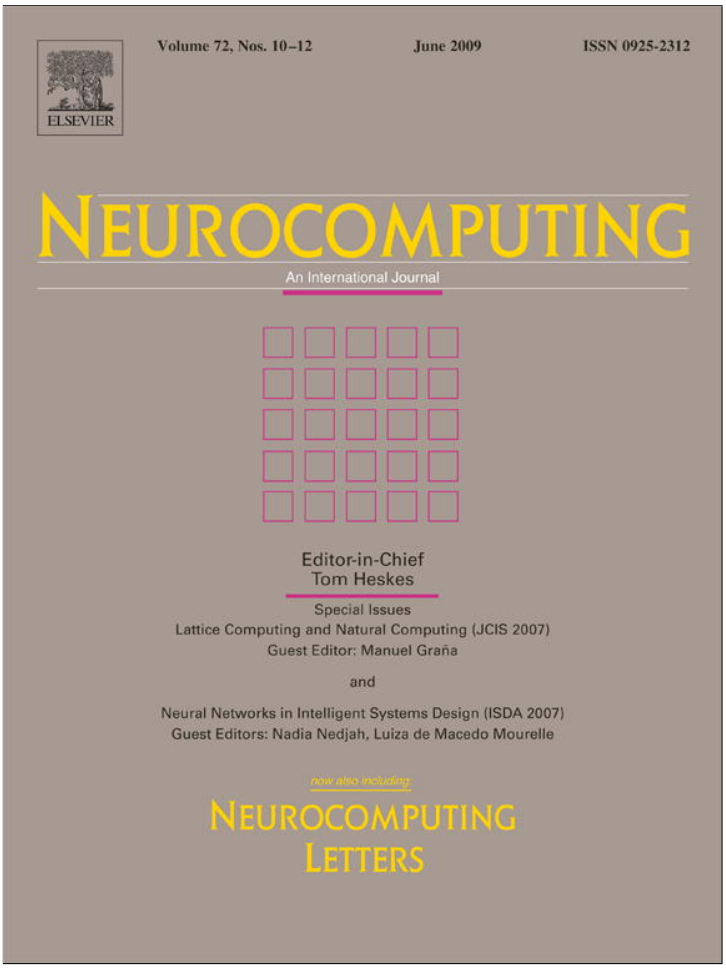

Neurocomputing 72 (2009) 2406–2417

Contents lists available at ScienceDirect

Neurocomputing

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

TDM modeling and evaluation of different domain transforms for LSI

Tareq Jaber [a,*], Abbes Amira [b], Peter Milligan [a]

[a] School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Belfast BT7 1NN, UK
[b] Electronic and Computer Engineering, School of Engineering and Design, Brunel University, West London, Uxbridge, Middlesex UB8 3PH, UK

ARTICLE INFO

Article history:
Received 18 January 2008
Received in revised form
2 December 2008
Accepted 14 December 2008
Communicated by D. Tao
Available online 7 January 2009

Keywords:
Latent semantic indexing
Information retrieval
Discrete cosine transform
Singular value decomposition
Cohen Daubechies Feauveau 9/7
Hard thresholding
Soft thresholding

ABSTRACT

Latent semantic indexing (LSI) is a popular technique used in information retrieval (IR) applications. This paper presents a novel evaluation strategy based on the use of image processing tools. The authors evaluate the use of the discrete cosine transform (DCT) and Cohen Daubechies Feauveau 9/7 (CDF9/7) wavelet transform as a preprocessing step for the singular value decomposition (SVD) step of the LSI system. In addition, the effect of different threshold types on the search results is examined. The results show that accuracy can be increased by applying both transforms as a preprocessing step, with better performance for the hard-threshold function. The choice of the best threshold value is a key factor in the transform process. This paper also describes the most effective structure for the database to facilitate efficient searching in the LSI system.

1. Introduction

Over the last decade the volume of information available to web, and other, users has increased dramatically; this is referred to as the data explosion. Consequently, there is a need to provide access to these data as efficiently as possible. Information retrieval (IR) examines the process of extracting relevant information from a dataset based on a user's query [1]. Latent semantic indexing (LSI) is a well-known technique used in IR. LSI has proved popular in IR as the technique can cope with the problems and inaccuracies associated with the fundamental problems of both synonymy and polysemy [2]. Synonymy is the situation where there are many ways to express a given concept, such as car and automobile. Therefore the literal terms in a user's query may not match those of relevant documents. Polysemy refers to single words that have more than one meaning, e.g. plane could refer to an aeroplane, or a flat surface. The main assumption of LSI is that this variation in word usage partly obscures the underlying semantic structure of a document [2]. By comparing word usage across documents, it has been found that certain sets of words are often shared among many documents while being absent in others, these words and the documents they share are semantically close to each other, whereas those which share fewer words are semantically distant [3]. Traditionally LSI is implemented in several stages [2]. The first stage is to preprocess the database of documents. This process includes removing all punctuation and 'stop words' such as the, as, and, etc., those without distinctive semantic meaning from a document. The following step is the construction of a term document matrix (TDM) which represents the relationship between the documents in the database and the words that appear in them. A suitable matrix decomposition algorithm is then used to decompose the TDM. The original decomposition algorithm proposed by Berry et al. [1], and by far the most widely used, is the singular value decomposition (SVD) [4,5]. The decomposition is used to remove noise (sparseness) from the matrix and reduce the dimensionality of the TDM, in order to ascertain the semantic relationship among terms and documents in an attempt to overcome the problems of polysemy and synonymy. Finally, the document set is compared with the query and the documents which are closest to the user's query are returned.

This paper presents a new approach to the LSI process based on the use of image processing techniques. In particular, the effect of using the discrete cosine transform (DCT) and Cohen Daubechies Feauveau 9/7 (CDF9/7) wavelet transform as preprocessing steps to the SVD is studied. Moreover, a comparison between the two transforms, to test the performance over a range of threshold values, is presented. This paper presents an investigation about the features for the best structure of the TDM that have major

* Corresponding author.
E-mail addresses: tjaber01@qub.ac.uk (T. Jaber), abbes.amira@brunel.ac.uk (A. Amira), p.milligan@qub.ac.uk (P. Milligan).

0925-2312/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2008.12.010

impact on the search result returned. A range of parameters and performance metrics including accuracy or precision (defined as the number of relevant documents returned) and the threshold values or dimensionsretained are used to evaluate the proposed LSI system. The paper is organized as follows. Section 2 introduces the existing work. The investigation method is presented in Section 3. This gives an overview of the proposed LSI system and the processes involved. Section 4 presents the results of the new methods, which are evaluated by comparison with standard baseline system. Moreover an understanding is achieved of the important features for the best TDM structure. Concluding remarks are given in Section 5.

2. Existing work

There is now a large body of research involving the area of LSI. For example, research into the preprocessing stage looks at how to determine what constitutes a 'stop word'. The list devised by Fox [5] has been widely accepted. A large amount of research has been carried out into speeding up the LSI process at Telecordia [6] by focusing on the calculation of the most computationally expensive task, the SVD. Research has moved beyond the basics of the LSI process. Several alternative decomposition algorithms to SVD have also been suggested, including QR factorization [1] and semi-discrete matrix decomposition (SDD) [7]. In unitary operators on the document space [8] Hoenkamp shows that the decomposition underlying LSI is an example of a unitary operator. Hoenkamp proposed the use of the Haar wavelet transform (HWT) as an alternative as this transpose shares the unitary property and has a much reduced computational cost. This line of research showed some promising initial results. Furthermore, the concept of representing the TDM as a gray scale image was postulated. In such a model the white dots in the image (non-zero values) represent the keywords in the document sets. In addition, it has been argued that using the HWT to remove noise from an image is equivalent to using the HWT to remove lexical noise from the TDM. However, this is theoretical work that needs to be proved in practice.

There are several studies using LSI in tandem with other techniques, such as neural nets [9] and document clustering [10]. In [9] the LSI technique was incorporated into a competition-based neural network model. The results show that the new LSI model outperformed the standard one and produced approximately 5.4% improvement in the precision–recall performance over the standard model. As recent studies report that the SVD algorithm strategy can be less effective for large non-homogeneous text collections [11], the clustering technique was used to split the original TDM, which represents the database, into a number of clusters and then performs the SVD on the clustered datasets individually. The results reported in [10] show that the accuracy of the LSI technique may be improved when retrieving from clustered subsets, and the improvement from the clustered SVD strategies is more on the less homogeneous database, while on already highly homogeneous databases the additional clustering does not help very much. However, the number of clusters to choose remains an unsolved problem which affects the performance of a clustered SVD retrieval system.

