PKU at INEX 2010 XML Mining Track

Songlin Wang, Feng Liang, and Jianwu Yang[*]

Institute of Computer Sci. & Tech., Peking University,
Beijing 100871, China
{Wangsonglin,liangfeng,yangjianwu}@icst.pku.edu.cn

Abstract. This paper presents our participation in the INEX 2010 XML Mining track. Our classification and clustering solutions for XML documents have used both the structure and content information, where the frequent subtrees as structural units are used for content extraction from the XML document. In addition, we used the WordNet and the link information for better performance, and applied the structured link vector model for classification.

Keywords: XML Document, Classification, Clustering, Frequent Subtree, Structured Link Vector Model (SLVM).

1 Introduction

The two tasks in INEX 2010's XML mining track are categorization and clustering. The clustering task is an unsupervised process through which all the documents can be classified into clusters and the problem is to find meaningful clusters without any prior information. The categorization task is a supervised task for which, given a set of categories, a training set of reclassified documents is provided. Using this training set, the task keep on learning the categories description in order to be able to classify a new document into one or more categories. And in this XML mining tack, the corpus is a subset of the Wikipedia corpus with 144,625 documents that belong to 36 categories resulting in a multi-label classification task.

In this paper, the documents are represented according to the structured link vector model (SLVM) [1], which was extended from the conventional vector space model (VSM) [2], and used the closed frequent subtrees as structural units. More precisely, we focus on the preprocessing step, particularly on the feature selection, frequent subtrees selection, and the process of link information, and these steps can be essential for improving the performance of the categorization and clustering.

Moreover, the content information (the text of the documents), the structural information (the XML structure of the documents) and the links between the documents can be used for this task. We need to extract text from a subset of terms that can be used to represent the documents in view of their categorization efficiently. In this year, a file with a list of links between XML documents has been given, and we will show that how those links add relevant information for the categorization of documents.

[*] Corresponding author.

This paper is organized as follows. In section 2, document representation is discussed. In section 3, we show some parameters tuning in the corpus of INEX 2009 and INEX2010. Section 4 describes the approach taken to the classification task and the associated results. In section 5, we review the clustering task and discuss the results. The paper ends with a discussion of future research and conclusion in section 6.

2 System Overview

2.1 Document Model for Categorization (SLVM)

The Vector space model (VSM) [2] has been widely used for representing text documents as vectors which contain terms weights. Given a collection D of documents, a document $\mathrm{d_x}$ of D is represented by a vector $\mathrm{d_x} = (\mathrm{d}_{(x,1)}, \mathrm{d}_{(x,2)}, \mathrm{d}_{(x,3)} \ldots\ldots \mathrm{d}_{(x,n)})$, where $\mathrm{d}_{(x,i)}$ is the weight of the term $\mathrm{t_i}$ in the document $\mathrm{d_x}$. SLVM is extended from VSM. The basic idea is to extract structure units from XML documents, then to extract content information from each structure units. The content information of each structure units is regarded as a VSM model. Every XML document is modeled as a matrix in SLVM [1].

So, the document $\mathrm{d_x}$ can be represented by a document feature matrix $\chi \in R^{m \times n}$,

$$\mathrm{d_x} = \left(\mathrm{d}_{x(1)}, \mathrm{d}_{x(2)}, \mathrm{d}_{x(3)} \ldots\ldots \mathrm{d}_{x(m)}\right) \tag{1}$$

$$\mathrm{d}_{x(i)} = \left(\mathrm{d}_{x(i,1)}, \mathrm{d}_{x(i,2)}, \mathrm{d}_{x(i,3)} \ldots\ldots \mathrm{d}_{x(i,n)}\right) \tag{2}$$

Where m is the number of unit of document $\mathrm{d_x}$, and $\mathrm{d}_{x(i,j)}$ is the weight of the term $\mathrm{t_j}$ in the unit $\mathrm{d}_{x(i)}$ of document $\mathrm{d_x}$. SLVM combines both structure information and content information. In order to calculate the weight of the terms, TFIDF and BM25 formula can be used.

