Expert Systems with Applications 36 (2009) 12023–12035

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

A new document representation using term frequency and vectorized graph connectionists with application to document retrieval

Tommy W.S. Chow *, Haijun Zhang, M.K.M. Rahman

Department of Electronic Engineering, City University of Hong Kong, 83 Tat Chee Avenue, Kowloon, Hong Kong

ARTICLE INFO

Keywords:
Graph representation
Multiple features
Document retrieval
Self-organizing map

ABSTRACT

This paper presents a new document representation with vectorized multiple features including term frequency and term-connection-frequency. A document is represented by undirected and directed graph, respectively. Then terms and vectorized graph connectionists are extracted from the graphs by employing several feature extraction methods. This hybrid document feature representation more accurately reflects the underlying semantics that are difficult to achieve from the currently used term histograms, and it facilitates the matching of complex graph. In application level, we develop a document retrieval system based on self-organizing map (SOM) to speed up the retrieval process. We perform extensive experimental verification, and the results suggest that the proposed method is computationally efficient and accurate for document retrieval.

1. Introduction

Internet access, such as World Wide Web (WWW), has made document retrieval increasingly demanding as collection and searching of documents has become an integral part of many people's lives. Accuracy and speed are two key measurements of effective retrieval methodologies. Existing document retrieval systems use statistical methods and natural language processing (NLP) approaches combined with different document representation and query structures. Document retrieval has created many interests in the information retrieval community. Document retrieval refers to finding similar documents for a given user's query. A user's query can be ranged from a full description of a document to a few keywords. Most of the extensively used retrieval approaches are keywords based searching methods, e.g., www.google.com, in which untrained users provide a few keywords to the search engine finding the relevant documents in a returned list. Another type of document retrieval is to use a query document to search similar ones. Using an entire document as a query performs well in improving retrieval accuracy, but it is more computationally demanding compared with the keywords based method. In addition to retrieval task, document classification and clustering has also become important in organizing the massive amount of document data, which also uses similar feature extraction approaches to facilitate the classification and clustering process. Until now, most conventional models use rough document features, such as terms in documents as feature units. Usually the connections among terms are overlooked which results in losing important semantic information of documents. Thus, there is a need of developing more effective document representation scheme to enhance the performance of relevant document data mining.

Most currently used methods of document representation in text data mining are based on vector space, probabilistic and language models. The vector space model (VSM) (Salton & McGill, 1983), the most popular and widely used *tf-idf* scheme, uses a basic vocabulary of "words" or "terms" for feature description. The term frequency (*tf*) is the number of occurrences of each term, and the inverse document frequency (*idf*) is a function of the number of document where a term took place. A term weighted vector is then constructed for each document using *tf* and *idf*. Similarity between two documents is then measured using 'cosine' distance or any other distance functions (Zobel & Moffat, 1998). Thus, this VSM scheme reduces arbitrary length of term vector in each document to fixed length. But a lengthy vector is required for describing the frequency information of terms, because the number of words involved is usually huge. This causes a significant increase of computational burden making the VSM model impractical for large corpus. In addition, VSM scheme reveals little statistical structure about a document. To overcome these shortcomings, researchers have proposed several dimensionality reduction methods such as latent semantic indexing (LSI) (Deerwester & Dumais, 1990), probabilistic latent semantic indexing (PLSI) (Hofmann, 1999), latent Dirichlet allocation (LDA) (Blei, Ng, & Jordan, 2003) and exponential family harmonium model (EFHM) (Welling, Rosen-Zvi, & Hinton, 2004). LSI maps the documents and terms to a latent space representation by employing a linear projection to compress the feature vector of the VSM model into low dimension. In addition

* Corresponding author.
E-mail address: eetchow@cityu.edu.hk (T.W.S. Chow).

0957-4174/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2009.03.008

to feature compression, the LSI model is useful in encoding the semantics (Berry, Dumais, & O'Brien, 1995). A step forward in probabilistic models is PLSI that defines a proper generative model of data to model each word in a document as a sample from a mixture distribution and develop factor representations for mixture components. By realizing overfitting problems and lack of description at the level of documents, Blei et al. (2003) introduced a further extension in this regard, latent Dirichlet allocation. LDA is viewed as a three-level hierarchical Bayesian model, in which each document is modeled as a finite mixture over an underlying set of topics. Using probabilistic approach then provides an explicit representation of a document. Compared with LDA, exponential family harmonium model is an alternative two-layer model using exponential family distributions and the semantics of undirected models. EFHM is able to reduce the feature dimension significantly using a few latent variables to represent a document. Apart from these probabilistic models, language model (Ponte & Croft, 1998), an alternative to the traditional *tf.idf* relevance models, has become quite popular. The relevance of a document to a given query is ranked by using statistical techniques and underlying language model of the document. Moreover, Erkan (2006) introduced a language model-based document representation using random walks for document clustering.

Speeding up a retrieval system for real time application is another equally important issue. In addition to the above mentioned techniques for reducing feature dimension that are able to improve the retrieval speed in some extent, other attempts such as clustering methods and novel system structures are suggested to speed up the retrieval operation. Rooney, Patterson, Galushka, and Dobrynin (2006) introduced an idea that clustering of large document corpus could be used for speeding up document retrieval. In order to reduce the searching effort, the scheme is to narrow the searching scope by comparing a query to a group of documents that are clustered according to the document nature. Fuzzy concept employing clustering techniques (Horng, Chen, Chang, & Lee, 2005; Rldvan, Tutuncu, & Allahverdi, 2007) was also used in document retrieval. Other than clustering techniques, a new file structure (Du, Ghanta, Maly, & Sharrock, 1989) was also suggested to speed up the retrieval process.

Despite the progress on the area of document retrieval, most reported techniques are largely based on typical term frequency information of "bag of words" model. This approach ignores the connections among terms. In all the above mentioned approaches, it is noticed that they all use independent word as feature unit. These feature schemes are a rough representation of a document. For example, two documents containing similar term frequencies may be contextually different when the spatial distribution of terms are very different, i.e., *school*, *computer*, and *science* means very different when they appear in different parts of a document compared to the case of *school of computer science* that appear together. In addition, with the evolution of natural language, there are increasing combinatorial words emerged such as *computer network*, *neural network*, *complex network*, *nature network*, etc. Thus, only using term frequency information from the "bag of words" model is not the most effective way to account contextual similarity that includes the word inter-connections and spatial distribution of words throughout the document. The semantics may be very different whether considering the term-connections or not.

To address the above shortcomings and improve the retrieval accuracy, we in this paper introduce graphs for document representation that resulting in more semantic information to be included. It is worth mentioning that graph representation for document is not new. An interesting application of graph representation describing words links with a perspective of evolving complex network for human language study can be found in Dorogovtsev and Mendes (2001), Cancho and Sole (2001). In Schenker, Last, Bunke, and Kandel (2003), Schenker, Last, Bunke, and Kandel (2004), different directed graphs with a few most frequent terms as nodes were defined to represent a document, k-Nearest Neighbor algorithm (k-NN) with different graph matching distances based on maximum common subgraph was applied to web document classification. Although it is quite successful to enhance the classification accuracy, graph matching can be accomplished in polynomial time making it impractical for large data sets. Apart from the computation time limitation, there may be difficulties in finding maximum common subgraph (subgraph isomorphism) between two documents. It is much difficult to define reasonable common subgraphs that are able to catch the semantic similarity between documents because portions in documents are usually not exactly similar (generally, quite few parts are similar). In this paper, in order to avoid time consuming matching process, first, we extract term-connections from graph representations with extensive feature extraction methods. Each document is then projected into feature vector space forming term-connection-frequency (*tcf*) together with term frequency (*tf*). This approach enables more semantic information to be utilized for document data mining. In application to retrieval process, we employ SOM to accelerate the searching operations by matching each document to topologically ordered neurons, and we further use query feedback to improve the retrieval accuracy. In summary, the contribution of this paper is twofold. First, we propose a new composite vector for representing a document combined with traditional term frequency and term-connection-frequency extracted from graphs. Multiple features are able to express more semantic information of the word inter-connections and spatial distribution of words throughout the document. As a result, it enhances the document retrieval accuracy. Second, Vectorized graph connectionists facilitate the matching of complex graph especially when the system handles large datasets. The vectorized graph uses a fixed length vector, resulting in substantial reduction in computational cost. We employ SOM together with relevance feedback approach to improve the computational efficiency and the document retrieval accuracy. The method using vectorized multiple features can serve as a unified feature extraction framework for performing both document retrieval and document classification.