Using the same technique as the previous work, but focussing on the homogeneity of the subset databases, various distributed implementations have been considered [12]. But this work depends on the semantic heterogeneity of the original document corpus and the degreeto which it can be successfully partitioned into smaller and more conceptually homogeneous document sets. Perhaps the most surprising applications of LSI research have been in fields other than IR. SVD has been used with water-marking algorithms to solve the problem of copyright protection of multimedia documents [13]. The principles underlying LSI have been applied to cross language retrieval [14]. Some have even gone further; suggesting that LSI-based techniques may be able to imbue machines with human-like learning capabilities [15,16]. In [17] the use of both keywords and image features to represent documents has been presented in order to improve the retrieval performance. One of the most recent works has focussed on dimension reduction in the LSI system [18]. Other researchers have used LSI in the field of image retrieval [19,20].

3. The hybrid approach

This section presents the different components of the proposed hybrid approach. To enable evaluation of the modified approach, the proposed method has been tested to four sample databases.

3.1. Database description

In this research, the database contains sets of document titles on which the search is performed. This section describes the structure and the contents of the databases used in this work.

3.1.1. Database structure

Each of the databases used is held as a simple binary table in Microsoft Access. The tables are in the form: *ID*, *Title* the 'Title' field holds the document title from which the keywords are generated. The 'ID' field acts as a unique key for each entry in the table, allowing documents to be referenced easily.

3.1.2. Database content

The documents used in the experiments are held as a set of four databases. The *Memos* database is a very small database consisting of nine Bellcore technical memos which is widely used as a worked example in many papers dealing with LSI [1,2]. The *Memos* database was constructed so that the first five documents deal with human computer interaction and the other four are related to data structures. Such a well defined structure has proved useful for outlining the main principles behind LSI. We have included this sample database in our study to provide a baseline reference. The *Cochrane* database is a small database of 135 documents containing the titles of medical studies into drug administration which is another commonly used test system in the LSI literature. It can be found at the *Cochrane* website [21]. The third is a larger dataset containing the titles of 658 electronic books held by the Science Library at Queens University [22]. It has been chosen for both the size and the good structure, as will be explained later in this section. The final database is the *Reuters*-21578 text categorization collection (TCC). It is the most widely used test collection for text categorization research. The data were originally collected and labeled by Carnegie Group, Inc. and Reuters, Ltd. in the course of developing the CONSTRUE text categorization system [23]. In common with other research studies a smaller subset of approximately 1000 titles is used. The actual test collection contains a considerable volume of information, but for simplicity only the titles of the documents have been used. It can be found at [23].

3.2. Document preprocessing description

The database-style document table needs to be converted to a TDM. Before this can be achieved, preprocessing has to be carried out on the document set. Punctuation and meaningless words need to be removed, and the keywords necessary for construction

have to be extracted [1]. A list that comprises all keywords in the document set is obtained, along with a list of keywords and phrases in each individual document. This stage is computationally insignificant.

3.2.1. Memos database example

To illustrate the preprocessing step consider the Memos database as example. The titles in the Memos database are:

Preprocessing produces the following set of unique keywords (above in bold): {**human**, **computer**, **interface**, **survey**, **user**, **system**, **response**, **time**, **EPS**, **trees**, **graph**, **minors**} (Table 1).

3.3. Term document matrix

Once preprocessing is complete, the TDM is constructed from a list of terms that characterizes the structure of all the documents and the keyword list for each document that was generated in the previous step. Each row of the matrix is assigned to a term, and each column of the matrix is assigned to a document. The value that appears in position (i,j) is the number of times that the keyword assigned to the ith row appears in the document assigned to the jth document. Most values in the matrix are (therefore) zero, as only a subset of keywords appears in any given document. It is interesting to see the relationship of terms across documents. Words that appear only in one document are removed as they add no information on this relationship. Similarly, words that appear in all documents are also removed, as their ubiquity renders them meaningless. These removals are achieved by removing rows with only one non-zero value, and those where less than two elements are zero. The TDM generated for the Memos example is shown in Table 2.

Each column in the database can be considered as a vector describing the document it represents, each row can be considered as a vector describing the term that it represents. Documents are described in terms of the keywords that make them up, and keywords are expressed in terms of the documents they appear in. There is undoubtedly a great deal of redundancy in this process, as illustrated by the sparseness of matrix. The LSI process seeks to eliminate this redundancy by decomposing the TDM and extracting only the most significant values.

Table 1
Memo document set.

B1	**Human computer interface** for ABC **computer** applications
B2	A **survey** of user opinion of **computer system response time**
B3	The EPS **user interface** management **system**
B4	**System** and **human system** engineering testing of **EPS**
B5	Relation of **user** perceived **response time** to error measurement
B6	The generation of random, binary and ordered **trees**
B7	The intersection of paths in **trees**
B8	**Graph minors** IV: widths of **trees** and well-quasi ordering
B9	**Graph minors**: a **survey**

Table 2
TDM for Memos example.

	B1	B2	B3	B4	B5	B6	B7	B8	B9
Computer	2	1	0	0	0	0	0	0	0
Eps	0	0	1	1	0	0	0	0	0
Graph	0	0	0	0	0	0	0	1	1
Human	1	0	0	1	0	0	0	0	0
Interface	1	0	1	0	0	0	0	0	0
Minors	0	0	0	0	0	0	0	1	1
Response	0	1	0	0	1	0	0	0	0
Survey	0	1	0	0	0	0	0	0	1
System	0	1	1	2	0	0	0	0	0
Time	0	1	0	0	1	0	0	0	0
Trees	0	0	0	0	0	1	1	1	0
User	0	1	1	0	1	0	0	0	0

3.4. Query vector

In order for searches to be carried out, queries have to be represented in vector form also. This is achieved by the same process that is used to convert documents into columns in the TDM. Keywords are extracted from the query, and if a keyword also appears in the document set then the number of times it appears in the query is recorded using the same format as one of the document vectors in the TDM. For example the query 'response time' would be converted to the form $(0,0,0,0,0,0,1,0,0,1,0,0)$ as 'response' corresponds to the seventh row of the TDM, and 'time' corresponds to the tenth row, and each word appears once in the query. In effect, the query is a pseudo-document.

3.5. Matrix decomposition and transformation

This subsection presents a number of decompositions which will be used for the evaluation of the results obtained using our proposed LSI system.

3.5.1. Singular value decomposition

A matrix M can be decomposed into an approximate, reduced form as

$$M = U^* S^* V^{\mathrm{T}} \tag{1}$$

where U is the singular row vectors of M, S is a diagonal matrix holding the singular values of M in ascending order and V^{T} is the transpose of the singular value column vectors of M [1,2,4]. The diagonal elements in S are stored in ascending order [1]. The higher order values are larger and this means that they represent more of the semantic content of M (Fig. 1). By contrast, the lower order values are small and can be viewed as 'lexical noise' [8].

In Fig. 1, t is the number of terms in the TDM, d the number of documents in the TDM, and r the rank of M.

At the heart of LSI is that the latent semantic structure of the document set is identified by the matrix of diagonal values (Fig. 2).

The original TDM can be approximated by multiplying the three matrices U, S and V. However, if the lowest singular values of S are discarded, then the TDM can be approximated by

$$M_a = U_k^* S_k^* V_k^{\mathrm{T}} \tag{2}$$

where $k < r$, M_a is the approximated TDM, U_k the first k columns of U, S_k the new matrix of singular values and V_k^{T} the transpose of the first k columns of V [1].

The resultant approximated matrix has the same dimensions as the original TDM and represents the best k-rank approximation of M in terms of the Frobenius norm and p-norm [24,25]. The query can then be compared to each document in the new approximated matrix. With 'lexical noise' removed, this should

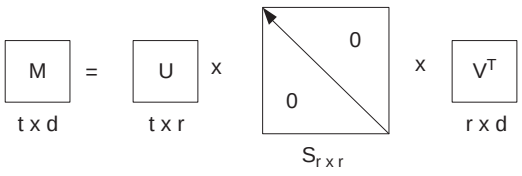

Fig. 1. SVD decomposition of $(t \times d)$ TDM.

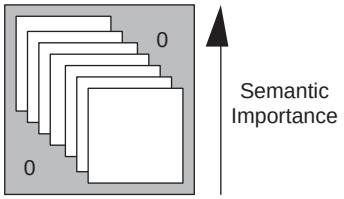

Fig. 2. Diagonal matrix *S*. The inner boxes represent singular values.

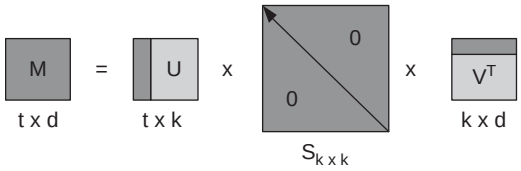

Fig. 3. Approximated TDM.

lead to improved results when the query is compared to the approximated documents. However, the TDM is typically not fully approximated. Instead, Cartesian co-ordinates for the documents are generated by multiplying the first k singular values in S with the first k columns of V (the transpose of V^T). Similarly, co-ordinates for the terms can be found by multiplying the first k singular values in S with the first k columns of U [1,2]. This can be used to achieve dimension reduction. For example if $k=2$, the resultant document vectors are two-dimensional. In this case, the vectors could be plotted on an axis to give a visual representation of the position of the terms and documents relative to each other. This visualization, for clustering of the documents, can help in understanding why the LSI outperforms the traditional keyword matching search techniques (Fig. 3).