2.2 BM25 Formula

Our system is based on the BM25 weights function [3].

$$\mathrm{BM25(q,d)} = \mathrm{TF}(q,d) \times \mathrm{IDF}(q) \tag{3}$$

$$\mathrm{TF(q,d)} = \frac{f(q,d) \times (\mathrm{k_1}+1)}{f(q,d) + \mathrm{k_1} \times \left(1 - \mathrm{b} + \mathrm{b} \times \frac{\mathrm{d}}{\mathrm{D_{avg}}}\right)} \tag{4}$$

$$\mathrm{IDF(q)} = \log \frac{N - n(q) + 0.5}{n(q) + 0.5} \tag{5}$$

With:

- $f(q,d)$: the frequency of term q in article d.
-N: the number of articles in the collection.

-n (q): the number of articles containing the term q.

-$\frac{d}{D_{avg}}$: the ratio between the length of articles d and the average article length.

-k_1 and b: the classical BM25 parameters.

Parameter k_1 is able to control the term frequency saturation. Parameter b allows setting the importance of $\frac{d}{D_{avg}}$.

2.3 Chi-Square Feature Selection

The main idea of the feature selection is to choose a subset of input variables by eliminating features with little or no predictive information, and the feature selection can significantly improve the comprehensibility of the classifier models. The Chi-square test is a commonly used method, which evaluates the features individually by measuring their chi-square statistic with respect to the classes.

3 Parameters Tuning

We have used both INEX2010 and INEX2009 collection of XML Mining Track for this experiment. As the collection of INEX2010 has been described above, the collection of INEX2009 is composed of about 54,889 XML documents of the Wikipedia XML Corpus, and this subset of Wikipedia represents 39 categories, each corresponding to one subject or topic. The training set with the category annotations is composed of 11,028 elements, which is about 20% of the whole collection. On the training set, the average the number of category per document is 1.46 and 9809 documents belong to only one category.

First we list all the features encountered in the documents of the collection, where about one million features are built for all of the documents. Then the chi-square test has been used to select the features, and we find that about 2,000 features for each categorization can acquire a good value.

The INEX 2009 tuning results are shown in table1, where all the value with the BM25 have been improved, so that our system is based on the BM25 weights function for this year task.

Table 1. The result of BM25(INEX2010)

	Ma_acc	Mi_acc	Ma_roc	Mi_roc	Ma_prf	Mi_prf	Map
TFIDF	0.966	0.953	0.855	0.863	0.496	0.543	0.709
BM25	0.967	0.952	0.932	0.933	0.537	0.579	0.745

3.1 Pruning the Tree

In this paper, we utilize the frequent subtrees as structural units to extract the content information from the XML documents, and a series of pre-processing have been done for the subtrees.

As the last year, we use the closed frequent subtrees as structural units. The documents of INEX2009 and INEX2010 is more complex than the collection of ever before, so we employ some pruning strategies for mining closed frequent subtrees and use the chi-square test to select a part of subtrees as useful structure. We obtain the document of INEX2009 vectors at three different pruning levels, 0, 1/3, and 1/2, which yield a better performance for almost all the evaluation measures, and INEX2010 has five as shown in table 3.

Table 2 presents the effectiveness comparison of classification structures at various pruning levels of INEX2009, while used no more than 40,000 features Table 3 presents the effectiveness comparison of classification structures at various pruning levels of INEX2010, 80,000 features for all categories, as concede that calculation of the similarity between the words is very slowly, we have not used the information of WordNet in this part.

