A New Fuzzy Information Retrieval Method Based on Document Terms Reweighting Techniques

Yih-Jen Horng  
National Chiao Tung University  
R.O.C.

Shyi-Ming Chen  
National Taiwan University of Science and Technology  
R.O.C.

Chia-Hoang Lee  
National Chiao Tung University  
R.O.C.

Abstract

In this paper, we present a new method for fuzzy information retrieval based on document terms reweighting techniques. The proposed method modifies the weights of document terms in document descriptor vectors based on the user’s relevance feedback. After modifying the weights of terms in document descriptor vectors, the degrees of satisfaction of relevant documents with respect to the user’s query will increase, and the degrees of satisfaction of irrelevant documents with respect to the user’s query will decrease. Then, the modified document descriptor vectors can be used as personal profiles for future query processing. The proposed method can make fuzzy information retrieval systems more flexible and more intelligent to deal with documents retrieval. It can increase the retrieval effectiveness of the fuzzy information retrieval systems for document retrieval.

Keywords: Document Descriptor Vectors, Fuzzy Information Retrieval, Personal Profile, Relevance Feedback, Document Terms Reweighting.

1. Introduction

Automatic terms weighting is an important research topic of modern information retrieval systems [2]. Since different index terms have different degrees of importance in a document, an importance indicator (i.e., the term weight) is associated with each index term. Two main components that affect the importance of an index term in a document are the term frequency factor \((tf)\) and the inverse document frequency factor \((idf)\) [16], [17]. However, the terms weighting methods based on these two statistic factors may
be not suitable enough from the human being’s viewpoint. Therefore, the degrees of similarity between relevant documents and the user’s query may not be large enough. As a result, the retrieval effectiveness of the information retrieval system is not good enough. In order to increase the retrieval effectiveness, some “terms weighting” methods have been proposed [10], [18]. In [10], Jung et al. proposed a terms weighting scheme which not only considers “occurrence terms”, but also “absence terms” in finding the degrees of similarity among document descriptor vectors, where the “absence terms” means terms not appearing in the content of a specific document and they are negatively weighted rather than zero weighted in the document descriptor vector of this document. In [18], Singhal et al. proposed a pivoted document length normalization method to improve the drawbacks of the conventional document length normalization method, and the documents with different lengths can be fairly retrieved.

Another approach to increase the retrieval effectiveness is by modifying the user’s query [3], [7], [9], [11]. In [3], Chan et al. proposed a query expansion method which applies clustering techniques to the initial search results to provide concept-based browsing and helps the user to reduce the browsing labor. In [7], we proposed a method for query expansion based on the cluster centers of the document clusters. In [11], Kim et al. proposed a query term expansion and reweighting method which considers the term co-occurrence within the feedbacked documents. Among these methods, the most used one is the “relevance feedback” method which requires the users to provide the relevance judgment of the retrieved documents. It modifies the user’s query based on the set of relevant documents and the set of irrelevant documents among the retrieved documents. Although most of the relevance feedback methods are used for user’s query expansion, the effect of the relevance feedback methods is only restricted to deal with the current query processing and it can not deal with the future queries. If the user’s relevance feedback can be recorded and be used for the future queries, then the users of the information retrieval systems do not need to perform the process of relevance feedback in the future.

In this paper, we present a new method for fuzzy information retrieval based on document terms reweighting techniques to modify the weights of terms in document descriptor vectors based on the user’s relevance feedback. After modifying the weights of terms in document descriptor vectors, the degrees of satisfaction of relevant documents with respect to the user’s query will increase, and the degrees of satisfaction of irrelevant documents with respect to the user’s query will decrease. Then, the modified document
descriptor vectors can be used as personal profiles for future query processing. The proposed method can make fuzzy information retrieval systems more flexible and more intelligent to deal with documents retrieval. It can increase the retrieval effectiveness of the fuzzy information retrieval systems for document retrieval.

The rest of this paper is organized as follows. In Section 2, we present an automatic terms weighting method and a query processing method for document retrieval. In Section 3, we present a method to derive modified document descriptor vectors based on the user’s relevance feedback. In Section 4, we present the experimental results. The conclusions are discussed in Section 5.

