Document Similarity Search Based on Manifold-Ranking of TextTiles

Xiaojun Wan, Jianwu Yang, and Jianguo Xiao

Institute of Computer Science and Technology,
Peking University, Beijing 100871, China
{wanxiaojun, yangjianwu, xiaojianguo}@icst.pku.edu.cn

Abstract. Document similarity search aims to find documents similar to a query document in a text corpus and return a ranked list of similar documents. Most existing approaches to document similarity search compute similarity scores between the query and the documents based on a retrieval function (e.g. Cosine) and then rank the documents by their similarity scores. In this paper, we proposed a novel retrieval approach based on manifold-ranking of TextTiles to re-rank the initially retrieved documents. The proposed approach can make full use of the intrinsic global manifold structure for the TextTiles of the documents in the re-ranking process. Experimental results demonstrate that the proposed approach can significantly improve the retrieval performances based on different retrieval functions. TextTile is validated to be a better unit than the whole document in the manifold-ranking process.

1 Introduction

Document similarity search is to find documents similar to a query document in a text corpus and return a ranked list of similar documents to users. The typical kind of similarity search is K nearest neighbor search, namely K-NN search, which is to find K documents most similar to the query document. Similarity search is widely used in recommender systems in library or web applications. For example, *Google*[1] can perform an advanced search with “related” option to find similar web pages with a user-specified web page and *CiteSeer.IST*[2] provides a list of similar papers with the currently browsed paper.

Document similarity search differs from traditional keyword-based text retrieval in that similarity search systems take a full document as query, while current keyword-based search engines usually take several keywords as query. The keyword-based short query is usually constructed by users and can well reflect users’ information need, while the document-based long query may contains more redundant and ambiguous information and even greater noise effects stemmed from the presence of a large number of words unrelated to the overall topic in the document.

[1] http://www.google.com

[2] http://citeseer.ist.psu.edu/cs

H.T. Ng et al. (Eds.): AIRS 2006, LNCS 4182, pp. 14 – 25, 2006.

The popular retrieval functions used in current text retrieval systems include the Cosine function, the Jaccard function, the Dice function [2, 14], the BM25 function in the Okapi system [10, 11] and the vector space model with document length normalization in the Smart system [12, 13], among which the standard Cosine measure is considered as the best model for document similarity search because of its good ability to measure the similarity between two documents. To our knowledge, almost all text search engines find relevant documents to a given query only by the pairwise comparison between the document and the query, thus neglecting the intrinsic global manifold structure of the documents. In order to make up for this limitation, we employ a manifold-ranking process [15, 16] for the initially retrieved documents and make full use of the relationships between the documents. In the manifold-ranking process, documents can spread their ranking scores to their nearby neighbors via a weighted network. Moreover, inspired by passage retrieval [6, 7], the manifold-ranking process is applied at a finer granularity of TextTile [4, 5] instead of the whole document, because a document is usually characterized as a sequence of subtopical discussions (i.e. TextTiles) that occur in the context of a few main topic discussions, and comparison of subtopics is deemed to be more accurate than comparison of the whole document.

Specifically, the proposed retrieval approach consists of two processes: initial ranking and re-ranking. In the initial ranking process, a small number of documents are retrieved based on a popular retrieval function. In the re-ranking process, the query document and the initially retrieved documents are segmented into TextTiles, and then the manifold-ranking algorithm is applied on the TextTiles and each TextTile obtains its ranking score. Lastly, a document gets its final retrieval score by fusing the ranking scores of the TextTiles in the document. The initially retrieved documents are re-ranked and the re-ranked list is returned to users. Experimental results show the encouraging performance of the proposed approach and it can significantly improve the retrieval performances based on different retrieval functions. TextTile is validated to be a better text unit than the whole document in the manifold-ranking process.

The rest of this paper is organized as follows: Popular retrieval functions are introduced in Section 2. The proposed manifold-ranking based approach is described in detail in Section 3. Section 4 gives the experiments and results. Lastly, we present our conclusion in Section 5.