Table 2. The results of pruning the document tree (2009)

Prune	Ma_acc	Mi_acc	Ma_roc	Mi_roc	Ma_prf	Mi_prf	Map
0	0.956	0.940	0.879	0.861	0.428	0.438	0.606
1/3	0.961	0.944	0.896	0.872	0.464	0.465	0.626
1/2	0.961	0.942	0.876	0.873	0.447	0.475	0.633

Table 3. The results of pruning the document tree (2010)

	Prune	Mi_p	Ma_p	Mi_r	Ma_r	Mi_f	Ma_f
Split_0	0%	0.453	0.39	0.632	0.55	0.524	0.454
Split_3	35%	0.458	0.409	0.635	0.555	0.534	0.466
Split_5	47%	0.467	0.407	0.639	0.558	0.535	0.467
Split_10	58%	0.474	0.412	0.635	0.560	**0.539**	**0.472**
Split_20	68%	0.454	0.392	0.633	0.557	0.525	0.457

In addition, we find that the average of subtrees for each XML document has increased, which is useful for the classification, when some pruning strategies are used for mining closed frequent subtrees.

3.2 WordNet

In some documents, there are only a few words, which can't express the information of the categories completely. So we use the WordNet to extend the information of the documents.

Given the closed frequent subtrees $T(t_1, t_2, t_3 \dots t_s)$, the features $W(w_1, w_2, w_3 \dots w_m)$, and two documents D_i, D_j that contain the tree t_k of T, and suppose their parts of the t_k are $D_{i_{tk}}(w'_1, w'_2, w'_3 \dots w'_m)$ and $D_{j_{tk}}(w''_1, w''_2, w''_3 \dots w''_n)$.

The similarity between the t_k parts of the documents $D_i$and D_j is given as:

$$S_{ij_{tk}} = \sum_{a=1}^{\mathrm{Max}\{m,n\}} w'_a * w''_a \tag{6}$$

After using the WordNet, we can get the similarity matrix between the words, and the similarity between the t_k parts of the documents given as:

$$S_{ij_{tk}} = \sum_{a=1}^{m}\sum_{b=1}^{n} S_{w'_a w''_b} w'_a * w''_b \tag{7}$$

Where $S_{w'_a w''_b}$ is the similarity between the word w'_a and w''_b.

Given the similarity matrix between the words, we can change the $D_{j_{tk}}$ as follow:

$$D_{j_{tk}}(w''_1 + \sum_{a}^{m} S_{w''_1 w'_a}, w''_2 + \sum_{a}^{m} S_{w''_2 w'_a}, w''_3 + \sum_{a}^{m} S_{w''_3 w'_a} \dots w''_n + \sum_{a}^{m} S_{w''_n w'_a}) \tag{8}$$

When $D_{j_{tk}}(w''_1, w''_2, w''_3 \dots w''_n)$ have all features of $W(w_1, w_2, w_3 \dots w_m)$, this method is equal to (7). However, no document can contain all of the selected features, and in this way we can increase the feature number of the document, especially the words that don't appear in the collection, which is important to some small documents for classification and clustering, it equal to use the similarity matrix between the words to do smoothing. For example, when the features of $D_{j_{tk}}$ don't contain word A, but it contains word B and word C, which the similarity with A is greater than 0, then we can use $S_{w_A w_B} + S_{w_A w_C}$ to smoothing the weight of word A in $D_{j_{tk}}$.

Definition. $S_{w_a w_b}$ means the similarity between the two words $w_a$and w_b , if $S_{w_a w_b} > \gamma$, where γ is the parameter of similarity, $w_a$and w_b are synonym, we can merge the word $w_a$and w_b together.

3.3 Using the Link Information to Improve the Result

In this section, we show that the classification accuracy can be improved based on word features and out-linked features. And the XML documents of INEX can provide much richer information to classifiers for classification. Links between the documents can be used for this task. In this year a file with a list of links between XML documents has been given, and we will show that those links add relevant information for the categorization of documents. As how to use the link information, we have tried three ways:

3.3.1 Change the Document Structural

After a series of pre-processing for the documents, we can change the documents in the following format:

```
<article>
    <content_information> ……</ content _information>
    <frq_subtrees>……</frq_subtrees>
</article>
```