The remaining sessions of this paper are organized as follows. Undirected and directed graph representations of documents are introduced in Section 2. In Section 3, three extraction schemes for term-connection features are described in details, multiple features are projected into vector space. Section 4 presents the SOM implementation for multiple features of documents. SOM based retrieval system is described in Section 5. Extensive simulation results followed by discussions are presented in Section 6. The paper ends with conclusions and future work propositions in Section 7.

2. Graph representations of documents

2.1. Directed graph representation

In our work, we use graphs to represent each document in corpus. It is quite straightforward to apply directed graph to express the semantics using terms in sequence appearing in the document. First, we remove the stop words (set of common words such as "in", "the", "are", etc.) which deliver little discriminate information. Then, we use the rest of terms to form a directed graph. A directed graph $\vec{G}$ for a document is denoted by $\vec{G} = (\vec{V}, \vec{E}, \vec{\phi}, \vec{\theta})$, where, $\vec{V}$ represents a set of vertices (i.e. terms), $\vec{E}$ is a set of edges or connections between terms, $\vec{\phi} : \vec{V} \to \vec{L}_V$ assigns an attribute (i.e. term frequency) to each vertex of $\vec{V}$, similarly, $\vec{\theta} : \vec{E} \to \vec{L}_E$ assigns an attribute (i.e. term-connection-

frequency) to each edge of $\vec{E}$. For example, Fig. 1 illustrates how such a graph would look like for a sentence "*we found it significantly more expensive for sending money to Mexico, but slightly less for sending money to the United Kingdom*". Note that we use only a single vertex for each term even if a term appears more than once in the document. In an early implementation, we used a single vertex to represent a term chain consisting of two and three words that appear together throughout the document, but later found that using only a single vertex for each term is sufficient and improves the performance of our application. Each vertex is labeled with term frequency measure that indicates how many times the related term appears in the document. Similarly, each edge is labeled with term-connection-frequency measure that indicates how many times the connected terms appear together in the document. Here, "connected" means that two terms are adjacent to each other in the specified sequence in the document.

2.2. Undirected graph representation

Using a directed graph according to the sequence of words is able to represent the semantics of a document. In many cases the sequence of words is convertible, although it conveys the same semantics for human language. For example, "*computer science*" can be expressed as "*science of computer*", which delivers the same meaning. Thus, in this paper we propose to use an undirected graph for representation of each document. Similarly, first we remove the stop words, and develop an undirected graph G for each document denoted by $G = (V, E, \phi, \theta)$ where notations are similar to directed graph G. Fig. 2 illustrates the same example discussed in the previous section by using undirected graph. Likewise, each vertex is labeled with term frequency measure that indicates how many times the related term appears in the document. Each edge is labeled with term-connection-frequency measure that indicates how many times the connected terms appear together in the document. Here, "connected" means that two terms are adjacent to each other without differing the word sequence.

3. Multiple features extraction

In this section, we describe the multiple features (terms and term-connections) extraction approaches to extract more information from each document for better document analysis.

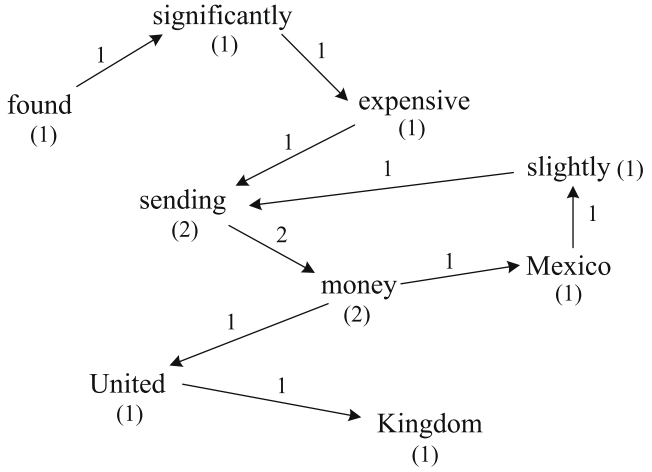

Fig. 1. Directed graph as an example: "*we* found *it* significantly *more* expensive *for* sending money *to* Mexico, *but* slightly *less for* sending money *to the* United Kingdom. (Here, 'we', 'it', 'more', 'for', 'to', 'but', 'less', 'for', 'the' are stop words that are removed.).

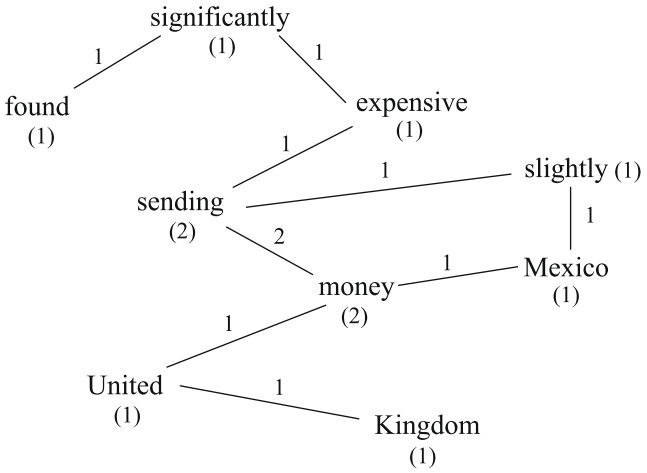

Fig. 2. Undirected graph as an example: "*we* found *it* significantly *more* expensive *for* sending money *to* Mexico, *but* slightly *less for* sending money *to the* United Kingdom (here, 'we', 'it', 'more', 'for', 'to', 'but', 'less', 'for', 'the' are stop words that are removed).

3.1. Term-frequency-based feature extraction

First, extract all the words from all documents except for stop words in a database and apply stemming algorithm to each word. Here, Porter stemming algorithm (Porter, 1980) is applied to extract stem of each word, and stems are used as basic features instead of original words. Thus, "send", "sent" and "sending" are all considered the same word. Store the stemmed words together with the information of term-frequency f_t and the document-frequency f_d^t. Then, construct the vocabulary based on term-frequency features. We use a term-weighting measure in calculating the weight of each word, which is similar to VSM (Salton & Buckley, 1996)

$$W_t = \sqrt{f_t} \times idf, \tag{1}$$

where the inverse-document-frequency $idf = \log_2\left(\frac{N}{f_d^t}\right)$, and N is the total number of documents in the corpus. Then, the words are sorted in descending order according to the weights and the first N_t words are selected to construct the vocabulary. The choice of N_t depends on the database.

3.2. Term-connection-frequency-based feature extraction

Feature extraction of terms-connection-frequency is based on the word vocabulary, which is constructed in Section 3.1. We use terms in the word vocabulary to build a directed or undirected graph for each document. Based on graph representations, if we directly use graph matching methods to calculate the semantic similarity like references (Schenker et al., 2004), much time and storage space will be wasted for large datasets because the adjacent matrix of each document is so sparse. The adjacent matrix A^k ($k = 1, 2, \ldots, N$) for graph G^k (or $\vec{G}^k$) is denoted by $A^k = [A_{ij}^k]_{N_t \times N_t}$ where $A_{ij}^k = f_{ij}^{k,tc}$ represents the term-connection-frequency between term i and term j in document k. Then, we calculate the total term-connection-frequency adjacent matrix for all the documents (i.e. $A = \sum_{k=1}^{N} A^k$). We also store the document frequency $f_{d,ij}^{tc}$ for term-connection between term i and term j in the database. We then use three schemes to extract the term-connection-frequency to construct a term-connection based vocabulary.

The first one, the simplest way, is top term-connection-based method that is to select the most frequent N_{tc} term-connections from matrix A. Second, we use the same weighting measure to calculate the weight of each term-connection for a pair of terms

$$W_{ij}^{tc} = \sqrt{f_{ij}^{tc}} \times idf_{ij}^{tc}, \tag{2}$$

where the inverse-document-frequency $idf_{ij}^{tc} = \log_2\left(\frac{N}{f_{d,ij}^{tc}}\right)$. Then, we sort the term-connections by using the weights in descending order and select the first N_{tc} term-connections. Finally, we use a similar entropy-based measure (Lochbaum & Streeter, 1989) to weight each term-connection

$$\text{entropy}_{ij} = \sum_{k=1}^{N} \frac{A_{ij}^{k}}{f_{d,ij}^{tc}} \log_2\left(\frac{f_{d,ij}^{tc}}{A_{ij}^{k}}\right). \tag{3}$$

3.3. Projection to vector space

After features are extracted from documents, we can use appropriate data reduction methods to obtain lower dimensional feature due to 'the curse of dimension'. Here, we apply PCA, a popular tool to project higher dimensional data into lower dimensional feature without losing much statistical information, to construct document histogram vector combined with term-frequency feature and term-connection-frequency feature. The overall procedures of extracting multiple features are summarized as follows.

