A Hybrid Method for XML Clustering by Structure and Content

Yong Piao and Xiu-kun Wang
School of Electronic and Information Engineering, Dalian University of Technology, Dalian, China
Email: {piaoy, jsjwxk}@dlut.edu.cn

Abstract—An effective XML cluster method called neighbor center clustering algorithm (NCC) is presented in this paper, whose similarity is obtained through both structural and content information contained in XML files. Structural similarity is firstly measured by frequency-path model and its similarity calculation algorithm with position and frequency weight by longest common subsequence is introduced. In order to improve the performance and precision, the frequency-path model is further extended by considering the structure and content information simultaneously. Experiments show that the NCC embed with hybrid similarity calculation method can obtain high purity and F-measure value and is effective and applicable for clustering XML with both homogenous and heterogeneous structures.

Index Terms—neighbor center clustering, position and frequency weight, longest common subsequence, hybrid similarity calculation

I. INTRODUCTION

XML (eXtensible Markup Language), as a common data representation and exchange format on the Internet, contains a rich entailment of information. Also, data mining on XML has become an important part in text mining research, in which large-scale text clustering is one of the effective solutions for massive texts. An efficient and fast XML clustering mechanism, which can provide better data for decision support, will greatly shorten the information retrieval time, improve the efficiency of data query and help find out potential value of information. The most important feature of XML document, which is different from other textural ones, is its structural character. For this reason, we believe that the structure of XML should also play important role in XML clustering.

Considering the structural character of XML documents, many traditional text clustering methods are not suitable for XML. Currently, partitioning and hierarchical methods are most widely used in XML clustering [1-3], but the effect of these two traditional methods in dealing with irregular non-spherical document clustering is not so satisfactory, and besides, they are not well in distinguishing noise or isolated points effectively. In terms of computational complexity, the searching time of traditional methods for cluster centers increase rapidly, this is an obstacle to get better performance in XML clustering. In addition, traditional partitioning methods represented by K-Means and K-Medoids have to be specified the clusters number K in advance. Due to these reasons, a neighbor center clustering algorithm with similarity (NCC) is proposed in this situation. It is not only simple, but can find non-spherical structure documents and distinguish noise or isolated point effectively as well.

Similarity among documents is the key issue in the field of document clustering. So far, the methods proposed for this purpose can be roughly classified into three types, namely by the graph matching, by the edit distance or by the tree path model. Reference [4] describes an XML document with a directed graph and calculates similarity between XML documents by graph matching in order to cluster XML with similar structure. But the result is not satisfactory due to its low accuracy. Reference [5] improves [4], in which its method leads to some limitations to clustering results without considering the order relationships between edges in the discussion of equal direct edges. Reference [6] and [7] introduce a concept of edit distance. Reference [6] calculates the similarity with graph matching algorithm by describing XML as a directed graph, while [7] uses tree editing distance to calculate the similarity, so do [1,8,9]. However, it is not suitable for document processing due to its high computational complexity. As the graph cannot express XML structure well, Reference [10] proposes a tree path model representation, which is simpler than the tree editing distance with a lower time complexity, but it uses a complete path matching method widely while handling the matching procedures, so do references [11,12] in frequent path mining and matching, including its improvement [13]. The complete path matching method is useful in XML clustering in tree path models because XML structure information ignored by the complete path matching has little impact on clustering XML which have the same DTD, but it is not true if their DTDs are different, e.g. they have various structures.

In this paper our similarity measurement among XML documents with different structures is firstly presented,
which is a similarity calculation algorithm with position and frequency weight by longest common subsequence (PFWLCS). On the basis of expressing XML document structures by correspondent paths using DOM tree, we extend the original tree path model to the frequency-path model by which we not only preserve label information of correspondent nodes, which decrease the original tree path model scale consumedly on the condition of not losing meaningful information and reduce the burden for later calculation, but also save the frequency structure of the original XML to improve the accuracy of the similarity. It makes the calculated similarity closer to the actual value by using the longest common subsequence method with position and frequency information. Furthermore, we continue to improve the frequency-path model by also considering node textural content, which makes the result more accurate and applicable, e.g. the hybrid method.

In Section II, Frequency-Path model and basic idea of hybrid similarity calculation methods are briefly introduced, followed by main steps of cluster algorithm NCC in detailed description in Section III. In Section IV experiments and its results are given showing better performance of our methods. Finally we summarize the whole work and provide future applications and research directions.

