Citation-Based Retrieval for Scholarly Publications

Y. He, *University of Cambridge*

S.C. Hui, *Nanyang Technological University*

A.C.M. Fong, *Massey University*

Scholarly publications are available online and in digital libraries, but existing search engines are mostly ineffective for these publications. The proposed publication retrieval system is based on Kohonen's self-organizing map and offers fast retrieval speeds and high precision in terms of relevance.

Many scholarly publications are available on the Internet or in digital libraries.[1–3] However, the information is not always well organized, which makes searching for relevant publications difficult and time consuming. Commercial search engines such as Yahoo!, Lycos, and Excite help users locate specific information by matching queries against a database of stored, indexed documents. However, many of these search engines have proved ineffective for searching scholarly publications accurately.

Researchers have developed autonomous *citation indexing agents*, such as CiteSeer,[4] to search computer-science-related literature online. These agents extract citation information from the literature and store it in a database. CiteSeer can convert PostScript and PDF documents to text using the pstotext program from the Digital Virtual Paper project (see www.research.digital.com/SRC/virtualpaper/home.html and www.research.digital.com/SRC/virtualpaper/pstotext.html). The citation indices created are similar to the Science Citation Index.[5] As such, you can generate a *citation database* to store citation information. When new publications become available, the citation indexing agent can detect them and store them in the database.

Citation databases contain rich information that you can mine to retrieve publications. Our intelligent-retrieval technique for scholarly publications stored in a citation database uses Kohonen's self-organizing map (KSOM).[6] The technique consists of a training process that generates cluster information and a retrieval process that ranks publications' relevance on the basis of user queries.

The scholarly publication retrieval system

Figure 1 shows our scholarly publication retrieval system's architecture. Without losing generality, we focus on scholarly publications found on the Internet. The term *publications repository* emphasizes the fact that we could apply the same retrieval technique to documents in a digital library.

In our system, the two major components are the citation indexing agent and the *intelligent retrieval agent*. The citation indexing agent finds scholarly publications in two ways. The first is similar to CiteSeer, which uses search engines to locate Web sites containing publication keywords. The other lets users specify publication Web sites through *indexing clients*. The indexing clients let users determine how often the citation indexing agent visits these sites. The indexing agent then downloads the publications from the Web sites and converts them from PDF or PostScript format to text using the pstotext tool. It identifies the Web publications' bibliographic section through keywords such as "bibliography" or "references," then extracts citation information from the bibliographic section and stores it in the citation database.

In addition to using indexing clients, the system incorporates several *retrieval clients*. A retrieval client provides the necessary user interface for inputting queries that will be passed to the intelligent retrieval agent for further processing. The intelligent retrieval agent mines the citation database to identify hidden relationships and explore useful knowledge to improve the efficiency and effectiveness of scholarly literature retrieval.

The citation database

Published papers generally contain some cited references for readers to probe further. These citations provide valuable information and directives for

1094-7167/03/$17.00 © 2003 IEEE
Published by the IEEE Computer Society

researchers in the exchange of ideas, current trends, and future developments in their respective fields. A citation index contains the references a paper cites, linking the source literature to the cited works. You can use citation indices to identify existing research fields or newly emerging areas, analyze research trends, discover the scholarly impact, and avoid duplicating previously reported works.

A citation database is a data warehouse for storing citation indices. Some of the stored information includes the author name, title, and journal name. The database contains all the cited references (footnotes or bibliographies) published with the articles. These references reveal how the source paper is connected to prior relevant research because the citing and cited references have a strong link through semantics. Therefore, citation indices can help facilitate the search for and management of information. Some commercial citation index databases, such as those the Institute for Scientific Information (ISI) provides, are available online (see www.isinet.com).

As we discussed earlier, the citation indexing agent can generate a citation database. For our experiment, we set up a test citation database by downloading papers published from 1987 to 1997 that were stored in the information retrieval field of ISI's Social Science Citation Index, which includes all the journals on library and information science. We selected 1,466 IR-related papers from 367 journals with 44,836 citations.

Figure 2 shows our citation database's structure, which consists of a source table and a citation table. The source table stores information from the source papers, and the citation table stores all the citations extracted from the source papers. Most of these two tables' attributes are identical—for example, paper title, author names, journal name, journal volume, journal issue, pages, and year of publication. You can access a linked article's full text through the URL stored as one of the attributes. The primary keys in the two tables are paper_ID in the source table and citation_ID in the citation table. In the source table, no_of_citation indicates the number of references the source paper contains. In the citation table, source_ID links to paper_ID in the source table to identify the source paper that cites the particular publication the citation table is storing. If two different source papers cite a publication, it is stored in the citation table with two different citation_IDs.