(1) Extract words from all the documents in the corpus excepting for stop words and apply stemming to each word. Calculate the weight of each word according to Eq. (1), and select the first N_t words to construct term-frequency-based vocabulary.
(2) Build graph for each document using selected words as nodes and calculate the total adjacent matrix A. Select the first N_{tc} term-connections (or the indexes of edges in graph) based on above mentioned three schemes (i.e. top tcf-based, weighted tcf-based, and entropy tcf-based) to construct the term-connection-frequency-based vocabulary.
(3) Calculate term histograms and term-connection histograms for documents that represent the multiple features of documents. Each element of the histograms indicates the number of times that the corresponding term or term-connection appears in a document. Finally, we normalize the histogram as follows.

$$H = \left[\, h_1^t \quad h_2^t \quad \cdots \quad h_{N_t}^t \quad h_1^{tc} \quad h_2^{tc} \quad \cdots \quad h_{N_{tc}}^{tc} \,\right], \quad \left(h_i^t = f^t(n_i^t), h_j^{tc} = f^{tc}(n_j^{tc})\right), \tag{4}$$

where n_i^t is the frequency of ith term in the vocabulary, similarly, n_j^{tc} is the frequency of jth term-connection in the vocabulary, $f^t(n_i^t)$ and $f^{tc}\left(n_j^{tc}\right)$ are normalization functions. In fact, there are many alternative normalization approaches such as mean normalization, maximum normalization, or traditional *tf-idf* (i.e. VSM) normalization. In this paper, we use *tf-idf* normalization method on the histogram vector

$$h_i^t = \frac{n_i^t}{\sum_{i=1}^{N_t} n_i^t} \times \log_2\left(\frac{N}{(f_t)_i}\right), \qquad h_j^{tc} = \frac{n_j^{tc}}{\sum_{j=1}^{N_{tc}} n_j^{tc}} \times \log_2\left(\frac{N}{(f_d^{tc})_j}\right). \tag{5}$$

(4) Use the normalized histogram to construct the PCA projection matrix. We use the MATLAB toolbox (Esben, 2005) to compute the PCA projection matrix.
(5) Project the normalized histogram into the lower dimensional PCA feature by using PCA projection matrix. The PCA features are computed as follows

$$F_h = \left[\, F_h^t \quad F_h^{tc} \,\right], \quad F_h^t = \left[\, h_1^t \quad h_2^t \quad \cdots \quad h_{N_t}^t \,\right] * B^t, \quad F_h^{tc} = \left[\, h_1^{tc} \quad h_2^{tc} \quad \cdots \quad h_{N_{tc}}^{tc} \,\right] * B^{tc}. \tag{6}$$

where B^t and B^{tc} are the projection matrixes with dimension $N_t \times m_F^t$ and $N_{tc} \times m_F^{tc}$, respectively, m_F^t and m_F^{tc} are the dimension of the projected feature from term-frequency and term-connection-frequency respectively. The projected features in F_h are ordered according to their statistical importance. In our application, both m_F^t and m_F^{tc} are set to be 100, respectively.
(6) Save the projection matrixes for making the features of a new query document. The multiple features of a query document are extracted in the same way except for steps (1), (2) and (4) that are computed only once over the database.

The above extracted hybrid feature vector can actually be used in various document applications such as document classification, categorization and retrieval. In this paper, we only used document retrieval as an application example to provide insights to the spatial structure of documents and exhibit its performance. Other applications will be easily combined to this framework using above hybrid document feature.

4. SOM implementation for document retrieval

Self-organizing map (SOM) (Kohonen, 1997) is a versatile unsupervised neural network used for dimension reduction, vector quantization and visualization. It is able to preserve a topologically ordered output map, where input data are mapped into a small number of neurons. In this way, SOM can form a mapping from document corpus to topologically ordered neurons. There are many attempts to employ SOM for document feature projection to reduce the data dimensionality (Ampazis & Perantonis, 2004; Honkela, Kaski, Lagus, & Kohonen, 1997). SOM has been used for document organization and web mining (Antonio et al., 2008; Georgakis, Kotropoulos, Xafopoulos, & Pitas, 2004). Document clustering and browsing using SOM are introduced in Isa, Kallimani, and Lee (2008), Freeman and Yin (2005). In this paper, we employ SOM to speed up the retrieval process.

SOM consists of M neurons located at a regular low dimensional grid that is usually in a 2-D grid. The lattice of the grid is either hexagonal or rectangle. The SOM algorithm is iterative. Each neuron i contains a d-dimensional feature vector $w_i = [\, w_{i1} \quad w_{i2} \quad \cdots \quad w_{id} \,]^T$. At each training step t, a sample data vector $x(t)$ is randomly chosen from a training set. Distances between $x(t)$ and all the feature vectors in the grid are computed. The winning neuron, denoted by c, is the neuron with the feature vector closest to $x(t)$

$$c = \arg\max_i (F(x(t), w_i)), \quad i \in \{1, 2, \ldots, M\}, \tag{7}$$

where $F(\cdot)$ is distance function to compute the similarity between $x(t)$ and w_i. In this paper, we use *cosine* distance to define the similarity

$$F(x(t), w_i) = C \cdot \frac{X^t \cdot W^t}{\|X^t\| \cdot \|W^t\|} + (1 - C) \cdot \frac{X^{tc} \cdot W^{tc}}{\|X^{tc}\| \cdot \|W^{tc}\|},$$

where

$$X^t = \left[\, (x(t))_1, \quad \cdots \quad (x(t))_{m_F^t} \,\right]^T, \quad W^t = \left[\, w_{i1}, \quad \cdots \quad w_{im_F^t} \,\right]^T, \quad X^{tc} = \left[\, (x(t))_{m_F^t+1}, \quad \cdots \quad (x(t))_{m_F^t+m_F^{tc}} \,\right]^T, \quad W^{tc} = \left[\, w_{i(m_F^t+1)}, \quad \cdots \quad w_{i(m_F^t+m_F^{tc})} \,\right]^T, \tag{8}$$

where $\cdot$ indicates the dot product operation, and $C (0 \leqslant C \leqslant 1)$ is a weight parameter to balance the importance of term-frequency feature and term-connection-frequency feature. The first part of the

expression computes the similarity based on *tf* feature, and the second part computes the similarity based on *tcf* feature. It is worth noting that the expression provides flexibility to users to change the value of C to balance this similarity measure according to their expectations. In this paper, the effect of the parameter C is also included and studied in Section 6.

Here, the neighborhood kernel function is taken as a Gaussian function, and the learning rate decreases monotonically with iteration. We do not include the details of the training process of SOM. Readers are referred to any standard book such as Kohonen (1997).

5. SOM-based retrieval system framework

5.1. Pre-processing of the retrieval system

The pre-processing for document retrieval can be summarized as follows.

(1) Save previously constructed vocabulary base and PCA projection matrix for multiple features.
(2) Train the SOM with all the documents of the database.
(3) Save the indexes of training data with their winning neurons, which is used for the document retrieval.
(4) Save the feature vectors (i.e. weights) of the SOM neurons. Save the SOM inputs of all documents constructed during training process, which is used for relevance feedback. Thus the trained SOM is ready to perform retrieval task for any new query document.

5.2. Document retrieval

A document is represented by a histogram vector combined with term-frequency feature and term-connection-frequency feature. Each document is indexed against its winning neuron at the grid of SOM. This association between document and neuron that is constructed by the pre-processing stage is prepared for document retrieval. The overall SOM-based retrieval system can be summarized as follows.

(1) For a given query document, extract its multiple features. Compute the projected feature using pre-stored vocabulary base and PCA projection matrix.
(2) Match the projected feature to find the most similar neurons on the SOM grid and return their attached documents.
(3) Go through the sorted neurons in descending order and add their attached documents into the retrieval list until at least user-defined N_{ret} documents are appended. Here, the total number of documents can be larger than N_{ret}.
(4) Sort the documents in the retrieval list by comparing the query according to Eq. (8). Return the first N_{ret} documents to users.

5.3. Relevance feedback

In order to make this system practical, we also provide an inference to allow users to browse through the preliminarily retrieved documents and give the relevance feedback to the retrieval system. In this study, we use query modification (Chow, Rahman, & Wu, 2006) for relevance feedback operation. The modified query data X_{new} is obtained by averaging all the features of the query and relevant documents

$$X_{new} = \frac{1}{N_R + 1}\left(X_q + \sum_{r=1}^{N_R} X_r\right), \tag{9}$$

where N_R is the number of relevant documents, X_q is the feature vector of the query document, and X_r is the feature vector of the rth relevant document. After query modification, the retrieval process can be summarized as follows.