2 Popular Retrieval Functions

2.1 The Cosine Function

The Cosine measure is the most popular measure for document similarity based on the vector space model (VSM). In VSM, a document *d* is represented by a vector with each dimension referring to a unique term and the weight associated with the term *t* is calculated by the $tf_{d,t} * idf_t$ formula, where $tf_{d,t}$ is the number of occurrences of term *t* in document *d* and $idf_t=1+log(N/n_t)$ is the inverse document frequency, where *N* is the total number of documents in the collection and n_t is the number of documents containing term *t*. The similarity *sim(q,d)*, between the query document *q* and the document *d*, can be defined as the normalized inner product of the two vectors $\vec{q}$ and $\vec{d}$:

$$\mathrm{sim}_{\mathrm{cosine}}(q,d)=\frac{\vec{q}\cdot\vec{d}}{|\vec{q}||\vec{d}|}=\frac{\sum_{t\in q\cap d}(w_{q,t}\cdot w_{d,t})}{\sqrt{\sum_{t\in q}w_{q,t}{}^{2}\times\sum_{t\in d}w_{d,t}{}^{2}}} \tag{1}$$

where *t* represents a term. $q\cap d$ gets the common words between *q* and *d*. Term weight $w_{d,t}$ is computed by $tf_{d,t}$* idf_t.

2.2 The Jaccard Function

The Jaccard function is similar to the Cosine function as follows:

$$\mathrm{sim}_{\mathrm{Jaccard}}(q,d)=\frac{\sum_{t\in q\cap d}(w_{q,t}\cdot w_{d,t})}{\sum_{t\in q}w_{q,t}{}^{2}+\sum_{t\in d}w_{d,t}{}^{2}-\sum_{t\in q\cap d}(w_{q,t}\cdot w_{d,t})} \tag{2}$$

2.3 The Dice Function

The Dice function is defined similarly to the Cosine function as follows:

$$\mathrm{sim}_{\mathrm{Dice}}(q,d)=\frac{2\times\sum_{t\in q\cap d}(w_{q,t}\cdot w_{d,t})}{\sum_{t\in q}w_{q,t}{}^{2}+\sum_{t\in d}w_{d,t}{}^{2}} \tag{3}$$

2.4 The BM25 Function

The BM25 measure is one of the most popular retrieval models in a probabilistic framework and is widely used in the Okapi system. Given the query document *q*, the similarity score for the document *d* is defined as follows:

$$\mathrm{sim}_{\mathrm{BM25}}(q,d)=\sum_{t\in \mathrm{q}}f_{q,t}\times\log(\frac{N-n_t+0.5}{n_t+0.5})\times\frac{(K+1)\times f_{d,t}}{K\times\left\{(1-b)+b\frac{dlf_d}{avedlf}\right\}+f_{d,t}} \tag{4}$$

where *t* represents a unique term; *N* is the number of documents in the collection; n_t is the number of documents in which term t exists; $f_{q,t}$ is the frequency of term *t* in *q*; $f_{d,t}$ is the frequency of term *t* in *d*; dlf_d is the sum of term frequencies in *d*; *avedlf* is the average of dlf_d in the collection; *K*=2.0, *b*=0.8 are constants.

2.5 The Vector Space Model with Document Length Normalization

The vector space model with document length normalization (NVSM) is also a popular retrieval model and is used in the Smart system. Given the query document *q*, the similarity score for the document *d* is defined as follows:

$$\text{sim}_{\text{NVSM}}(q,d) = \sum_{t\in q}(1+\log(f_{q,t}))\times idf_t \times \frac{1+\log(f_{d,t})}{1+\log(avef_d)} \times \frac{1}{avedlb + S\times(dlb_d - avedlb)} \quad (5)$$

where idf_t is the inverse document frequency of term t; dlb_d is the number of unique terms in d; $avef_d$ is the average of term frequencies in d (i.e. "dlf_d/dlb_d"); $avedlb$ is the average of dlb_d in the collection; S=0.2 is a constant.

3 The Manifold-Ranking Based Approach

3.1 Overview

The aim of the proposed approach is two-fold: one is to evaluate the similarity between the query and a document by exploring the relationship between all the documents in the feature space, which addresses the limitation of present similarity metrics based only on pairwise comparison; the other is to evaluate document similarity at a finer granularity by segmenting the documents into TextTiles, which addresses the limitation of present similarity metrics based on the whole document.

The proposed approach first segments the query and the documents into TextTiles using the TextTiling algorithm [4, 5], and then applies a manifold-ranking process on the TextTiles. All the TextTiles of a document obtain their ranking scores and the ranking score of the document is obtained by fusing the ranking scores of its TextTiles.

