Integrating Element and Term Semantics for Similarity-Based XML Document Clustering

Jianwu Yang[1], William K. Cheung[2], Xiaoou Chen[1]

Institute of Computer Sci. & Tech.[1]
Peking University
Beijing 100871, China
{yjw, cxo}@icst.pku.edu.cn

Department of Computer Science[2]
Hong Kong Baptist University
Kowloon Tong, Hong Kong
william@comp.hkbu.edu.hk

Abstract

Structured link vector model (SLVM) is a recently proposed document representation that takes into account both structural and semantic information for measuring XML document similarity. Its formulation includes an element similarity matrix for capturing the semantic similarity between XML elements – the structural components of XML documents. In this paper, instead of applying heuristics to define the similarity matrix, we proposed to learn the matrix using pair-wise similar training data in an iterative manner. In addition, we extended SLVM to SLVM-LSI by incorporating term semantics into SLVM using latent semantic indexing, with the element similarity related properties of the original SLVM preserved. For performance evaluation, we applied SLVM-LSI to similarity-based clustering of two XML datasets and the proposed SLVM-LSI was found to significantly outperform the conventional vector space model and the edit-distance based methods. The similarity matrix, obtained as a by-product via the learning, can provide higher-level knowledge about the semantic relationship between the XML elements.

1. Introduction

XML has widely been used as a mark-up language for describing different categories of semi-structured information. Examples include the W3C recommended ones, e.g., NewsML, MathML, SVG, as well as those privately used within enterprises. The rapid growth in XML adoption has led to a great need for techniques specialized in analyzing and managing semi-structured documents so as to support precise and efficient XML document retrieval.

To contrast with ordinary unstructured documents, XML documents carry additional information about their syntactic structure via the use of XML elements, each with a user-specified tag. Making the best use of the structural information as well as their hidden semantic relationships is crucial for the effectiveness of the corresponding document analysis. Under this agenda, the major issues become (1) how to represent XML documents with the aforementioned characteristics taken into consideration and (2) how to compute a corresponding XML document similarity. In the literature, unordered labeled trees[1] were commonly used for modeling XML documents that are free of reference tags. The structural similarity between a pair of XML documents can thus be computed based on different tree edit distances [1,2] which differ from each others in terms of the set of allowed edit operators and their support for repetitive and optional XML elements. It has been proved in [3] that computing the edit distance for unordered labeled trees is NP-complete, and yet the distance is not optimal in any sense related to the elements' semantics. Other than trees, the structure of an XML document can be represented as a time series, as in [4], with each occurrence of a tag corresponding to an impulse. The degree of similarity between documents can then be computed by analyzing their Fourier transform coefficients. This approach does not take into account the order in which the elements appear and is adequate only when the XML documents are drastically different from each other, i.e., they have few tags in common.

Another promising approach for addressing the problem is using the kernel methods where the semantics between XML elements can be incorporated into the document similarity formulation in the form of a similarity matrix. In [5], Yang and Chen extended the conventional vector space model and proposed a structured link vector model (SLVM) for representing XML documents. SLVM takes into account the document structure, referencing links and element similarity for representing XML documents. The similarity between two particular elements was pre-set to be related to the path difference between the two elements as well as their depth in the corresponding document schema (specified in, say, DTD).

The contribution of this paper is four-fold. (1) We extended SLVM by automatically learning its XML element similarity matrix in an unsupervised (using

[1] A labeled unordered tree refers to a tree structure where each tree node is assigned a label and the order of the children of any parent node is not maintained.

unlabelled training data) as well as a semi-supervised (using pair-wise similar training data) manners. (2) While the basic SLVM assumes that document terms are independent of each other with no semantic relationship, we incorporated term semantics into SLVM using latent semantic indexing [6]. The enhanced model, called SLVM-LSI, still has all the nice properties of SLVM preserved and the corresponding similarity learning method can still be defined. (3) We applied the proposed approach to similarity-based document clustering and demonstrated its superior performance when compared with existing approaches based on two real-world datasets. SLVM-LSI was found to outperform significantly the basic SLVM, showing the effectiveness of considering term semantics in the XML element similarity learning. (4) We demonstrated that the element similarity matrix, obtained as a by-product, is a useful piece of discovered knowledge about the semantic relationships between the XML elements for subsequent conceptual analysis.