2. Automatic Terms Weighting and User’s Query Processing

In traditional information retrieval systems, the contents of documents are typically represented by some index terms extracted from the texts of the collected documents [16]. The most direct approach is to use all words appearing in the contents of the collected documents as index terms. However, since each document may contain a large amount of words, these documents should be preprocessed to reduce the set of words into a manageable size for processing. The selected documents are preprocessed in two steps. Firstly, the words appearing on a stoplist (i.e., a list of words thought to have no indexing value such as function words like “and,” “or,” “of,” “but,” ..., etc.) are eliminated [15]. Then, the word extractor stems each remaining word to its “root form” [6]. The collection of these root-formatted words forms a set of index terms $T$ for the document set. In this paper, the weight $w_{\text{term-document}}(t, d_i)$ of term $t$ in document $d_i$ is calculated as follows [8]:

$$w_{\text{term-document}}(t, d_i) = \frac{0.5 + 0.5 \frac{tf_{it}}{\max_{k=1,2,...,L} tf_{ik}}}{\max_{j=1,2,...,L} \left( 0.5 + 0.5 \frac{tf_{ij}}{\max_{k=1,2,...,L} tf_{ik}} \right) \log \frac{N}{df_t}}, \quad (1)$$

where $tf_{it}$ denotes the frequency of term $t$ appearing in document $d_i$, $df_t$ denotes the number of documents containing term $t$, $L$ denotes the number of terms contained in document $d_i$ and $N$ denotes the number of collected documents. The larger the value of $w_{\text{term-document}}(t, d_i)$, the more important the term $t$ to document $d_i$. From formula (1), we can see that the value of $w_{\text{term-document}}(t, d_i)$ is between zero and one.
After the weight of each term in each document has been calculated, each document \( d_i \) is represented as a fuzzy set [19]. Roughly speaking, a fuzzy set is a set with fuzzy boundaries. Let \( U \) be the universe of discourse, \( U = \{u_1, u_2, \ldots, u_n\} \). A fuzzy set \( A \) of the universe of discourse \( U \) can be represented by

\[
A = \{(u_1, \mu_A(u_1)), (u_2, \mu_A(u_2)), \ldots, (u_n, \mu_A(u_n))\},
\]

(2)

where \( \mu_A \) denotes the membership function of the fuzzy set \( A \), \( \mu_A : U \rightarrow [0, 1] \), and \( \mu_A(u_i) \) indicates the grade of membership of \( u_i \) in the fuzzy set \( A \). Therefore, we can represent each document \( d_i \) as a fuzzy set of index terms shown as follows:

\[
d_i = \{(t_1, w_{i1}), (t_2, w_{i2}), \ldots, (t_s, w_{is})\},
\]

where \( w_{ij} \) denotes the weight of term \( t_j \) in document \( d_i \), \( 0 \leq w_{ij} \leq 1 \), \( 1 \leq j \leq s \), and \( s \) denotes the number of terms in the set of index terms. Each document \( d_i \) can then be represented by a document descriptor vector \( \overline{d_i} \) shown as follows:

\[
\overline{d_i} = (w_{i1}, w_{i2}, \ldots, w_{is}),
\]

where \( w_{ij} \) denotes the weight of term \( t_j \) in document \( d_i \), \( 0 \leq w_{ij} \leq 1 \), \( 1 \leq j \leq s \), and \( s \) denotes the number of terms in the set of index terms.

Assume that the user's query \( q \) is represented by a query vector \( \overline{q} \) shown as follows:

\[
\overline{q} = (w_{q1}, w_{q2}, \ldots, w_{qs}),
\]

where \( w_{qi} \) indicates the degree of strength that the desired documents contain term \( t_j \), \( 0 \leq w_{qi} \leq 1 \), and \( 1 \leq i \leq s \). Then, we apply a fuzzy query processing method to calculate the degree of satisfaction \( DS(d_i) \) of document \( d_i \) with respect to the user's query as follows [4]:

\[
DS(d_i) = \frac{\sum_{j=1,2,\ldots,s} T(w_{qj}, w_{ij})}{s},
\]

