} (w_{ik} \times RD_{ik}) \qquad (7)$$
$$w_{ik} = TF_{ik} \times IDF_i$$

where, w_{ri} is the weight of term t_i in the document set, w_{ik} is the weight of term t_i in document d_k, TF_{ik} is the frequency of term t_i in document d_k, IDF_i is the inverse document frequency of term t_i, and n is the number of example documents.

The equation (7) is a modification of the Rocchio's in Section 2. Different from that equation, we additionally use the term relevance degree between initial representative terms and a candidate term. Let us assume that the IDF value of the candidate term t_1 is 1.0 and it occurs 3, 2, and 1 within document *d1*, *d2* and *d3*, respectively. If the relevance degrees for three documents are also assumed to 0.3, 0.5, and 0.7, respectively, then the weight of candidate term t_i is calculated as below.

$$w_{r1} = ((3.0 \times 0.4) + (2.0 \times 0.5) + (1.0 \times 0.7)) = 1.82$$

Finally, the weights of the FRKs are calculated by the following equation:

$$w_i = w_{ki} + w_{ri} \qquad (8)$$

where, w_{ki}, is the initial weight of term t_i. Instead of using the weight obtained by fuzzy inference, the initial weight w_{ki} of term t_i is recalculated by the equation (9), if the term is in IRKs and otherwise 0. The equation is the one introduced to assign a weight to an initial query term in IR systems based on the vector space model (Baeza-Yates and Ribeiro-Neto, 1999).

$$w_{ki} = \left(0.5 + \frac{0.5 \times freq_i}{\max_j freq_j}\right) \times \log\left(\frac{N}{n_i}\right) \qquad (9)$$

where, $freq_i$ is the frequency of initial representative keyword t_i, n_i is the frequency of documents in

which t_i appear, and N is the total number of documents.

Let K = {t1, t3, t4} be the set of IRKs, WK = {3.0, 2.0, 1.0} be the set of their weights calculated by the equation (9), T = {t1, t2, t3, t4, t5} be the set of FRKs, and WT = {5.0, 4.0, 3.0, 2.0, 1.0} be their weights through the equation (7). Then, we can get the final weights of FRKs, {8.0,4.0, 5.0, 3.0, 1.0}.

4 Experiments

We used Reuters-21578 data as an experimental document set. This collection has five different sets of contents related categories. They are EXCHANGES, ORGS, PEOPLE, PLACES and TOPICS. Some of the categories set have up to 265 categories, but some of them have just 39 categories. We chose the TOPICS categories set which has 135 categories. We divided the documents according to the "ModeApte" split. There are 9603 training documents and 3299 test documents. Among the 135 categories, we first chose only 90 ones that have at least one training example and one testing example. Then, we finally selected 21 categories that have from 10 to 30 training documents. The 3019 documents of those categories are used as testing documents. The document frequency information from 7770 training documents in 90 categories is used to calculate IDF values of terms. We did not consider negative documents under the assumption that only positive documents coincident with users' preferences were given implicitly or explicitly .

Documents are ranked by the cosine similarity and the following F-measure (Baeza-Yates and Ribeiro-Neto, 1999), which is a weighted combination of recall and precision and popularly used for performance evaluation. Since the maximum value for F can be interpreted as the best possible compromise between recall and precision, we use this maximum value.

$$F_j = \frac{2}{\frac{1}{P_j}+\frac{1}{R_j}} = 2P_jR_j/(P_j+R_j) \qquad (10)$$

where, R_j and P_j are the recall and precision for the j'th document in the ranking and F_j is their harmonic mean.

First, our method was compared to the Rocchio and Widrow-Hoff algorithms. To see the effect of the number of FRKs, we made experiments by varying it from 5 to 30 in increment 5 and for the case that all terms are used. Table 2 shows the overall or summary result of the proposed method compared to the two existing algorithms for 21categories. The result shows that our method is better than the others in all cases, especially when 10 terms are used to represent user preferences. Table 3 shows the detail result in that case, i.e. the F-values and the performance improvement ratios when 10 terms are used. The proposed method has achieved about 20% over Rocchio algorithm and 10% over Widrow-Hoff algorithm on the average. When 5 terms are used to represent user preferences, 19 categories among 21 categories are used because "strategic-metal" and "pet-chem" categories do not satisfy the constraint in Section 3.2, i.e., 5 terms are too few to cover all training documents.

Table 2. Performance of 21 categories in the REUTERS corpus and comparison with two existing algorithms.