The rest of the paper is organized as follows. Section 2 describes SLVM and its extended version SLVM-LSI. Then, related work on document similarity metrics and the proposed similarity metric for SLVM/SLVM-LSI are presented. Section 3 describes the proposal iterative algorithm for learning the element similarity. Section 4 shows the experimental results and Section 5 concludes the paper with some future research directions.

2. Modeling XML Document Using SLVM and LSI

2.1. SLVM – An XML document representation

Vector Space Model (VSM) [7] has long been used to represent unstructured documents as document feature vectors containing term occurrence statistics. In particular, assume that there are n distinct terms for a given set of documents D. Let doc_x denote the x^{th} document and d_x denote the corresponding feature vector such that

$$d_x = [d_{x(1)}, d_{x(2)} \cdots\cdots, d_{x(n)}]^T$$
$$d_{x(i)} = TF(w_i, doc_x) \bullet IDF(w_i)$$

where $TF(w_i, doc_x)$ is the frequency of the term w_i in doc_x, $IDF(w_i) = log(|D|/DF(w_i))$ is the inverse document frequency of w_i for discounting the importance of the frequently appearing words, $|D|$ is the total number of the documents, and $DF(w_i)$ is the number of documents where the term w_i appears at least once.

Applying VSM directly to represent semi-structured documents, like those in XML, is obviously inadequate as the document structure is ignored. For example, it cannot differentiate a word in the title field and the same word in a semantically unrelated field, say, the author field. Structured Link Vector Model (SLVM) was proposed in [5] as an extended vector space model for representing XML documents. Intuitively speaking, SLVM represents an XML document as a collection of VSMs, each being specific for an XML element (specified by the `<element>` tag in DTD).[2] Mathematically speaking, an XML document doc_x is represented as a matrix $d_x \in R^{n \times m}$, given as

$$d_x = [d_{x(1)}, d_{x(2)}, \cdots\cdots, d_{x(m)}]^T$$

where m is the number of distinct XML elements, $d_{x(i)} \in R^n$ is the feature vector for the i^{th} XML element (e_i), $d_{x(i,j)}$ is the TFIDF feature specific to the term w_j and the element e_i, given as

$$d_{x(i,j)} = TF(w_j, doc_x.e_i) \cdot IDF(w_j)$$

and $TF(w_i, doc_x.e_j)$ is the frequency of the term w_i in the element e_j of doc_x. In order to discount the factor caused by different numbers of terms appearing in different elements, each $d_{x(i,j)}$ is normalized by $\sum_j d_{x(i,j)}$.

2.2. Integrating term semantics into SLVM

By taking the vector space approach for representing XML documents, SLVM inherits the limitation of VSM --- terms are assumed to be independent of each other. Lacking the capability to represent terms' semantic relationships could result in problematic cases caused by polysemies and synonyms.

Latent Semantic Indexing (LSI) [6] is a technique commonly used in information retrieval for overcoming the aforementioned problems caused by synonyms and polysemies. In particular, LSI projects a document from the original document feature space onto a corresponding "semantic" space via singular value decomposition (SVD) so that more robust semantic-based document similarity measure can be resulted.

Using LSI, the original term document matrix $D_{n \times |D|} = [d_1, d_2, \cdots d_{|D|}]$ is first decomposed into three matrices:

$$D_{n \times |D|} = U_{n \times n} \cdot S_{n \times |D|} \cdot V_{|D| \times |D|}^T$$

where U and V contain orthonormal columns and S is diagonal. By restricting the matrices U, V and S to their first $k < \min(m, n)$ columns, one obtains the matrix

$$\tilde{D}_{n \times |D|} = \tilde{U}_{n \times k} \cdot \tilde{S}_{k \times k} \cdot \tilde{V}_{k \times |D|}^T$$

where $\tilde{D}$ is the best square approximation of D by a matrix of rank k [6]. The newly defined term document matrix will contain document feature vectors with term semantics

[2] In the current version of SLVM, only the elements corresponding to the leaf nodes of the XML DOM tree are modeled.

