XML Mining Track: XML clustering using WEKA

Jaturawit Sooksompong and Surachate Disapirom

School of Computer Science and Information Technology,
RMIT University, Melbourne, Australia
E-mail : s3159078@student.rmit.edu.au, s3144736@student.rmit.edu.au

Abstract.

In this paper, we describe the attempt at the INEX 2007: XML Mining Track, which continue from INEX 2005, using an approach that combine principles of Information Retrieval with Data Mining to find a solution for categorising the types of documents. This experiment focused exclusively on the content of documents.

We tested the approach by several random sets of data with the hypothesis that in the same category of documents, they should have the similar set of terms.

The result of this system is not considered succeeding to classify document types. The correctness ratio of the experiment is approximately 50%. To improve the efficiency of the clustering, the pre-processing step is the most importance by removing general words

1. Introduction

Since the world realize that documents in paper format are unlikely to last long for decades; so most documents has been transformed into digital formats. As one of the digital format, XML is a semistructure like many languages on the Web such as HTML, SGML and it promises to be compatible with any applications which support XML standard. With its great benefit, this format has been chosen to use on the Web such as Wikipedia website. As we all know that Wikipedia allow everyone post data in the website, this make the website contains variety of useful information and rapidly increase the size of the website's data; however, the large volume of data needs to be managed in efficient way.

One of the most popular techniques is data mining. Data mining basically is a process to analysedata from various dimension to extract the useful information [1]. Data mining mainly used these days in business intelligent organisations to find out customer's needs and in financial analysis to predict the marketing trend [2]; however, in science field, data mining has been increasingly used to experiment and observe large data set to generate new methods. The software for data mining is considered as analytical tool such as Clemmentine, Darwin, MineSet and Weka

Initiative for the Evaluation of XML Retrieval (INEX) has been challenged since 2005 to develop structured data mining by machine learning methods, and to evaluate these methods for XMLdocument mining task. This track is focused both on classification and clustering. Clustering is a technique to classify entities into several groups. INEX 2007 provides the dataset as Wikipedia documents which is a set of 48035 XML files.

This paper focuses only on clustering approach for particular XML dataset emphasise on mining from content of documents due to the fact that structure of documents in this collection does not give meaningful information [5]. We propose a method to make use of Inverted List [4] in Information Retrieval theory (IR) and Data Mining by Weka tool. The experiments will compare the efficiency (Purity of the result clusters) of clustering algorithms or threshold level applied.

2. Background

2.1 Weka (Waikato Environment for Knowledge Analysis) – Analytical tool

This tool consists of a collection of several small graphical programs with algorithms for data analysis and predictive modeling. Originally, this tool was developed with TCL/TK for user interactive section and C for the data processing [3, 6]. Weka in current version has been moved to Java and issued under GNU General Public License.
Weka comes with many features in data mining such as pre-processing, clustering, classification, regression, visualisation and feature selection. Techniques are based on the assumption that the data is available as a single flat file or relation, where each data point is a fixed number of attributes. There are some options to input data to Weka such as text file and SQL database through Java Database Connectivity

Weka has two main interfaces on both command line interface (CLI) and graphic user interface (GUI). The main graphic user interface (GUI) is the 'Explorer' [7]. Weka also provides an option to operate by command line through the component-based 'Knowledge Flow' interface. There are two additional interfaces 'Simple CLI' and 'Experimenter'. In this paper, the 'Explorer' was chosen as the mode to use in the experiment task.

There are five clustering algorithms provides in this program.

- Cobweb (Incremental Clustering)
- Simple K-means
- EM
- FastestFirst
- MakeDensityBasedClusterer

2.2 Data Corpus

INEX 2007's data corpus is a collection of Wikipedia documents in XML format. The documents in this collection consist of several types of content such as music, language, history and science, and some of which written in languages other than English. This corpus provides two sections of data for training and testing [2].
In mining process, train data is used in data mining program to estimates the parameters. When the model achieves good results, it can proceed to run the model on the test data.