	Our	**Rocchio**	**W.H.**
5	0.582	0.511	0.566
10	0.594	0.496	0.540
15	0.571	0.490	0.529
20	0.552	0.489	0.522
25	0.545	0.491	0.493
30	0.541	0.495	0.500
All	0.490	0.467	0.483

It is not clear which component of our method mainly contributes to such improvement since our method consists of two main components - one is for extracting IRKs, the other for expanding and reweighting of IRKs. To analyze our method, we made several variants of the proposed method and did experiments with them. The variants are named by the sequence of the following symbols.

IF, **IR**, **IW**: mean that IRKs are selected based on the weight obtained by the method in Section 3.1, the Rocchio algorithm, and the Widrow-Hoff algorithm, respectively.

RC, **RR**, **RW**: mean that terms are reweighted by the method in Section 3.3, the Rocchio algorithm, and the Widrow-Hoff algorithm, respectively.

EC, **EF**, **ER**, **EW**: mean that expanded terms are selected based on the weight obtained by applying the method in Section 3.3, the method in Section 3.1, the Rocchio algorithm, and the Widrow-Hoff algorithm, respectively.

For example, the proposed method in Section 3 is named as IF_EF_RC, which means IRKs, and expanded terms are selected based on the weight calculated by the method in Section 3.1 and then reweighted by the method in Section 3.3. For another example, the method called by IF_RC_EC means that IRKs are selected based on the weight obtained by the method in Section 3.1 and then all terms are reweighed by the method in Section 3.3 before expanded terms are selected.

In the proposed method, fuzzy inference technique is used to extract IRKs. So, we tried two variants, IR_ER_RC and IW_EW_RC, where the Rocchio and Widrow-Hoff algorithms are used respectively to calculate the representativeness (or weights) of terms instead of the method in Section 3.1, and then IRKs and expanded terms are selected based on these weights. The variants all use the reweighting scheme in Section 3.3. Table 4 shows that other keyword extraction algorithms do not show any benefit over the fuzzy inference approach. We can also observe that when one of the existing algorithms is combined with the second component of our method, the performance improvement over the case that the algorithm solely is used is negligible.

The method to extract IRKs reflecting user's preference directly affects the result of the term reweighting process because the process is based on the term co-occurrence similarity with the IRKs. If the terms that are far from user's preference are extracted as IRKs, then some terms that actually are improper in representing user's information needs may be assigned with high weights during the reweighting process and then the final vector generated from the results may be disqualified from representing user's preferences. So, we can know that our fuzzy inference technique is effective to extract IRKs from the results in Table 4.

To demonstrate the usefulness of the second part of our method, i.e., the expansion and reweighting technique, we also tried the 5 variants of our method (IF_RC_EC, IF_RR_ER, IF_RW_EW, IF_EF_RR, IF_EF_RW). Table 5 shows the all variants are not better than the original though they outperform Rocchio and Widrow-Hoff algorithms.

5 Conclusions

In this study, we apply fuzzy inference technique and term reweighting scheme based on the term co-occurrence similarity to the problem that extract important keywords representing contents of documents presented by users. We have conducted extensive experiments on the Reuters-21578 collection. The results show that our method outperforms two well-known training algorithms for linear text classifiers. Moreover, some variants of our method have been explored to analyze the characteristics of our method. Though this paper only describes how to extract user preferences from example documents, the technique will be applicable to several areas such as query modification in IR, user profile modification in information filtering, text summarization and so forth directly or with some modifications.

Since only positive examples are considered in our method, the method is not applicable to a document set containing negative examples. For covering negative examples, it needs to modify the fuzzy inference rules with considering additional input variables. The proposed method was also designed for a small set of documents. So, we could not achieve performance improvement as described in this paper when our method is applied to a large set of documents. However, such a problem will be alleviated if clustering techniques are used together as in (Alberto et al., 2001; Lam and Ho, 1998; Ugur et al., 2000).

Table 3. The detail result when 10 terms are used for user preferences

	Our	**Rocchio**	**W.H.**
lumber	0.7273	0.4444	0.6667
dmk	0.4	0.4444	0.4
sunseed	0.5714	0.3333	0.3333
lei	1	0.8	1
soy-meal	0.6667	0.5143	0.5185
fuel	0.4615	0.4615	0.4615
heat	0.75	0.75	0.75
soy-oil	0.3704	0.2692	0.32
lead	0.5625	0.5	0.5
strategic-	0.13333	0.1053	0.1408
hog	0.8	0.6	0.8
orange	0.9091	0.9091	0.8571
housing	0.5714	0.6667	0.5714
tin	0.96	0.7857	0.9231
rapeseed	0.6154	0.5714	0.6154
wpi	0.5714	0.5882	0.5882
pet-chem	0.3704	0.2727	0.2759
silver	0.381	0.4	0.5

zinc	0.8966	0.6667	0.6842
retail	0.1667	0.0548	0.0548
sorghum	0.5882	0.2727	0.3871
Average	0.5940	0.4957	0.5404

Table 4. The performance of our method and its two variants that use Rocchio and Widrow-Hoff algorithms instead of fuzzy inference, respectively.