In the latter case, the query must be converted to the same space as the document vectors for useful comparison to be made. This is achieved by using this equation

$$q = queryvector^* U_k^* S_k^i \tag{3}$$

where q is the new query vector, *queryvector* is the original query vector, U^k is the first k columns of U, and S_k^i is the inverse of the new matrix of singular values. The approach adopted in this work is to approximate the TDM and compare the documents in the approximated TDM to the query.

3.5.2. Cohen Daubechies Feauveau 9/7 (CDF9/7)

The CDF9/7 is an effective biorthogonal wavelet, as wavelets are capable of quickly capturing the essence of a data set with only a small number of coefficients. Therefore, this wavelet is used for signal approximation [26]. Signal approximation is the problem of representing a signal with as few components as possible. This is similar to lossy image compression. JPEG2000, which is a wavelet-based image compression standard, uses the CDF9/7 wavelet as a default wavelet for lossy compression [26]. The approach is used in many applications, e.g. face recognition [27], and by the FBI for fingerprint compression [26]. JPEG2000 supersedes the original JPEG standard which uses DCT in the compression. The JPEG2000 has not only improved compression performance over JPEG but also added (or improved) features such as scalability and editability, by decomposing the image into a multiple resolution representation.

3.5.3. The DCT

The DCT is a transform from a different domain, and in the last decade DCT has emerged as the de facto image transformation in many image processing applications [28]. As in wavelets, the DCT has the property that, for a typical image, most of the visually significant information about the image is concentrated in just a few coefficients of the DCT. For this reason, the DCT is often used in image compression applications. For example, the DCT is used in the standard lossy image compression algorithm JPEG [29], which has been used in many applications such as watermarking multimedia [30].

The two-dimensional transform of both CDF9/7 and DCT is equivalent to a one-dimensional transform, in which a one-dimensional transform is performed along a single dimension followed by a one-dimensional transform along the second dimension. In image processing, an image is transformed using transform technique, and then a thresholding function, at a certain threshold value, is applied to remove some unimportant components from the image; the new image results when the image is reconstructed after thresholding. The most common thresholding functions are the hard-thresholding function and the soft-thresholding function [31]. The hard-thresholding function chooses all wavelet coefficients that are greater than the given threshold λ and sets the others to zero, as described in the following equation:

$$f_h(x) = \begin{cases} x & \text{if } |x| \geqslant \lambda \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

The soft-thresholding function has a somewhat different rule from the hard-thresholding function. It shrinks the wavelet coefficients by λ towards zero, as described in the following equation:

$$f_s(x) = \begin{cases} x-\lambda & \text{if } x \geqslant \lambda \\ 0 & \text{if } |x| < \lambda \\ x+\lambda & \text{if } x \leqslant -\lambda \end{cases} \tag{5}$$

Both the hard- and soft-threshold functions [31] were tested and the results of applying each thresholding model are presented later in this paper.

For our present purposes, the TDM can be considered as a gray scale image, usually a binary image (sparse TDM with 0, 1 probabilities). By applying the transform and a threshold to the TDM, we can also remove 'noise' from our image, in this case we argue that this represents the removal of lexical noise.

3.6. TDM analysis and lexical noise

In this approach image processing techniques are used for TDM analysis. The first step involves representing the sparse TDM as a binary image to visualize the noise in the TDM. Within this binary representation of the TDM the occurrence of zeros represents the presence of noise. Correspondingly, the significant data will be represented by non-zero values. Fig. 4 shows the binary representation of the TDM for the Memos database (it is possible to view it, because of its small size), and Fig. 5 shows the image generated by visualizing the TDM for the Memos database.

If the images are examined, the white dots represent the data. When dots are close to each other, forming a cluster, it is possible, by looking at appropriate columns and rows, to say that there is a relationship between these documents because they contain the same terms.

Although the ordering of TDM rows and columns would not change the results, it is worth noting that visualizing the TDM as an image enables large datasets to be examined and analyzed more easily [32]. The distribution on the TDM, which can be noticed easily on the visualized image, depends on the structure, content and the size of the database. More investigation on these issues is presented in the Results section.

2	1	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0
0	0	0	0	0	0	0	1	1
1	0	0	1	0	0	0	0	0
1	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1
0	1	0	0	1	0	0	0	0
0	1	0	0	0	0	0	0	1
0	1	1	2	0	0	0	0	0
0	1	0	0	1	0	0	0	0
0	0	0	0	0	1	1	1	0
0	1	1	0	1	0	0	0	0

Fig. 4. Left: original TDM for Memos database.

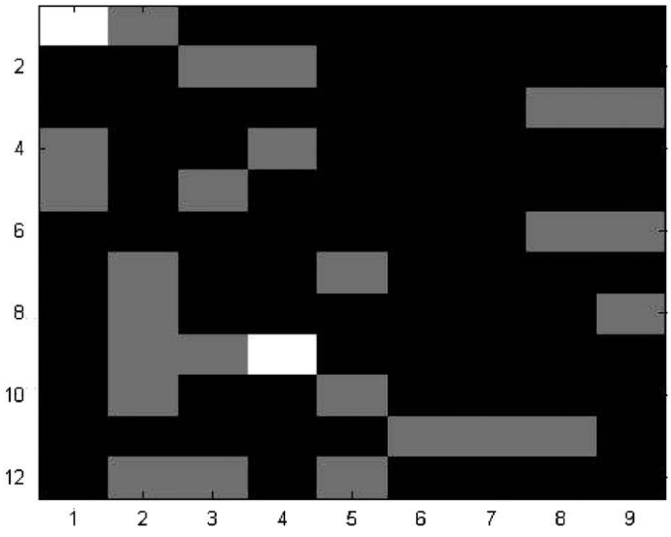

Fig. 5. TDM as an image for Memos database.

Candidates can be found in image processing applications, which have similar aims as LSI, but under the guise of lossless and lossy compression [8]. In this research some image processing transforms have been suggested for testing in LSI. The choice of such techniques can be explained by the following:

- One of the advantages of LSI is that it seems to remove (lexical) noise from the TDM by dimension reduction. In SVD, dimension reduction is achieved by zeroing the smallest eigenvalues in the diagonal matrix *S*.
- The analog for the transform is to zero the smallest coefficients.
- In both cases this is interpreted as removing (lexical) noise.

As mentioned before the lossless and lossy compression techniques have similar goals and mechanism as SVD in the LSI system. Therefore, and in this work, the CDF9/7 and DCT have been chosen as they have been used in the image compression standard algorithms JPEG2000 and JPEG. Maybe there are other techniques applicable, and would be suitable for future research.

3.7. (CDF9/7 and DCT)-SVD-based hybrid approach

As shown before, Hoenkamp proposed the use of the HWT as an alternative for SVD in the LSI system. As part of this research, this suggestion was investigated (the results are not presented in the paper) to determine the viability and value of the approach. The Haar transform does not fare well as a substitute for SVD in the LSI process, as it is unable to produce any more results than the standard model, it clearly fails as a straight substitute for SVD in the LSI process.

How can this be explained? If we return to Hoenkamp's analogy of the TDM as a gray scale image, we can shed some light on the process. In image processing, the HWT can be used to reveal the structure of an image; different levels of resolution show different features of an image: edge structure, background detail, etc. If we consider the TDM as an image, then the same rules must apply. The HWT must show the topology of the TDM.

The problem is that this structure is illusory. Consider the case of four one-dimensional documents. They can be represented in a TDM like this:

D1	D2	D3	D4
0	1	1	0

However, it is equally valid to arrange them in a different order:

D3	D1	D2	D4
1	0	1	0

Both these cases represent the same document set, but have different edge structure. A HWT of each produces different results. Different information is removed when the threshold is applied; the same query may produce different results.

Following on Hoenkamp's work, we have chosen to exploit this suggestion in different way, by including the HWT as a preprocessing stage for the SVD. This is a feasible approach as it is well known that the ordering of the TDM rows and columns would not change the results [33,34]. Therefore, the effect of the image processing transform on matrix decomposition and approximation and the quality of results returned is broadly studied.

A commonly used approach in image processing is to combine different techniques in order to improve noise reduction. The visualization of the TDM as a gray scale image invites a similar technique. The system allows DCT and CDF9/7 techniques to be combined with SVD, as shown in Fig. 6, to investigate their combined effect on the TDM and the quality of the results.