II. SIMILARITY CALCULATION

A. Structural Similarity

1) Frequency-Path Model

Definition 1: \( FPath = (f, v_1, v_2, ..., v_m, c_1, c_2, ..., c_m) \), where \( f \) denotes the number of occurrences of path in the current document; \((v_1, v_2, ..., v_m)\) is a node sequence from the root of XML DOM tree to one of its leaves; the \( c_i \) in \((c_1, c_2, ..., c_m)\) denotes the number of occurrences of \( v_i \), whose ancestor nodes \( v_i, v_{i-1}, ..., v_{i-l} \) are the same; \( m \) denotes the length of FPath (the node sequence).

Definition 2: \( XMLDoc = (FPath_1, FPath_2, ..., FPath_n) \), \( FPath \) and \( FPath_i \) are not the same FPath. We call two FPaths same only if the nodes at corresponding location of the two FPaths are completely identical. \( n \) denotes the number of various FPath.

Before calculating the similarity of XML, we extract the structure information of XML into FPaths where only the label of the node (structure) is considered. Other information such as data type and constraints are ignored. In Fig. 1 is an XML tree model and in Fig. 2 is a path model with its statistical information.

2) Frequency-Path Model Generation

Fig. 3 is the pseudo code of the algorithm to create the FPath model from XML document. The input of this algorithm is an XML document and output is an FPath model of it.

We also consider the semantic meanings that a node name can have during the structure matching. It is necessary since we are aiming at XML documents from different DTDs, which may not use the same word to express the similar meaning. For expressing the similar meaning only one word was taken from the synonymous word sets provided by WordNet. Besides, we assume that the node in higher hierarchy contributes more to the similarity than in lower hierarchy during the FPath matching. We will cover that in detail later.

For reducing the complexity of getting semantic FPath, we use a parameter \( \zeta \) to denote the depth of the node hierarchy being considered. For example, if \( \zeta = 1 \), we just consider the meaning of the first node (root) of FPath.

3) Algorithm PFWLCS

In this part the algorithm PFWLCS (Position and Frequency Weight by Longest Common Subsequence) used to calculate structural similarity of XML documents is described.

Definition 3: Subsequence: we call \( \langle a_{i_1}, a_{i_2}, ..., a_{i_k} \rangle \) a subsequence of \( \langle a_1, a_2, ..., a_n \rangle \), only if \( 1 \leq i_1 < i_2 < ... < i_k \leq n \).

Definition 4: Common subsequence: we call \( \langle c_1, c_2, ..., c_k \rangle \) one common subsequence of \( \langle a_1, a_2, ..., a_n \rangle \) and \( \langle b_1, b_2, ..., b_m \rangle \), only when \( \langle c_1, c_2, ..., c_k \rangle \) is a subsequence of \( \langle a_1, a_2, ..., a_n \rangle \) and also a subsequence of \( \langle b_1, b_2, ..., b_m \rangle \), \( k \) denotes the length of the common subsequence \( \langle c_1, c_2, ..., c_k \rangle \).

In our method, we use Longest Common Subsequence (LCS) in matching two given paths. In Table I different situations are illustrated by xPath [10-12], PCXSS [13] and LCS[14-15] respectively, showing more information can be kept by using LCS than other methods.
Figure 3. FPath model generation

We introduce a position-frequency weight vector 
\[ W(i) = \left( T(i) + F(i) \right) / 2 \]
and the position-frequency weight function \( W(i) \) is,
\[ W(i) = (T(i) + F(i)) / 2 \]
We prove that (2) satisfies the features of \( W(i) \), e.g.
\[ W(i) > 0 \text{ and } \sum W(i) = 1 \]

<table>
<thead>
<tr>
<th>No.</th>
<th>Path1</th>
<th>Path2</th>
<th>LCS</th>
<th>PCXSS</th>
<th>xPath</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(a,b)</td>
<td>(a,b)</td>
<td>(a,b)</td>
<td>(a,b)</td>
<td>NULL</td>
</tr>
<tr>
<td>2</td>
<td>(a,b)</td>
<td>(a,b)</td>
<td>(a,b)</td>
<td>(a,b)</td>
<td>NULL</td>
</tr>
<tr>
<td>3</td>
<td>(x,a,b)</td>
<td>(x,a,b)</td>
<td>(a,b)</td>
<td>(a,b)</td>
<td>NULL</td>
</tr>
<tr>
<td>4</td>
<td>(a,b,x)</td>
<td>(a,b,y)</td>
<td>(a,b)</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>(a,b,x)</td>
<td>(a,b,y)</td>
<td>(a,b)</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>(a,b,x)</td>
<td>(a,b,y)</td>
<td>(a,b)</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>(a,b,x)</td>
<td>(a,b,y)</td>
<td>(a,b)</td>
<td>NULL</td>
<td></td>
</tr>
</tbody>
</table>

To further explain the necessity for position-frequency weight vector, the followings are discussed.