In the above table, the text of the document is included by the tag of "<content_information>", "<frq_subtrees>" tag contain sthe frequent subtrees of the document, which is the XML structure of the documents. As well, in order to join the link information of the document, we should add another tag, which is named "<link_informatiom>", and the table can be changed as follow:

```
<article>
    <content_information> ……</ content _information>
    <frq_subtrees>……</frq_subtrees>
    <link_information>……</link_information>
</article>
```

3.3.2 Change the Vector of the Document

The closed frequent subtrees $T(t_1, t_2, t_3 \dots t_s)$, and a given document$D$, which contain the tree t_k of T, and suppose the part of the t_k is

$$D_{tk}(w'_1, w'_2, w'_3 \dots w'_m) \tag{9}$$

Where m is the total number of features appearing in the tree t_k part of the document collection, w'_i is the weight of term i in document D and $0 \le w'_i \le 1$. Hence, if $w'_i > 0$,it means that word i exist in the tree t_k part of the document D. Therefore, document D will be represented by a vector of features

$$D_{tk}(w'_1, w'_2, w'_3 \dots w'_m, l'_1, l'_2, l'_3 \dots l'_n) \tag{10}$$

Where n is the total number of links appear in the tree t_k part of the document, l'_i is the weight of link l'_i in document D and $0 \le l'_i \le 1$. Hence, if $l'_i > 0$,it means that link l'_i exists in the tree t_k part of the document D.

3.3.3 Using the Link for Tuning

Build SVM model base on the link information, get the correlativity between the documents and categories, and improve the correlativity of documents and categories that calculate form the text information:

$$S(c|d) = \alpha \times S_l(c|d) + (1-\alpha) \times S_t(c|d) \tag{11}$$

Where $S(c|d)$ is the final correlativity between the document d and the category c; $S_l(c|d)$ is the correlativity between the document d and the category c by using the link information only; and $S_t(c|d)$ is the correlativity between the document d and the category c by using the text information only; parameter α allows setting the importance of $S_l(c|d)$.

In this experiment, we use 80000 features for the text content, and use BM25 for the text representation. The SVM based text classification results are listed in Table 4 and 5. In the following tables, we can notice that, out-linked features work very well in the experiment, and this simply means that adding in out-link features can improve the performance of classification.

Table 4. The result of using the link for tuning (2009)

	Ma_acc	Mi_acc	Ma_roc	Mi_roc	Ma_prf	Mi_prf	Map
α=0	0.967	0.952	0.932	0.932	0.537	0.579	0.746
Using link for tuning	0.968	0.956	0.912	0.916	0.570	0.608	0.780

Table 5. The result of using the link for tuning (2010)

	Prune	Mi_p	Ma_p	Mi_r	Ma_r	Mi_f	Ma_f
Split_10	58%	0.474	0.412	0.635	0.560	0.539	0.472
Split_10_tuning	58%	0.491	0.433	0.644	0.572	0.554	0.490

3.4 Other Tuning

In addition to the above methods, we also made some other optimization, for example smoothing IDF value in the formula of BM25, normalized the vector of the units in the document, and in particular the result of normalized the matrix of the documents performance better, so using this case, we pruning the trees of documents and using the out-link information, and the we get our result in the table 6, as follow.

Table 6. The result of using the link for tuning

runs	idf	Norm	Mi_p	Ma_p	Mi_r	Ma_r	Mi_f	Ma_f
	log	N	0.473	0.414	0.646	0.566	0.544	0.476
	1+log	N	0.476	0.416	0.641	0.563	0.543	0.476
	1+ log(f-v)/v	N	0.474	0.414	0.636	0.559	0.541	0.473
	1+log	Y	0.528	0.485	0.647	0.572	0.578	0.521
Result_link_ tuning	1+log	Y	0.543	0.499	0.656	0.583	**0.591**	**0.535**

3.5 Compare the Results

Table 7. Compare the result with INEX 2009

Team	Micro F1	Macro F2	APR
University of Wollongong+	51.2	47.9	68
University of Peking+	51.8	48	70.2
XEROX Research center+	60	57.1	67.8
University of Saint Etienne+	56.4	53	68.5
University of Granada+	26.2	25.3	72.9
Our result*	**60.9**	**57.1**	**78.0**

+ Result of INEX09; * The wordNet is not used.

