Naive clustering of a large XML document collection*

Antoine Doucet
Department of Computer Science
P.O. Box 26 (Teollisuuskatu 23)
00014 University of Helsinki
Finland
GREYC CNRS-UMR 6072
University of Caen, France
antoine.doucet@cs.helsinki.fi

Helena Ahonen-Myka
Department of Computer Science
P.O. Box 26 (Teollisuuskatu 23)
00014 University of Helsinki
Finland
helena.ahonen-myka@cs.helsinki.fi

ABSTRACT
In this paper, we address the problem of clustering a homogenous collection of text-centric XML documents. We present some experiments we have led on clustering the INEX[1] structured document collection. Our claim is that element tags provide additional information that must help improve the quality of clustering. We have implemented and experimented various ways to account for document structure, and used the well-known k-means algorithm to validate these principles.

Keywords
Document Clustering, XML, Information Retrieval

1. INTRODUCTION
Document clustering has been applied to information retrieval following the **cluster hypothesis**, which states that relevant documents tend to be highly similar to each other, and subsequently they tend to belong to the same clusters[3]. The theory behind this is that document clustering should permit to improve the effectiveness of an IRS by permitting to recall more of the relevant documents; Notably, in a best-match approach, some very relevant documents might receive a low rank simply because they miss one of the keywords of the query. Based on the cluster hypothesis however, these documents are to be clustered together with the best-ranked documents and can be found this way [1]. Document clustering can be performed prior to the query, in which case it is used to form a document taxonomy similar to that of the well-known "Yahoo" search engine. An alternative application of document clustering to IR is **post-retrieval clustering** [11], which is not performed on the whole document collection, but solely on the candidate subcollection retrieved in answer to a query. In this case, the clustering is used to ameliorate the quality of the final answer.

Nowadays, Internet is a repository for huge amounts of data. The quantity of XML data shared over the World Wide Web is increasing drastically. A large majority of this XML data is data-centric, but text-centric XML document collections are now getting more and more frequent. As a consequence, it became necessary to provide means to manage these collections. This can be done by automatically organizing very large collections into smaller subcollections, using document clustering techniques. Unfortunately, most of the research on structured document processing is still focused on data-centric XML (see for example [2] and [13]).

In this paper, based on the conjecture that "As structure is supplementary information to raw text, there must exist a way to use it, that improves the clustering quality", we present various naive approaches to represent text-centric XML documents (section 2) and experiment with them using a well-known partitional clustering algorithm. The results presented in section 3 are emphasizing the difficulty of this task and calling for discussion of the results and a description of the eventual directions of our future work (section 4).

2. PROCEDURE OF THE EXPERIMENTS
2.1 Document representation
As a representation of the documents, we have used the vector space model. In this representation, each document is represented by an N-dimensional vector, with N being the number of **document features** in the collection. In most approaches, the

*This work is supported by the Academy of Finland (project 50959; DoReMi - Document Management, Information Retrieval and Text Mining)
[1]Initiative for the evaluation of XML Retrieval (http://qmir.dcs.qmw.ac.uk/inex/)

features have been the most significant words of the collection. All the words are not selected as features, as the number of dimensions of the vector would easily place the computational efficiency at stake. For this reason, in the case of very large document collections, **feature selection** techniques are applied. We have used three different feature sets along our experiments: text features (i.e., words), tag features, and finally a combination of both.

- "Text features only": For the text feature set, as the size of the document collection is very large, we have used a few feature selection techniques. First, we have ignored words of less than three characters, and used a **stoplist** to delete longer words with a weak discriminative power (such as articles, pronouns, conjunctions and auxiliary verbs). We also pruned all words containing a numerical character. This simple heuristic diminished the feature set of about 50,000 word terms! The last step has been to **stem** the words, that is, to reduce them to a canonical form (for example, 'brought' 'bring' and 'brings' can be reduced to 'bring'), using the Porter algorithm[8]. The resulting set contained 188,417 features.
- "Tag features only": The clustering method we are willing to develop for clustering structured documents aims to be general. Therefore, we have made the choice to not manually group any tag labels. In practice, this means that all tag labels are distinct (e.g., 'ss1' and 'ss2' for sub-section of level 1 and 2 are distinct). The only preprocessing we made was to prune the closing tags, as we decided to account as much for 'complete' tags (with both a starting and an ending tag) as for the non-closed ones (e.g., 'art', 'entity', 'colspec'). Finally, we found 183 different tag features.
- "Text+tags": This last method combines both feature sets, by simply merging them. The total number of features is then 188,600.