4. Results and analysis

This section presents a number of experiments, in which a range of searches are performed on the sample databases to compare the basic LSI-SVD approach with the proposed hybrid technique. (The standard SVD, when mentioned in the work, refers to the standard LSI-SVD system.)

4.1. Metrics methodology

This section explains the methodologies used to generate the results. Each column of the TDM represents a document in the

original document set in vector form as shown in Table 3. This is also true for the approximated TDM. The query is a row vector constructed such that its transpose can be considered equivalent to a document vector containing only the words that appear in the query. In effect, the query is a pseudo-document. For example the query (0 1 0) is a three-dimensional row vector.

Each document vector in the approximated TDM can then be compared to the query by calculating the cosine between them. The cosine is calculated from the following equation:

$$\cos\theta = \frac{a_j^{\mathrm{T}} q}{\|a_j^{\mathrm{T}}\|\|q\|} \tag{6}$$

where a_j^{T} is the transpose of the jth document vector in the approximated matrix a, q is the query vector, $\|a_j^{\mathrm{T}}\|$ is the modulus of a_j^{T}, $\|q\|$ is the modulus of q.

The modulus is equivalent to the Euclidean norm: $\|q\| = \sqrt{(q_1^2 + q_2^2 + q_3^2 + \cdots + q_{n-1}^2 + q_n^2)}$.

A cosine value of 1 means that both vectors exist in exactly the same dimensional space. Below this value the vectors become steadily less similar. In order to determine which documents are similar enough to be returned in response to a user's query, a threshold of 0.5 is set which is usually selected by the most researchers in this area. A suitable threshold value could be determined based on certain heuristics and experiments, which could be an interesting topic for future study [10,33].

Computation time is also an important factor when considering the performance of an algorithm. The system in this research generates the time taken to perform a query by comparing the time immediately before and after the query is performed.

4.2. Metrics used

There are several different measures for evaluating the performance of IR systems. The most common properties that are widely accepted by the research community are *recall* and *precision* [34]. *Precision* is the fraction of the documents retrieved that are relevant to the user's query,

$$precision = \frac{|\{relevant\ documents\} \cap \{retrieved\ documents\}|}{|\{retrieved\ documents\}|} \tag{7}$$

and *recall* is the fraction of the documents that are relevant to the query and successfully retrieved

$$recall = \frac{|\{relevant\ documents\} \cap \{retrieved\ documents\}|}{|\{relevant\ documents\}|} \tag{8}$$

Both *recall* and *precision* are needed for measuring issues in the IR. It is common to achieve recall of 100% by returning all relevant documents in response to any query. Therefore recall alone is not enough but one needs to measure the number of irrelevant documents also, to measure the precision or accuracy of the results returned. On the other hand, precision of 100% can also be achieved in many cases by returning only relevant results, but again, one needs to count all the relevant documents in the database to measure the recall. As addressed by many researchers, the SVD remains the best technique in terms of number of documents returned, which indicates a high recall level. This work will be focused on improving the accuracy of results returned.

In order to demonstrate the measures clearly, the graphs of the search results in this and the next sections show two lines or columns for each algorithm, the total number of documents returned and the number of relevant documents returned, where graphs show only one line or column for each algorithm, the total number of documents returned is equal to the number of relevant documents returned. The performance of each algorithm is affected by user entered parameters, e.g. the rank (k) value for SVD, and the threshold value for image processing transforms. Performance across a range of values is considered, with a heavy emphasis on determining optimal values, i.e. which values that give the best volume and accuracy of the returned results.

4.3. CDF9/7-SVD LSI

In this and the next section, a number of searches are performed on the sample databases to compare the basic LSI-SVD approach with the proposed hybrid technique.

- *eBooks database*: Searching for 'plastics engineering', 'xml transformations', 'health and safety' and 'advanced java programming'.
  In Fig. 7, for the first query, the CDF9/7-SVD outperforms the standard approach by having a higher precision value. The CDF9/7(soft)-SVD returns two extra unrelated results, while the basic LSI returns seven. Both approaches have recall of 100%, while a precision value of 93% is indicated for the soft-thresholding approach and a precision value of 80% for the standard SVD. The CDF9/7(hard)-SVD returns only one less relevant result and does not produce any irrelevant documents, which results in precision of 100% and recall of 96%. At the second query the CDF9/7(hard)-SVD approach returns one less irrelevant result than the other approaches and obtains a higher precision value, and thus performs better. All the approaches have recall of 100%. For the third query, a lower precision value is indicated for the standard method (71%), by returning four unrelated extra documents. A precision of 100% is achieved with the hard function-based new approach, by returning only the relevant results with one less relevant document resulting in a recall of 90%. The new method, with the soft thresholding, failed in this query by returning only four related results. For the last query, the new approach again performs well in improving the precision action by removing unrelated documents returned by the standard method. The CDF9/7(hard)-SVD returns one more irrelevant documents than the soft function. Precision values of 94% for the CDF9/7(soft)-SVD, 89% for the CDF9/7(hard)-SVD and finally 72% for the standard SVD are obtained. The three approaches achieve recall values of 100%.
- *Reuters database*: Searching for 'Japan', 'bank', 'money market' and 'sales tax'.
  In Fig. 8, for the first query, the standard method produces poor accuracy or precision, by returning 11 extra irrelevant documents with a precision of 67%. The new hybrid approach, with the hard function, shows excellent performance and removes the all irrelevant results while keeping all the relevant ones in the database with recall and precision of 100%. The new approach, with the soft function, also produces good results and achieves precision of 100%, while one less relevant result is returned which results in recall of 95%. The hybrid approach clearly outperforms the standard method in the second query. The standard SVD produces a relatively large volume of irrelevant documents, while the CDF9/7(hard)-SVD returns the same number of relevant results, with only extra three irrelevant documents. Recall of 100% for both, precision of 57% for the standard SVD and precision of 91% for the CDF9/7(hard)-SVD are obtained. The CDF9/7(soft)-SVD performs well and achieves precision of 93%, and produces a slightly lower volume of relevant results achieving recall of 93%. On the third query, the hybrid novel approach, with both thresholding methods, keeps improving the accuracy of the results returned. Although the CDF9/7(hard)-SVD and the standard SVD achieve recall values of 100%, a precision value of 82% is achieved by

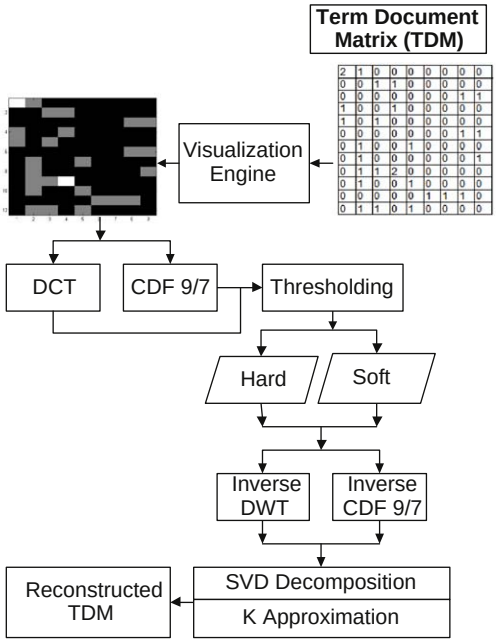

Fig. 6. The hybrid method.

Table 3
Each column represents a document as a three-dimensional column.

0	0	1	1
0	1	1	0
1	0	0	0

the hard-thresholding function approach and a precision value 75% is achieved for the standard SVD. The CDF9/7(soft)-SVD has precision of 100%, but at the same time produces one related document less, resulting in a recall of 94%. The hybrid approach with the soft-thresholding function failed in the last search, by producing a low recall value of 44%. While the CDF9/7(hard)-SVD remains powerful in obtaining higher accuracy (precision) than the standard SVD.

And, as in most of the search queries, the CDF9/7(hard)-SVD performs better than the CDF9/7(soft)-SVD in terms of recall action.

- *CDF9/7-SVD Analysis*: The results show that, the hybrid method, at a specific threshold value, can remove many irrelevant documents returned by the standard LSI. Thus, the accuracy or precision can be improved. The hybrid technique with the hard-thresholding function shows some steadiness in all cases, while some inconstancy can be noticed with the soft-thresholding performance. The best threshold values vary from database to database. Consequently a generic value for the threshold cannot be determined at this line.

