Learning the Kernel Matrix for 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

The rapid growth of XML adoption has urged for the need of a proper representation for semi-structured documents, where the document structural information has to be taken into account so as to support more precise document analysis. In this paper, an XML document representation named "structured link vector model" is adopted, with a kernel matrix included for modeling the similarity between XML elements. Our formulation allows individual XML elements to have their own weighted contribution to the overall document similarity while at the same time allows the between-element similarity to be captured. An iterative algorithm is derived to learn the kernel matrix. For performance evaluation, the ACM SIGMOD Record dataset as well as the CEDB dataset have been tested. Our proposed method outperforms significantly the traditional vector space model and the edit-distance based methods. In addition, the kernel matrix obtained as a by-product provides knowledge about the conceptual 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 and thus plays an important role in supporting system interoperability. Examples include those W3C recommended ones, e.g., NewsML, MathML, SVG, as well as those used privately in companies. The rapid growth in XML adoption has led to a great need in semi-structured document management systems, where functionalities like retrieval, classification and clustering of XML documents are fundamentally important.

To contrast with the ordinary unstructured documents, XML documents carry additional information about their syntactic structure. Making best use of the structure information is crucial for the corresponding effectiveness of the document management systems. One of the underlying research issues is to determine the similarity between XML documents. In the literature, trees are commonly used for modeling XML documents without reference elements and the structural similarity between a pair of XML documents can then be defined as some edit distance between the corresponding labeled trees [1,2]. Various tree distance algorithms differ from each other according to the set of edit operations allowed and whether repetitive and optional fields being handled or not. Zhang, Statman and Shasha proved in [3] that computing the edit distance for unordered labeled trees is NP-complete, and yet not optimized in any sense related to the elements' semantics. A rather different approach has recently been proposed in [4], where the structure of an XML document is represented as a time series with each occurrence of a tag corresponding to an impulse. The degree of similarity among documents is computed by analyzing the corresponding Fourier transform coefficients. This approach does not take into account the order in which the element tags 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 kernel methods to incorporate XML element similarity into the formulation of the document similarity for capturing the underlying semantics between the elements. In [5], Yang and Chen extended the vector space model for document representation and proposed a structured link vector model (SLVM) for representing XML documents. The model takes into account the document structure, referencing link and element similarity for representing XML documents. The similarity between an element pair is pre-set in [5] to be related to the path difference between the two elements as well as the depth of the elements in the document structure. In this paper, the optimality and adaptability of such a similarity model is addressed. In particular, we extend the SLVM model by treating the element similarity matrix as a kernel and formulate the corresponding kernel-learning problem so that the element similarity can be adapted based on a set of unlabelled training data with pairwise similarity information. We studied both semi-supervised (making use of the pairwise similarity information) and unsupervised kernel learning for clustering XML documents and compared the results with other existing approaches using one benchmarking and one real-world datasets. The results obtained demonstrate that the proposed kernel learning methodology can greatly improve the clustering performance.

The rest of the paper is organized as follows. Section 2 provides a brief review on some related work in the area of document similarity metric. The proposed kernel-learning method for modeling XML documents is described in Section 3. Section 4 shows the experimental results and Section 5 concludes the paper with some future research directions.

2. Background

2.1. Document Representation

Vector Space Model (VSM) [6] has long been used to represent documents as a set of terms where a document vector space spanned by the features of the n distinct terms is defined. Let doc_x denote the x^{th} document with the corresponding feature vector d_x 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 the document doc_x, $IDF(w_i)$ is the inverse document frequency of w_i based on a document collection D, $IDF(w_i) = log(|D|/DF(w_i))$ for discounting the importance of the frequently appearing words, $|D|$ is the total number of the documents in the collection and $DF(w_i)$ is the number of documents where the term w_i appears at least once.

VSM is used to be applied for representing unstructured text documents and does not consider at all the document structure. For example, it does not make the difference between a word in title and the same word in the main text. Thus, directly applying it to represent XML documents is inadequate. Structured Link Vector Model (SLVM) was proposed by Yang and Chen [5], which can be considered as an extended vector space model for representing XML documents. In the model of SLVM, each document, doc_x, is represented as a matrix $d_x \in R^{n \times m}$, given as

$$d_x \;=\; < d_{x(1)}, d_{x(2)} \cdots\cdots, d_{x(n)} >^T$$

$$d_{x(i)} \;=< d_{x(i,1)}, d_{x(i,2)} \cdots\cdots d_{x(i,m)} >$$

where m is the number of elements, $d_{x(i)} \in R^m$ is a feature vector related to the term w_i for all the elements, $d_{x(i,j)}$ is a feature related to the term w_i and specific to the element e_j, given as

$$d_{x(i,j)} \;= TF(w_i,doc_x.e_j) \cdot IDF(w_i)$$

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

One can interpret SLVM as extending VSM by keeping the term statistics for each element instead of the whole document.