Suppose there are several paths when calculating the XML document similarity: \( P(f_1, a, b, c, 1, 1, 1), P(f_2, a, b, x, 1, 1, 1), P(f_3, a, b, x, 1, 1, 1), P(f_4, 2, a, x, b, x, 2, 2, 1) \).

For the similarity comparison of actual data, the nodes in higher hierarchy have greater effect in the XML document tree, namely the more front position the node is, the more contributions to the similarity during FPath matching. Therefore, the similarity between \( P_1 \) and \( P_2 \) is significantly higher than it between \( P_1 \) and \( P_3 \). It shows the weight function \( W(i) \) is closely related to the position factor, namely position weight \( T(i) \).

We also notice that the similarity degree between \( P_1 \) and \( P_2 \) is significantly greater than it between \( P_1 \) and \( P_2 \). That is because in the case of FPaths with the same node position, higher frequency of the same node indicates its role is more dominant than other nodes. Hence, the weight function \( W(i) \) is also closely related to node’s frequency factor, namely frequency weight \( F(i) \).

From the above, the position-frequency weight function \( W(i) \) is composed of \( T(i) \) and \( F(i) \), where the position weight function \( T(i) = 1/2^i \). As for the frequency weight function \( F(i) \), we first define frequency equation of node \( \nu_i = c \log(\nu_i / f_i + 1) \) \((c_i \text{ is from the definition} 1; f_i \text{ is } \sum \text{for all FPaths in the document}) \). Then the normalized function of frequency weight can be expressed as (1).

\[ F(i) = c \log(\nu_i / f_i + 1) / \sum c \log(\nu_i / f_i + 1) \]

and the position-frequency weight function \( W(i) \) is, \[ W(i) = (T(i) + F(i)) / 2 \]

Definition 5: FPath similarity: Suppose two FPaths, \( FPath_1 = (f_{j_1}, x_1, x_2, ..., x_m, c_{y_1}, c_{y_2}, ..., c_{y_n}) \), \( FPath_2 = (f_{j_2}, y_1, y_2, ..., y_m, c_{y_1}, c_{y_2}, ..., c_{y_n}) \), the longest common subsequence (LCS) is \( LCSPath=(z_1, z_2, ..., z_k) \), the hierarchy of the nodes in LCSPath in \( FPath_1 \) is \( Hierarchy_1=(l_1, l_2, ..., l_k) \) orderly, the hierarchy of the nodes in LCSPath in \( FPath_2 \) is \( Hierarchy_2=(h_1, h_2, ..., h_k) \) orderly. Then the similarity of \( FPath_1 \) and \( FPath_2 \) is described as (3) below.

\[ \text{similarity} = \left( \frac{\sum W(Q) + \sum W(H_1) + \sum W(H_2)}{2} \right) / 2 \]

In some practical situations, the occurrence number of the same path is also an important component of XML structure information, but (3) does not contain the path frequency information and just uses label information, thus leading to the situation that the calculated result does not represent the true sense of XML similarity. So we integrate the path frequency into (3) in order to make the result closer to actual value.

From definition 5, \( f_{p1} \) and \( f_{p2} \) are the occurrence numbers of \( FPath_1 \) and \( FPath_2 \), while \( f_{p1}/f_{s1} \) and \( f_{p2}/f_{s2} \)
are the path frequencies of $FPath_1$ and $FPath_2$ ($f_{ij}$ is the sum of all path frequency in $FPath_i$, and $f_{j2}$ is the sum of all path frequency in $FPath_j$). On the basis of the TF-IDF statistics theory widely used in text mining, we assume that, the larger the recurrent number of a path in XML is, the higher path frequency it is. That is to say, the more important the path is, the more structure information the path contains. Therefore, the similarity of path should be appropriate to be improved, in which the path frequency is large and the degree of increasing similarity should be within the scope of $[1 - \sum W(i,j)/\sum W(j), i=1..k, j=1..n]$, so (3) is modified as follows.