(1) Match the modified query feature vector X_{new} with the neurons on the SOM grid.
(2) Sort the neurons in descending order according to distance with the query X_{new}.
(3) Go through the sorted neurons in descending order and add their attached documents into the retrieval list until at least user-defined N_{ret} documents are appended.
(4) Sort the documents in the retrieval list by comparing the new query with the preliminary retrieval results in step (3) and then return the first N_{ret} documents to users.

In fact, this relevance feedback process can be executed in several times. In this study, we use this process only once.

6. Experimental results and discussion

6.1. Database and experimental setup

In this study, the document database, "Html_CityU1", which consists of 25 categories, were used for all simulations. Each category includes 400 documents making a total number of 10,000 documents. The corpus was split into a training set and a test set that is used for query. One thousand test documents were randomly selected from the 25 categories, i.e. 25×40. The remaining 9000 documents were used for training. In order to provide a more real-life testing platform, we established this database consisting of documents with size ranged from few hundred words to over 20 thousand words. For each category, 400 documents were retrieved from "Google" using a set of keywords. Some of the keywords are shared among different categories, but the set of keywords for a category is different from that of other categories. The database can be found online at www.ee.cityu.edu.hk/~twschow/Html_CityU1.rar. Parameters of the SOM algorithm are set as follows. The size of the SOM grid was set at 30×30; the initial learning rate μ_0 was set to 0.3; the initial radius of the neighborhood function was set to half-length of the SOM square grid; the number of total training iterations was set to 27,000 (i.e. three epochs multiply 9000 documents); and the balance weight parameter C for term-frequency and term-connection-frequency feature was set to 0.75. All the above parameters were found to deliver good performance. But it was also noticed that a mild deviation from these settings would not have noticeable effect on the overall performance. After SOM training, the test set was used to verify the performance of this work. All the simulations were performed on a PC with Intel Core-2 2.13 GHz and 2 GB memory. The feature extraction programs were written in Java programming language, and all the document retrieval programs were tested in Matlab 7.1.

6.2. Results and discussion

In this section, we present our simulation results using multiples features and SOM-based retrieval system. In our comparative study, we compare the results of SOM with that of direct method by using single feature and multiple features. In direct method (that is similar to LSI), the query data with projected feature vector is compared directly with the documents in the training set by using cosine distance like Eq. (8). To quantify the retrieval results, we used averaged precision and recall values for each query document from the test set and retrieving unto 360 documents. The precision and recall measure are defined as follows

$$\text{Precision} = \frac{\text{No. of correctly retrieved documents}}{\text{No. of total retrieved documents}}, \quad (10)$$

$$\text{Recall} = \frac{\text{No. of correctly retrieved documents}}{\text{No. of total documents in relevant category}}. \quad (11)$$

First, we summarize the results by using undirected graph for document representation with different feature extraction approaches in Tables 1–3. Retrieved documents, the most similar training documents from the datasets for every query, vary from 1 to 360, the precision and recall values in the case of 10, 40, and 360 documents retrieved are listed. Performances of different retrieval approaches like SOM, RF in SOM, and direct method are compared in different feature extraction schemes. In order to show the effect of term-connection-frequency, we summarized the values of precision and recall with different number of retrieved documents when term-connection-frequency (*tcf*) is used as a single feature (i.e. balance weight parameter $C = 0$). We list the results with term-frequency (*tf*) as a single feature (i.e. $C = 1$).

From Table 1, based on top-frequency feature extraction method, it is observed that different retrieval tools using features combined *tf* with *tcf* achieve significant improvement of retrieval accuracy with 10 retrieved documents. The results based on the approach of multiple features consistently delivers better results than those using single feature (either *tf* or *tcf*) when the number of retrieved documents increases from 10 to 360. Similar results are also obtained for the recall measurement. On the other hand, through relevance feedback SOM delivers the best retrieval results among three different query types, and pure SOM approach performs better than direct method. For weighted-frequency based feature extraction method shown in Table 2, using multiple

Table 1
Retrieval results with top-frequency-based feature extraction method by using undirected graph for document representation.

Feature extraction scheme	Query type	Feature	No. of retrieved documents					
			10	40	360	10	40	360
			Precision (%)			Recall (%)		
Top-frequency-based method	SOM	*tf* + *tcf*	89.05	87.81	78.78	2.47	9.76	78.78
		tf	86.79	85.68	77.64	2.41	9.52	77.64
		tcf	79.56	75.69	58.24	2.21	8.41	58.24
	RF in SOM	*tf* + *tcf*	92.36	90.24	80.34	2.57	10.03	80.34
		tf	90.37	88.26	79.00	2.51	9.81	79.00
		tcf	82.78	77.91	61.36	2.30	8.66	61.36
	Direct method	*tf* + *tcf*	87.87	85.86	76.74	2.44	9.54	76.74
		tf	85.39	83.79	76.01	2.37	9.31	76.01
		tcf	79.48	75.68	58.13	2.21	8.41	58.13

Table 2
Retrieval results with weighted-frequency-based feature extraction method by using undirected graph for document representation.

Feature extraction scheme	Query type	Feature	No. of retrieved documents					
			10	40	360	10	40	360
			Precision (%)			Recall (%)		
Weighted-frequency-based method	SOM	*tf* + *tcf*	89.04	87.66	78.58	2.47	9.74	78.58
		tf	86.57	85.01	76.57	2.40	9.45	76.57
		tcf	80.88	76.73	59.65	2.25	8.53	59.65
	RF in SOM	*tf* + *tcf*	92.40	90.31	80.31	2.57	10.03	80.31
		tf	89.93	87.63	77.87	2.50	9.74	77.87
		tcf	83.71	78.77	62.81	2.33	8.75	62.81
	Direct method	*tf* + *tcf*	87.88	85.99	76.98	2.44	9.55	76.98
		tf	85.39	83.79	76.01	2.37	9.31	76.01
		tcf	80.90	76.83	59.54	2.25	8.54	59.54

Table 3
Retrieval results with entropy-based feature extraction method by using undirected graph for document representation.

Feature extraction scheme	Query type	Feature	No. of retrieved documents					
			10	40	360	10	40	360
			Precision (%)			Recall (%)		
Entropy-based method	SOM	*tf* + *tcf*	88.85	87.39	79.44	2.47	9.71	79.44
		tf	86.75	85.55	77.46	2.41	9.51	77.46
		tcf	73.50	70.27	53.14	2.04	7.81	53.14
	RF in SOM	*tf* + *tcf*	92.18	90.15	81.09	2.56	10.02	81.09
		tf	90.03	88.06	78.95	2.50	9.78	78.95
		tcf	77.87	72.97	56.41	2.16	8.11	56.41
	Direct method	*tf* + *tcf*	87.57	85.52	76.44	2.43	9.50	76.44
		tf	85.39	83.79	76.01	2.37	9.31	76.01
		tcf	73.09	69.40	53.17	2.03	7.71	53.17

features obtains about 2.5% improvement compared with only using *tf* feature. RF in SOM encouragingly delivers better performance with 92.40% precision and 2.57% recall for 10 documents retrieved. From Table 3, using multiple features consistently perform well based on entropy feature extraction approach. SOM and RF in SOM achieve 79.44% and 81.09% in precision measurement, respectively even for 360 documents retrieved.

Fig. 3 visually summarizes the precision results against number of retrieved documents and the precision results against recall values with top-frequency-based feature extraction scheme. Fig. 3 shows the results of using different features for document retrieval, i.e. single *tf* feature and combined *tcf* with *tf* feature, and it also shows the results of the first retrieval as well as the retrieval results after relevance feedback for each feature combination. It is observed that retrieval precision decreases with the increase of number of retrieved documents. RF with *tf* and *tcf* achieves about 98% precision when only one document retrieved. Using multiple features is able to obtain the highest improvement of accuracy when 20 documents are retrieved. The sharp slope approximates a right-angle in Fig. 3(b) due to the relationship between precision and recall measurement. Number of retrieved documents varying from 50 to 200 shows interesting results. The precision value decreases in an insignificant rate using *tf* and *tcf* features, whilst it increases slightly for the case of using only *tf*. Similar visual results are shown in Figs. 4 and 5 by using weighted-frequency-based feature extraction and entropy-based feature extraction scheme, respectively, with different feature combinations.