(obtained based on term co-occurrence statistics) taken into consideration. To deal with novel documents not included in the term-document matrix D, one can project the novel document vector onto the "semantic space" of dimension k and measure distance directly in the semantic space. According to [6], a novel document d's projection can be computed as:

$$d_{LSI}^{T} = d^{T} \cdot U \cdot S^{-1}$$

where $U \cdot S^{-1}$ is the transformation for the projection. Another alternative is to use simply U and the corresponding pseudo document projection becomes

$$d_{LSI}^{T} = d^{T} \cdot U$$

which is equivalent to put $d_{LSI}^{T} = d^{T} \cdot U \cdot S^{0}$.

To apply LSI to SLVM, XML documents are first partitioned into segments based on the element tags. SVD is then applied to the segment-term matrix. Thus, an XML document will eventually be represented as a matrix $d_x \in R^{k \times m}$, with each column being the projection of the element-specific feature vector on the semantic space. The rationale is that each XML element instance should be a semantically self-contained unit. We call this version of SLVM as SLVM-LSI in the subsequent sections.

2.3. Similarity measures

In VSM, the similarity between two documents doc_x and doc_y is commonly defined using the cosine similarity:

$$sim(doc_x, doc_y) = d_x * d_y = \sum_{i=1}^{n} d_{x(i)} \cdot d_{y(i)} \quad (1)$$

where n is the total number of terms, the sign * indicates the vector dot product, and d_x and d_y are the normalized document feature vectors of doc_x and doc_y so that $|d_x|^2=|d_y|^2=1$. Using LSI, the similarity between two documents can be computed similarly, but using the projections of d_x and d_y instead.

For SLVM, with the objective to model semantic relationships between XML elements, the corresponding document similarity can be defined with an element similarity matrix introduced, given as

$$sim(doc_x, doc_y) = \sum_{i=1}^{n} d_{x(i)}^{T} \bullet M_e \bullet d_{y(i)} \quad (2)$$

where M_e is an mxm matrix that captures both the similarity between a pair of XML element tags as well as the contribution of the pair to the overall document similarity (*i.e.*, the diagonal elements of M_e are not necessarily equal to one). An entry in M_e being small means that the two XML elements should be semantically unrelated and same words appearing in the two elements should not contribute to the overall similarity. One can easily extend the aforementioned formulation to SLVM-LSI, with the exception that the per-element semantic projections of d_x and d_y are used instead. To obtain M_e, we adopt the similarity learning approach.

3. Learning XML Document Similarity

3.1. Related works on similarity learning

In the literature, there exist a number of algorithms proposed for learning document similarity (c.f. distance) metrics. The metric learning problem was posed as a convex optimization problem in [8,9]. The Mahalanobis distance (derived based on training data's covariance matrix) was adopted in [10] to define the similarity. In addition, an iterative algorithm was proposed in [11] for deriving document similarity in some non-orthogonal feature space.

3.2. Learning SLVM-LSI

Inspired by [11], we derive an element similarity learning algorithm based on the notion that different element tags have different contributions to the overall XML document similarity and the contribution should depend on the elements' semantics rather than the relative position of the elements in the XML documents as used in [5]. For example, in Figure 1, it is obvious that the contribution of the "confYear" tag should be much less than that of the "authors" tag to the overall similarity of documents. Instead, words appearing in both document A's "title" tag and document B's "abstract" tag should be considered to be highly correlated. This intuitive requirement is by no means related to merely the XML data structure and thus cannot be fully satisfied using the edit distance measure. The adopted SLVM-LSI takes into account both term semantics as well as elements' semantic relationship in document similarity.