$$
\text{similarity} = \frac{\sum f_{ij} + \sum f_{j2} [1 - (1 - \sum f_{ij}/f_{j2})] \cdot \frac{1}{k}}{\sqrt{\sum_{i=1}^{n} S_i}}
$$

(4)

Definition 6: XML document similarity: Suppose two XMLDocs, $XMLDoc_1 = (FPath_{11}, FPath_{12}, \ldots, FPath_{1m})$, and $XMLDoc_2 = (FPath'_{21}, FPath'_{22}, \ldots, FPath'_{2m})$, $m > n$. Each $FPath$ in $XMLDoc_1$ finds LCS with every $FPath$ in $XMLDoc_2$, and calculates the similarity according to (4). We denote the biggest similarity as $s_b$, the similarity between $XMLDoc_1$ and $XMLDoc_2$ is formed with (5) below.

$$
\text{Similarity} = \left[ \sum_{i=1}^{n} \frac{S_i}{n} + \sum_{i=1}^{n} S_i / m \right] / 2
$$

(5)

**Figure 4. PFWLCS algorithm**

Name: PFWLCS
Input: two $FPath$ model: mypathModel_1, mypathModel_2
Output: similarity of two Documents
Pseudo code: double getPFWLCSSimilarity

```java
{ //n1, n2: number of paths in mypathModel_1, mypathModel_2
    similarity = double getPFWLCSSimilarity(mypathModel_1, mypathModel_2)
    /n1, n2: number of paths in mypathModel_1, mypathModel_2
    //f1, f2: sum of path frequencies in the two models
    Initialize f1, f2;
    //fs1, fs2: sum of all path frequencies
    /fs1, fs2: sum of path frequencies in the current document;
    Initialize fs1, fs2;
    if(n1!=n2)
    getSimilarity(mypathModel_2, mypathModel_1);
    else{
        for(each path p1_i in mypathModel_1){
            for(each path p2_j in mypathModel_2){
                //a1, a2: position-frequency weights of p1_i,p2_j
                Initialize a1, a2;
                //get the longest common path of p1_i,p2_j
                lcs=getLCS(p1_i,p2_j);
                //w1,w2: position-frequency weights of the lcs
                Initialize w1, w2;
                im1=w1/a1;
                sim2=w2/a2;
                //f1,f2: path frequency of p1_i,p2_j
                Initialize f1,f2;
                sim=(sim2+a1*im1)/(1+1-(1-sim2)*f2/fs2);\n                similarity += sim;
            }
        }
        similarity=(similarity/n1+similarity/n2)/2;
        return similarity;
    }
}
```

Figure 5. An improved FPath model

Example in Fig. 5 is improved based on the model introduced before. A content vector $(e_1,e_2,\ldots,e_t)$ is supplemented, by which the XML content information is contributed to the result of data mining, which should not be ignored.

The content information is the textual part between tags. A method called SCSC (Similarity Calculation with Structure and Content) is presented in this subsection. The Frequency-path model is firstly improved in SCSC by adding the element content vector under the same path, making the presentation of XML documents richer. Moreover, a level ratio is introduced in XML similarity calculation, considering both the structural and element content similarity and making the result more sensible.

### 1) Improved frequency-path model

Definition 7: Improved tree path model: $IFPath=\langle f, v_1,v_2,\ldots,v_n, /E/c_1,c_2,\ldots,c_m \rangle$, where $f$ denotes the number of occurrences of path in the current document; $\langle v_1,v_2,\ldots,v_n \rangle$ is a node sequence from the root of XML DOM tree to one of its leaves; $E$ is a content vector $(c_1,c_2,\ldots,c_m)$ under structural path $(v_1,v_2,\ldots,v_n)$; the $c_i$ in $(c_1,c_2,\ldots,c_m)$ denotes the number of occurrences of $v_i$ whose ancestor nodes $v_1,v_2,\ldots,v_n$ are the same; $m$ denotes the length of IFPath (the node sequence).

1. automobile manufacturer / Cadillac / 20 1
2. automobile model / Cadillac CTS / 20 1
1 automobile year / 2004 / 20 1
1 automobile engine type / 255 hp / 20 1 1
1 automobile transmission / Speed Manual / 20 1 1
1 automobile feature seat / RWD / 20 1
1 automobile feature / Driver-Passenger airbags / 20 15 1 0
1 automobile feature / ABS Air Base / 20 15 1 2
1 automobile feature / Transmission Changer / 20 15 2

Figure 5. An improved FPath model
further kept, making hybrid computation of XML similarity of both structure and content possible.