(3)

where \( s \) is the number of terms in the set of index terms and \( T \) is a similarity function [4] to calculate the degree of similarity between two real values between zero and one defined as follows:

\[
T(w_{qj}, w_{ij}) = 1 - |w_{qj} - w_{ij}|,
\]

(4)

where \( 0 \leq w_{qj} \leq 1 \), \( 0 \leq w_{ij} \leq 1 \), and \( 1 \leq j \leq s \). After the degree of satisfaction \( DS(d_i) \) of each document \( d_i \) with respect to the user’s query is obtained, we normalize
the value of $DS(d_i)$ by dividing it by the maximum value among the values of $DS(d_1)$, $DS(d_2), \ldots,$ and $DS(d_N)$, where $N$ is the number of collected documents. The user can set a query threshold value $\alpha$, where $\alpha \in [0, 1]$. The documents are retrieved only when their degrees of satisfaction with respect to the user’s query are larger than or equal to $\alpha$, where $\alpha \in [0, 1]$.


In this section, we propose a new method for document terms reweighting for fuzzy information retrieval. The goal of the proposed method is to reduce the degrees of satisfaction of irrelevant documents and to increase the degrees of satisfaction of relevant documents with respect to the user’s query according to the user’s “relevance feedback”.

First, the documents and the user’s query can be represented as points in a vector space as shown in Figure 1, respectively, where each “○” means a relevant document with respect to the user’s query, each “×” means an irrelevant document with respect to the user’s query and “●” means the user’s query.

![Figure 1](image)

An intuitive idea of reducing the degrees of satisfaction of irrelevant documents and increasing the degrees of satisfaction of relevant documents with respect to the user’s
query, respectively, is to move each relevant document closer to the user’s query \( q \) and move each irrelevant document away from the user’s query in the vector space as shown in Figure 2.

![Figure 2. Relevant documents move toward the user’s query and irrelevant documents move away from the user’s query in the vector space.](image)

However, since the amount of modification for each retrieved document will be recorded for the future use, if each document has its own amount of modification, then there are lots of data to be stored. Therefore, we let each document move in the same direction with the same distance. That is, we let each document have a uniform movement. We hope that the relevant documents are more close to the user’s query and the irrelevant documents are more far away from the user’s query. This uniform movement of each document is transformed into a vector, which is defined as the document modification vector \( \Delta \) shown as follows:

\[
\Delta = \langle \delta_1, \delta_2, \ldots, \delta_s \rangle,
\]

where \( \delta_i \) indicates the amount of modification to the \( i \)th term of each document, \( 1 \leq i \leq s \), and \( s \) is the number of terms extracted from the collected documents. When the document modification vector \( \Delta \) is used to modify each document descriptor vector \( d_i \) to derive a modified document descriptor vector \( \overrightarrow{d_i} \), where \( \overrightarrow{d_i} = \overrightarrow{d_i} + \Delta \), most of the relevant documents will move toward the user’s query to increase their degrees of satisfaction with
respect to the user’s query, and most of the irrelevant documents will move away from the user’s query to decrease their degrees of satisfaction with respect to the user’s query, as shown in Figure 3. Therefore, when all the retrieved documents are resorted according to their new degrees of satisfaction with respect to the user’s query, most of the relevant documents will be listed in front of the irrelevant documents. It will be useful to the users when browsing the retrieved documents since most relevant documents are listed in the front part of the document list and the user can ignore the irrelevant documents by a query threshold value $\alpha$, where $\alpha \in [0, 1]$.

![Figure 3](image-url)

**Figure 3.** Each document in the vector space is modified by a document modification vector $\Delta$.

However, there are many candidates for the document modification vector $\Delta$. For example, from Figure 4 and Figure 5, we can see that when the document modification vector $\Delta_1$ and the document modification vector $\Delta_2$ are applied to modify the document descriptor vector of each document, both of them will cause most of the relevant documents to move toward the user’s query and cause most of the irrelevant documents to move away from the user’s query.

In order to choose a better document modification vector, we use the “Relevant Document Ranking Score” (RDRS) to judge the performance of the document modification vector based on the resorted document list, where

$$RDRS = \sum_{i=1,2,...,m} \frac{1}{\text{Rank}_{d_i}},$$

where

- $\text{RDRS}$ is the Relevant Document Ranking Score,
- $m$ is the number of documents,
- $\text{Rank}_{d_i}$ is the rank of document $d_i$.