4.4. DCT-SVD LSI

- *eBooks database*: Searching for 'plastics engineering', 'xml transformations', 'health and safety' and 'advanced java programming'.
  In Fig. 9, in the first query, the hybrid technique performs very well by removing unrelated results and achieving higher precision. The DCT(hard)-SVD returns two less irrelevant results than the standard method, resulting in precision of 85% and 80% for the standard SVD. The power of the new technique is demonstrated clearly by the DCT(soft)-SVD in removing seven unrelated documents and returning only the relevant documents that were returned by the standard method. This results in a precision value of 100%, which is a considerable improvement over the standard method. All three approaches obtain recall of 100%. Again in the second query, the DCT(hard)-SVD approach returns 10 documents, all of which are relevant, and the DCT(soft)-SVD returns one less relevant result, generating a recall value of 91%. Both approaches do not return any irrelevant result, achieving precision of 100%. A precision value of 85% is obtained for the standard SVD. In the third query, the standard method returns four unrelated extra documents with a precision of 71%. The new approach returns three documents with the hard function method achieving a precision of 77%. The new method with the soft thresholding failed in this query by returning only four related results, resulting in a low recall value (40%). For the last query, the DCT(hard)-SVD returns three less irrelevant documents than the standard SVD, and thus, values of precision 73% and 84% are achieved for the standard SVD and DCT(hard)-SVD, respectively. The soft function failed again by returning only three documents all of which are related, which decreases the recall value to less than 19%.
- *Reuters database*: Searching for 'Japan', 'bank', 'money market' and 'sales tax'.
  In the first query for the search in Fig. 10, the standard method as shown before performs inefficiently in terms of precision. A considerable volume of unrelated documents are retrieved, generating a low value of precision (67%). Excellent results are returned by the new hybrid approach with the hard function. The new method returns all the relevant results in the database and does not produce any irrelevant documents with recall of 100% and precision of 100%. A less efficient performance can be noticed by the new approach with the soft-thresholding function. Although precision of 100% is achieved, this method misses 10 related documents decreasing the recall value to 55%. A strong performance is shown for the DCT(hard)-SVD in the second query, as the method returns 20 irrelevant documents less than the standard SVD, with recall of 100% and precision of 91%. As shown in the previous section for this search, a considerable volume of irrelevant documents are returned by the standard SVD, resulting in precision of 53%. As in the previous query, the DCT(soft)-SVD again performs less well than the hard one. All the documents returned are relevant but it produces a lower volume of relevant results, with a recall of 83% and precision of 100%. The standard SVD in the third and fourth queries keeps showing a lower level of accuracy for the results returned when compared with the hybrid novel approach with the hard-thresholding function. Both approaches have recall of 100%. A lower number of relevant results is returned by DCT(soft)-SVD, resulting in a lower recall value.
- *DCT-SVD Analysis*: Again the results for this hybrid technique show that, the accuracy of the results returned has been improved by applying a transform as preprocessing step for the SVD in the LSI process. The hybrid technique with the hard-thresholding function again performs better than the soft one. As noticed in most of the cases, the DCT(soft)-SVD misses more relevant results and as a result the recall value decreases.

4.5. DCT vs. CDF9/7

The results in the previous sections have shown that the CDF9/7-SVD performs slightly better than the DCT-SVD in terms of both accuracy (precision) and recall, in the searches that have been

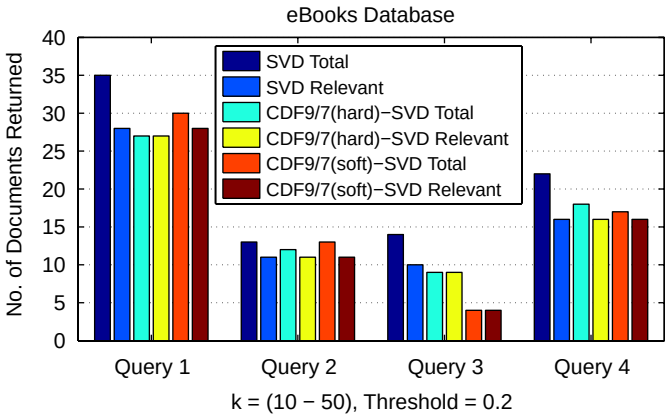

Fig. 7. CDF9/7-SVD LSI search results for eBooks database.

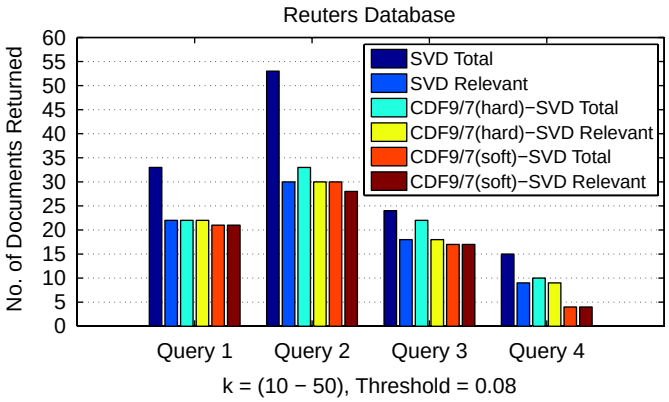

Fig. 8. CDF9/7-SVD LSI search results for Reuters database.

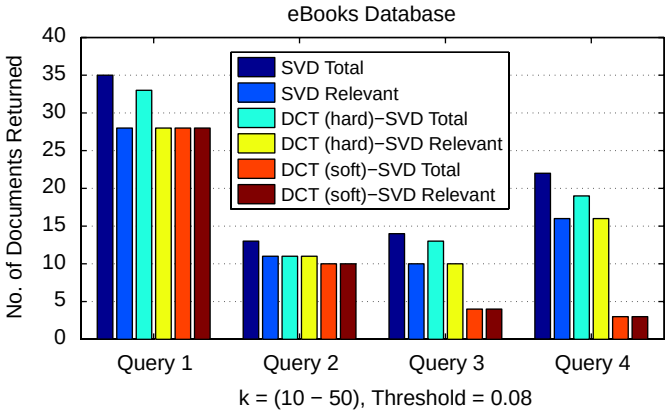

Fig. 9. DCT-SVD LSI search results for eBooks database.

carried out at the best threshold values. The criteria for selecting the best threshold value in the previous sections depend on finding a common threshold value for all the queries in a given database, for which we obtain the best results. This section presents a comparison between the two transforms used in the hybrid technique, to test performance over a range of threshold values. The results in the previous sections show that, in many cases, the hybrid methods with the hard-thresholding function perform better than the soft one. Consequently, the thresholding function used for the reminder of this investigation is the hard

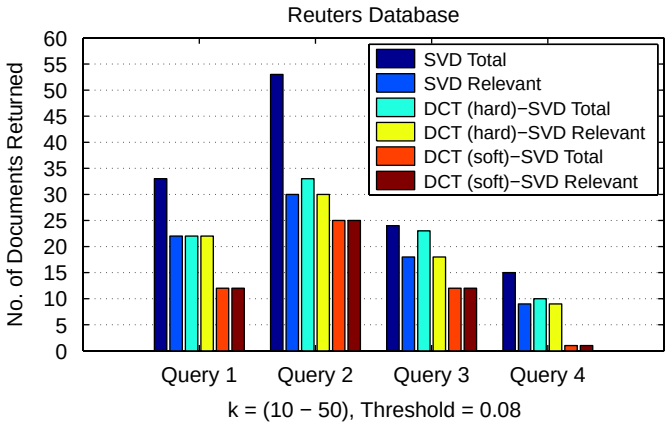

Fig. 10. DCT-SVD LSI search results for Reuters database.

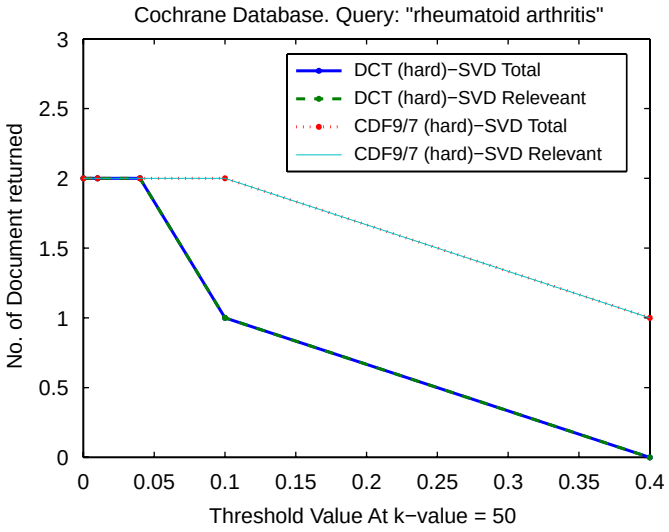

Fig. 11. DCT vs. CDF9/7 for Cochrane database.

function. (The threshold value 0 in the figures indicates that no thresholding is used, which means the result presented at this point refers to the standard SVD.)

Fig. 11 shows that, at small threshold values both methods keep returning the same results, and as the threshold increases, particulary at the threshold value 0.4, the CDF9/7-SVD returns one document, while the DCT-SVD does not return any result. All the results returned are relevant.