	IF_EF_RC	**IR_ER_RC**	**IW_EW_RC**
5	0.582	0.509	0.571
10	0.594	0.505	0.528
15	0.571	0.502	0.537
20	0.552	0.491	0.526
25	0.545	0.487	0.518
30	0.541	0.497	0.510
All	0.490	0.478	0.490

Table 5. The performance of our method and its five variants that use different reweighting and expanding approaches.

	IF_EF_RC	**IF_RC_EC**	**IF_RR_ER**	**IF_RW_EW**	**IF_EF_RR**	**IF_EF_RW**
5	0.582	0.571	0.546	0.580	0.545	0.570
10	0.594	0.520	0.498	0.549	0.551	0.561
15	0.571	0.514	0.491	0.508	0.518	0.517
20	0.552	0.513	0.495	0.533	0.497	0.538
25	0.545	0.509	0.498	0.503	0.491	0.521
30	0.541	0.515	0.506	0.512	0.498	0.511
All	0.490	0.488	0.478	0.494	0.465	0.483

References

Alberto Diaz Esteban, Manuel J. Mana Lopez, Manuel de Buenaga Rodriguez, Jose Ma Gomez Hidalgo and Pablo Gervas Gomez-Navarro. 2001. *Using linear classifiers in the integration of user modeling and text content analysis in the personalization of a web-based Spanish news servic*, In Proceedings. of the Workshop on Machine Learning, Information Retrieval and User Modeling, 8th International Conference on User Modeling.

Baeza-Yates, R. and Ribeiro-Neto B.. 1999. *Modern Information Retrieval*, ACM Press, USA.

David D. Lewis, Robert E. Schapire , James P. Callan and Ron Papka. 1996. *Training algorithms for linear text classifiler*, In Proc. of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval.

Goldberg D., Nichols D., Oki B. M., and Terry D.. 1992. *Using collaborative filtering to weave an information tapestr*, Communication of the ACM, 35(12), p61-70.

Kim, J., Oard, D.W., and Romanik, K.. 2000. *User modeling for information filtering based on implicit feedback*, In Proceedings. of ISKO-France.

Konstan J. A. , Miller B. N., Maltz D., Herlocker J. L., Gordon L.R. and Riedl J.. 1997. *GroupLens: Applying collaborative filtering to Usenet News*, Communication of the ACM, 40(3), p 77-87.

Lam K. and Ho C.. 1998. *Using a generalized instance set for automatic text categorization*, In 21th Ann. Int. ACM SIGIR Conference on Research and Development in Information Retrieval, p81-89.

Lee C.C.. 1990. *Fuzzy logic in control systems: fuzzy logic controller-part I*, IEEE Trans. On Systems, Man, and Cybernetics, 20 (2) , p408-418.

Mitra, M., Singhal, A., and Buckley, C.,. 1998. *Improving Automatic Query Expansion*, In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, p206-214, 1998.

Nichols D. M.. 1997. *Implicit ratings and filteri*", In Proceedings of the 5th DELOS Workshop on Filtering and Collaborative Filtering, p10-12.

Pazzani, M. and Billsus, D.. 1997. *Learning and revising user profiles: the identification of interesting Web site*, Machine Learning, 1997.

Seo, Y. and Zhang, B.. 2001. *Personalized Web Document Filtering Using Reinforcement Learning*, Applied Artificial Intelligence.

Soltysiak, S. J. and Crabtree, I. B.. 2000. *Automatic Learning of User Profiles—Towards the Personalization of Agent Services*, BT Technology Journal, 16 (3), p110–117.

Ugur Çetintemel, Franklin Michael J. and Lee Giles C.. 2000 . *Self-Adaptive User Profiles for Large-Scale Data Delivery*, ICDE, p622-633.

Xu Jinix and Croft W. B.. 1996. *Query Expansion Using Local and Global Document Analysis*, In Proceeding of ACM SIGIR International Conference on Research and Development in Information Retrieval, p4-11.

Yan T. W. and Garcia-Molin H.. 1995. *SIFT- A tool for wide-area information dissemination*, In Proceedings of the 1995 USENIX Technical Conference, p177-186.

Yang, Y. and Pedersen, J.. 1997. *A comparative study on feature selection in text categorization*, In Proceedings of the 14th International Conference on Machine Learning, p412-420.