Based on SLVM-LSI described in Section 2.2 with a set of $|D|$ XML documents, each containing m distinct XML elements and represented as a k-dimensional vector resulting from SVD, we can define $B_{(i)} \in R^{m \times |D|}$ as a matrix with its x^{th} column corresponding to $d_x = [d_{x(1)}, d_{x(2)}, \cdots\cdots, d_{x(m)}]^T$ of the x^{th} XML document and denote $\vec{B} = [B_{(1)}, B_{(2)}, \cdots\cdots, B_{(k)}]^T$. To recall, $d_x = [d_{x(1)}, d_{x(2)}, \cdots\cdots, d_{x(m)}]^T$ is a feature vector (TF-IDF in our case) corresponding to the i^{th} dimension of the "semantic space" of all the elements. Thus, $B_{(i)}$ can be interpreted as the matrix storing statistics of the i^{th} concept in each XML element among all the documents. The similarity matrix for the whole set of documents can then be defined as

$$S_d = (\sum_{i=1}^{k} B_{(i)}^{T} \bullet M_e \bullet B_{(i)}) / k \cdot \quad (3)$$

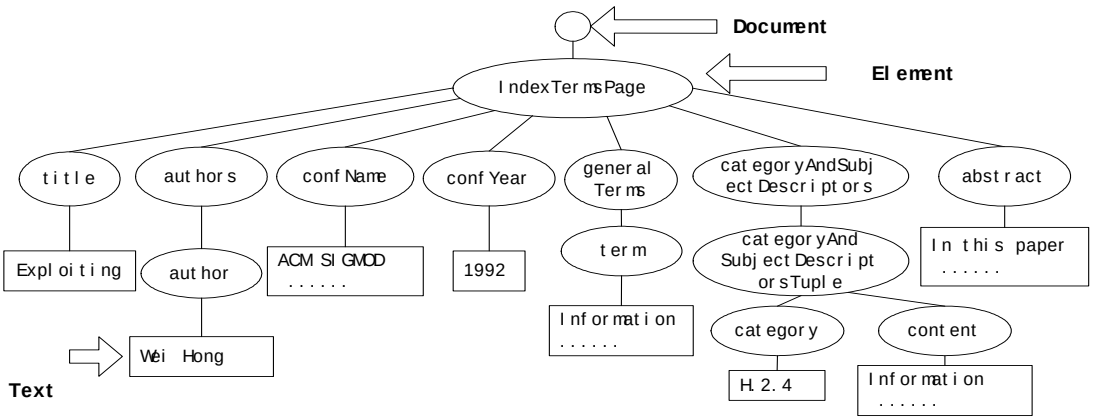

Figure 1. The DOM tree of an XML document extracted from the ACMSIGMOD dataset.

Note that M_e is not a matrix with its diagonal elements all equal to 1. It captures not only cross-element similarity but also their individual contributions to the overall document similarity at the same time.

Based on Eq.(3), the remaining task is how to estimate the similarity matrix *Me* based on a set of unlabelled XML documents as the training data (*unsupervised* learning). With the notion that element similarity should affect document similarity and vice versa, we propose an iterative algorithm learning the element similarity matrix M_e in SLVM-LSI, given as

$$S_d = (\sum_{i=1}^{k} B_{(i)}{}^T \bullet M_e \bullet B_{(i)}) / k \quad (4)$$

$$M_e = (\sum_{i=1}^{k} B_{(i)} \bullet S_d \bullet B_{(i)}{}^T) / k \quad . \quad (5)$$

Note that we here assume that all the document similarity measurements are normalized to one. In other words, all the entries' values of matrix S_d should be between zero and one. Two totally different documents should have a similarity value equal zero and two identical documents should have a similarity value of one; otherwise, the similarity should take some value in-between. To satisfy this constraint and at the same time to guarantee the convergence of the iterative equations, we modify Eq.(4) and Eq.(5) by normalizing them using a set of parameters $\lambda_i = 0.9 / \max(|B_i|_1, |B_i|_\infty)$ and estimate S_d and M_e iteratively, given as

$$S_d{}^{g+1} = (\sum_{i=1}^{k} \lambda_i \cdot B_i{}^T \bullet M_e{}^g \bullet B_i) / k \quad (6)$$

$$M_e{}^{g+1} = (\sum_{i=1}^{k} \lambda_i \cdot B_i \bullet S_d{}^g \bullet B_i{}^T) / k \quad (7)$$

A similar normalization step has been adopted in [11] with a convergence proof. We extended that to our case for SLVM-LSI. While our proposed iterative equations were found to converge for all the cases we have tested, we are still working on the corresponding convergence proof. Also, note that the pair of iterative equations Eq.(6) and Eq.(7) has an obvious trivial solution of having both matrices with all zero elements. Thus, an additional constraint for getting a non-trivial solution is required to force the diagonal elements of S_d (i.e., the similarity of identical documents) to take the value of one. Up to this step, the iterative algorithm is still essentially unsupervised, as the information about how the documents should be grouped is not used.