2) **Content similarity calculation**

Suppose we have two paths \( P_1 \) and \( P_2 \) belonging to documents \( D_1 \) and \( D_2 \) respectively, \( E_i \) is content vector of \( P_i \) and \( E_j \) is content vector of \( P_j \).

Before processing, we use TF-IDF theory to complete feature word selection and extraction against each \( E_i \) and \( E_j \), resulting in feature word vector \( E_i' \) and \( E_j' \). Then we use cosine similarity equation to get content similarity.

\[
\text{Similarity(content)} = \frac{\theta}{\sqrt{n_1+n_2}}
\]

Where \( n_1 \) is the dimension of \( E_i' \) and \( n_2 \) is the dimension of \( E_j' \), \( \theta \) is the number of common feature words \( E_i' \) and \( E_j' \) have.

3) **Overall Similarity Calculation**

Due to the reason that a new part is added to the frequency-path model, the original similarity calculation method is about to be modified accordingly. Similarity of a complete document is defined:

\[
\text{Sim} = \alpha \text{Sim}_{\text{content}} + (1-\alpha) \text{Sim}_{\text{structure}}
\]

The final similarity is determined by both structure and content similarity, where \( \alpha \) is called the level ratio parameter, representing the structure layer of the element content. Structural similarity is still calculated using PFWLCS algorithm.

The higher level an element is at, the more closely it is to root node and has more contribution. That is, the level ratio factor \( \alpha \) is greater and so has close relation with the position factor \( T(i) \):

\[
\alpha = (T(i) + T(j))/2.
\]

\( T(i) \) and \( T(j) \) are two position weights of the last nodes in the compared paths separately.

III. XML DOCUMENTS CLUSTERING

A. Steps of Algorithm NCC

1) A point is chosen as the initial cluster center \( O_1 \) from a data set.

2) Set the center threshold parameter \( (\xi_1, \xi_2 \geq 0) \). Then the similarity of \( O_1 \) between each remaining points from the data set is calculated and compared. If the similarity is greater than \( \xi_1 \), the point will be put into the cluster \( C_1 \), where \( O_1 \) is in.

3) Set the neighbor threshold parameter \( (\xi_3, \xi_4 \geq \xi_3) \). At this time, the similarity of the points except \( O_1 \) in \( C_1 \) with the remaining points from the data set is calculated and compared again. If the similarity is greater than \( \xi_3 \), the point will be put into the cluster \( C_1 \).

4) Set the isolated threshold parameter \( (\phi, \phi = 1, 2... n) \). If the number of points in \( C_1 \) is less than \( \phi \), the cluster \( C_1 \) will be discarded.

5) A point is chosen as another cluster center \( O_2 \) from the remaining points in the data set. Repeat 2) 3) 4) steps until there are no points in the data set left.

B. Analyses of NCC

The basic idea of NCC algorithm is trying to identify the actual cluster from the data set in each iterative step greedily. Below explains the main idea of NCC algorithm.

Considering the XML document set \( D = \{d_1, d_2, d_3, d_4, d_5, d_6, d_7\} \) (\( d_i \) denotes the \( i \)th XML document), where \( d_1, d_2, d_3, d_4, d_5 \) are actually in the same cluster, while \( d_6 \) is in another set. The parameters \( (\xi_1 = 0, \xi_2 = 0.1, \phi = 1) \).

NCC arbitrarily selects \( d_1 \) as the initial cluster center and then finds the similarity between \( d_1 \) and \( d_2 \) greater than \( \xi_1 \), so \( d_2 \) is put into the cluster \( C_1 \), where \( d_1 \) is. So does \( d_3 \). At the moment, in order to avoid the situation that the chosen cluster center may be irrelevant resulting in imperfect clusters, it calculates and compares the similarity of the points except \( d_1 \) in \( C_1 \), in this case \( d_4 \) and \( d_5 \), with the remaining points. The purpose here is to spread the function of cluster center out over the neighborhood points in one cluster, which tries the number of matched points into one cluster, e.g. \( C_1 \), as many as possible. In Fig. 7, \( d_4 \) and \( d_5 \) are also put into the cluster because of \( d_2 \) and \( d_3 \).