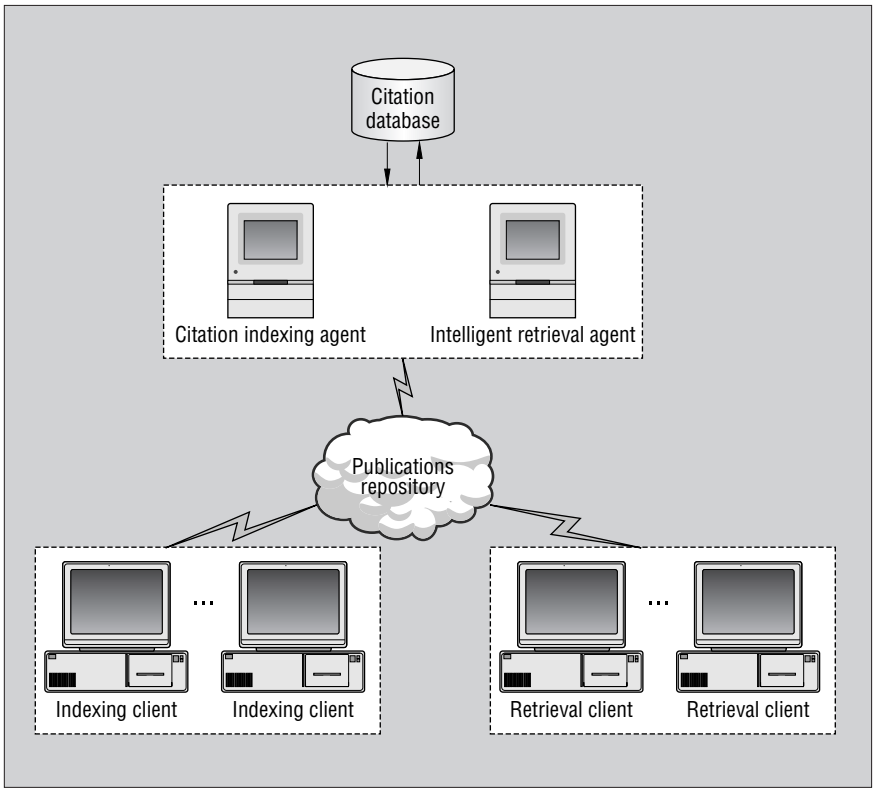


Figure 1. The scholarly publication retrieval system architecture.

Document clustering using KSOM

Various artificial neural network models have been applied to document clustering. In particular, KSOM is a popular unsupervised tool for ordering high-dimensional statistical data in a way similar to how input items are mapped close to each other. With KSOM, users can display a colorful map of topic concentrations that they can further explore by drilling down to browse the specific topic, as Websom demonstrates.[7] AuthorLink, which provides a visual display of related authors, demonstrates the benefits of using KSOM to generate a visualization interface for co-citation mapping (see http://cite.cis.drexel.edu and http:faculty.cis.drexel.edu/~xlin/authorlink.html).

KSOM is essentially a stochastic version of K-means clustering (see the "Document Clustering Techniques" sidebar). What distinguishes KSOM from K-means is that KSOM updates not only the closest model but also the neighboring models. Because the KSOM algorithm is a nonlinear projection of the patterns of arbitrary dimensionality into a one- or two-dimensional array of neurons, the neighborhood refers to the best-

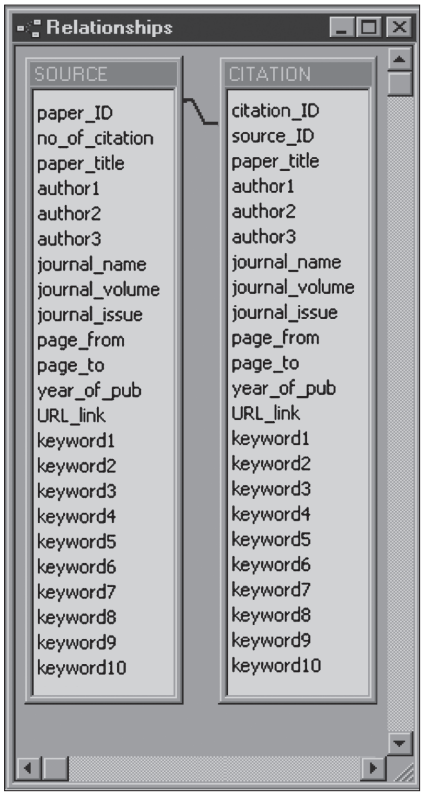


Figure 2. The citation database.

Document Clustering Techniques

Clustering approaches using TFIDF (term frequency × inverse document frequency) representations for text are the most popular.[1] Each component of a document vector is calculated as the product of TF (term frequency, or the number of times word w_i occurs in a document) and IDF (log[D/DF(w_i)], where D is the number of documents and DF(w_i), or document frequency, is the number of documents where word w_i occurs at least once). We use the cosine measure, which is a popular similarity measure, to compute the angle between any two sparse vectors. Using this method, we can classify the documents into different groups according to the distance between them.