For *semi-supervised* learning, one possibility is to collect document similar pairs (i.e., a set of pair-wise similar document pairs but without their class labels). Then, instead of forcing only the diagonal elements of S_d to one, we force also the value of $S_{d(i,j)}$ corresponding to those similar document pairs to one throughout the iterations. The proposed iterative algorithm is summarized in Figure 2.

Our preliminary experiments showed that M_e normally converges within 4 to 5 iterations. Note that we do not force the diagonal elements of M_e to one as that for S_d. The objective is to allow each element to have different contribution to the overall similarity measure.

Step 1) $\lambda_i = 0.9 / \max(|B_i|_1, |B_i|_\infty)$

$$S^0_{d(i,j)} = \begin{cases} 1 \text{ if } \text{doc}_i \text{ and } \text{doc}_j \text{ are in same class} \\ 0 \text{ else} \end{cases}$$

Step 2) $M_e{}^{g+1} = (\sum_{i=1}^{k} \lambda_i \cdot B_i \bullet S_d{}^g \bullet B_i{}^T) / k$

Step 3) $S_d{}^{g+1} = (\sum_{i=1}^{k} \lambda_i \cdot B_i{}^T \bullet M_e{}^{g+1} \bullet B_i) / k$

Step 4)

$$S^{g+1}_{d\ (i,j)} = \begin{cases} 1 \text{ if } \text{doc}_i \text{ and } \text{doc}_j \text{ are in same class} \\ S^{g+1}_{d\ (i,j)} \text{ else} \end{cases}$$

Step 5) Repeat Step 2 to Step 4 until M_e converges.

Figure 2. The iterative algorithm for learning the element similarity matrix.

4. Experiments

4.1. Datasets and preprocessing

We applied the proposed similarity learning method to similarity-based clustering for performance evaluation. Two datasets were used, namely ACMSIGMOD Record [12] and "Chinese Encyclopedia Database" (CEDB) [13].

ACMSIGMOD Record is composed of around one thousand documents obtained from the past issues of SIGMOID Record. In order to compare the performance between unsupervised and semi-supervised similarity learning, we extracted only the data with category information. Besides, we remove the categories which contain less than ten documents for fair evaluation of subsequent clustering performance. Thus, the final subset of data used in our experiments contains 461 documents.

"Chinese Encyclopedia Database" (CEDB) is a digital archive, which contains millions of XML documents from a Chinese encyclopedia with 74 categories. In order to test the sensitivity of the proposed algorithm on datasets with different sizes, we prepared a number of data subsets as shown in Table 2 for our experiments.

After preparing the datasets for our experiments, we also noted that both ACMSIGMOD Record and CEDB contain elements with category information. In order to fully explore the effectiveness of semantic learning capability, we deliberately remove those tags from the datasets for all the experiments reported in this paper. Also, all the documents are preprocessed by (1) converting all the words to lower case (for ACMSIGMOD), (2) going through the Porter stemming algorithm (for ACMSIGMOD), and (3) removing stop-words (for both ACMSIGMOD and CEDB).

Table 1. Data subsets used in our experiments.

Datasets	Sources	Num. of categories	Total num. of documents
ACM-8	ACMSIGMOD	8	96
ACM-12	ACMSIGMOD	12	461
CEDB-8	CEDB	8	320
CEDB-16	CEDB	16	640
CEDB-32	CEDB	32	960

4.2. Experiment setup

To compare the performance of the proposed SLVM and SLVM-LSI with that of other related works, we have implemented the traditional VSM as well as a version of SLVM but with the element similarity estimated using the edit distance approach. In order to fairly compare the performance of SLVM-LSI with others, we also performed experiments with LSI applied to the VSM and the edit distance approaches. Both the unsupervised and semi-supervised versions of the similarity learning were evaluated. Cross-validation was adopted to avoid bias in training data sampling. All the algorithms were implemented in C++ and all experiments were run on a PC with a 2.66GHz Intel CPU and 512M RAM.