Compared with some traditional algorithms, the NCC algorithm reduces the repeated calculation complexity on choosing cluster center in each iterative step, and improves overall efficiency. The clustering result from Fig.
7 is non-spherical, and besides, isolated points or isolated clusters are also identified, e.g. the point $d_6$ in the Figure.

NCC algorithm is suitable for XML clustering considering the XML structural character, while some vectorization methods ignore this kind of information during their calculation. It not only retains XML document’s structural information, but also evaluates XML’s textual meaning when with the method SCSC embedded. Therefore, the NCC algorithm based on SCSC has advantages in XML document clustering.

C. Evaluation of NCC

At present the two parameter indexes used widely to evaluate the overall performance of a clustering algorithm are the Purity and F-measure. The experimental data are the known document set or have usually been categorized before evaluation [16]. The purity of the cluster $r$ is defined as follows:

$$P(S_r) = \frac{1}{n_r} \cdot \max(n'_i)$$

The purity of the overall clustering is defined as:

$$\text{Purity} = \frac{1}{n} \sum_{i} n'_i \cdot P(S_r)$$

Where $n'_i$ is the number of document belonging to type $i$, which is assigned to cluster $r$; $n_i$ is the number of documents in cluster $r$; $n$ is the number of the whole document set.

High purity can be easily achieved when the number of clusters becomes larger. In particular, the purity will be 1 if each document gets its own cluster. Thus, F-measure is introduced as a harmonic mean to combine both precision and recall factors.

Recall: $\text{Recall}(i, r) = \frac{n(i, r)}{n_i}$

Precision: $\text{Precision}(i, r) = \frac{n(i, r)}{n_r}$

where $n(i, r)$ is the number of document belonging to type $i$ in the cluster $r$; $n_i$ is the number of document in cluster $r$; $n_r$ is the number of documents of type $i$. So the F-measure between cluster $r$ and type $i$ is defined as follows.

$$f(i, r) = \frac{2 \cdot \text{Recall}(i, r) \cdot \text{Precision}(i, r)}{\text{Recall}(i, r) + \text{Precision}(i, r)}$$

The F-measure of the overall clustering is defined as:

$$F = \frac{1}{n} \sum_i \frac{n'_i}{n} \max \{f(i, r)\}$$

IV. EXPERIMENT AND DISCUSSION

The goal of our experiments is to examine the effectiveness of the SCSC by calculating the similarity of XML documents using both structure and content information. Further, SCSC is embedded into NCC in clustering XML and we obtained satisfactory results.

Two data sets are used in the following experiments. The first data set used in our experiment is a real life data set introduced in [17], which is generated in the XML/XSLT version of web pages from 20 different sites belonging to 4 categories, labeled as “automobile”, “movie”, “reference” and “software”. There are a total of 120 documents: 24 in “automobile”, 24 in “movie”, 48 in “reference” and 24 in “software”. There is no cross-labeling and the depth of the DOM tree of these XML documents is 5. The second data set is from Sigmod XML collection, 84 files and 4 DTDs are included.

In the experiments, NCC method is used to cluster the two data sets and evaluate results according the recall ratio, accuracy and F-measure introduced before.

First we use improved frequency-path model to extract information from all documents, then complete feature selection and extraction. We randomly choose one file from two data sets as cluster center and calculate similarities of other files against the cluster center respectively, this process is repeated 10 times and the result is as following:

<table>
<thead>
<tr>
<th>Table II.</th>
<th>Experiment result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Values</td>
<td>DataSet1</td>
</tr>
<tr>
<td>Recall Ratio</td>
<td>81.13%</td>
</tr>
<tr>
<td>Accuracy</td>
<td>94.08%</td>
</tr>
<tr>
<td>F-measure</td>
<td>87.13%</td>
</tr>
<tr>
<td>Purity</td>
<td>94.33%</td>
</tr>
<tr>
<td>Clusters</td>
<td>3.72</td>
</tr>
</tbody>
</table>

From the Table II, results of dataset1 and dataset 2 are all satisfactory. Specifically, results of the first two columns, e.g. dataset1 (SCSC) considers both the structure and content information, while dataset1 (PFWLCS) is calculated by considering only structure information. Each item of dataset1 (SCSC) is a little lower than dataset1 (PFWLCS), that is because in dataset1 XML files are from many different DTDs, even in the same cluster. Therefore structural information plays more important role than content information in this situation. However, considering both the structure and content information is more nature and reasonable in practice and can reflect more real feature of data.