We broadly divide clustering algorithms into two basic categories: hierarchical and nonhierarchical algorithms.[2] As the name implies, hierarchical clustering algorithms involve a tree-like construction process. Agglomerative hierarchical clustering algorithms are among the most commonly used. These algorithms are typically slow when applied to large document collections. AHC algorithms are sensitive to halting criteria because the stopping point greatly affects the results: a cluster combination created too early or too late often causes poor results. Nonhierarchical clustering algorithms select the cluster seeds first and assign objects into clusters on the basis of the seeds specified. The seeds might be adjusted accordingly until all clusters are stabilized. These algorithms are faster than AHC algorithms. The K-means algorithm[3] is a nonhierarchical clustering algorithm that can produce overlapping clusters. However, its disadvantage is that the selection of initial seeds might greatly impact the final result.

Recently, many other document-clustering algorithms have been proposed, including Suffix Tree Clustering,[4] Distributional Clustering,[5] and supervised clustering.[6] Suffix Tree Clustering is a linear-time clustering algorithm based on identifying the phrases that are common to groups of documents, as opposed to other algorithms that treat a document as a set of unordered words. Distributional Clustering clusters words into groups on the basis of the distribution of class labels associated with each word. This way, the dimensionality of a document's feature space is reduced while maintaining the document classification accuracy. In contrast to all other clustering algorithms, which are unsupervised clustering, supervised clustering starts with a set of seeds that represent the classes in the original taxonomy. The subsequent clustering process is independent of any further supervision. The number of clusters is maintained by either merging two clusters—if the similarity of their seeds is higher than a predefined threshold—or by discarding a cluster, if the number of documents in the corresponding cluster is less than a predefined value.

References

1. G. Salton and M.J. McGill, *Introduction to Modern Information Retrieval*, McGraw-Hill, 1983.
2. L. Kaufman and P.J. Rousseeuw, *Finding Groups on Data: An Introduction to Cluster Analysis*, John Wiley & Sons, 1990.
3. J.J. Rocchio, *Document Retrieval Systems—Optimization and Evaluation*, doctoral dissertation, Computational Laboratory, Harvard Univ., 1966.
4. O. Zamir and O. Etzioni, "Web Document Clustering: A Feasibility Demonstration," *Proc. 21st Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval*, ACM Press, 1998, pp. 46–54.
5. L. Baker and A. McCallum, "Distributional Clustering of Words for Text Classification," *Proc. 21st Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval*, ACM Press, 1998, pp. 96–103.
6. C. Aggarwal, S. Gates, and P. Yu, "On the Merits of Building Categorization System by Supervised Clustering," *Proc. 5th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, ACM Press, 1999, pp. 352–356.

matching neuron's spatial neighbors, which are also updated to react to the same input.

Adapting KSOM for retrieving scholarly publications

Citation information lets us judge document relevance because authors cite articles that are related. The measure CCIDF (common citation × inverse document frequency) is analogous to the word-oriented TFIDF (term frequency × inverse document frequency) word weights.[8] CCIDF assigns a weight to each citation that equals the inverse of the citation's frequency in the entire database. Then, for every two documents, the weights of their common citations are summed. The resulting value indicates how related the two documents are—the higher the summed value, the greater the relationship. However, we find that this method is not very practical; identifying citations to the same article is difficult because they can be formatted differently in different source papers. Therefore, we propose a new way of calculating documents' relatedness in the citation database.

Instead of extracting keywords from the document as the feature factors, we can actually extract the keywords from its citations. If two documents share the same citation, they must also share the same keywords. In the citation database, the full-text content of cited articles is not available. The keywords are extracted solely from the titles of all citations. Each extracted keyword forms an element of a document vector. If $\mathbf{d}$ denotes the document vector, then each keyword will be denoted by d_i where i is between 1 and N, and N is the total number of distinct keywords. For each document, we extract the 20 most frequently occurring keywords from its citations as the feature factors. (We determined experimentally that using 20 keywords gives us the best result.) We can then adopt the TFIDF method (see the sidebar) to represent the document vector. After solving the document representation problem, we use KSOM to categorize documents in the citation database.

Figure 3 shows the citation-based retrieval technique using KSOM. The citation indexing agent generates the citation database. The KSOM retrieval technique consists of two processes: training and retrieval. The training process mines the citation database to generate cluster information, and the retrieval process retrieves and ranks the publications