The number of documents for testing is 48035 files classified in five categories

Corpus	
Train	***Test***
Train Collection	Test Collection
Train Categories	Test Categories

Table 1. *Presenting the details of 2007's corpus*

Detail of Corpus		
Train	**Test**	###############
48,306	48,305	Number of Documents
17,261,996	16,682,466	Number of Words
360 Mbytes	360 Mbytes	Size of the Corpus
470,293		Number of Distinct Words

Table 2. *Statistics of Training and Testing data in Corpus.*

3. Approach

3.1 Pre-processing Data

To cluster the XML documents based on its content, our approach assumes that the documents in the same cluster should have some words which are in common among them. Those words of interest should appear only or more frequency in the document in a specific cluster. They should not appear or rarely appear in the documents from other clusters. Then, if we assume again that the document in the collection can be divided equally into each cluster, so the term frequency or Tf of the words used to differentiate the document should be around N/C where N is the number of document in the collection and C is the number of categories which the document can be allocated to.

In practical, because of the variety of the document contents, even if they are in the same category, their content may differ from the use of synonyms. Moreover, the document in a different category may contain keyword of another category because of their detail covered in the document. Therefore, in this situation, even if there are, for example, n documents contain word t_i (Ft = n) and we know that, there are n documents in each category for this collection, but we still cannot conclude that those n document are in the same category.

In ideal case, if we select the words with Ft = N/C, there will be more than or equal to C words selected and they will cover all document because each word appears in the documents belong to a specific category only. However, in reality, according to the reason stated above, the coverage check has to be performed before using those words to clustering the collection. This check is to ensure that the selected words are covered all documents in the collection. If they do not cover all the documents in the collection, the selected words must increase by expanding the scope of Ft from Ft=N/C upward and downward (usually downward is to prevent interfering from the general used words) until all document are covered.
At this point, those selected words will be used to build a vector space for each document and ready to be input into clustering process.

3.2 Clustering

For clustering process, we decided to run our experiment by a software tool rather than implement it by ourselves because of the complexity of clustering algorithm and the limitation of time. The mining tool we chose is WEKA. As mention in section 2.1, WEKA has many advantages which make us comfortable to use it. In WEKA, there are five algorithms provided for clustering process, but, in this paper, we will use only four algorithms. Because, in the COBWEB algorithm, we cannot set the number of output cluster, so its result is difficult to analyse.

We use an Explorer mode of WEKA to run our experiments, because the ease of use. But, there is a disadvantage about amount of data input, because this mode loads everything into memory.

4. Result

To demonstrate the effect of document content to the range of F_t used for word selection, six groups of documents have been randomly selected from INEX

2007's corpus. These sample data consisted of 100 and 250 documents in 5 categories from 21 different categories provided by INEX 2007's train data Corpus. The result of the experiment is shown in Table 3.

Data Collection	Ft range	Distinct Words	Selected Words
100_1	17-28	9435	102
100_2	13-30	10818	210
100_3	14-21	10844	132
250_1	37-56	20243	143
250_2	11-53	18776	1088
250_3	14-53	18678	978

***Table 3.** Result of experiments to define ranges of F_t that satisfies coverage check.*

From the result, it shows that the number of distinct words in the collection (or, in the other word, size of the collection) do not affect the number of selected word. Indeed, the characteristic of the collection itself is the main factor which affects the width of F_t range and number of selected words. The evidence is the comparison of collection 100_2 and 250_1. Although, the size and number of distinct words of 250_1 is larger and the width of range is roughly equal, but the number of selected words need to cover all document of 250_1 is less than of 100_2. About the bad results of 250_2 and 250_3 collection, we found that a few documents in the collection are very short and contain very rare words, so they make the range wider. Those numbers of selected words are not acceptable to be the input of clustering, so we decide to cut those bad documents off and find the new acceptable range and selected words to be used in clustering those two collections.

In clustering process, we used the data collection and modified version of selected words from previous experiment as input into WEKA. For each algorithm, we set the expected cluster (numCluster) to five and vary the seed number to find out the best result. The outputs of these experiments are shown in Table 4.