Table 8. Compare the result with INEX 2010

Team	Run	Mi_p	Ma_p	Mi_r	Ma_r	Mi_f	Ma_f
Peking	T1_S3_linkN+	0.432	0.356	0.656	0.613	0.517	0.444
	T1_S1_link0+	0.582	0.570	0.363	0.252	0.400	0.320
Qut	Inex10_sub+	0.561	0.527	0.523	0.440	0.536	0.473
	Our result*	0.543	0.499	0.656	0.583	**0.591**	**0.535**

+ Result of INEX2010; * The wordNet is not used.

4 The Clustering Methods

AHC Clustering Algorithm and K-means Algorithm are two popular clustering algorithms in statistics and machine learning, but these methods do not work well in the massive data sets. So we modify these algorithms, considering the structure of the document and the massive data set. We have approached the simple countering technique based on batch of the complete document collection; the process followed is present in Figure 1.

Another algorithm is based on AHC Clustering Algorithm, and the process of the Local AHC Clustering Algorithm is present as Figure 2.

```
• Given
• X: a set of N vectors
• K:desired number of clusters
• Begin
    • Select K random seeds $\{\overrightarrow{s_1}, \overrightarrow{s_2}, \dots\dots, \overrightarrow{s_k}\}$ from X
    • Each time load part of the documents, and computer the distance
    between the document and clusters
    •  Do
        • Do all of the document
            • Load a batch of documents
            • Assign each document to cluster which is the minimal
            distance, and calculate the cluster for next time.
        • End_do
        • Calculate the clusters and the distance between the clusters
        and document.
    • End_do (if the gap of distance between two consecutive cycles
    exceeds the given threshold.)
    • Load the documents and assign each document to clusters.
    • Print_result.
• Done
```

Fig. 1. An algorithm for clustering

```
• Let D be the whole complete document collection, split D into subset $D_i$
containing 1000 documents, and for each $D_i$, we do the follow process, after we
combine the result ,then continue, until the number of clusters meet our
requirement:
    • Given:
    • Subsets $D_i$ and a threshold $\tau$
    • Calculate similarity matrix M
    • Do  (if max element in M > threshold $\tau$)
        • Find the max element of M: $m_{i,j}$ and constructs a cluster
          $C_i$ made up of the document $d_i$ and $d_j$ , it marks these
          documents as assigned.
        • Do  all unassigned document $d_k$
            • If  $m_{i,k} > \tau$
                •  add $d_k$ to cluster $C_i$ and mark $d_k$ assigned
            • End_if
        • End_do
        • Update row i and column j, set $m_{k,j} = m_{i,k} = \text{avg}(m_{c,k})$,
        where $d_c \in C_i$.
        • Update row c and column c, set $m_{k,c} = m_{c,k} = -1$,
        where $d_c \in C_i$.
    • End_do
```

Fig. 2. The Local AHC Clustering Algorithm

```
• Given:
• The list of xml documents D
• Calculate similarity matrix M
• While (Loop < MAX_Loop)
        • Split D into subsets $D_i$ containing 1000XML documents
        • For all $D_i$ of D do
                • Apply the Local AHC Clustering Algorithm to discover
              k clusters $C_{(Loop,k)}$
        • End for
        • If(Loop>1) then
                • Merge $C_{(final,k)} = C_{(Loop,k)} \cup C_{(Loop-1,k')}$ where
                    $C_{(Loop,k)} \cup C_{(Loop-1,k')} > 1$
        • End if
        • Select a representative centroid $R_j$ for each cluster
        • Let D be the set of representative centroid $R_j$
        • Loop = Loop + 1
• End while
• Output the set of clusters $C_{final}$
```

Fig. 3. Simple algorithm for clustering

Experimental results and analysis of these clustering algorithms will be showed in section 5.