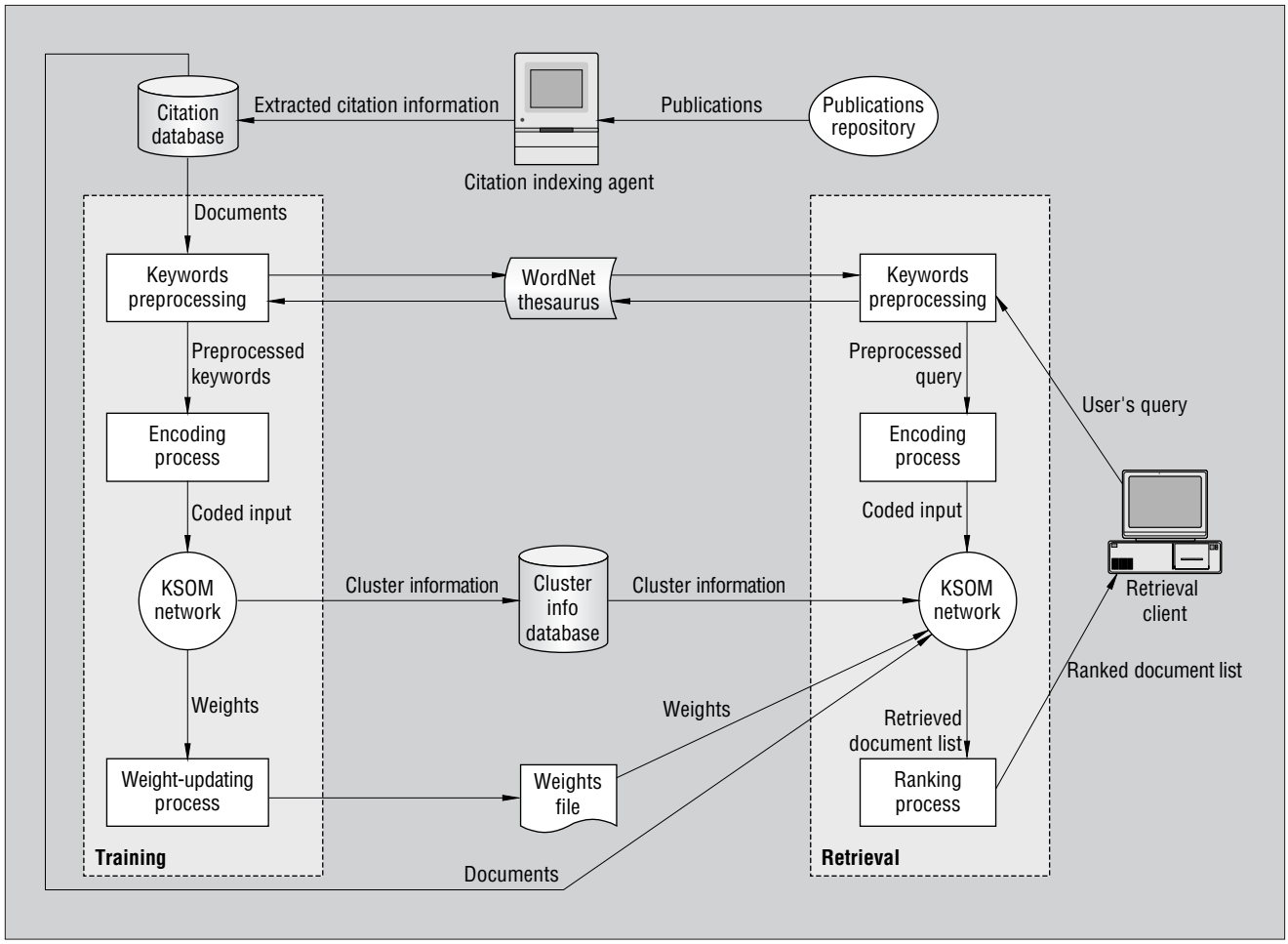


Figure 3. The citation-based retrieval technique using the Kohonen's self-organizing map (KSOM) network.

according to user queries through the retrieval client.

Training process

The system is first trained using existing documents from the citation database to group the potentially relevant documents. During the training process, the system preprocesses the keywords of each document stored in the citation database and encodes them into tuples in floating-point representations. It then feeds them into the KSOM neural network to determine the winning clusters (those closest to the input vector). The system then updates the winning clusters' weights and saves them to a file.

Keyword preprocessing uses the WordNet[9] thesaurus to remove stop words—for example, "the," "a," and "to"—using a stop list and to stem extracted keywords into their root forms. For example, "deletion," "deletes," and "deleted" are various inflectional forms of the stem word "delete." WordNet also identifies antonym forms of words and converts them to the form of "not + verb" using a logical-not list. For example, "not delete" and "cannot delete" are converted to "not + delete." Figure 4 shows the keyword preprocessing algorithm, which we implemented using algorithms from WordNet.

1. Sort all records in the Web citation database in ascending order of source `paper_ID`.
2. For each document entry read from the Web citation database, do steps 3 through 6.
3. If the source `paper_ID` is the same—that is, the citations belong to the same source paper—extract all keywords from citations and store in a temporary database.
4. Stem extracted keywords to their root forms, identify logical phrases, and remove stop words.
5. For every keyword, accumulate the number of occurrences.
6. If the source `paper_ID` is different—that is, the keywords from the citations of the previous source paper are all extracted—sort the keywords for the previous source paper on the basis of their occurrence; only take the first 20 keywords as the feature factors of the previous source paper.
7. Go to step 3 for the next source paper.