The results for the search query in Fig. 12 show that, at small threshold values the results for the two approaches once again remain the same as the standard SVD producing a number of irrelevant documents. These threshold values do not remove the small values in the TDM (noise) which represent the unrelated results. A threshold value of 0.1 in both approaches removes some irrelevant documents. However, the CDF9/7-SVD retrieves slightly more irrelevant results than the DCT-SVD. While at the same time, and for threshold value of 0.4, the CDF9/7-SVD keeps producing all the relevant documents in the database with no irrelevant results, and the DCT-SVD failed to retrieve any results. This failure may be due to the fact that the coefficients for the DCT-SVD are smaller and thus, the high threshold values remove important information from the TDM.

For Fig. 13, and at threshold values in the range (0.01–0.08), the CDF9/7-SVD returns slightly more results, but these results are not relevant. Both methods return the same number of related

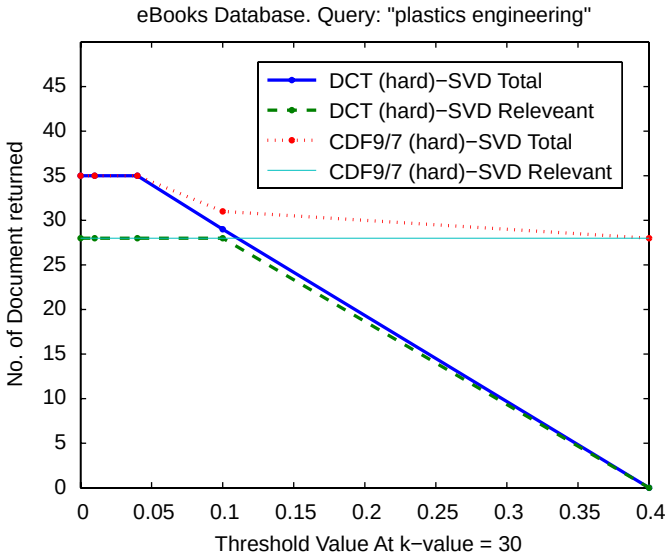

Fig. 12. DCT vs. CDF9/7 for eBooks database.

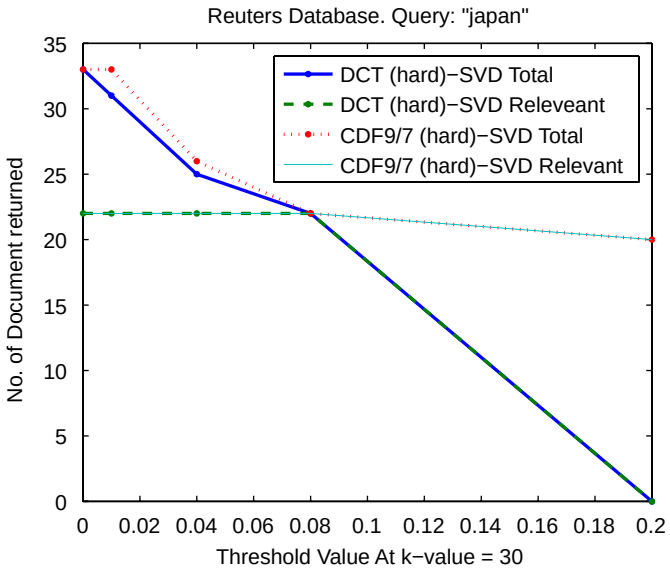

Fig. 13. DCT vs. CDF9/7 for Reuters database.

documents. Again, at higher threshold values, in particular 0.2, the CDF9/7-SVD keeps returning a good number of relevant documents, while the DCT-SVD returns no results and obviously fails at this threshold value.

4.6. TDM modeling

This section presents different investigations for the LSI system, more precisely on the TDM, and in addition provides a simple illustration for the SVD algorithm at the decomposition stage in the LSI process. A number of TDMs and query vectors, with different structures and degrees of sparsity, are generated and tested in the LSI system. The aim behind this analysis is to study the effect of structure, degree of sparsity, distribution and any other attributes of the TDM on the search results. With the obvious goal of determining the best characteristics of TDM that will give the best results.

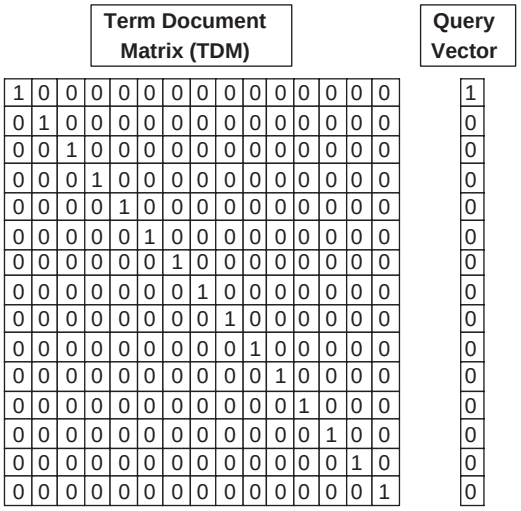

Fig. 14. Diagonal TDM of 1's and one term query vector.

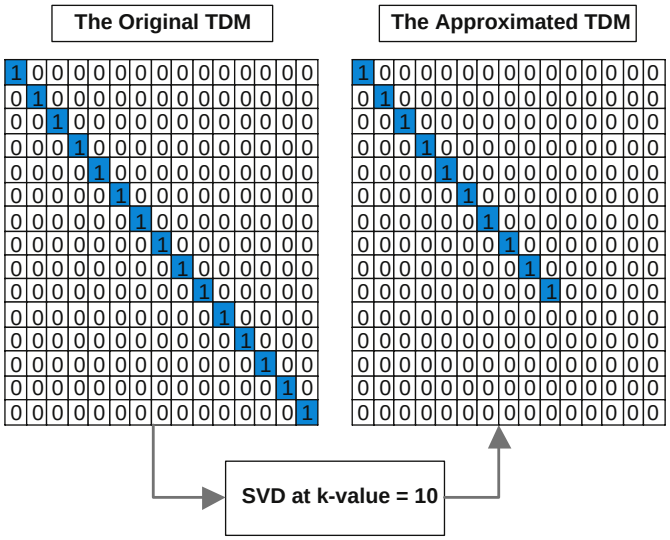

Fig. 15. The original diagonal TDM and the approximated TDM.

The LSI search is carried out for the different random TDMs which are presented in the following sections. A number of figures for the TDMs and the queries are generated to present clear illustrations.

- *Example one*: In this example a 15×15 diagonal matrix, where the diagonal elements are ones and non-diagonal elements are zeros, is generated to test in the LSI system, a pseudo-query vector is also created.
  For Fig. 14, the results of the search always return one document at different k-values for the SVD algorithm in the LSI process, and it is the first document in the TDM.
  At the decomposition step in the LSI system, the SVD decomposes the TDM to three matrices, one of them the diagonal matrix S, that holds the singular values of the original TDM in ascending order, in order to apply the dimension reduction on this matrix to remove the small singular values which represent the noise. The same procedure is applied for the above example, the TDM is diagonal matrix of ones, and it is decomposed into three diagonal matrices. Since there is no change to the TDM, the singular diagonal values of the S matrix are still ones, and applying the dimension reduction at

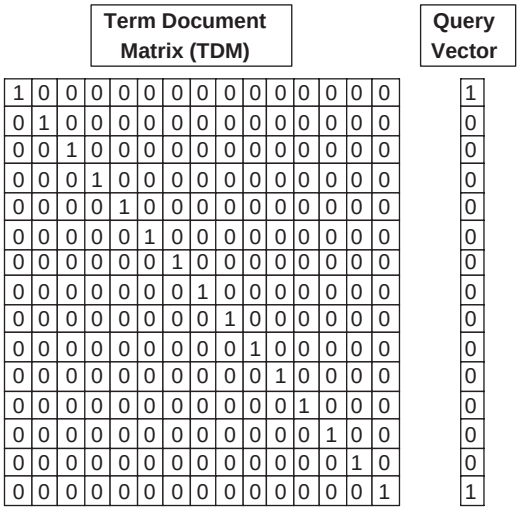

Fig. 16. Diagonal TDM of 1's and two terms query vector.

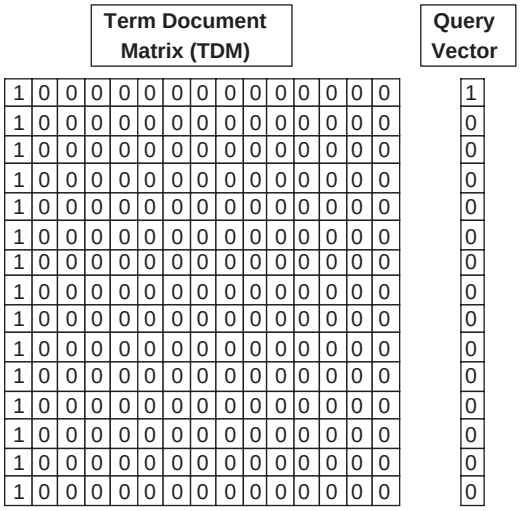

Fig. 17. TDM with one column of 1's and one term query vector.

different *k*-values on the matrix will result in ignoring the same diagonal values in the original TDM. Fig. 15 clarifies this point.