Data collection	Ft range	Selected Words	Clustering Algorithms			
			EM	FastestFirst	MakeDensityBasedClusterer	Simple K-means
			% Incorrect (Seed)	% Incorrect (Seed)	% Incorrect (Inner Algorithm)	% Incorrect (Seed)
100_1	17-28	102	59 (36)	72(22)	59 (EM seed = 36)	64 (83)
100_2	13-30	210	66 (62)	76(*)	65 (S K-means seed = 31)	66 (31)
100_3	14-21	132	60 (51)	74(75)	59 (EM seed = 51)	69 (75)
250_1	37-56	143	52.0 (12)	78.4(37)	51.6 (EM seed = 12)	57.6 (18)
250_2	25-53	193	66.9355 (99)	71.7742(40)	67.7419 (EM seed = 99)	71.7742 (14)
250_3	28-53	145	46.371 (71)	76.6129(9)	45.9677 (EM seed = 71)	58.871 (85)

***Table 4.** Output of datasets from clustering algorithms*

It can be noticed that the result is better when the number of selected word need to cover the entire collection is lower. Among the primary algorithms (EM, MDBC, S K-means), EM gives the best result in clustering for all collections. Then, after applied the best algorithm and parameter setting to the MDBC, The clustering efficiency will increase a bit (approximately 1%).

0	1	2	3	4 ← assigned to cluster
3	16	0	27	4 | cat1
2	23	0	25	0 | cat2
2	0	35	4	9 | cat3
0	2	1	10	37 | cat4
0	10	1	36	1 | cat5

Figure 1. *The best result fromMakeDensityBasedClusterer algorithm*

When consider deeply into the cluster allocation of the best result, it can be noticed that the distribution of document is not balance among every cluster. A large portion of document allocates to cluster 3 which is a major factor of incorrectness in clustering. This may come from the effect of the very small size documents and general words which included in the selected words. But it tends to improve when the size of collection is bigger, so the behavior of bad document is outweighed.

5. Conclusion

We tested the approach by several random sets of data with clustering algorithms provided in Weka, and define the hypothesis that in the same categories of documents, they should have the similar set of terms. This experiment focused exclusively on the content of documents. Incorrectly clustered instances: 114.0 45.9677 %

From the experimental results, we can conclude that the main factor which affects the performance of clustering is the characteristic of the collection, i.e. mixed of very small size document, documents contain very rare word. But the performance may increase when the size of collection is grown up, because the effect of bad document may be neglected when appeared in very large collection.

To improve the efficiency of the clustering, the pre-processing step is the most importance. If the general words can be indentified and removed from the selected word, this should dramatically decrease the interference from this kind of word while clustering.

References

[1] Anderson UCLA Website (2007), “Data Mining What is Data Mining” , viewed 2 June 2008, http://www.anderson.ucla.edu/faculty/jason.frand/teacher/technologies/palace/datamining.html

[2] DMReview Website (2004), “The Future of Data Mining – Predictive Analytics”, viewed 30 June 2008, http://www.dmreview.com/issues/20040801/1007209-1.html

[3] G. Holmes; A. Donkin and I.H. Witten (1994). "Weka: A machine learning workbench" http://www.cs.waikato.ac.nz/~ml/publications/1999/99IHW-EF-LT-MH-GH-SJC-Tools-Java.pdf

[4] J. Zobel and A. Moffat (2006), "Inverted Files for Text Search Engines", ACM Computing Surveys, 38 (2) P. 1-56, 2006

[5] R. Nayak, T. Tran,"Document Clustering using Incremental and Pairwise Approaches", Pre-Proceedings of INEX 2007, P. 215-223

[6] XML Mining Track Home Page (2008), "XML Mining Track 2007", viewed 1 June 2008, http://xmlmining.lip6.fr/pmwiki.php?n=Main.2007

[7] The University of Waikato Home Page (2008), " Weka 3 – Data Mining with Open Source Machine Learning software in Java", viewed 28 May 2008, (http://www.cs.waikato.ac.nz/ml/weka/)