Figure 4. The keyword preprocessing algorithm.

1. For each document, extract the first 20 of the most frequently occurring keywords from the documents' citations.
2. Using the vector space model, represent each document as a vector with 1,614 dimensions (corresponding to the 1,614 distinct keywords for all the citations in the citation database).
3. Each element of the vectors is weighted by TFIDF (term frequency × inverse document frequency), given by $w_i = f(t_i, d) \times \log(N/n)$, where $f(t_i, d)$ is the frequency of term t_i in document d, N is the total number of documents in the collection, and n is the number of documents containing t_i.
4. Multiply document vector by a random matrix R to reduce vector dimension.

Figure 5. The encoding algorithm.

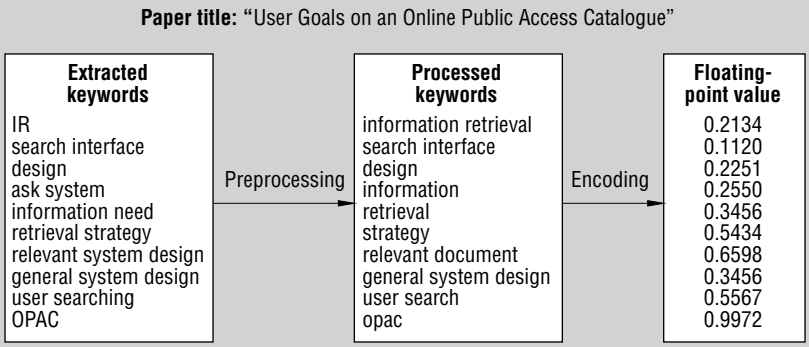


Figure 6. An example of the training process.

For each encoded input vector **x**, do steps 1 through 3, and repeat the same process for the whole training set 5,000 times:

1. Obtain the similarity measure between the input vector and the weight vectors of the output nodes, and compute the winning output node as the one with the shortest Euclidean distance given as

$$|\mathbf{x}-\mathbf{w}_m| = \min\{|\mathbf{x}-\mathbf{w}_m|\},$$

where $\mathbf{w}_i$ is the weight vector and $\mathbf{w}_m$ is the weight of the heaviest node.

2. Update weight vectors as

$$\Delta\mathbf{w}_i(t) = \alpha(N_i, t)[\mathbf{x}(t) - \mathbf{w}_i(t)] \quad \text{for } i \in = N_m(t),$$

where $N_m(t)$ denotes the current spatial neighborhood, α is a positive-valued learning function, and $0 < \alpha(N_i t) < 1$. The function of α can be represented as

$$\alpha(N_i, t) = \alpha(t) \exp\left(-\frac{|(\mathbf{r}_i - \mathbf{r}_m|}{\sigma^2(t)}\right) \quad \text{for i} \in = N_m(t)$$

where $\mathbf{r}_m$ and $\mathbf{r}_i$ are the position vectors of the winning cell and of the winning neighborhood nodes respectively, and $\alpha(t)$ and $\sigma(t)$ are suitable decreasing functions of learning time t.

3. Update the cluster information database by adding an entry to record the link between the input vector and the winning cluster number.

Figure 7. The KSOM network training algorithm.

The encoding process converts documents to vectors before feeding them into the neural network for training. Traditionally, documents are represented using the vector space model. This model's major problem is its large vocabulary, which results in a vast dimensionality of the document vectors. Latent semantic indexing[10] attempts to reduce the document vectors' dimensionality by forming a matrix in which each column corresponds to a document vector. However, this method still incurs expensive computation time. Our system uses a random projection method to reduce the document vectors' dimensionality without losing the power of discrimination between documents.[11] The basic idea is to multiply the original document vector by a random matrix $\boldsymbol{R}$ normalize the Euclidean length of each column to unity.

Figure 5 shows our encoding algorithm. We extract the first 20 of the most frequently occurring keywords from each document's citations. Altogether, 1,614 distinct keywords exist for all the citations in the citation database. Using the vector space model, each document would be represented as a vector with 1,614 dimensions. By multiplying the original document vector with the matrix $\boldsymbol{R}$, the final document vectors obtained only have 300 dimensions. This increases the learning speed dramatically, compared to previously reported methods. Figure 6 demonstrates the training process.

The reduced-dimension document vectors are then fed into the KSOM neural network to determine the winning cluster. The weights of the network are initialized with random real numbers within the interval [0, 1]. In the testing citation database, there are 1,466 records in the source table and 44,836 records in the citation table. The KSOM neural network retrieval's performance depends on how many clusters it generates and the average number of documents in a cluster. However, deciding on the cluster map's best size requires some insight into the training data's structure. In our implementation, the system generates a number of clusters equal to the square root of the total number of documents to be categorized. This achieves fast retrieval. Therefore, the number of clusters the KSOM neural network generates is set to 100. The initial neighborhood size is set to half the number of clusters. The number of iterations and the initial learning rate are set to 5,000 and 0.5, respectively. Figure 7 summarizes the KSOM network training process.