Note that it is of high computational cost to apply the manifold-ranking process to all the documents in the collection, so the above manifold-ranking process is taken as a re-ranking process. First, we use a popular retrieval function (e.g. Cosine, Jaccard, Dice, BM25, NVSM, etc.) to efficiently obtain an initial ranking of the documents, and then the initial k documents are re-ranked by applying the above manifold-ranking process.

Formally, given a query document q and the collection C, the proposed approach consists of the following four steps:

1. **Initial Ranking:** The initial ranking process uses a popular retrieval function to return a set of top documents $D_{\text{init}}\subseteq C$ in response to the query document q, $|D_{\text{init}}|=k$. Each document $d_i\in D_{\text{init}}$ ($1\le i\le k$) is associated with an initial retrieval score $InitScore(d_i)$.
2. **Text Segmentation:** By using the TextTiling algorithm, the query document q is segmented into a set of TextTiles $\chi_q = \{x_1, x_2, \ldots, x_p\}$ and all documents in D_{init} are segmented respectively and the total set of TextTiles for D_{init} is $\chi_{D_{\text{init}}}=\{x_{p+1}, x_{p+2}, \ldots, x_n\}$.
3. **Manifold-Ranking:** The manifold-ranking process in applied on the whole set of TextTiles: $\chi = \chi_q \cup \chi_{D_{\text{init}}}$, and each TextTile x_j ($p+1\le j\le n$) in $\chi_{D_{\text{init}}}$ gets its ranking score f_j^*.

4. **Score Fusion:** The final score *FinalScore(d_i)* of a document $d_i \in D_{\text{init}}$ ($1 \le i \le k$) is computed by fusing the ranking scores of its TextTiles. The documents in D_{init} are re-ranked according to their final scores and the re-ranked list is returned.

The steps 2-4 are key steps in the re-ranking process and they will be illustrated in detail in next sections, respectively.

3.2 The Text Segmentation Process

There have been several methods for division of documents according to units such as sections, paragraphs, or fixed length sequences of words, or semantic passages given by inferred shift of topic [6, 7]. In this study, we adopt semantic passages to represent subtopics in a document. As mentioned in [4, 5], the text can be characterized as a sequence of subtopical discussions that occur in the context of a few main topic discussions. For example, a news text about China-US relationship, whose main topic is the good bilateral relationship between China and the United States, can be described as consisting of the following subdiscussions (numbers indicate paragraph numbers):

1 Intro-the establishment of China-US relationships
2-3 The officers exchange visits
4-5 The culture exchange between the two countries
6-7 The booming trade between the two countries
8 Outlook and summary

We expect to acquire the above subtopics in a document and use them in the manifold-ranking process instead of the whole document. The most popular TextTiling algorithm is used to automatically subdivide text into multi-paragraph units that represent subtopics.

The TextTiling algorithm detects subtopic boundaries by analyzing patterns of lexical connectivity and word distribution. The main idea is that terms that describe a subtopic will co-occur locally, and a switch to a new subtopic will be signaled by the ending of co-occurrence of one set of terms and the beginning of the co-occurrence of a different set of terms. The algorithm has the following three steps:

1) Tokenization: The input text is divided into individual lexical units, i.e. pseudo-sentences of a predefined size;
2) Lexical score determination: All pairs of adjacent lexical units are compared and assigned a similarity value;
3) Boundary identification: The resulting sequence of similarity values is graphed and smoothed, and then is examined for peaks and valleys. The subtopic boundaries are assumed to occur at the largest valleys in the graph.

For TextTiling, subtopic discussions are assumed to occur within the scope of one or more overarching main topics, which span the length of the text. Since the segments are adjacent and non-overlapping, they are called TextTiles.

The computational complexity is approximately linear with the document length, and more efficient implementations are available, such as Kaufmann [8] and JTextTile [3].

3.3 The Manifold-Ranking Process

Manifold-ranking [15, 16] is a universal ranking algorithm initially used to rank data points along their underlying manifold structure. The prior assumption of manifold-ranking is: (1) nearby points are likely to have the same ranking scores; (2) points on the same structure (typically referred to as a cluster or a manifold) are likely to have the same ranking scores. An intuitive description of manifold-ranking is as follows: A weighted network is formed on the data, and a positive rank score is assigned to each known relevant point and zero to the remaining points which are to be ranked. All points then spread their ranking score to their nearby neighbors via the weighted network. The spread process is repeated until a global stable state is achieved, and all points obtain their final ranking scores.