A good similarity calculation will make the similarity closer within one cluster and larger outside the cluster. In the following experiments, we pick up one file arbitrary from 4 clusters of the two datasets separately. We compare the similarity of this file to other files in its cluster and repeat this process 10 times. The following results are obtained.
with position frequency weight, based on frequency path structural similarity is calculated using method PFWLCS structure and content similarity. On one hand, the effectiveness will be achieved by considering both method NCC, when embed with SCSC higher performance, decrease complexity when embedded in heterogeneous XML documents. NCC, and is suitable to both homogenous and performance, decrease complexity when embedded in XML's content similarity is obtained through method and the position frequency weight vector. On the other hand, XML's content similarity become important. Anyway, SCSC model, in which more valuable information in XML information become more significant, while in dataset2 dataset1 are from different DTDs and structure information become important. Anyway, SCSC can handle both of these conditions and results showing that the hybrid method is more applicable and effective.

<p>| TABLE III. SIMILARITY COMPARISON IN DATASET1 |</p>
<table>
<thead>
<tr>
<th>SimAvg(Si, Sj)</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1 1.0000</td>
<td>0.4853</td>
<td>0.0054</td>
<td>0.0026</td>
<td>0.0018</td>
</tr>
<tr>
<td>S2 0.0054</td>
<td>0.4526</td>
<td>0.0021</td>
<td>0.4722</td>
<td>0.0012</td>
</tr>
<tr>
<td>S3 0.0026</td>
<td>0.0021</td>
<td>0.0012</td>
<td>0.4336</td>
<td></td>
</tr>
</tbody>
</table>

$S_i$ is the $i^{th}$ cluster in Dataset1, $SimAvg(S_i, S_j)$ is the average similarity between documents in cluster $S_i$ and $S_j$.

<p>| TABLE IV. SIMILARITY COMPARISON IN DATASET2 |</p>
<table>
<thead>
<tr>
<th>SimAvg(Ci, Cj)</th>
<th>C1</th>
<th>C2</th>
<th>C3</th>
<th>C4</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1 1.0000</td>
<td>0.9646</td>
<td>0.1326</td>
<td>0.1464</td>
<td>0.1318</td>
</tr>
<tr>
<td>C2 0.1326</td>
<td>0.9524</td>
<td>0.1298</td>
<td>0.1431</td>
<td></td>
</tr>
<tr>
<td>C3 0.1464</td>
<td>0.1298</td>
<td>0.9346</td>
<td>0.1373</td>
<td></td>
</tr>
<tr>
<td>C4 0.1318</td>
<td>0.1431</td>
<td>0.1373</td>
<td>0.9546</td>
<td></td>
</tr>
</tbody>
</table>

$C_i$ is $i^{th}$ cluster in Dataset2, $SimAvg(C_i, C_j)$ is the average similarity of documents in clusters $C_i$ and $C_j$.

From the two tables above, Similarities in the same cluster are all far greater than in other clusters using SCSC. We also notice that scores of dataset2 is a litter greater than in dataset1, this is because most XML files in dataset1 are from different DTDs and structure information become more significant, while in dataset2 XML files in one cluster are most from same DTD and content information become important. Anyway, SCSC can handle both of these conditions and results showing that the hybrid method is more applicable and effective.

V. CONCLUSION

We have demonstrated an XML document clustering method NCC, when embed with SCSC higher effectiveness will be achieved by considering both structure and content similarity. On one hand, the structural similarity is calculated using method PFWLCS with position frequency weight, based on frequency path model, in which more valuable information in XML clustering is kept using the longest common subsequence method and the position frequency weight vector. On the other hand, XML’s content similarity is obtained through TF-IDF method. Experiments showed SCSC method could greatly help to improve clustering precision and performance, decrease complexity when embedded in NCC, and is suitable to both homogenous and heterogeneous XML documents.

REFERENCES


Yong Piao, born in Liaoning province, China, 1975, received his B.A’s and M.A’s degrees in computer science from Dalian University and Technology, Dalian, Liaoning, China, in 1998 and 2001 respectively. From 2002 to 2003, he made an advanced study in EI department, University of Siegen, Germany. Since 2004 he has been a lecturer, 2010 an associate professor, in Dalian University of Technology. His current research interests mainly cover database and decision support systems.

Xiu-kun Wang, born in 1945. She has been a professor in Dalian University of Technology. Her main research interests are data mining, database and decision support systems.