The weight-updating process is the last step of the training process. Whenever the system finds a winning cluster, it must adjust the weights of the winning cluster together with its neighborhood to adapt to the input pattern. It stores the updated weights in a file. After the training process, the system writes the cluster information into the cluster information database to indicate which document belongs to which cluster. It also stores the links between documents in the cluster information database and the original papers in the citation database.

Retrieval process

During the retrieval process, a user first submits a query as free-text natural language. The system then preprocesses, parses, and encodes the query in a way similar to the keyword preprocessing and encoding in the training process. The encoded user query inputs are 300-dimensional vectors and are compatible with the document vectors from the training process. We feed these query vectors into the KSOM network to determine which clusters should be activated. This process resembles using indices in the document-retrieval process. In this scenario, instead of having an index file, the index is encoded into the KSOM neural network in the form of weight distribution.

Once the best cluster is found, the *ranking process* uses the query term's Euclidean distance from all the documents in the cluster. This is based on the observation that the documents with a lower Euclidean distance to the query will be semantically and conceptually closer to it.[8] Given a document vector **d** and a query vector **q**, their similarity function, sim(**d**,**q**), is

$$\mathrm{sim}(\mathbf{d},\mathbf{q}) = \frac{\sum_{i=1}^{n}\left(w_{\mathbf{d}i} \times w_{\mathbf{q}i}\right)}{\sqrt{\sum_{i=1}^{n}\left(w_{\mathbf{d}i}\right)^2 \times \sum_{i=1}^{n}\left(w_{\mathbf{q}i}\right)^2}},$$

where $w_{\mathbf{d}i}$ and $w_{\mathbf{q}i}$ are the weight of the ith element in the document vector **d** and query vector **q**, respectively.

This equation measures the Euclidean distance between the document vector and the query vector. The document with the minimum value of sim(**d**,**q**) in a cluster gets the highest ranking in that cluster. Other documents in the cluster are sorted on the basis of this principle. If two or more documents in the cluster have the same sim(**d**,**q**) value, the system ranks these documents arbitrarily.

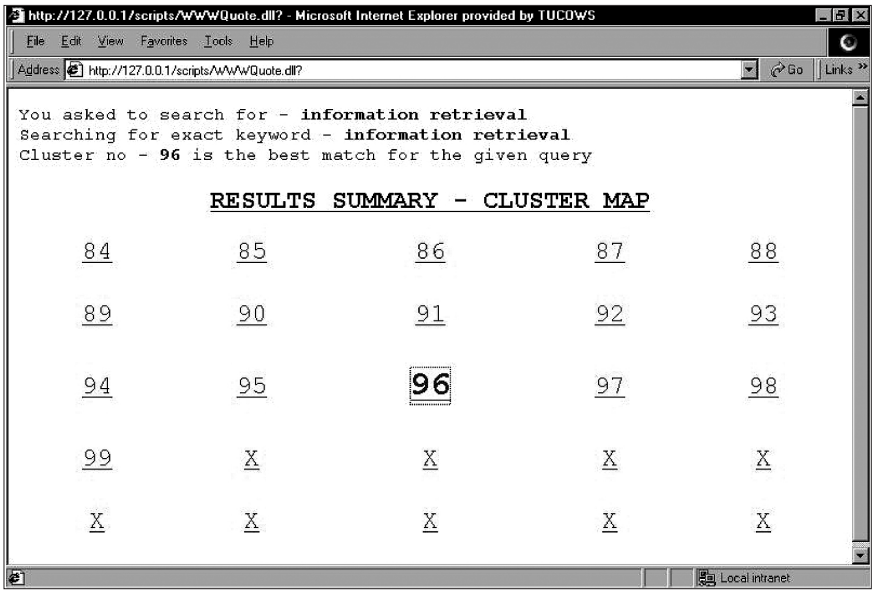


Figure 8. Cluster map for searching "information retrieval." The best matching cluster is number 96.