In our context, the data points are denoted by the TextTiles in the query document q and the top documents in D_{init}. The manifold-ranking process in our context can be formalized as follows:

Given a set of data points $\chi = \chi_q \cup \chi_{D_{\text{init}}} = \{ x_1, x_2, ... x_p, x_{p+1}, ..., x_n \} \subset R^m$, the first p points are the TextTiles in the query document q and the rest n-p points are the TextTiles in the documents in D_{init}. Let $f : \chi \rightarrow R$ denotes a ranking function which assigns to each point x_j ($1 \le j \le n$) a ranking value f_j. We can view f as a vector $f = [f_1, f_2, ..., f_n]^{\mathrm{T}}$. We also define a vector $y = [y_1, y_2, ..., y_n]^{\mathrm{T}}$, in which y_j=1 ($1 \le j \le p$) for the TextTiles in q and y_j= *InitScore(d_i)* (p+$1 \le j \le n$) for the TextTiles in any document d_i in D_{init}, where $x_{\text{j}} \in d_{\text{i}}$, $d_{\text{i}} \in D_{\text{init}}$, which means that the initial retrieval score of a document is used as the initial ranking scores of the TextTiles in the document. The manifold-ranking algorithm goes as follows:

1. *Compute the pairwise similarity among points (TextTiles) and using the standard Cosine function.*
2. *Connect any two points with an edge. We define the affinity matrix W by* $W_{ij}=sim_{cosine}(x_i,x_j)$ *if there is an edge linking* x_i *and* x_j*. Note that we let* $W_{ii}=0$ *to avoid loops in the graph built in next step.*
3. *Symmetrically normalize W by* $S=D^{-1/2}WD^{-1/2}$ *in which D is the diagonal matrix with (i,i)-element equal to the sum of the i-th row of W.*
4. *Iterate* $f(t+1) = \alpha S f(t) + (1-\alpha) y$ *until convergence, where* α *is a parameter in (0,1).*
5. *Let* f_j^* *denote the limit of the sequence* $\{f_j(t)\}$*. Each TextTiles* x_j (p+$1 \le j \le n$) *gets its ranking score* f_j^**.*

Fig. 1. The manifold-ranking algorithm

In the above iterative algorithm, the normalization in the third step is necessary to prove the algorithm's convergence. The fourth step is the key step of the algorithm, where all points spread their ranking score to their neighbors via the weighted network. The parameter of manifold-ranking weight α specifies the relative contributions to the ranking scores from neighbors and the initial ranking scores. Note that *self-reinforcement* is avoided since the diagonal elements of the affinity matrix are set to zero.

The theorem in [16] guarantees that the sequence $\{f(t)\}$ converges to

$$f^* = \beta(I-\alpha S)^{-1} y \tag{6}$$

where β=1-α. Although f^* can be expressed in a closed form, for large scale problems, the iteration algorithm is preferable due to computational efficiency. Usually the convergence of the iteration algorithm is achieved when the difference between the scores computed at two successive iterations for any point falls below a given threshold (0.0001 in this study).

Using Taylor expansion, we have

$$\begin{aligned} f^* &= \beta(I-\alpha S)^{-1} y \\ &= \beta(I+\alpha S+\alpha^2 S^2 + K)y \\ &= \beta(y+\alpha Sy+\alpha S(\alpha Sy)+K) \end{aligned} \tag{7}$$

From the above equation, if we omit the constant coefficient β, f^* can be regarded as the sum of a series of infinite terms. The first term is simply the vector y, and the second term is to spread the ranking scores of the TextTiles to their nearby TextTiles, and the third term is to further spread the ranking scores, etc. Thus the effect of the TextTiles in the documents is gradually incorporated into the ranking score.

3.4 The Score Fusion Process

The final retrieval score of a document $d_i \in D_{\text{init}}$ is computed by fusing the ranking scores of its TextTiles as follows:

$$FinalScore(d_i) = \frac{\sum_{x_j \in d_i} \lambda_j f_j^*}{|d_i|} \tag{8}$$

where $\lambda_j = \text{sim}_{\text{cosine}}(x_j, d_i)$ is the cosine similarity between the TextTile x_j and its associated document d_i, which measures the importance of the TextTile x_j in the document d_i. $|d_i|$ represents the number of TextTiles in the document d_i. This normalization avoids favoring long documents.