2.2. Similarity Measure

Based on VSM, the similarity between two documents doc_x and doc_y is commonly defined as:

$$\begin{aligned} sim(doc_x, doc_y) &= \cos(< d_x, d_y >) \\ &= d_x * d_y = \sum_{i=1}^{n} d_{x(i)} \cdot d_{y(i)} \end{aligned} \qquad (1)$$

where "*" 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 $|d_x|_2=1$.

For SLVM, the document similarity can be defined similarly with a kernel matrix introduced, given as

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

where M_e is an $m*m$ kernel matrix which captures both the similarity between a pair of XML elements as well as the contribution of the pair to the overall document similarity. 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 and vice versa. To determine the value of M_e, we adopt the kernel-learning approach in this paper.

2.3. Similarity Learning

Recently, there have been some algorithms proposed in the literature for learning similarity (c.f. distance) metrics. Among the different approaches, [7] and [8] posed the metric learning as a convex optimization problem. Some other approaches (e.g. [9]) used the Mahalanobis distance (with the use of covariance matrix) to describe the similarity. In [10], an iterative similarity learning approach was proposed to measure the similarity between objects defined in some non-orthogonal feature space. They assume the existence of a dual relationship between the object similarity and the feature similarity in their algorithm.

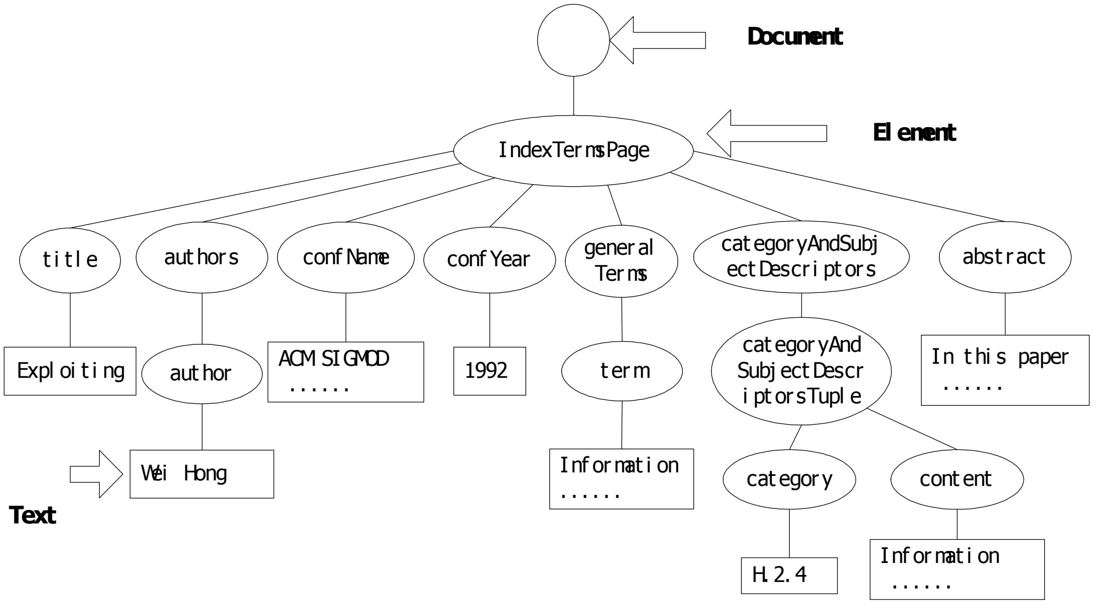

Fig. 1 The DOM tree of an XML document extracted from ACMSIGMOD dataset.

3. Learning the Kernel Matrix for XML Documents

In this section, we present the problem formulation which is based on kernel methods and provide the details about the kernel learning algorithm.