In summary, according to above quantitative and visual results, it is observed that using multiple features is able to obtain significant improvement of retrieval accuracy compared with using term-frequency feature only in all the cases. This indicates that the addition of term-connection feature is significant in providing discriminate information for document retrieval. Compared with single term-frequency feature, using multiple features improves

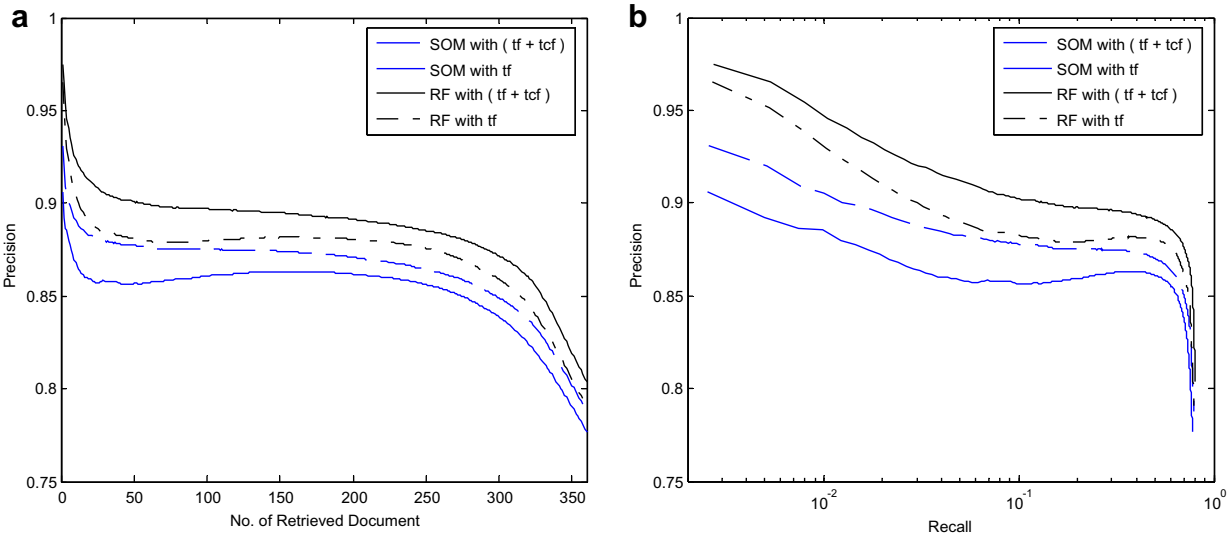

Fig. 3. Retrieval results based on top-frequency feature extraction using undirected graph: (a) precision against no. of retrieved document and (b) precision against recall.

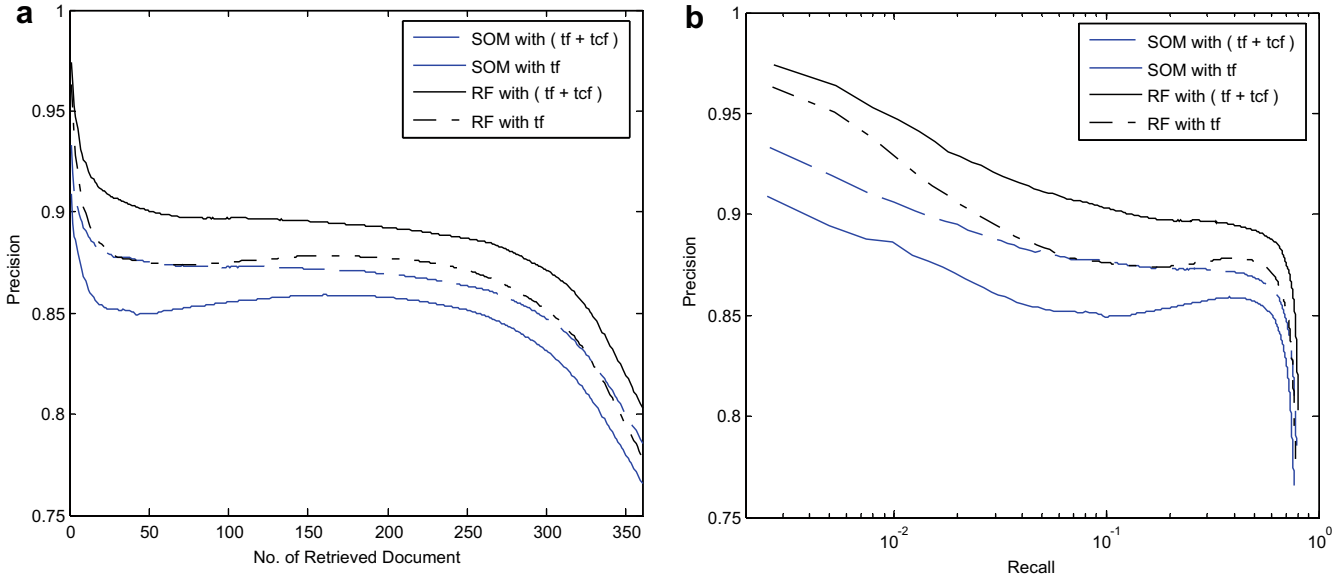

Fig. 4. Retrieval results based on weighted-frequency feature extraction using undirected graph: (a) precision against no. of retrieved document and (b) precision against recall.

at least 2% quantitatively in precision. It is more interesting to note that SOM approach delivers better performance than direct method after combining *tf* and *tcf* features compared with the case of using a single feature. This is believed to be mainly attributed to the choice of normalization methods (here, we used VSM normalization). The use of relevance feedback (i.e. RF-SOM) is useful as it is able to consistently improve the retrieval accuracy from about 2% to 3% compared with the first retrieval using SOM approach without relevance feedback. In our study, top-frequency-based method performs better than weighted-frequency-based method, when we used SOM approach without relevance feedback together with undirected graph for document representation. After relevance feedback is conducted, weighted-frequency-based method shows better results than top-frequency-based feature extraction scheme. Interesting results are observed for entropy feature extraction approach. Entropy-based method exhibits the worst performance compared with other two feature extraction approaches when a few documents are retrieved. As the number of retrieved documents increases to 360, entropy approach delivers the best performance compared to other two methods.

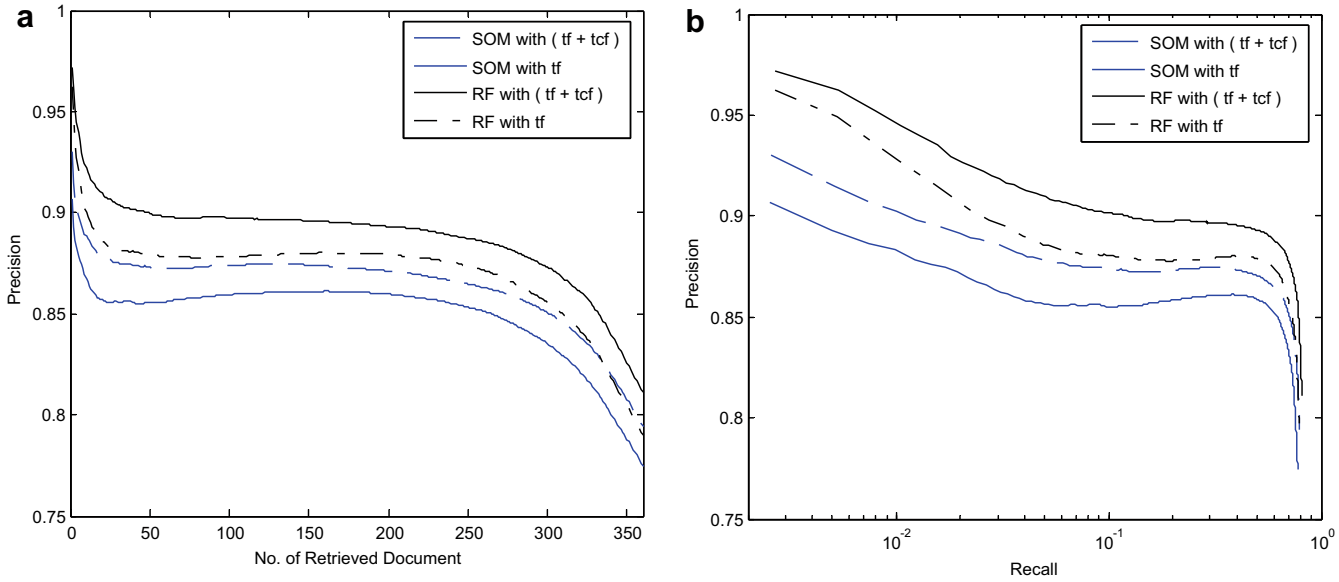

Fig. 5. Retrieval results based on entropy feature extraction using undirected graph: (a) precision against no. of retrieved document and (b) precision against recall.

Table 4
Retrieval results with top-frequency-based feature extraction method by using directed graph for document representation.