This formula calculates the sum of the reciprocals of the ranks of the relevant documents. The higher the RDRS value, the better the document modification vector performs.
where \( m \) is the number of retrieved documents with respect to the user’s query and \( \text{Rank}_{d_i} \) denotes the rank of document \( d_i \) in the resorted document list. The maximum value of \( RDRS \) occurs when all relevant documents are ranked before all irrelevant documents with respect to the user’s query. On the other hand, the minimum value of \( RDRS \) occurs when all irrelevant documents are ranked before all relevant documents with respect to the user’s query. Moreover, the more the relevant documents are listed before irrelevant documents, the larger the value of \( RDRS \). For example, assume that there are five documents \( d_1, d_2, d_3, d_4 \) and \( d_5 \) retrieved from a fuzzy information retrieval system with respect to the user’s query. Furthermore, assume that the documents \( d_1, d_2 \) and \( d_3 \) are judged by the user as relevant documents and the documents \( d_4 \) and \( d_5 \) are judged by the user as irrelevant documents. Assume that these five documents are ordered according to their degrees of satisfaction with respect to the user’s query shown as follows:

\[
d_1 > d_4 > d_5 > d_3 > d_2,
\]

then the value of \( RDRS \) of the retrieved documents is \( \frac{1}{4} + \frac{1}{4} + \frac{1}{5} = 1.45 \). Assume that a document modification vector \( \Delta_1 \) is used, and the relevant documents are moved closer to the user’s query than the irrelevant documents, where the retrieved documents are reordered according to their new degrees of satisfaction with respect to the user’s query \( q \) shown as follows:

\[
d_1 > d_3 > d_4 > d_2 > d_5,
\]

then the value of \( RDRS \) of the retrieved documents becomes \( \frac{1}{1} + \frac{1}{2} + \frac{1}{5} = 1.75 \). Moreover, assume that another document modification vector \( \Delta_2 \) is used and the relevant documents are moved closer to the user’s query than the irrelevant documents, where the retrieved documents are reordered according to their new degrees of satisfaction with respect to the user’s query shown as follows:

\[
d_1 > d_3 > d_2 > d_4 > d_5,
\]

then the value of \( RDRS \) of the retrieved documents becomes \( \frac{1}{1} + \frac{1}{2} + \frac{1}{3} = 1.83 \). Therefore, based on the above \( RDRS \) values, we can see that the document modification vector \( \Delta_2 \) is better than the document modification vector \( \Delta_1 \).
The purpose of the proposed method is to provide a method to derive a document modification vector $\Delta$ which can make the value of $RDRS$ of the retrieved documents as large as possible. However, since the degree of satisfaction of each document $d_i$ with respect to the user’s query $q$ is based on the relative position of the document descriptor vector $\vec{d_i}$ and the user’s query vector $\vec{q}$ in the vector space, the effect of modifying the document descriptor vector $\vec{d_i}$ of each document $d_i$ by a document modification vector $\Delta$ as shown in Figure 3 is equal to the effect of modifying the user’s query vector $\vec{q}$
by the inverse vector \( -\Delta \) of the document modification vector \( \Delta \) as shown in Figure 6. Since considering the modification of only one point (i.e., the user’s query vector \( \overline{q} \)) in the vector space seems to require less effort than considering the modification of many points (i.e., all document descriptor vectors of retrieved documents) in the vector space at the same time, the method of deriving a document modification vector can start by deriving a “virtual query modification vector” which virtually moves the user’s query vector \( \overline{q} \) to relevant document descriptor vectors as close as possible and moves the user’s query vector from irrelevant document descriptor vectors in the vector space as far away as possible. The difference between the virtually new position and the original position of the user’s query vector \( \overline{q} \) in the vector space is then viewed as the virtual query modification vector. Then, the inverse vector of the virtual query modification vector is used as the document modification vector.

\[
\begin{align*}
\bullet & \text{ means relevant documents} \\
\times & \text{ means irrelevant documents} \\
\bigcirc & \text{ means the user’s query}
\end{align*}
\]

Figure 6. The user’s query vector is modified by an inverse document modification vector \( -\Delta \).