This query will always return one document, that being the first document in the TDM, which has a cosine value of one, indicating an identical match with the query, it is also important to note that no other cosine values are returned above zero for the other documents. This occurs due to the query having only one term, and since the matrix is diagonal, the term will only appear in one document.

- *Example two*: The query vector is amended for this example by adding another term at the bottom of the vector as shown in Fig. 16.

  In this example, and when the LSI is applied at different *k*-values, only one document is returned which is the first in the TDM.

  The query intersects the TDM at the first and last documents, and as mentioned before the TDM is diagonal and therefore the *S* matrix is also diagonal, and so removing any elements from the diagonal values results in removing the same diagonal values in the original TDM. Thus, when the SVD is applied at range of *k*-values (1–14), only the first document is returned, because at this range of values the last document, which is document number 15, is ignored. It is important to note that, applying the LSI at *k*-value 15, which means no dimension reduction occurs, returns two documents, the first and the last documents in the TDM, which have a cosine value of 0.7071, indicating high match with the query, since the query differs from the both documents by one term.

  The above two examples clearly show that, such TDM structure does not support the LSI searching, since it represents very poor database, that contains 15 documents each of which consists of only one word. Bigger sizes of the same structure of TDM have been tested and produce the same results.

- *Example three*: This and the next examples test other structures of TDM, in order to determine more features of the efficient databases.

  For Fig. 17, no results are returned in this search at any of the *k*-values. As shown in the figure, the TDM has only one column of ones, which is the first, and zeros elsewhere, and the query for this search contains one term and it is the first one. The difference between the query and the first document in the TDM is big, therefore the cosine value for this document is less than 0.5, which is the threshold cosine value, all documents not exceeding the threshold are not returned. Making changes

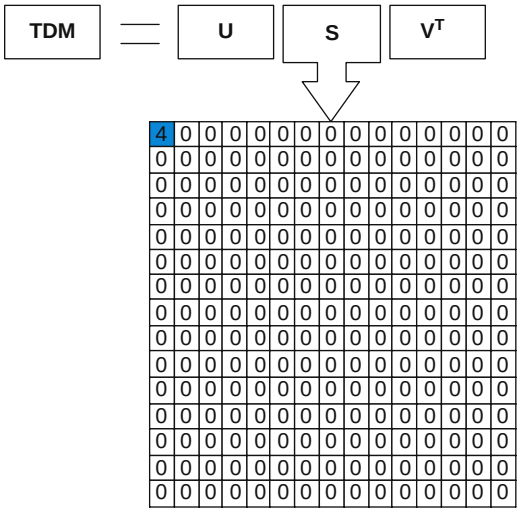

Fig. 18. The *S* diagonal matrix of the TDM.

to the query by adding some terms results in improving the similarity and increasing the cosine value between the query and the first document in the TDM, consequently, and for more than three terms the cosine value of this document exceeds 0.5, and the document is returned at the different *k*-values.

In the previous examples of TDM structure, the *S* diagonal matrix was the same as the original TDM, which has diagonal values of ones, therefore the stage of dimension reduction affects the original TDM by eliminating some values. In this example, and at the different *k*-values, the approximated TDM remains the same as the original TDM with no change. Fig. 18 shows the *S* diagonal matrix of the TDM.

As shown, the *S* diagonal matrix contains only one non-zero value and it is the first value, whereas all other diagonal values are zeros. As a result and since the *k*-value for the SVD algorithm must be at least one, the dimension reduction stage at different *k*-values has no effect on the TDM.

- *Example four*: At Fig. 19 for this example, another column of ones is added to the TDM, and the search is again applied for the same query.

  Again, the same scenario occurs as in the previous example, but there is one more extra column of ones, which is the third in the TDM, and no results return at the different *k*-values for

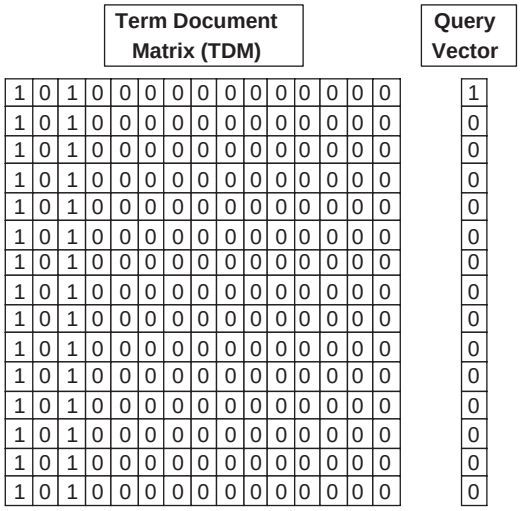

Fig. 19. TDM with two columns of 1's and one term query vector.

this query. For this example as well and for more than three terms, the cosine value of the two documents exceeds 0.5, and documents one and three in the TDM are returned at the different *k*-values. The *S* diagonal matrix for this TDM contains, as in the previous example, a high value and it is the first value of the diagonal, the second and the third values are very small negative values, and all the other diagonal values are zeros. The SVD at the small *k*-values (1, 2) has insignificant impact on the TDM.

For example three and four, the TDM contains only one and two columns of ones, respectively, this column represent a document in the database, in other word, all the keywords in the TDM appear only in these two documents. Such poor structure of databases does not reflect accurate trends in the LSI system, e.g. very bad distribution for the non-zero values in the matrix (the appearance for the keywords in the documents of the database), even though the volume of non-zero values in the TDM according to the size of the matrix is good.

4.7. Analysis

In this research, the TDM is a two-dimensional array, with most of the values being 0 or 1. So the high frequencies represent important data or details, and deleting these data will affect the structure of the TDM [35]. Whereas the low frequencies (0's values in a typical TDM) represent lexical noise. The DCT and CDF9/7 transform the image by applying a forward transform. Then a thresholding function is applied to remove the small values from the TDM. As mentioned before, the small values are interpreted as noise or reference to unrelated documents. Hence, their removal should not have a negative effect on the results. The new matrix can then be constructed by applying the inverse DCT and CDF9/7. For the hybrid approaches CDF9/7-SVD and DCT-SVD, the results show that by adding the transform as a preprocessing step the precision is improved by approximately 29% and 25% for the CDF9/7-SVD and DCT-SVD, respectively. Both hybrid approaches remove irrelevant documents, and in many cases return the same number of relevant documents as the standard SVD approach. The new methods use two thresholding functions to threshold the TDM. The results indicate that the hard-thresholding function shows more consistency and performs better than the soft-thresholding function. In the investigation of both transforms at different threshold values, the results show that at the optimal threshold values the performance remain the same. However, when the threshold value increases and becomes less optimal the CDF9/7-SVD performs better. This means that the high threshold values affect the DCT more than the CDF9/7. This may be due to the fact that the values of the coefficients for CDF9/7 are slightly larger than the values for DCT.

In the other part of the investigations, random TDMs with different structures and degree of sparsity were generated and tested in the LSI system with some random query vectors, in order to determine the mathematical description of the TDM structure which gives the best search results. LSI attempts to compare the conceptual meaning among documents rather than word alone, by returning documents that are similar in meaning, even though the keywords in the search query may not appear in the document's description. In order to achieve that, a good database structure is needed, the sparsity (volume of non-zero values) and good distribution in the TDM are very important attributes to obtain a good structure of database that being support the LSI searching. Database with more entries and longer document titles, produce better search results and supports the use of the LSI system.

5. Conclusions

A new hybrid modified approach to LSI for effective use in IR has been presented in this paper. The results of the investigation for different approaches show that, applying DCT and CDF9/7 as a preprocessing step prior to the SVD in the LSI process shows good results, the preprocessing step tends to remove irrelevant documents from the documents returned, causing enhancement on the accuracy of the results returned. It also offers the possibility of other combined approaches. It is beneficial to note that the precise action of the preprocessing step depends on the value of *k* used for the SVD and the threshold value used in the transform, but for most optimal cases the results show that adding CDF9/7 and DCT as preprocessing improves the precision of the documents returned. The investigations of the TDM modeling show that, the LSI search results rise with larger databases, and longer document descriptions increase the keywords appearing in more than a couple of documents. In turn, this will reduce the sparseness (lexical noise) in the TDM. Recent works have been published on tensor and subspace methods [36–38], which would form an excellent basis for future research, as the work presented in this paper is strongly relevant to subspace analysis.