Feature extraction scheme	Query type	Feature	No. of retrieved documents					
			10	40	360	10	40	360
			Precision (%)			Recall (%)		
Top-frequency-based method	SOM	*tf* + *tcf*	89.68	88.03	78.29	2.49	9.78	78.29
		tf	86.62	85.41	77.34	2.41	9.49	77.34
		tcf	79.00	75.90	57.90	2.19	8.43	57.90
	RF in SOM	*tf* + *tcf*	92.41	90.08	80.13	2.57	10.01	80.13
		tf	89.98	87.93	78.68	2.50	9.77	78.68
		tcf	82.24	77.73	60.52	2.28	8.64	60.52
	Direct method	*tf* + *tcf*	88.23	86.44	76.79	2.45	9.60	76.79
		tf	85.39	83.79	76.01	2.37	9.31	76.01
		tcf	78.89	75.65	56.66	2.19	8.41	56.66

Table 5
Retrieval results with weighted-frequency-based feature extraction method by using directed graph for document representation.

Feature extraction scheme	Query type	Feature	No. of retrieved documents					
			10	40	360	10	40	360
			Precision (%)			Recall (%)		
Weighted-frequency-based method	SOM	*tf* + *tcf*	89.22	88.05	79.08	2.48	9.78	79.08
		tf	86.79	85.68	77.64	2.41	9.52	77.64
		tcf	79.85	76.68	59.18	2.22	8.52	59.18
	RF in SOM	*tf* + *tcf*	92.44	90.40	81.01	2.57	10.04	81.01
		tf	90.37	88.26	79.00	2.51	9.81	79.00
		tcf	82.76	78.34	61.81	2.30	8.70	61.81
	Direct method	*tf* + *tcf*	88.20	86.68	77.08	2.45	9.63	77.08
		tf	85.39	83.79	76.01	2.37	9.31	76.01
		tcf	79.85	76.41	58.36	2.22	8.49	58.36

The reported results using recall measurement also deliver the similar performance with the above comparative analysis by using different types of features and retrieval approaches.

Tables 4–6 summarize the retrieval results by using directed graph for document representation with different feature extraction approaches. Figs. 6–8 illustrate the visual retrieval results

Table 6
Retrieval results with entropy-based feature extraction method by using directed graph for document representation.

Feature extraction scheme	Query type	Feature	No. of retrieved documents					
			10	40	360	10	40	360
			Precision (%)			Recall (%)		
Entropy-based method	SOM	*tf* + *tcf*	88.78	87.67	79.28	2.47	9.74	79.28
		tf	87.07	85.65	77.11	2.42	9.52	77.11
		tcf	70.94	67.53	50.31	1.97	7.50	50.31
	RF in SOM	*tf* + *tcf*	92.19	90.08	80.86	2.56	10.01	80.86
		tf	89.99	87.94	78.53	2.50	9.77	78.53
		tcf	75.16	70.08	53.03	2.09	7.79	53.03
	Direct method	*tf* + *tcf*	87.46	85.77	76.59	2.43	9.53	76.59
		tf	85.39	83.79	76.01	2.37	9.31	76.01
		tcf	70.64	67.24	51.59	1.96	7.47	51.59

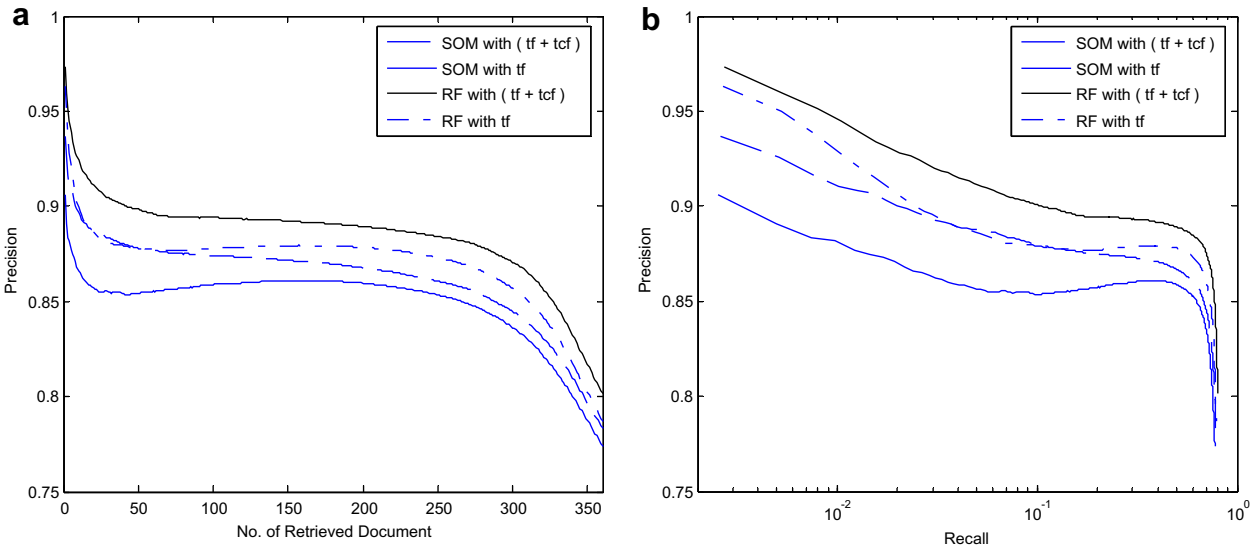

Fig. 6. Retrieval results based on top-frequency feature extraction using directed graph: (a) precision against no. of retrieved document and (b) precision against recall.

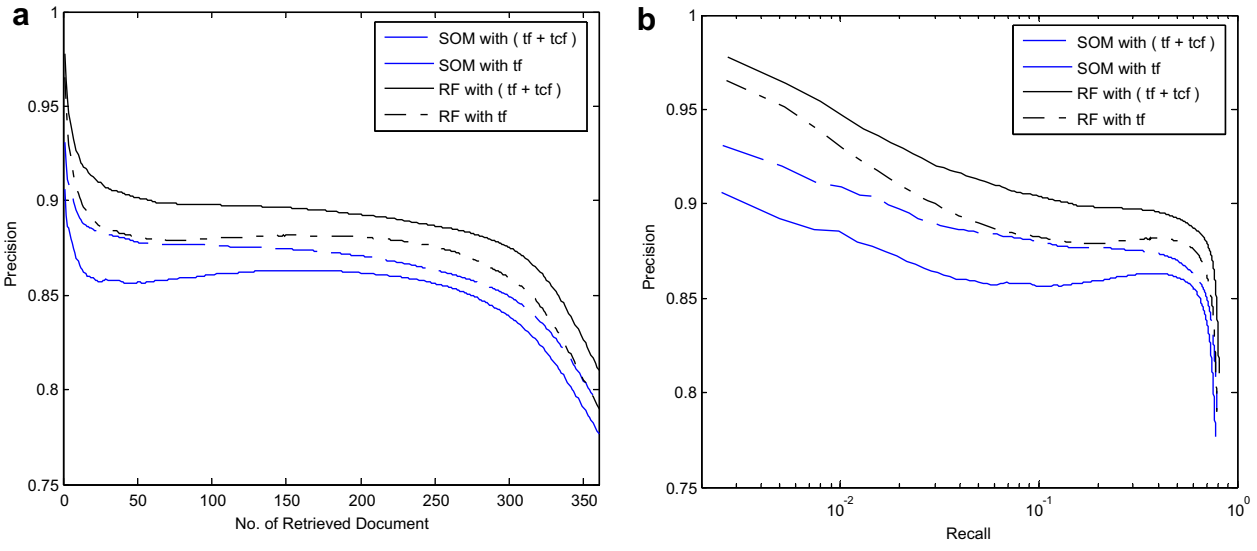

Fig. 7. Retrieval results based on weighted-frequency feature extraction using undirected graph: (a) precision against no. of retrieved document and (b) precision against recall.