In the following, we present an algorithm for document terms reweighting in document descriptor vectors.

**Document Terms Reweighting Algorithm:**

**Step 1:** Divide the retrieved documents into two clusters, where one cluster contains relevant documents and the other contains irrelevant documents.

**Step 2:** Let the new virtual user’s query vector \( \overline{vq} \) be equal to the cluster center of the cluster containing relevant documents in the vector space.
**Step 3**: Calculate the degree of satisfaction $DS(d_i)$ of each document $d_i$ with respect to the new virtual user’s query vector $\overline{vq}$ by formula (3).

**Step 4**: Sort the retrieved documents according to the new value of $DS(d_i)$ of each document $d_i$.

**Step 5**: If all relevant documents are listed before all irrelevant documents

- then **Stop**
- else Find the irrelevant document $d_{ir}$ which has the largest value of $DS(d_i)$ and find the first relevant document $d_r$ next to $d_{ir}$ in the ordered document list.

**Step 6**: Move the new virtual user’s query vector $\overline{vq}$ across the middle line between $d_r$ and $d_{ir}$ in the vector space.

**Step 7**: Calculate the degree of satisfaction $DS(d_i)$ of each document $d_i$ with respect to the new virtual user’s query vector $\overline{vq}$ using formula (3).

**Step 8**: Sort the retrieved documents according to the new $DS(d_i)$ of each document $d_i$.

**Step 9**: If the position of $d_{ir}$ is moved backward in the document list

- then go to Step 5
- else restore the former document list and find the relevant document next to $d_r$ in the ordered document list;
  - if no such documents exist then **Stop**
  - else use it as $d_r$ and go to Step 6.

**Step 10**: Calculate the difference between the new virtual user’s query vector $\overline{vq}$ and the original user’s query vector $\overline{q}$ and use it as the virtual query modification vector.

**Step 11**: Use the inverse vector of the virtual query modification vector as the document descriptor modification vector $\Delta$.

**Step 12**: Use the derived document descriptor modification vector $\Delta$ to reweight document terms in the set of $D$ document descriptor vectors of the retrieved documents, where $D = \{d_1, d_2, \ldots, d_m\}$.

- for $i = 1$ to $m$ do
  - let $d_i = d_i + \Delta$
- end.
The proposed document terms reweighting algorithm starts by dividing the retrieved documents into two clusters (i.e., two classes), where one cluster contains relevant documents and the other contains irrelevant documents. The idea behind this is based on the assumption that most of the relevant documents should be closer to the cluster center of the cluster containing the relevant documents than the irrelevant documents. Therefore, the cluster center of the cluster containing relevant documents should be a good starting point as the new virtual position of the user’s query vector \( \mathbf{q} \).

After virtually moving the user’s query vector \( \mathbf{q} \) to the cluster center of the cluster containing the relevant documents in the vector space to derive a new virtual user’s query vector \( \mathbf{vq} \) and calculating the degree of satisfaction \( DS(d_i) \) of each document \( d_i \) with respect to the new virtual user’s query, the system sorts the retrieved documents according to the new value of \( DS(d_i) \) of each document \( d_i \). However, since the two clusters are often overlapped, some irrelevant documents may be closer to the cluster center of the cluster containing relevant documents than some relevant documents as shown in Figure 7. That is, some irrelevant documents may be listed before some relevant documents in the ordered document list. Therefore, further adjustment of the position of the new virtual user’s query vector \( \mathbf{vq} \) in the vector space is required. We do this by finding an irrelevant document \( d_{ir} \) with the highest rank comparing to other irrelevant documents and by finding the first relevant document \( d_r \) ranked after \( d_{ir} \) in the ordered document list. For example, assume that there are seven retrieved documents \( d_1, d_2, \ldots, d_7 \) and assume that their order according to their degrees of satisfaction with respect to the new virtual user’s query from high to low is \( d_1 > d_2 > d_3 > d_4 > d_5 > d_6 > d_7 \) as shown in Table 1. Then, document \( d_3 \) is used as \( d_{ir} \) and document \( d_5 \) is used as \( d_r \).