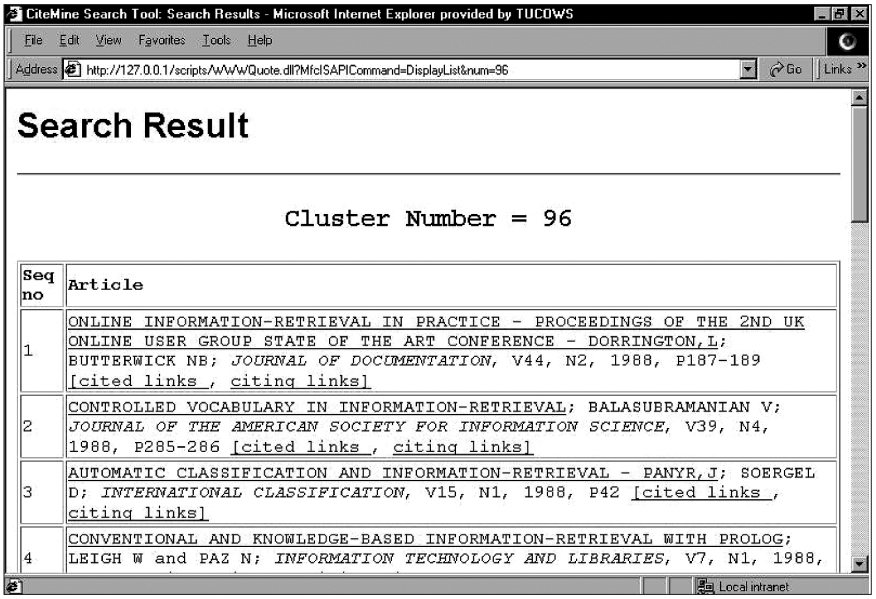


Figure 9. Search results for cluster 96.

Figure 8 shows the two-dimensional cluster map returned by searching the input string "information retrieval." In this case, the best-matching cluster is 96. The client interface lets the user browse through the cluster map. The system has grouped the documents into broad areas on the basis of the particular neighborhood relationships. Users can further examine each cluster to retrieve the documents in it. These documents are ranked on the basis of their similarity to the user's query, as described earlier.

Figure 9 shows the documents in cluster 96. The system ranks them in descending order of relevance to the query item on the basis of the similarity function. The under-

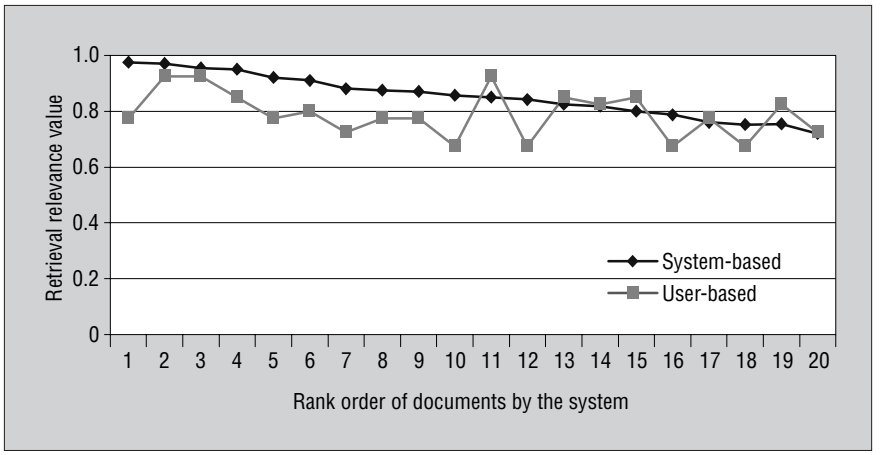


Figure 10. Comparison of retrieval relevance.

Table 1. Performance results of the KSOM network.

Training	
Preprocessing	2 min 34 sec
Number of iterations	5,000
Training time	52 min 47 sec
Total number of clusters	100
Retrieval	
Average retrieval speed	1.6 sec
System-based relevance	85.5%
User-based relevance	83.0%
Average retrieval precision	84.3%

lined paper titles are URL links that users can click to get a paper's full-text content. The "citing" and "cited" links let the user go deeper into a particular publication's citing or cited documents.

Performance evaluation

We evaluated our citation-based retrieval technique's performance on the basis of its training performance, retrieval speed, and retrieval precision. The experiments were carried out on a 450-MHz Pentium II machine with 128 Mbytes of RAM. During the experiments, we used the following test data:

- The number of keywords in the keyword list was 1,614.
- The number of words to be searched in the WordNet dictionary was 121,962.
- The number of clusters was predefined at 100.
- The total number of documents used for training (the training set) was 1,000.

We measured training performance on the basis of the number of iterations the neural network requires to reach the convergent state and the total training time. We measured the retrieval performance on the basis of the average online retrieval speed and the retrieval precision. While speeds of operation are easy to measure, retrieval precision involves considering the relevance between the query and the output. Our research used both system-based and user-based relevance measures. The former used the similarity function discussed earlier, whereas we derived the latter from the users' assessment. Miranda Lee Pao[12] defined three categories for relevance assessment: highly relevant, partially relevant, and not relevant. In our work, we introduced two more numerical categories for finer differentiation. We assigned the five categories the respective values of 1.0, 0.75, 0.5, 0.25, and 0.