Finally, the documents in D_{init} are re-ranked according to their final scores and the re-ranked list is returned[3].

4 Experiments

4.1 Experimental Setup

In the experiments, the manifold-ranking based approach ("MR+TextTile") is compared with two baseline approaches: "Cosine" and "MR+Document". The "Cosine"

[3] Only the top k documents (i.e. the documents in D_{init}) in the original ranked list are re-ranked and the rest documents in the original ranked list still hold their initial ranks.

baseline does not apply the manifold-ranking process and directly ranks the documents by their Cosine similarity with the query document, which is the popular way for document similarity search. The "MR+Document" baseline is adopted in [16], which uses the whole document instead of TextTile in the manifold-ranking process, and thus it does not need the steps of text segmentation and score fusion. The manifold-ranking process is also applied on top documents retrieved by other retrieval functions.

To perform the experiments, a ground truth data set is required. We build the ground truth data set from the TDT-3 corpus, which has been used for evaluation of the task of topic detection and tracking [1] in 1999 and 2000. TDT-3 corpus is annotated by Linguistic Data Consortium (LDC) from 8 English sources and 3 Mandarin sources for the period of October through December 1998. 120 topics are defined and about 9000 stories are annotated over these topics with an "on-topic" table presenting all stories explicitly marked as relevant to a given topic. According to the specification of TDT, the on-topic stories within the same topic are similar and relevant. After removing the stories written in Chinese, we randomly chose 40 topics as a test set, while the others were used as a training set.

Sentence tokenization was firstly applied to all documents. Stop words were removed and Porter's stemmer [9] was used for word stemming. The JTextTile tool with default setting was employed to segment each document into TextTiles. The total stories are considered as the document collection for search, the first document within the topic is considered as the query document and all the other documents within the same topic are the relevant (similar) documents, while all the documents within other topics are considered irrelevant (dissimilar) to the query document. A ranked list of 500 documents was required to be returned for each query document based on each retrieval approach. The higher the document is in the ranked list, the more similar it is with the query document. For the proposed manifold-ranking process, the number of initially retrieved documents is typically set to 50, i.e. $|D_{\text{init}}|=k=50$.

As in TREC[4] experiments, we use the average precisions at top N results, i.e. *P@5* and *P@10*, as evaluation metrics. The precision at top N results for a query is calculated as follows:

$$P@N = \frac{|C \cap R|}{|R|}, \tag{9}$$

where R is the set of top N retrieved documents, and C is the set of similar documents defined above for a given query document. The precision is calculated for each query and then the values are averaged across all queries.

Note that the number of documents within each topic is different and some topics contain even less than 5 documents or 10 documents, so its corresponding *P@5* or *P@10* may be low.

4.2 Experimental Results

The precision values of our proposed approach ("MR+TextTile") and two baseline approaches (i.e. "Cosine" and "MR+Document") are compared in Table 1, when the

[4] http://trec.nist.gov

manifold-ranking weight α is set to 0.3, which is tuned on the training set. The upper bounds are the ideal values under the assumption that all the relevant (similar) documents are retrieved and ranked higher than those irrelevant (dissimilar) documents in the ranked list. Seen from Table 1, the proposed approach significantly outperforms the two baseline systems. We can also see that the "MR+Document" baseline achieves almost the same *P@5* value with the "Cosine" baseline and the higher *P@10* value than the "Cosine" baseline, which demonstrates that manifold-ranking process can benefit document ranking.

Table 1. Performance comparison of the proposed approach and baseline approaches (* indicates that the performance change over baseline1-"Cosine" is statistically significant, and # indicates that the performance change over baseline2-"MR+Document" is statistically significant, i.e. p-value<0.05 for t-test)

	Baseline1 (Cosine)	Baseline2: (MR+Document)	Our Approach: (MR+TextTile)	Upperbound
P@5	0.830	0.825	0.855*$^{\#}$	0.935
P@10	0.720	0.738*	0.763*$^{\#}$	0.863

The performances of two MR-based approaches (i.e. "MR+TextTile" & "MR+Document") with different manifold-ranking weight α are shown and compared in Figures 2 and 3. Seen from the figures, with appropriate values of the manifold-ranking weight (i.e. α<0.5), the proposed approach (i.e. "MR+TextTile") can significantly outperform the approach of "MR+Document", which demonstrates that TextTile is a more appropriate unit than the whole document for the manifold-ranking process. This result can be explained by that a document is usually characterized as a sequence of subtopical discussions that occur in the context of a few main topic discussions and each TextTile can represent a subtopic with coherent text, and thus the manifold-ranking process can work at a finer granularity.