![Figure 7. The relevant cluster and the irrelevant cluster are often overlapping.](image-url)
Table 1. An ordered document list.

<table>
<thead>
<tr>
<th>Ordered Document List</th>
<th>Relevant or Irrelevant</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d_1$</td>
<td>Relevant</td>
</tr>
<tr>
<td>$d_2$</td>
<td>Relevant</td>
</tr>
<tr>
<td>$d_3$</td>
<td>Irrelevant</td>
</tr>
<tr>
<td>$d_4$</td>
<td>Irrelevant</td>
</tr>
<tr>
<td>$d_5$</td>
<td>Relevant</td>
</tr>
<tr>
<td>$d_6$</td>
<td>Relevant</td>
</tr>
<tr>
<td>$d_7$</td>
<td>Irrelevant</td>
</tr>
</tbody>
</table>

Then, we try to adjust the position of the new virtual user’s query vector $v_q$ to bring it closer to the document descriptor vector $d_r$ than to the document descriptor vector $d_{ir}$ in the vector space. It is achieved by deriving the middle line between $d_r$ and $d_{ir}$ in the vector space and move the new virtual user’s query vector $v_q$ across this middle line as shown in Figure 8. Therefore, after calculating the degree of satisfaction $DS(d_i)$ of each document $d_i$ with respect to the new virtual user’s query and resorting the retrieved documents according to the new value of $DS(d_i)$ of each document $d_i$, we can see that $d_r$ can get a higher rank than $d_{ir}$.

Figure 8. The new virtual user’s query vector is derived by virtually moving across the middle line between $d_r$ and $d_{ir}$. 
However, sometimes virtually moving the new virtual user’s query vector $\overrightarrow{vq}$ across the middle line between $d_r$ and $d_{ir}$ may cause other relevant documents to decrease their degrees of satisfaction with respect to the new virtual user’s query. Therefore, although the degrees of satisfaction of $d_r$ with respect to the new virtual user’s query vector $\overrightarrow{vq}$ is larger than the one of $d_{ir}$, the rank of $d_{ir}$ may still move forward in the document list when all retrieved documents are reordered according to their new degrees of satisfaction with respect to the new virtual user’s query. When this happens, the former ordered document list should be restored, and the relevant document next to $d_r$ in the ordered document list should be used as $d_r$. The operation repeats again until no further appropriate $d_r$ is found. Then, the difference between the new virtual user’s query vector $\overrightarrow{vq}$ and the original user’s query vector $\overrightarrow{q}$ is calculated and we use the difference as the virtual query modification vector. The inverse vector of the virtual query modification vector is used as the document descriptor modification vector. Finally, the document descriptor modification vector is used to reweight document terms in document descriptor vectors of the retrieved documents.

After reweighting the terms in the document descriptor vectors by the proposed algorithm, most of the relevant documents will be listed in front of the irrelevant documents. For example, assume that there are three retrieved documents $d_1$, $d_2$ and $d_3$ with respected to the user’s query $q$. The original document descriptor vectors of the three retrieved documents are:

\[
\overrightarrow{d_1} = (0.4, 0.6, 0.1, 0), \\
\overrightarrow{d_2} = (0.7, 0.6, 0, 0.2), \\
\overrightarrow{d_3} = (0.9, 1, 0.1, 0),
\]

and the user’s query descriptor vector $\overrightarrow{q}$ is

\[
\overrightarrow{q} = (0.5, 0.8, 0, 0).
\]

Based on formula (3), we can get

\[
DS(d_1) = \frac{(1 - |0.4 - 0.5|) + (1 - |0.6 - 0.8|) + (1 - |0.1 - 0|) + (1 - |0 - 0|)}{4} = \frac{0.36}{4} = 0.9,
\]

\[
DS(d_2) = \frac{(1 - |0.7 - 0.5|) + (1 - |0.6 - 0.8|) + (1 - |0 - 0|) + (1 - |0.2 - 0|)}{4} = \frac{0.34}{4} = 0.85,
\]
A New Fuzzy Information Retrieval Method

\[ DS(d_3) = \frac{(1 - |0.9 - 0.5|) + (1 - |1 - 0.8|) + (1 - |0.1 - 0|) + (1 - |0 - 0|)}{4} = \frac{0.33}{4} = 0.825. \]