4.3. Performance evaluation for similarity-based clustering

A similarity matrix that can accurately describe the semantic relationship between XML elements (and thus XML documents) should benefit all the existing similarity-based clustering algorithms. Without loss of generality, the Agglomerative Hierarchical Clustering (AHC) algorithm [15] was chosen in this paper as the clustering algorithm. AHC starts from one data point per cluster, computes the similarity between all pairs of clusters at each stage, and then merges the most similar pair. The process repeats until all the documents are finally merged as one cluster and then a hierarchical clustering result will be generated. We used the following measure as the similarity between a pair of clusters C_i and C_j:

$$SIM(C_i, C_j) = \frac{\sum_{k=1}^{|C_i|}\sum_{l=1}^{|C_j|} sim(d_k^i, d_l^j)}{|C_i| \cdot |C_j|}$$

where $|C_i|$ denotes the number of documents in the i^{th} cluster C_i and d_k^i denotes the k^{th} document in C_i.

Also, among the different quality measures for clustering, we used one of the most common ones F-measure [16] which combines the precision and recall rates as an overall performance measure. The measure first assumes that a cluster is the result of a query and a category is the desired set of the documents for the query. Then, the recall and precision rates, and thus the F-measure, for that cluster-category pair are computed. More specifically, for the j^{th} cluster and i^{th} category,

$$recall(i, j) = N_{ij} / N_i$$

$$precision(i, j) = N_{ij} / N_j$$

where N_{ij} is the number of the i^{th} category documents falling into the j^{th} cluster, N_j is the number of documents in the j^{th} cluster, and N_i is the number of documents in the i^{th} category. The F-measure associated to the j^{th} cluster and the i^{th} category is given as

$$F(i, j) = \frac{2 * recall(i, j) * precision(i, j)}{precision(i, j) + recall(i, j)}$$

The quality of the j^{th} cluster should then be measured as the maximum among the F-scores measured between it and all the possible categories. Thus, the overall weighted F-measure can then be computed, given as

$$F = \sum_i \frac{N_i}{N} \max_j \{F(i, j)\}$$

where N is the total number of documents.

4.4. Results and discussion

4.4.1. Performance comparison between learning-based and heuristics-based. According to Figure 3 and 4, the clustering performance based on the proposed SLVM and SLVM-LSI with similarity learning adopted was found to be significantly better than that based on the others, especially when the training data were sufficient. In particular, the conventional VSM was found to be the worst for all the datasets, with the F-score ranging from 38-52%. SLVM adopting the edit distance for the element similarity resulted in performance improvement for all the datasets by up to 13% when compared with VSM. SLVM adopting the proposed learning approach outperformed that using the edit distance approach by 10-20%.

4.4.2. Effectiveness of adding term semantics. By adopting LSI as explained in Section 2.2 (with the best semantic space dimension selected using the training data), significant performance improvement was observed for all the approaches (Figure 3 and 4). The projections $d_{LSI}{}^{T} = d^{T} \cdot U \cdot S^{-1}$ and $d_{LSI}{}^{T} = d^{T} \cdot U \cdot S^{0}$ have been tested and the former one was found to outperform the latter. Based on $d_{LSI}{}^{T} = d^{T} \cdot U \cdot S^{-1}$, the improvement was found to be 3-20% for VSM, 5-40% for the editing distance approach, and 3-30% for SLVM (i.e., SLVM-LSI).

4.4.3. Comparison between unsupervised and semi-supervised similarity learning. The results reported above were based on unsupervised training. For comparison between the semi-supervised and unsupervised learning of the similarity matrix, the former one was unexpectedly found to be similar to that of the latter one on average (as shown in Figure 5). We are currently investigating the reason and believe that the use of the pair-wise similar information has rooms for improvement.