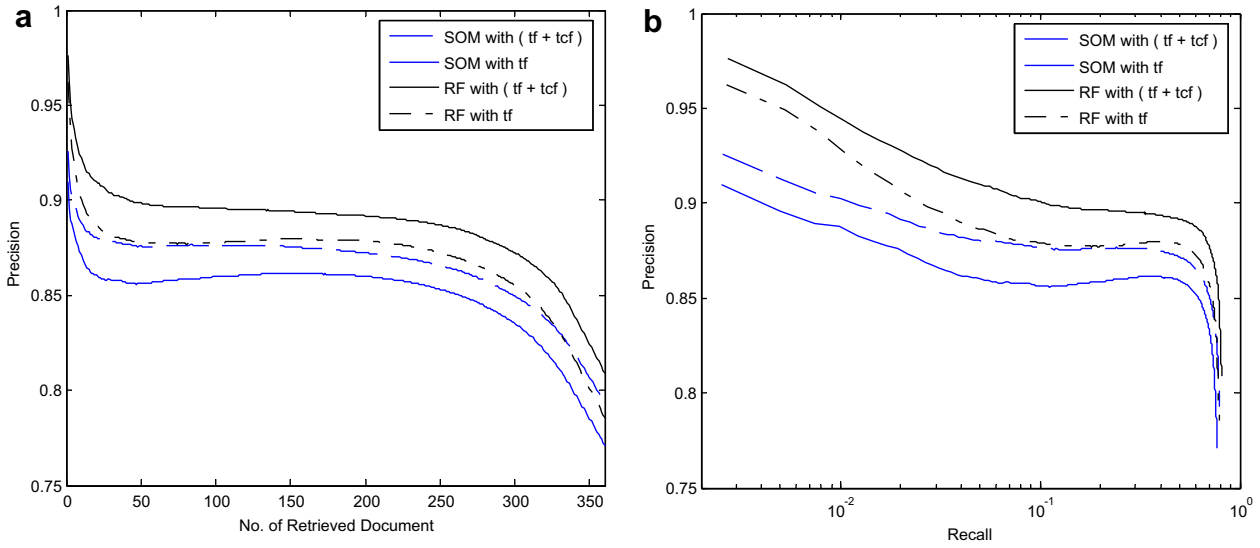

Fig. 8. Retrieval results based on entropy feature extraction using undirected graph: (a) precision against no. of retrieved document and (b) precision against recall.

Table 7
Query time with different methods.

Query type	SOM	RF in SOM	Direct method
Query time (s)	0.77	0.91	1.21

including different feature extraction methods. Similar comparative retrieval results are obtained by using SOM-based retrieval system. Using multiple features consistently performs better than using single feature with precision improvement from about 2% to 3%. SOM approach as well as relevance feedback exhibits better performance than direct method. For using directed graph for document representation, the top-frequency-based and the weighted-frequency-based feature extraction schemes deliver better retrieval results compared with entropy-based approach. It is observed that weighted-frequency-based feature extraction method exhibits better performance than top-frequency-based approach in directed graph representation. By comparative studies and analyzing the cases of different document representations, it is believed that using either undirected graph or directed graph is dependent on the testing datasets.

Our comparative study and analysis indicate that the superior performance delivered by SOM-based retrieval system is attributed to the co-existence of the SOM properties of self-organizing, topological ordering and non-linear projection. In order to show the searching speed of the retrieval system, Table 7 summarizes the average execution time from different retrieval approaches on the same simulation platform. It is obvious that the SOM approach is much faster than direct method with improvement of retrieval

Table 8
Retrieval results with SOM using undirected graph for document representation from 10-fold cross-validation sets.

Cross-validation set	Query type	No. of retrieved documents					
		10	40	360	10	40	360
		Precision (%)			Recall (%)		
1	SOM	89.31	86.99	78.02	2.48	9.67	78.02
	RF in SOM	92.02	89.11	79.76	2.56	9.90	79.76
2	SOM	87.39	85.47	76.89	2.43	9.50	76.89
	RF in SOM	90.58	88.24	79.31	2.52	9.80	79.31
3	SOM	89.12	87.45	78.94	2.48	9.72	78.94
	RF in SOM	92.48	90.49	80.93	2.57	10.05	80.93
4	SOM	89.91	88.55	78.93	2.50	9.84	78.93
	RF in SOM	92.78	90.65	80.67	2.58	10.07	80.67
5	SOM	86.57	84.76	76.47	2.40	9.42	76.47
	RF in SOM	90.49	88.10	78.90	2.51	9.79	78.90
6	SOM	88.79	87.24	78.29	2.47	9.69	78.29
	RF in SOM	91.97	89.80	80.29	2.55	9.98	80.29
7	SOM	89.64	88.21	78.54	2.49	9.80	78.54
	RF in SOM	92.66	90.18	80.37	2.57	10.02	80.37
8	SOM	87.99	86.42	78.00	2.44	9.60	78.00
	RF in SOM	91.09	88.76	79.92	2.53	9.86	79.92
9	SOM	88.08	86.88	78.41	2.45	9.65	78.41
	RF in SOM	91.53	89.68	80.49	2.54	9.96	80.49
10	SOM	89.04	87.66	78.58	2.47	9.74	78.58
	RF in SOM	92.40	90.31	80.31	2.57	10.03	80.31

speed of about 36%. Relevance feedback with SOM also improves the retrieval speed by about 25%. In order to ensure the generalization of the proposed system, we also examined the retrieval results from 10-fold cross-validation sets. Each validation set includes 1000 documents without any overlapped data. Tables 8 and 9 list the precision and recall results from 10-fold cross-validation sets by using different document representations. Similar results are obtained from 10 validation sets, which indicate the stability of the system. Finally, we studied the robustness of the proposed system over SOM initialization. We summarized the precision results in different training sessions in Tables 10 and 11 by using different graphs for document representations. It is observed that SOM approach is able to deliver similar retrieval results under different initializations.

Table 9
Retrieval results with SOM using directed graph for document representation from 10-fold cross-validation sets.

Cross-validation set	Query type	No. of retrieved documents					
		10	40	360	10	40	360
		Precision (%)			Recall (%)		
1	SOM	88.95	87.16	78.21	2.47	9.68	78.21
	RF in SOM	92.09	89.44	79.97	2.56	9.94	79.97
2	SOM	87.98	85.89	77.11	2.44	9.54	77.11
	RF in SOM	90.81	88.34	79.39	2.52	9.82	79.39
3	SOM	89.02	87.76	79.11	2.47	9.75	79.11
	RF in SOM	92.81	90.82	81.53	2.58	10.09	81.53
4	SOM	89.73	88.52	79.01	2.49	9.84	79.01
	RF in SOM	92.51	90.59	80.80	2.57	10.07	80.80
5	SOM	86.88	85.18	76.61	2.41	9.46	76.61
	RF in SOM	90.66	88.18	78.93	2.52	9.80	78.93
6	SOM	88.82	87.73	78.77	2.47	9.75	78.77
	RF in SOM	91.81	89.87	80.61	2.55	9.99	80.61
7	SOM	90.13	88.74	79.12	2.50	9.86	79.12
	RF in SOM	93.09	90.87	80.92	2.59	10.10	80.92
8	SOM	88.00	86.38	77.68	2.44	9.60	77.68
	RF in SOM	91.04	88.62	79.62	2.53	9.85	79.62
9	SOM	87.81	86.88	77.16	2.44	9.65	77.16
	RF in SOM	91.53	89.82	79.44	2.54	9.98	79.44
10	SOM	89.22	88.05	79.08	2.48	9.78	79.08
	RF in SOM	92.44	90.40	81.01	2.57	10.04	81.01

Table 10
Precision results with SOM from different training sessions using undirected graph for document representation (%).

Training sessions	Query type	No. of retrieved documents		
		10	40	360
1	SOM	89	88	79
	RF in SOM	92	90	80
2	SOM	89	88	79
	RF in SOM	92	90	81
3	SOM	89	88	79
	RF in SOM	92	90	81
4	SOM	89	88	78
	RF in SOM	93	90	80

Table 11
Precision results with SOM from different training sessions using directed graph for document representation (%).

Training sessions	Query type	No. of retrieved documents		
		10	40	360
1	SOM	89	88	79
	RF in SOM	92	90	81
2	SOM	89	88	79
	RF in SOM	92	90	80
3	SOM	90	88	79
	RF in SOM	93	91	81
4	SOM	89	88	78
	RF in SOM	92	90	80