Therefore, the order of the three retrieved documents according to their degrees of satisfaction with respect to the user’s query \( q \) from high to low is \( d_1 > d_2 > d_3 \). Assume that document \( d_1 \) and document \( d_2 \) are judged by the user as irrelevant documents and document \( d_3 \) is judged as a relevant document. Then, according to the proposed document terms reweighting algorithm, we can get a document descriptor modification vector \( \Delta \) shown as follows:

\[ \Delta = (-0.2, -0.1, 0, 0). \]

Therefore, the modified document descriptor vectors of the three retrieved documents are as follows:

\[ \overrightarrow{d_1} = (0.2, 0.5, 0, 0), \]
\[ \overrightarrow{d_2} = (0.5, 0.5, 0, 0.2), \]
\[ \overrightarrow{d_3} = (0.7, 0.9, 0.1, 0). \]

Based on formula (3), we can get

\[ DS(d_1) = \frac{(1 - |0.2 - 0.5|) + (1 - |0.5 - 0.8|) + (1 - |0.1 - 0|) + (1 - |0 - 0|)}{4} = \frac{0.33}{4} = 0.825, \]
\[ DS(d_2) = \frac{(1 - |0.5 - 0.5|) + (1 - |0.5 - 0.8|) + (1 - |0 - 0|) + (1 - |0.2 - 0|)}{4} = \frac{0.35}{4} = 0.875, \]
\[ DS(d_3) = \frac{(1 - |0.7 - 0.5|) + (1 - |0.9 - 0.8|) + (1 - |0.1 - 0|) + (1 - |0 - 0|)}{4} = \frac{0.36}{4} = 0.9. \]

Therefore, the order of the three retrieved documents according to their degrees of satisfaction with respect to the user’s query \( q \) from high to low is \( d_3 > d_2 > d_1 \). We can see that the only one relevant document (i.e., document \( d_3 \)) is moved from the last place of the original document list to the first place of the new document list.
4. Experimental Results

We have implemented the proposed terms reweighting algorithm for document retrieval on a Pentium 4 PC using Delphi Version 5.0 [5]. We choose 247 research reports [20] as the set of documents for clustering, which are a subset of a collection of research reports of the National Science Council, Taiwan, Republic of China. Each report consists of several parts, including a report ID, a title, researchers’ names, a Chinese abstract, an English abstract, ..., etc. Since the proposed method intends to deal with English documents, the system grabs the English abstracts of the reports to represent the contents of the documents. The automatic document indexing method described in Section 2 is used to represent the contents of the documents by the index terms extracted from the set of documents containing the 247 reports [20]. The documents are then represented by document descriptor vectors which are used for further user’s query processing.