4.4.4. High-level knowledge from the similarity matrix. The estimated similarity matrix obtained via learning can in fact provide higher-level knowledge about the conceptual relationships among the elements. The similarity matrix learned using the ACMSIGMOD Record dataset was shown in Figure 6, with the element indices shown in Table 2. The element pairs: {term, content}, {abstract, term}, {term, content}, {title, term}, {title, content} were found to be the most similar ones, whose validity can easily be validated based on the nature of those elements. In addition, the weightings of different elements contributing to the overall similarity measure were found to be relevant. For example, unimportant elements include conference year and initial page while important ones include abstract and categories' content.

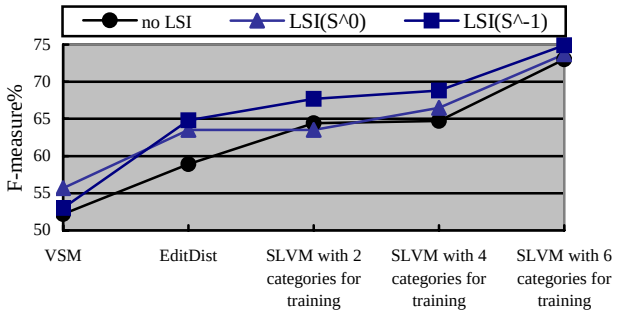

Figure 3. Performance using ACM-12 dataset.

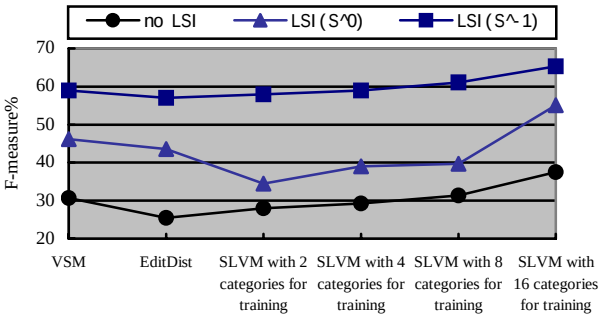

Figure 4. Performance using CEDB-32 dataset.

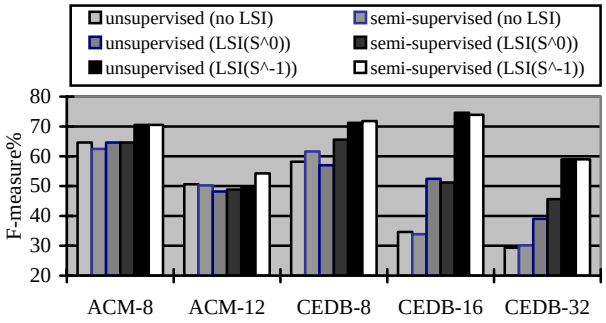

Figure 5. Performance comparison between unsupervised and semi-supervised similarity learning for SLVM-LSI (with 4 categories for training).

Table 2. Index to elements of the ACMRecord dataset

No.	Element Name	No.	Element Name
1	IndexTermsPage	11	fullTextURL
2	**title**	**12**	**Size**
3	authors	**13**	**abstract**
4	**author**	14	generalTerms
5	confName	**15**	**Term**
6	confYear	16	categoryAndSubjectDescriptors
7	volume	17	categoryAndSubjectDescriptorsTuple
8	number	**18**	**category**
9	InitPage	**19**	**content**
10	EndPage		

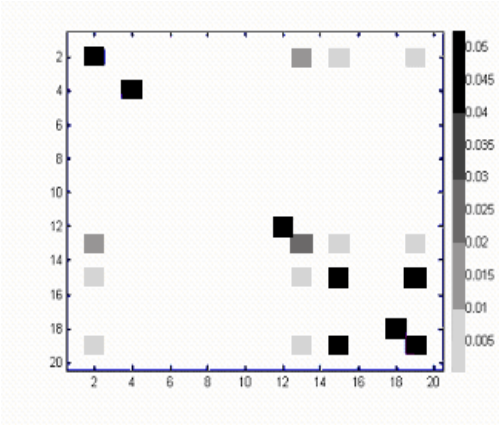

Figure 6. The similarity matrix learned using the ACMRecord dataset. The x and y axes show the element indices where their mappings to the exact elements are described in Table 2.