Figure 4 explores the influence of the number of initially retrieved documents (i.e. *k*) on the performance of the proposed approach (i.e. "MR+TextTile"). Seen from the figure, when *k* is larger than 75, the system performances almost do not alter any more. This shows that a small number of initially retrieved documents work well in the re-ranking process and it will not improve the retrieval performance by increasing the number of initially retrieved documents.

In addition to the Cosine function, other popular retrieval functions are explored in the experiments, including the Jaccard function, the Dice function, the BM25 function and the VSM with document length normalization (NVSM). We use these functions for initial ranking and get the initial retrieved documents associated with their initial retrieval scores, and then the proposed re-ranking process (steps 2-4) is applied. The performances of the systems based only on the retrieval functions and the performances ofthe systems using the re-ranking process are compared in Table 2. For example, "Jaccard" denotes the system using the Jaccard function to retrieve the documents, while "Jaccard+MR" denotes the system using the Jaccard function to retrieve the initial *k*

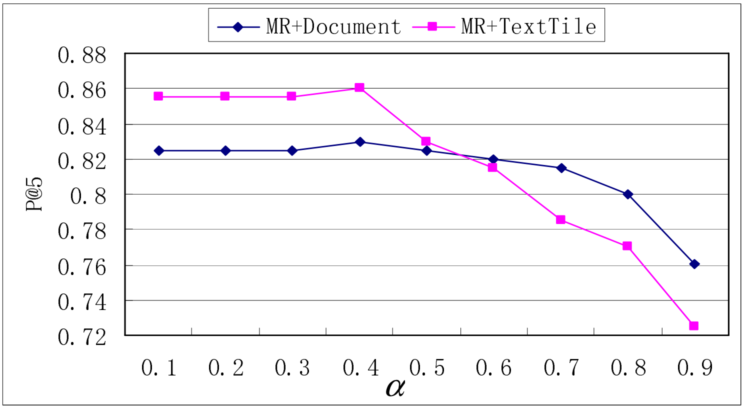

Fig. 2. P@5 comparison of MR-based approaches with different *α*

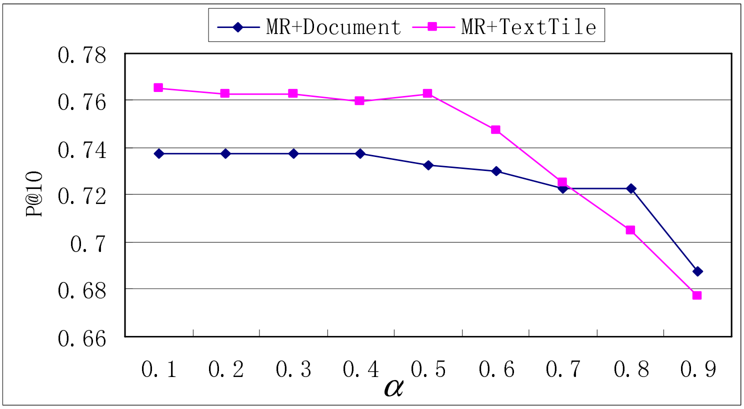

Fig. 3. P@10 comparison of MR-based approaches with different *α*

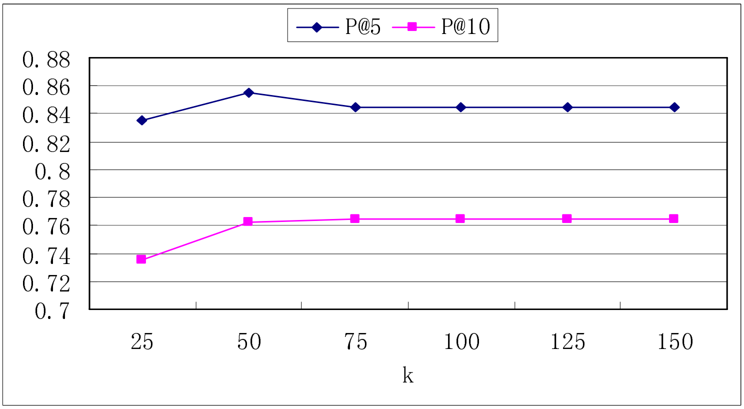

Fig. 4. Performance comparison of the proposed approach with different *k*

documents and then applying the re-ranking process to get the re-ranked document list. Here we still have k=50 and α=0.3. Seen from Table 2, the proposed re-ranking process improves all the retrieval performances based on different retrieval functions, which demonstrates the robustness of the proposed manifold-ranking process.

Table 2. Performance comparison based on different retrieval functions w/ and w/o the manifold-ranking process (* indicates that the performance change over the corresponding retrieval approach w/o MR is statistically significant, i.e. p-value<0.05 for t-test)

	Jaccard	Jaccard +MR	Dice	Dice +MR	BM25	BM25 +MR	NVSM	NVSM +MR
P@5	0.835	0.850	0.832	0.850*	0.820	0.850*	0.810	0.845*
P@10	0.740	0.758*	0.740	0.753*	0.720	0.740*	0.710	0.723

5 Conclusion

In this paper, we propose a novel retrieval approach for document similarity search. The proposed approach re-ranks a small number of initially retrieved documents based on manifold-ranking of TextTiles. The manifold-ranking process can make full use of the relationships among TextTiles to improve the retrieval performance. The experimental results demonstrate the favorable performance of the proposed approach.