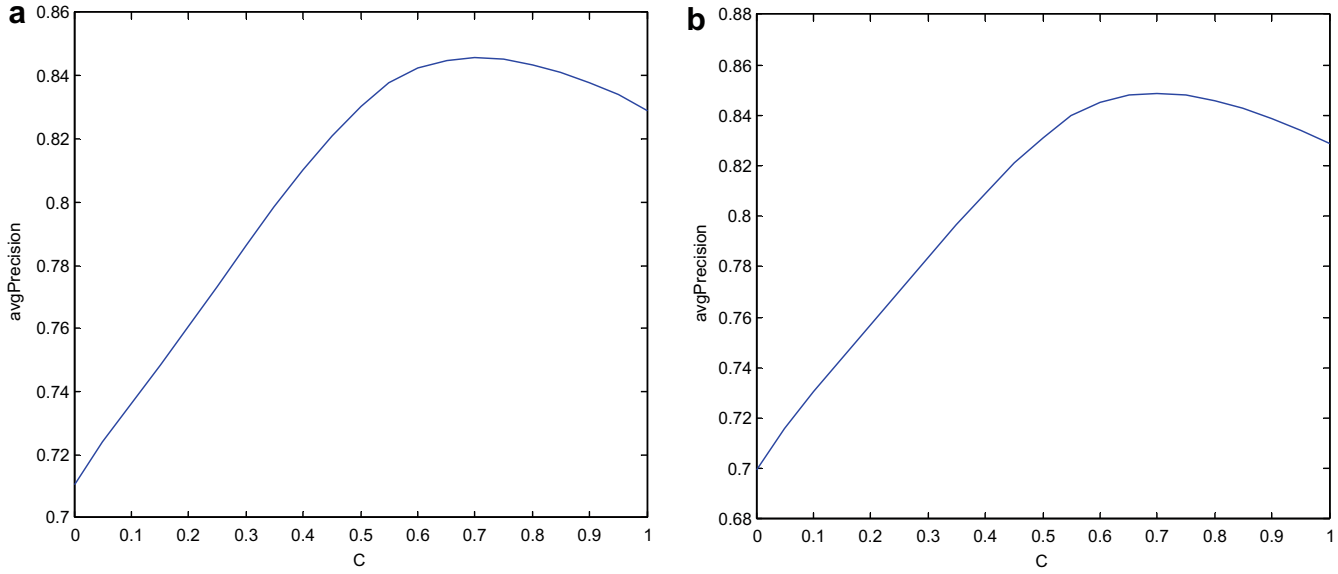

Fig. 9. Effect of weight parameter C: (a) for undirected graph and (b) for directed graph.

6.3. Parameter study

This section studies the effect of parameter of balance weight C. Based on the assumption that using combined multiple features is able to deliver superior performance, there must exist an optimal weight C to balance the effect of term-frequency feature and term-connection-frequency feature. We used the average precision measure to evaluate the effect of different value of C in Eq. (8). The average precision measure is defined as follows.

$$\text{avgPrecision}_C = \frac{\sum_{i=1}^{N_{max\,ret}} \text{precision}(i)}{N_{max\,ret}}, \tag{12}$$

where $N_{max\,ret}$ is the maximum number of retrieved documents, in our experiments, $N_{max,\,ret} = 360$, precision(i) is denoted as the average precision when retrieving i documents. We included the average precision results when $C = 0.00, 0.05, 0.10, \ldots, 1$ in Fig. 9. It is obvious that there exists an optimal C that is able to optimally combine the multiple features according to our experiments. The optimal value of C, around 0.7 in this study, is dependent upon different data sets. Users can specify the relative emphasis between term-frequency feature and term-connection-frequency feature by choosing an appropriate value of C according to the nature of documents.

7. Conclusion

A new document representation using multiple features including term frequency and vectorized graph connectionists is proposed. This representation turns complex graph matching process into a fixed length vector that contains more semantic information, and it can serve a unified feature extraction framework for various document mining tasks. Different feature extraction methods are extensively examined in this study. We then develop a SOM-based retrieval system in the application level of the new document representation. Experimental results show that vectorized multiple features extracted from different graphs of document representation are able to enhance the retrieval accuracy. SOM is used to speed up the retrieval process and also improves the accuracy of retrieval in our experiments. With the support of relevance feedback, the SOM-based system consistently further enhances the retrieval accuracy. It is suggested that when dealing with a large dataset like our applications, SOM-retrieval system is able to save much computation cost and can be a practical tool for real time application.

References

Ampazis, N., & Perantonis, S. (2004). LSISOM: A latent semantic indexing approach to self-organizing maps of document collections. *Neural Processing Letters, 19*(2), 157–173.

Antonio, S. A. et al. (2008). Web mining based on growing hierarchical self-organizing maps: Analysis of a real citizen web portal. *Expert Systems with Applications, 34*, 2988–2994.

Berry, M. W., Dumais, S. T., & O'Brien, G. W. (1995). Using linear algebra for intelligent information retrieval. *SIAM Review, 37*(4), 573–595.

Blei, D., Ng, A., & Jordan, M. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research, 3*, 993–1022.

Cancho, R. F. I., & Sole, R. V. (2001). The small-world of human language. *Proceedings of the Royal Society B: Biological Sciences, 268*(1482), 2261–2265.

Chow, T. W. S., Rahman, M. K. M., & Wu, S. (2006). Content based image retrieval by using tree-structured features and multi-layer SOM. *Pattern Analysis and Applications, 9*(1), 1–20.

Deerwester, S., & Dumais, S. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science, 41*(6), 391–407.

Dorogovtsev, S. N., & Mendes, J. F. F. (2001). Language as an evolving word web. *Proceedings of the Royal Society B: Biological Sciences, 268*(1485), 2603–2606.

Du, D. H. C., Ghanta, S., Maly, K. J., & Sharrock, S. M. (1989). An efficient file structure for document retrieval in the automated office environment. *IEEE Transactions on Knowledge and Data Engineering, 1*(2), 258–273.

Erkan, G. (2006). Language model-based document clustering using random walks. In *Proceedings of the human language technology conference of the North American Chapter of the ACL, New York, America*, pp. 479–486.

Esben HØgh-Rasmussen (2005). BBTools – a Matlab toolbox for black-box computations. Neurobiology Research Unit, Copenhagen University Hospital. URL: http://nru.dk/software/bbtools/.

Freeman, Richard T., & Yin, Hujun (2005). Web content management by self-organization. *IEEE Transactions on Neural Networks, 16*(5), 1256–1268.

Georgakis, A., Kotropoulos, C., Xafopoulos, A., & Pitas, I. (2004). Marginal median SOM for document organization and retrieval. *Neural Networks, 17*(3), 365–377.

Hofmann, T. (1999). Probabilistic latent semantic indexing. In *Proceedings of the twenty-second annual international SIGIR conference*.

Honkela, T., Kaski, S., Lagus, K., Kohonen, T. (1997). WEBSOM—self-organizing maps of document collections. In *Proceedings of WSOM'97, workshop on self-organizing maps, Espoo, Finland, Helsinki University of Technology, Neural Networks Research Centre*, pp. 310–315.

Horng, Yih- Jen, Chen, Shyi-Ming, Chang, Yu-Chuan, & Lee, Chia-Hoang (2005). A new method for fuzzy information retrieval based on fuzzy hierarchical clustering and fuzzy inference techniques. *IEEE Transaction on Fuzzy Systems, 13*(2), 216–228.

Isa, D., Kallimani, V. P., Lee, L. H. (2009). Using the self-organizing map for clustering of text documents. *Expert Systems with Applications, 36*(5), 9584–9591.

Kohonen, T. (1997). *Self-organizing maps*. Berlin, Germany: Springer-Verlag.
Lochbaum, K. E., & Streeter, L. A. (1989). Comparing and combining the effectiveness of latent semantic indexing and the ordinary vector space model for information retrieval. *Journal of Information Science, 6*, 25–33.
Ponte, J. M., Croft, W. B. (1998). A language modeling approach to information retrieval. In *Proceedings. of the 21st annual international ACM SIGIR conference on research and development in information retrieval, Melbourne, Australia*, pp. 275–281.
Porter, M. F. (1980). An algorithm for suffix stripping. *Program, 14*(3), 130–137.
Rldvan, Saracoglu, Tutuncu, Kemal, & Allahverdi, Novruz (2007). A fuzzy clustering approach for finding similar documents using a novel similarity measure. *Expert Systems with Applications, 33*, 600–605.
Rooney, N., Patterson, D., Galushka, M., & Dobrynin, V. (2006). A scalable document clustering approach for large document corpora. *Information Processing and Management, 42*, 1163–1175.
Salton, G., & Buckley, C. (1996). Term weighting approaches in automatic text retrieval. *Information Processing and Management, 32*(4), 431–443.
Salton, G., & McGill, M. (Eds.). (1983). *Introduction to modern information retrieval*. McGraw-Hill.
Schenker, A., Last, M., Bunke, H., Kandel, A. (2003). Classification of web document using a graph model. In *Proceedings of the 7th international conference on document analysis and recognition (ICDAR'03)*.
Schenker, A., Last, M., Bunke, H., & Kandel, A. (2004). Classification of web documents using graph matching. *International Journal of Pattern Recognition and Artificial Intelligence, 18*(3), 475–496.
Welling, M., Rosen-Zvi, M., & Hinton, G. (2004). Exponential family harmoniums with an application to information retrieval. *Advances in neural information processing systems* (vol. 17, pp. 1481–1488). Cambridge, MA: MIT Press.
Zobel, J., & Moffat, A. (1998). Exploring the similarity space. *ACM SIGIR Forum, 32*(1), 18–34.