3.1. Problem Description

As in [10], we derive our algorithm based on the notion that different elements have different contributions to the overall XML document similarity and the contribution is dependent on the elements' semantic rather than the relative position of the elements in the XML documents. For example, in Figure 1, it is obvious to understand that the contribution of the element "confYear" should be much less than that of the element "authors" to the overall similarity of documents. Instead, words appearing in both document A's "title" element and document B's "abstract" element should be considered to be more relevant. However, this intuitive requirement is by no mean related to merely the XML data structure and thus cannot be satisfied using the edit distance measure. The adopted SLVM can take into account not only the terms in the documents, but also the elements they belong to. The formulation is flexible enough to represent the contribution of different elements to the overall document similarity and at the same time can capture similarity between elements.

Based on the SLVM described in Section 2.1, we denote a set of XML documents as $\vec{B} =< B_{(1)}, B_{(2)}, \cdots\cdots, B_{(n)} >^T$ where $B_{(i)} \in R^{m\times p}$ is a matrix with its k^{th} column corresponding to $d_{k(i)}$ of the k^{th} document and p is the number of documents. To recall, $d_{k(i)}$ is a feature vector (TF-IDF in our case) related to the i^{th} term for all the elements. The similarity matrix of the document set *{doc$_x$}* can then be defined as

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

Note that M_e is not restricted to have diagonal elements all equal to 1, implying that it is not a similarity matrix and thus we call it a kernel matrix. The matrix captures not only the elements' 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 kernel matrix M_e based on a set of XML documents.

3.2. Learning the Kernel Matrix

In this paper, we extend the iterative algorithm proposed by Ning Liu *et al.* in [10] for clustering unstructured documents to semi-structured documents.

With the notion that term similarity should be affecting document similarity and vice versa, we propose a similar iterative algorithm for learning the kernel matrix M_e in the SLVM model, given as

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

$$M_e = (\sum_{i=1}^{n} B_{(i)} \bullet S_d \bullet B_{(i)}{}^T)/n \cdot \qquad (5)$$

Note that we here assume that all the similarity measurements are normalized. 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 be between zero and one. In order to satisfy this constraint, 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

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

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

Note the iterative equations Eq.(6) and Eq.(7) have 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 yet used.

In order to incorporate supervised learning, one possibility is to collect document pairs that are known to be similar. Then, instead of only forcing the diagonal elements of S_d to one, we can 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.

```
Step 1) $\lambda_i = 0.9/\max(|B_i|_1, |B_i|_\infty)$
        $S^0_{d(i,j)} = \begin{cases} 1 \text{ if doc}_i \text{ and doc}_j \text{ are in same class□} \\ 0 \text{ else} \end{cases}$
Step 2) $M_e^{k+1} = (\sum_{i=1}^{n} \lambda_i \cdot B_i \bullet S_d^k \bullet B_i^T)/n$
Step 3) $S_d^{k+1} = (\sum_{i=1}^{n} \lambda_i \cdot B_i^T \bullet M_e^{k+1} \bullet B_i)/n$
Step 4)
        $S^{k+1}_{d\ (i,j)} = \begin{cases} 1 \text{ if doc}_i \text{ and doc}_j \text{ are in same class□} \\ \mathrm{S}^{k+1}_{\mathrm{d}\ (i,j)} \text{ else} \end{cases}$
Step 5) Repeat Step 2 to Step 4 until $M_e$ converges.
```

Fig. 2 The iterative algorithm for learning the kernel matrix.

Our preliminary experiments show that M_e normally converges within 5 iterations.

4. Experiments

4.1. Datasets and Experiment Design

In our experiment, we first used the benchmarking dataset --- ACMSIGMOD Record [11], which is composed of hundreds of documents of issues of SIGMOID Record. In addition, we also tested the proposed algorithm on part of a real world dataset "Chinese Encyclopedia Database" (CEDB) [12], which is one of the earliest large-scale national projects in China adopting XML for representing documents. The dataset 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 extracted a number of data subsets as shown in Table 1 for our experiments. Note that for all the datasets, the number of documents per class is identical.

Table 1 Data subsets used in our experiments.

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

Before running the experiments, all 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). To compare the performance of the proposed method 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. For our proposed supervised and unsupervised versions of the kernel learning approach, we have also tested their performance using training sets of different sizes. Cross-validation is used for conducting the experiments 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.

4.2. Evaluation of Clustering Performance

Among the existing similarity-based clustering algorithms, such as k-means, CLARANS, AHC, we choose the Agglomerative Hierarchical Clustering (AHC) algorithm [13] in this paper and more extensive empirical evaluation for the others will be conducted in the future. AHC computes the similarity between all pairs of clusters at each stage and merges the most similar pair. The process repeats until all the documents are merged as a cluster and then a hierarchical clustering result will be generated. We use 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 $|C_i|$ represents the number of documents in the i^{th} cluster C_i and d_k^i represents the k^{th} document in C_i. Conventional AHC possesses the possibility of considering each isolated data point as a cluster. In our implementation, clusters of isolated points are merged with their corresponding nearest clusters.