5. Conclusion and future work

Measuring XML document similarity is a fundamental issue for XML document retrieval. In this paper, we proposed the model called SLVM-LSI for representing XML documents so that the term semantics, element similarity, as well as elements' relative importance for a given set of documents can all be taken in account. Also, we formulated an iterative estimation procedure for automatically learning an element similarity matrix associated to SLVM-LSI. Both semi-supervised and unsupervised versions for the similarity learning have rigorously been tested for their performance when applied to clustering. The proposed similarity learning approach was found to outperform the conventional vector space model and the commonly adopted edit-distance approach. For future work, we are investigating how the elements not at the leaf nodes of an XML document should further be modeled to enrich the structural representation of SLVM-LSI. In addition, we are also interested to see how the similarity matrix obtained via learning can be related to ontology generation in general.

6. Acknowledgement

This work was partially supported by RGC Central Allocation Group Research Grant (HKBU 2/03/C).

7. References

[1] Z.P. Zhang, R. Li, S.L. Cao, and Y.Y. Zhu, "Similarity Metric for XML Documents", *Proceedings of the 2003 Workshop on Knowledge and Experience Management (FGWM 2003)*, Karlsruhe, Oct. 2003.

[2] A. Nierman and H. V. Jagadish, "Evaluating Structural Similarity in XML Documents", *Proceedings of the Int. Workshop on the Web and Databases (WebDB)*, Madison, WI, Jun. 2002.

[3] K. Zhang, R. Statman, and D. Shasha, "On the editing distance between unordered labeled trees", *Information Processing Letters*, 42(3):133--139, 1992.

[4] S. Flesca, G. Manco, E. Masciari, L. Pontieri, and A. Pugliese, "Detecting structural similarities between xml documents", *Proceedings of the International Workshop on the Web and Databases (WebDB)*, Madison, WI, Jun. 2002.

[5] J.W. Yang, and X.O. Chen, "A semi-structured document model for text mining", *Journal of Computer Science and Technology*, 17(5): 603-610, 2002.

[6] S. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman, "Indexing by latent semantic analysis", *Journal of the Society for Information Science*, 41(6), 391-407, 1990

[7] G. Salton, and M. J. McGill, *Introduction to Modern information Retrieval*. McGraw-Hill, 1983.

[8] M. Schultz, and T. Joachims, "Learning a Distance Metric from Relative Comparison", *Proceedings of the Neural Information Processing Systems (NIPS)*, Whistler, B.C., 2003.

[9] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart Russell, "Distance Metric Learning, with Application to Clustering with Side-Information", *Proceedings of the Neural Information Processing Systems (NIPS)*, Whistler, B.C., 2003.

[10] J. Kandola, J. Shawe-Taylor, and N. Cristianini, "Learning Semantic Similarity", *Proceedings of the Neural Information Processing Systems (NIPS)*, Canada, 2002.

[11] N. Liu, B.Y. Zhang, J. Yan, Q. Yang, S.C. Yan, Z. Chen, and W.Y. Ma, "Learning Similarity Measures in the Non-orthogonal Space", *Proceedings of the Thirteenth Conference on Information and Knowledge Management (CIKM 2004)*, Washington D.C., U.S.A., Nov. 2004.

[12] http://www.acm.org/sigs/sigmod/record/xml/XMLSigmodRecordMarch1999.zip

[13] http://www.ecph.com.cn/

[14] J.W. Yang, W.K. Cheung and X.O Chen, "Learning Element Kernel for XML Document Clustering," *Proceedings of IEEE International Conference on E-Technology, E-Commerce and E-Service*, Hong Kong, March, 2005

[15] P. Sneath, and R. R. Sokal, *Numerical Taxonomy - The Principles and Practice of Numerical Classification*, W. H. Freeman, San Francisco, 1973.

[16] B. Larsen and C. Aone, "Fast and Effective Text Mining Using Linear-time Document Clustering", *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Diego, California, August 1999.