In future work, we will explore the influence of different text segmentation methods on the retrieval performance. We will also adapt the proposed retrieval approach to search of semi-structured documents, such as XML documents and web pages.

References

1. Allan, J., Carbonell, J., Doddington, G., Yamron, J. P. and Yang, Y. (1998) Topic detection and tracking pilot study: final report. In Proceedings of DARPA Broadcast News Transcription and Understanding Workshop, 194-218
2. Baeza-Yates, R., and Ribeiro-Neto, B. (1999) Modern Information Retrival. ACM Press and Addison Wesley
3. Choi, F. (1999) JTextTile: A free platform independent text segmentation algorithm. http://www.cs.man.ac.uk/~choif
4. Hearst, M. A. (1994) Multi-paragraph segmentation of expository text. In Proceedings of the 32nd Meeting of the Association for Computational Linguistics, Los Cruces, NM
5. Hearst, M. A. (1997) TextTiling: Segmenting text into multi-paragraph subtopic passages. Computational Linguistics, 23(1): 33-64
6. Hearst, M. A. and Plaunt, C. (1993) Subtopic structuring for full-length document access. In Proceedings of the 16th Annual International ACM/SIGIR Conference, Pittsburgh, PA
7. Kaszkiel, M. and Zobel, J. (1997) Passage retrieval revisited. In Proceedings of the 20th Annual International ACM/SIGIR Conference, Philadelphia, Pennsylvania
8. Kaufmann, S. (1999) Cohesion and collocation: using context vectors in text segmentation, Proceedings of the 37th conference on Association for Computational Linguistics, 591-595
9. Porter, M. F. (1980) An algorithm for suffix stripping. Program, 14(3): 130-137

10. Robertson, S., Walker, S. (1994) Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In Proc. of the 17th International ACM/SIGIR Conference on Research and Development in Information Retrieval , 232-241
11. Robertson, S., Walker, S. and Beaulieu, M. (1999) Okapi at TREC–7: automatic ad hoc, filtering, VLC and filtering tracks. In Proceedings of TREC'99
12. Salton, G. (1991) The SMART retrieval system: experiments in automatic document processing. Prentice-Hall
13. Singhal, A., Buckley, C. and Mitra, M. (1996) Pivoted document length normalization. In Proceedings of SIGIR'96.
14. van Rijsbergen, C. J. (1979) Information Retrieval. Butterworths, London
15. Zhou, D., Bousquet, O., Lal, T. N., Weston, J. and SchÖlkopf, B. (2003) Learning with local and global consistency. In Proceedings of NIPS-2003
16. Zhou, D., Weston, J., Gretton, A., Bousquet, O. and SchÖlkopf, B. (2003). Ranking on data manifolds. In Proceedings of NIPS-2003