Also, among the different quality measures for clustering, we use one of the most common ones F-measure [14] which combines the precision and recall rates as an overall performance measure. The measure first assumes that each cluster is the result of a query and each class is the desired set of the documents for the query. Then, the recall and precision rates of each cluster for each given class are computed. More specifically, for the j^{th} cluster and i^{th} class,

$$recall(i, j) = n_{ij} / n_i$$

$$precision(i, j) = n_{ij} / n_j$$

where n_{ij} is the number of items of the i^{th} class falling into the j^{th} cluster, n_j is the number of items in the j^{th} cluster and n_i is the number of items in the i^{th} class. The F-measure associated to the j^{th} cluster and the i^{th} class is then given by

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

and 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 number of the documents. In other words, we only consider the cluster with the largest value of F-measure for each class when computing the overall F-measure.

4.3. Results and Discussion

According to the experimental results reported in Table 2-6, the clustering performance of the proposed kernel-learning method is significantly better than that based on the others by 21-60% at maximum for each dataset. In particular, the conventional vector space model performs the worst for both ACMSIGMOD and CEDB. The adoption of structure information using the edit distance results in significant improvement in performance. The proposed kernel learning approach again outperforms the edit distance approach significantly.

For comparison between the supervised and unsupervised learning of the kernel matrix, the supervised version outperforms the unsupervised one significantly for CEDB but not ACMSIGMOD. We are currently investigating the reason behind and believe that the use of the supervised information still has rooms for improvement.

5. Conclusion and Future Work

Measuring the similarity of XML document is a fundamental issue for XML document management. Based on the use of SLVM for representing XML documents, we formulated an iterative estimation procedure for learning a kernel matrix which captures both the element similarity and the elements' relative importance. Both semi-supervised and unsupervised versions have rigorously been studied and tested for their clustering performance using two datasets ACMSIGMOD Record and Chinese Encyclopedia Database. The proposed kernel-based learning approach is found to outperform the conventional vector space model and the commonly adopted edit-distance approach significantly.

As a by-product, it is interested to note that the estimated kernel matrix itself provides the knowledge about the semantic relationship among the elements. For example, if a word appears in different elements and is found to be not similar as revealed by the kernel matrix, this may imply that there exist more than one senses for the word (polysemy). We are currently investigating how this can be related to ontology generation in general.

Acknowledgement

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

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.

Table 2: F-measure (%) computed for different methods based on ACM-8.

VSM	Edit Dist	Unsupervised (# classes for training)		Supervised (# classes for training)	
		2	4	2	4
39.9	52.7	53.6±2.1	**64.6±7.3**	54.1±1.9	62.5±9.2

Table 3: F-measure (%) computed for different methods based on ACM-16.

VSM	Edit Dist	Unsupervised (# classes for training)			Supervised (# classes for training)		
		2	4	8	2	4	8
52.2	58.9	64.4±4.7	64.7±8.4	**73.0±9.6**	63.7±3.5	63.4±5.2	71.7±13.7

Table 4: F-measure (%) computed for different methods based on CEDB-8.

VSM	Edit Dist	Unsupervised (# classes for training)		Supervised (# classes for training)	
		2	4	2	4
40.0	89.0	90.8±6.6	90.4±6.2	99.6 ±0.3	**100±0.0**

Table 5: F-measure (%) computed for different methods based on CEDB-16.

VSM	Edit Dist	Unsupervised (# classes for training)			Supervised (# classes for training)		
		2	4	8	2	4	8
37.3	93.3	94.1±1.4	94.5±2.3	94.7±4.2	96.2±3.1	**97.0±2.6**	95.7±4.4

Table 6: F-measure (%) computed for different methods based on CEDB-32.

VSM	Edit Dist	Unsupervised (# classes for training)				Supervised (# classes for training)			
		2	4	8	16	2	4	8	16
41.5	85.1	91.8 ±1.6	91.4 ±1.2	90.6 ±1.4	92.1 ±0.5	96.5 ±0.8	96.3 ±1.2	95.4 ±2.3	**96.6 ±3.3**

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

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

[8] 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.

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

[10] 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.

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

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

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

[14] 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.