5 INEX 2010 Results

We present in this section the official results obtained by our system during INEX 2010 on the new INEX 2010 collection, we submitted 9 runs, and all of our submit results are based on the BM25, reference run given by the INEX organizers.

5.1 System Settings

Most of the settings given in section 3 have been reused for our INEX 2010 runs, except:

- BM25 parameters $k_1 = 2.0$ and $b = 0.75$;
- For each category, we set the frequent tree number is 10, and the threshold of support is 200;
-To improve the result of classification, we used the link information, and α we set 0.40 and 0.46, which is obtained from tuning in section 3;
-The threshold for each category, we tuning from the training set, and we get three groups.

5.2 Results

Table 9 present the official results of classification:

Table 9. The results of classification

Team	Run	Mi_p	Ma_p	Mi_r	Ma_r	Mi_t
Peking	T1_S3_link0	0.553	0.525	0.436	0.334	0.931
	T1_S3_link67	0.422	0.345	0.653	0.612	0.874
	T1_S3	0.433	0.368	0.635	0.582	0.886
	T1_S3_linkN	0.432	0.356	**0.656**	**0.613**	0.878
	T1_S2_linkN	0.456	0.389	0.615	0.559	0.893
	T1_S1_link0	**0.582**	**0.570**	0.363	0.252	**0.934**
	T2_S2_link0	0.562	0.536	0.422	0.321	0.934
	T2_S2	0.48.	0.414	0.574	0.510	0.910
	T2_S3_linkN	0.436	0.359	0.652	0.614	0.882
Qut	Inex10_sub	0.561	0.527	0.523	0.440	0.932

Team	Run	Ma_t	Mi_a	Ma_a	Mi_f	Ma_f
Peking	T1_S3_link0	0.974	0.891	0.942	0.518	0.446
	T1_S3_link67	0.928	0.860	0.914	0.508	0.435
	T1_S3	0.936	0.869	0.920	0.518	0.444
	T1_S3_linkN	0.931	0.864	0.917	0.517	0.444
	T1_S2_linkN	0.944	0.873	0.926	0.522	0.454
	T1_S1_link0	**0.980**	0.888	0.943	0.400	0.320
	T2_S2_link0	0.976	0.892	0.943	0.452	0.372
	T2_S2	0.954	0.883	0.932	0.521	0.452
	T2_S3_linkN	0.933	0.886	0.918	0.518	0.446
Qut	Inex10_sub	0.970	**0.896**	**0.944**	**0.536**	**0.473**

Firstly, for all of the runs , we use 80,000 features for all categories, and frequent subtrees have selected two groups: one is T1, in which the trees can be a subtree of another; another is T2, in which the frequent subtree cannot be a subtree of another. We also use different thresholds for each category, S1, S2, S3. And we use the link information to improve the result of the castigation, the run 1, 2, 4,5,6,7 and 9 have used, but the run 3 and 8 have not. In addition, in order to save time, we just use the WordNet to calculate the similarity matrix between the words that selected form INEX2009 collection, so there is a certain deviation between the words.

Secondly, from the result table, we can obtain some conclusions. The threshold of each category has played an important roe, and the results including the information are not as good as last year. We suspect that this might be caused by the fact that we tuned the parameters in the collection of INEX 2009, which were not suitable in this year's data. For this year, the number of categories that each document belonging to is more than for last year, and the standard evaluation of classification has changed, thus, future experiments should do this parameters adjustment. In the part of frequent subtrees we use two extreme cases, T1 and T2, and the result are not so much difference as we expected, though they have a different effect on the INEX2009 data set, so this part needs to do further research experiments. Tables 10 present the official results of clustering.