We conducted the experiment as follows. We submitted 50 queries to the system, and we measured the average online retrieval speed from the time difference between the moment of query submission and the point when the user received the results. For each query result, we examined only the first 20 publications to check their relevance to the query. We obtained the user-based relevance measure from an average of 10 users. We then derived a collective measure of retrieval precision from the system-based and user-based relevance measures. Figure 10 compares system-based and user-based retrieval relevance. Table 1 gives the overall performance results. Evidently, the KSOM network can perform quite efficiently, with high-retrieval precision. Also, little difference exists between the system-based and user-based relevance measures.

Although we used Web publications as a case study for analysis, our proposed method is generic enough to retrieve relevant publications from other repositories, such as digital libraries. In terms of scalability, you can easily adapt our method to cope with large document collections. Although the number of training samples will increase with larger document collections, the average length of each document vector is not affected much. This means you only need to redefine the number of clusters KSOM generates, while other procedures remain unchanged.

Apart from the proposed citation-based document retrieval technique, we are investigating other techniques, such as co-citation analysis,[13] to support document clustering and author clustering in the publication retrieval system. Additionally, we are using KSOM to explore techniques that can enhance the visual representation of cluster maps. ❐

References

1. B. Schatz and H. Chen, "Building Large-Scale Digital Libraries," *Computer*, vol. 29, no. 5, May 1996, pp. 22–26.
2. D.M. Levy and C.C. Marshall, "Going Digital: A Look at Assumptions Underlying Digital Libraries," *Comm. ACM*, vol. 38, no. 4, Apr. 1995, pp. 77–84.
3. E.A. Fox et al., "Digital Libraries," *Comm. ACM*, vol. 38, no. 4, Apr. 1995, pp. 22–28.
4. S. Lawrence, C.L. Giles, and K.D. Bollacker, "Digital Libraries and Autonomous Citation Indexing," *Computer*, vol. 32, no. 6, June 1999, pp. 67–71.
5. E. Garfield, *Citation Indexing: Its Theory and Application in Science, Technology, and Humanities*, John Wiley & Sons, 1979.
6. T. Kohonen, *Self-Organizing Maps*, Springer-Verlag, 1995.
7. K. Lagus et al., "WEBSOM for Textual Data Mining," *Artificial Intelligence Rev.*, vol. 13, nos. 5–6., Dec. 1997, pp. 310–315.
8. G. Salton, "Developments in Automatic Text Retreival," *Science*, vol. 253, 30 Aug. 1991, pp. 974–979.
9. G. Miller, *WordNet: An Electronic Lexical Database*, C. Fellbaum, ed., MIT Press, 1998, preface.

10. S. Deerwester et al., "Indexing by Latent Semantic Analysis," *J. Am. Soc. for Information Science*, vol. 41, no. 6, June 1990, pp. 391–407.

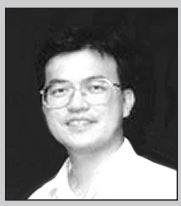
11. S. Kaski, "Dimensionality Reduction by Random Mapping: Fast Similarity Computation for Clustering," *Proc. Int'l Joint Conf. Neural Networks* (IJCNN 98), vol. 1, IEEE Press, 1998, pp. 413–418.

12. M.L. Pao, "Term and Citation Retrieval: A Field Study," *Information Processing & Management*, vol. 29, no. 1, Jan./Feb 1993, pp. 95–112.

13. H.D. White and K.W. McCain, "Visualizing a Discipline: An Author Co-citation Analysis of Information Science, 1972–1995," *J. Am. Soc. for Information Science*, vol. 49, no. 4, Apr. 1998, pp. 327–355.

The Authors

Y. He is a PhD student in Cambridge University's Speech, Vision, and Robotics Group. The work this article describes was carried out while she was a senior tutor at Nanyang Technological University's School of Computer Engineering. Her research interests include data mining and information retrieval. She received her BASc and MEng from Nanyang Technological University. Contact her at the Speech, Vision, and Robotics Group, Cambridge Univ. Eng. Dept., Trumpington St., Cambridge, CB2 1PZ; yh213@cam.ac.uk.



S.C. Hui is an associate professor in Nanyang Technological University's School of Computer Engineering. His research interests include data mining, Internet technology, and multimedia systems. He received his BSc in mathematics and DPhil in computer science from the University of Sussex. Contact him at the School of Computer Eng., Nanyang Technological Univ., Nanyang Ave., Singapore 639798; asschui@ntu.edu.sg

A.C.M. Fong is a lecturer at Massey University. The work this article describes was carried out while he was with Nanyang Technological University's School of Computer Engineering. His research interests include various aspects of Internet technology, data mining, information and coding theory, and signal processing. He received his BEng in electronic and electrical engineering and computer science and his MSc in EEE from Imperial College, London, and a PhD in EEE from the University of Auckland. He is a member of the IEEE and IEE and is a chartered engineer. Contact him at the Inst. of Information and Mathematical Sciences, Massey Univ., Albany Campus, Private Bag 102-904, North Shore Mail Ctr., Auckland, New Zealand; a.c.fong@massey.ac.nz.