Ten queries as shown in the first column of Table 2 are submitted to the information retrieval system, where the query threshold value $\alpha$ given by the user is 0.4. Then, a set of documents is retrieved and the retrieved documents are sorted according to their degrees of satisfaction with respect to the queries. The values of $RDRS$ of each set of retrieved documents are calculated and recorded as shown in the second column of Table 2. Then, for each query, the retrieved documents are judged by the user as relevant or irrelevant. Based on the user’s relevance feedback, the system modifies the document descriptor vectors of the retrieved documents by using the proposed document terms reweighting algorithm presented in Section 3 and calculates their new degrees of satisfaction with respect to the query. The retrieved documents for each query are resorted according to their new degrees of satisfaction with respect to the query. The values of $RDRS$ of each set of retrieved documents are calculated and recorded again as shown in the third column of Table 2. From Table 2, we can see that by modifying the document descriptor vectors using the proposed document terms reweighting algorithm, most of the $RDRS$ values of the retrieved documents of the 10 queries have improved except the ones of the second query and the fifth query. It is because the $RDRS$ values of the retrieved documents using the original document descriptor vectors with respect to the second query and the fifth query, respectively, are larger than 3, which means that most of the relevant documents are listed in the front part of the document list, and there is no much room for further improvement.
Table 2. RDRS values of the retrieved documents.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_1 = \text{natural language processing}$ (The Weights of the Terms “natural”, “language” and “processing” are 0.8, 0.8 and 0.7, Respectively)</td>
<td>2.246</td>
<td>2.659</td>
</tr>
<tr>
<td>$q_2 = \text{fuzzy set}$ (The Weights of the Terms “fuzzy” and “set” are 0.8 and 0.8, Respectively)</td>
<td>3.010</td>
<td>3.010</td>
</tr>
<tr>
<td>$q_3 = \text{heterogeneous database}$ (The Weights of the Terms “heterogeneous” and “database” are 0.9 and 0.8, Respectively)</td>
<td>3.018</td>
<td>3.060</td>
</tr>
<tr>
<td>$q_4 = \text{database management}$ (The Weights of the Terms “database” and “management” are 0.8 and 0.7, Respectively)</td>
<td>1.136</td>
<td>2.599</td>
</tr>
<tr>
<td>$q_5 = \text{expert system}$ (The Weights of the Terms “expert” and “system” are 0.9 and 0.7, Respectively)</td>
<td>3.173</td>
<td>3.173</td>
</tr>
<tr>
<td>$q_6 = \text{image processing}$ (The Weights of the Terms “image” and “processing” are 0.8 and 0.7, Respectively)</td>
<td>1.905</td>
<td>2.368</td>
</tr>
<tr>
<td>$q_7 = \text{machine learning}$ (The Weights of the Terms “machine” and “learning” are 0.8 and 0.8, Respectively)</td>
<td>2.413</td>
<td>3.131</td>
</tr>
<tr>
<td>$q_8 = \text{object oriented database}$ (The Weights of the Terms “object”, “oriented” and “database” are 0.9, 0.9 and 0.8, Respectively)</td>
<td>2.782</td>
<td>3.030</td>
</tr>
<tr>
<td>$q_9 = \text{image restoration}$ (The Weights of the Terms “image” and “restoration” are 0.8 and 0.8, Respectively)</td>
<td>3.111</td>
<td>3.141</td>
</tr>
<tr>
<td>$q_{10} = \text{multimedia database}$ (The Weights of the Terms “multimedia” and “database” are 0.9 and 0.8, Respectively)</td>
<td>2.050</td>
<td>3.112</td>
</tr>
</tbody>
</table>
Since a large RDRS value of the retrieved document indicates that most of the relevant documents are in the front part of the ordered document list, it will result in a better retrieval effectiveness if the user only considers the top ranked documents of the ordered document list. For the 10 queries submitted to the information retrieval system, assume that the user considers only the top 10 documents of each set of the retrieved documents with respect to the 10 queries. A comparison of the recall rates of the top 10 retrieved documents with respect to each query using the original document descriptor vectors with the ones using the modified document descriptor vectors is shown in Figure 9. A comparison of the precision rates of the top 10 retrieved documents with respect to each query using the original document descriptor vectors with the ones using the modified document descriptor vectors is shown in Figure 9. From Figure 9 and Figure 10, we can see that when the system uses the modified document descriptor vectors, it can get a higher or the same precision rate and recall rate regarding the top 10 retrieved documents than those using the original document descriptor vectors.

![Graph](image)

**Figure 9.** The recall rate of the top 10 documents with respect to each user’s query.
5. Conclusions

In this paper, we have presented a new method for fuzzy information retrieval based on document terms reweighting techniques to modify the weights of document terms in document descriptor vectors based on the user’s relevance feedback. After modifying the weights of document terms in document descriptor vectors, the degrees of satisfaction of relevant documents with respect to the user’s query will increase, and the degrees of satisfaction of irrelevant documents with respect to the user’s query will decrease. The modified document descriptor vectors then can be used as personal profiles for future query processing. The proposed method can make fuzzy information retrieval systems more flexible and more intelligent to deal with documents retrieval. It can increase the retrieval effectiveness of the fuzzy information retrieval systems for document retrieval.

Acknowledgements

The authors would like to thank the referees for providing very helpful comments and suggestions. Their insight and comments led to a better presentation of the ideas expressed in this paper. This work was supported in part by the National Science Council, Republic of China, under Grant NSC-91-2213-E-011-052.

References