Table 10. The results of clustering

	Runs	Mi_p	Ma_e	Mi_e	Ma_n	Mi_n
Pek	Result_1000_4	0.531	3.89	3.89	0.202	0.219
	Result_500_4	0.518	3.99	3.97	0.201	0.218
	Result_200_4	0.467	4.00	4.05	0.314	0.212
	result502_0.15	0.364	4.36	4.52	0.133	0.118
	Result_100_4	0.481	4.06	4.10	0.212	0.206
	result_0.18_1004	0.336	4.38	4.48	0.118	0.119
	Result_50_4	0.463	4.11	4.18	0.203	0.191
	Result_100_3	0.483	4.07	4.11	0.209	0.204
	Result_200_2	0.501	4.01	4.00	0.213	0.213
	Runs	**Ma_f1**	**Mi_f1**	**Nmi**	**M_NCCG**	**S_NCCG**
Pek	Result_1000_4	0.072	0.150	0.093	0.524	0.220
	Result_500_4	0.063	0.140	0.088	0.584	0.235
	Result_200_4	0.056	0.130	0.084	0.661	0.219
	result502_0.15	0.019	0.053	0.020	0.391	0.221
	Result_100_4	0.048	0.120	0.082	0.741	0.193
	result_0.18_1004	0.022	0.060	0.024	0.290	0.169
	Result_50_4	0.038	0.101	0.075	0.185	0.157
	Result_100_3	0.040	0.119	0.081	0.745	0.190
	Result_200_2	0.057	0.132	0.085	0.662	0.212

In table 10, the runs 4 and 6 we use the Local AHC Clustering Algorithm, and the rest we used k-means based on batch of the complete document collection.

6 Conclusions

In this paper, we applied SLVM to XML documents classification and clustering, and we used the frequent subtrees as structural units, both the structure and content information have been used, especially the information of link, which performance well, and in order to extend the information of features, we used the WordNet to smooth the weight of the words in the documents.

It is not so easy to reuse setting of parameters tuned on the different collection. In the experiments of INEX2009, we show that using link information can improve the result of classification by 2%, but the results in part 5 that we presented are not good, so we experiment more deeply on 2010 collection. Furthermore, Information provided by the inlinks (links received by one file) is also useful, a symmetric procedure can be defined with inlinks instead, and being tested. If the inlinks could be available, that information could result in a better performance. Some of those experiments will probably be included in the final version of this paper although, as we exposed in the introduction, we not consider that approach to be very realistic.

Acknowledgment. The work reported in this paper was supported by the National Natural science Foundation of China Grant 60642001 and 60875033.

References

1. Yang, J., Chen, X.: Chen, X.: A semi_structured document model for text mining. Journal of computer science and Technology 17(5), 603–610 (2002)
2. Salton, G., McGill, M.: Introduction to Modern information Retrieval. McGraw-Hill, New York (1983)
3. Robertson, S.E., Spark Jones, K.: Relevance weighting of search terms. JASIST 27(3), 129–146 (1976)
4. Chi, Y., Nijssen, S., Muntz, R.R., Kok, J.N.: Frequent Subtree Mining – An Overview. Fundamenta Information (2005)
5. Chi, Y., Yang, Y., Xia, Y., Muntz, R.R.: CMTreeMiner: Mining Both Closed and Maximal Frequent Subtrees. In: The Eighth Pacific Asia Conference on Knowledge Discover and Data Mining (2004)
6. Xie, W., Manmadov, M., Yearwood, J.: Using Links to Aid Web Classification. In: ICIS (2007) 0-7695-2841-4/07
7. Yang, J., Zhang, F.: XML Document Classification using Extended VSM. In: Pre-Proceedings of the Sixth Workshop of Initiative for the Evaluation of XML Retrieval, Dagstuhl, Germany (2007)
8. Berry, M.: Survey of Text Mining: Clustering. Springer, Heidelberg (2003)
9. Joachims, T.: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, Springer, Heidelberg (1998)
10. Yang, J., Wang, S.: Extended VSM for XML document classification using frequent subtrees. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2009. LNCS, vol. 6203, pp. 441–448. Springer, Heidelberg (2010)
11. Shin, K., Han, S.Y.: Fast clustering algorithm for information organization. Proc. of the CICLing 2003 Conference. In: Gelbukh, A. (ed.) CICLing 2003. LNCS, vol. 2588, pp. 619–622. Springer, Heidelberg (2003)