References

[1] M. Berry, S. Dumais, G. O'Brien, Using linear algebra for intelligent information retrieval, SIAM Review 37 (1995) 573–595.
[2] M. Bell, N. Degani, Latent semantic indexing, parallel svd and its applications, in: Proceedings of ALGORITMY 2002, 2002, pp. 113–120.
[3] S. Deerwester, S. Dumais, T. Landauer, G. Furnas, R. Harshman, Indexing by latent semantic analysis, Journal of the Society for Information Science 41 (1990) 391–407.
[4] D. Kalman, A singularly valuable decomposition: the svd of a matrix, College Mathematics Journal 27 (1996) 2–23.
[5] C. Fox, Lexical analysis and stoplists, in: Information Retrieval—Data Structures and Algorithm, Prentice-Hall, Englewood Cliffs, NJ, 1992, pp. 102–130.
[6] C. Chen, N. Stoffel, M. Post, C. Basu, D. Basu, C. Behrens, Telcordia lsi engine: implementation and scalability issues applied, in: Proceedings of the International Workshop on Research Issues in Data Engineering (RIDE), 2001, pp. 51–58.
[7] T. Kolda, D. O'Leary, A semi-discrete matrix decomposition for latent semantic indexing in information retrieval, ACM Transactions on Information Systems 16 (1998) 322–346.

[8] E. Hoenkamp, Unitary operators on the document space source, Journal of the American Society for Information Science and Technology 54 (2003) 314–320.
[9] I. Syu, S. Lang, N. Deo, A neural network model for information retrieval using latent semantic indexing, in: The IEEE International Conference on Neural Networks, vol .2, 1996, pp. 1318–1323.
[10] J. Gao, J. Zang, Clustered svd strategies in latent semantic indexing, Laboratory for High Performance Scientific Computing and Computer Simulation, University of Kentucky, 2004.
[11] P. Husbands, H. Simon, C. Ding, On the use of the singular value decomposition for text retrieval, in: Computational Information Retrieval (SIAM), 2001, pp. 145–156.
[12] D. Bassu, C. Behrens, Distributed lsi: scalable concept-based information retrieval with high semantic resolution, in: Proceedings of the 2003 Text Mining Workshop, 2003, pp. 72–82.
[13] R. Liu, T. Tan, An svd-based watermarking scheme for protecting rightful ownership, IEEE Transactions on Multimedia 4 (2002) 121–128.
[14] M. Littman, S. Dumais, T. Landauer, Automatic cross-language information retrieval using latent semantic indexing, in: SIGIR'96—Workshop on Cross-Linguistic Information Retrieval, 1996, pp. 16–23.
[15] T.K. Landauer, D. Laham, P. Foltz, Learning human-like knowledge by singular value decomposition, in: Proceedings of the 10th Conference on Advances in Neural Information Processing Systems, 1997, pp. 45–51.
[16] T.A. Letsche, M.W. Berry, Large-scale information retrieval with latent semantic indexing, Information Sciences: International Journal 100 (1997) 105–137.
[17] R. Zhao, W.I. Grosky, Narrowing the semantic gap-improved text-based web document retrieval using visual features, IEEE Transactions on Multimedia 4 (2002) 189–200.
[18] A. Kontostathis, Essential dimensions of latent semantic indexing (lsi), in: Proceedings of the 40th Hawaii International Conference on System Sciences—2007, 2007, pp. 73–80.
[19] H. Ito, H. Koshimizu, Keyword and face image retrieval based on latent semantic indexing, in: IEEE interactional Conference on Systems, Man and Cybernetics, vol. 1, 2004, pp. 358–363.
[20] M.M. Rahman, B.C. Desai, P. Bhattacharya, Visual keyword-based image retrieval using latent semantic indexing, correlation-enhanced similarity matching and query expansion in inverted index, in: Proceedings of the 10th International Database Engineering and Applications Symposium, 2006, pp. 201–208.
[21] Cochrane, URL: ⟨http://www.cochrane.org⟩.
[22] eBooks, URL: ⟨http://www.library.qub.ac.uk⟩.
[23] Reuters, URL: ⟨http://www.daviddlewis.com/⟩.
[24] J. Yu, Singular value decomposition with applications to ir and text clustering, Technical Report, School of Electronics Engineering and Computer Science, Peking University, Beijing, 2003.
[25] R. Baeza-Yates, B.R. Neto, Modern Information Retrieval, ACM Press, New York, 1999, pp. 126–127.
[26] M. Unser, T. Blu, Mathematical properties of the jpeg2000 wavelet filters, IEEE Transactions on Image Processing 12 (2003) 1080–1090.
[27] K. Delac, M. Grgic, S. Grgic, Towards face recognition in jpeg2000 compressed domain, in: Proceedings of the 14th International Workshop on Systems, Signals and Image Processing (IWSSIP) and the 6th EURASIP Conference Focused on Speech and Image Processing, Multimedia Communications and Services (EC-SIPMCS), 2007, pp. 148–152.
[28] S.A. Khayam, The discrete cosine transform (dct): theory and application, Technical Report, DCT Tutorial, 2003.
[29] G.K. Wallace, The jpeg still picture compression standard, IEEE Transactions on Consumer Electronics 38 (1992) xviii–xxxiv.
[30] J. Cox, J. Kilian, F.T. Leighton, T. Shamoon, Secure spread spectrum watermarking for multimedia, IEEE Transactions on Image Processing 6 (1997) 1673–1687.
[31] B. Yoon, P.P. Vaidyanathan, Wavelet-based denoising by customized thresholding, in: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 2, 2004, pp. 925–928.
[32] T. Jaber, A. Amira, P. Milligan, A novel approach for lexical noise analysis and measurement in intelligent information retrieval, in: Proceedings of IEEE International Conference on Pattern Recognition ICPR, Hong Kong, vol. 3, 2006, pp. 370–373.
[33] E.R. Jessup, J.H. Martin, Taking a new look at the latent semantic analysis approach to information retrieval, Computational Information Retrieval (2001) 121–144.
[34] A. Singhal, Modern information retrieval: a brief overview, IEEE Data Engineering Bulletin 24 (2001) 35–43.
[35] T. Jaber, A. Amira, P. Milligan, Empirical study of a novel approach to lsi for text categorisation, in: Proceedings of the IEEE Symposium on Signal Processing and Its Applications ISSPA, Sharjah, UAE, 2007.
[36] D. Tao, X. Li, X. Wu, S. Maybank, Geometry mean for subspace selection in multiclass classification, IEEE Transactions on Pattern in Analysis and Machine Intelligence 30.
[37] J. Sun, D. Tao, S. Papadimitriou, P. Yu, C. Faloutsos, Incremental tensor analysis: theory and applications, ACM Transactions on Knowledge Discovery from Data 2
[38] D. Tao, X. Li, X. Wu, S. Maybank, Supervised tensor learning. Knowledge and Information Systems 13 (2007) 1–42.

Tareq Jaber has received his PhD in Computer Science from Queen's University, Belfast, UK, in 2008. He has worked on field of Information Retrieval (IR), on Latent Semantic Indexing (LSI) as a technique used for intelligent IR as an alternative to traditional keyword matching techniques. He has submitted and presented four papers at major conferences already.

Abbes Amira is a senior lecturer at Brunel University, West London, UK, within the division of Electronic and Computer Engineering in the School of Engineering and Design. Before he joined Brunel University in May 2006 he has held a lectureship in Computer Science at Queen's University, Belfast (QUB) since November 2001. He received his PhD in Computer Science from Queen's University Belfast in 2001. He has been awarded a number of grants from government and industry, has published over 100 publications and supervised five PhD students during his career to date. Dr. Amira has been invited to give talks at universities in UK, Europe, USA and North Africa, at international conferences, workshops and exhibitions and being chair, program committee for a number of well-known conferences. He is a senior member of IEEE, member of ACM, IET and Fellow of the Higher Education Academy. His research interests include information retrieval, reconfigurable computing, image and vision systems, system on chip, custom computing using FPGAs, medical image analysis, multi-resolution analysis and biometrics technologies.

Peter Milligan was awarded a BSc (Honours) degree in Computer Science and a PhD (The Synthesis of Parallel Programs) by the Queen's University of Belfast (QUB). Dr. Milligan is a senior lecturer in the School of Electronics, Electrical Engineering and Computer Science at QUB where he is the Adviser of Studies for postgraduate students and the Chair of the School Postgraduate Research Committee. Dr. Milligan has been awarded a number of grants from industry (Cray Research), local government (TBNI), SERC/EPSRC, and the EU and has published over 120 papers in peer reviewed conferences and journals. He has served on the editorial boards of four journals and has been a member of over 60 international conference and programme committees. In addition, Dr. Milligan was a member and director of the Euromicro organization for many years. Dr. Milligan is a member of the ACM and IEEE. Dr. Milligan has supervised 15 PhD and MPhil students, over 40 MSc students and currently has 10 PhD students working on various research projects. Dr. Milligan's research interests include information retrieval, reconfigurable computing, and aspects of peer-to-peer and grid systems focusing on the location and retrieval of resource information.