The document vectors were then filled in with normalized tf-idf measures. Tf-idf combines term frequency (tf) [6] and inverted document frequency (idf) [4]. Term frequency is simply the number of occurrences of the feature words in a document. Its weakness is that it does not take the specificity of the terms into account. A term which is common to many documents is less useful than a term common to only a few documents. This is the motive for combining a term's frequency with its inverse document frequency, which is the division of the total number of documents in the collection by the total number of documents where this term occurs. In short, term frequency is a measure of the importance of a term in a document and inverted document frequency is a measure of its specificity within the collection.

2.2 Similarity measure

Clustering techniques group items based on their pairwise similarity. Thus, the first task is to find the right similarity measure. Following the vector space model, two measures are commonly used. The first one is the Euclidean distance, which has the advantage of being easily understandable. The other frequent measure is the cosine similarity. Its strength is very efficient computation for normalized vectors, since in that case $cosine(\vec{d_1}, \vec{d_2})$ simplifies to the dot product $(d_1 \cdot d_2)$. Because their results are very similar in nature, cosine similarity can be prefered to Euclidean distance (see for example [14]).

2.3 Clustering technique

There are two main families of clustering algorithms. Given n documents, hierarchical clustering produces a nested sequence of partitions, with a single cluster containing all documents at the top, and n singleton clusters at the bottom. This result can be displayed as a dendrogram (a subclass of the tree family). In partitional clustering, where **k-means** is the most common technique, the number k of desired clusters is either given as input, or determined as part of the process. The collection is initially partitioned into clusters whose quality is repeatedly optimized, until a stable solution is found.

In general, hierarchical clustering has been considered as the best quality clustering approach, and its quadratic complexity seen as its main weakness. For large documents, the linear time complexity (w.r.t. the number of documents) of partitional techniques has made them more popular. This is especially true for IR systems where the clustering is often aimed to improve the system's efficiency. Furthermore, Steinbach et al. [10] have made large scale experiments with numerous datasets and evaluation metrics which finally pointed out as a result that the cluster-quality of the bisecting k-means technique was at least as good as that of the hierachical approaches they tested. In these experiments, we have decided to use the k-means algorithm, both for its linear time complexity and the simplicity of its algorithm.

Given a number k of desired clusters, k-means techniques provide a one-level partitioning of the dataset in linear time ($O(n)$ or $O(n(log\ n))$ where n stands for the number of documents[12]). The *base* algorithm presented in figure 1 assumes the number of desired clusters be given and relies on the idea that documents are seen as data points.

1. *Initialisation*:
   - k points are chosen as initial centroids
   - Assign each point to the closest centroid
2. *Iterate*:
   - Compute the centroid of each cluster
   - Assign each point to the closest centroid
3. *Stop condition*:
   - As soon as the centroids are stable

Figure 1: Base k-means algorithm

2.4 Discussion on evaluation

2.4.1 Internal and External Quality.

There are two main families of quality measures. The **external** quality measures use an (external) manual classification of the document classification, whereas the **internal** quality measures do not. The principle of an external quality measure is to compare the clustering to existing testified classes. The better the clustering and the classification "match", the better the external quality measure evaluates the clustering.

In this work, we have used entropy and purity, two frequent external quality measures.

- The **entropy** is an information theoretic measure presented by Shannon [9]. It measures how the classes (manually tagged) are distributed within each cluster. This provides a quality evaluation for un-nested clusters (for hierarchical clustering, this means an entropy value can be computed only per level of the dendrogram). Note from the nature of entropy that its optimal score is obtained with singleton clusters and therefore entropy can hardly be used to compare clustering solutions of different sizes.

  The technique consists of first calculating the class distribution of the document collection, that is the number of documents in each class. The entropy of each cluster C is based on the probability that a document of C belongs to each class. The overall entropy is the average per cluster entropy weighted by the size of each cluster.

- The **purity** of a cluster measures how much that cluster is "specialised" in a class. It is simply its largest class divided by its size. The overall purity of a clustering solution is then a weighted average of the purity of each of its individual clusters.

- There exist many more measures. For example, the well-known IR F-measure has been adapted to clustering [5]. We did not use it, however, as it is by definition adapted for the case where the evaluation classes are query answers (this evaluation method was used with various TREC collections and their assessment results).

Internal quality measures are used when no manual classification is provided. They are computed by calculating average inter- and intra-cluster similarities. An example of an internal quality measure is **cohesiveness** (a.k.a. "overall similarity"), which is defined for each cluster as the average similarity between each two documents of that cluster.

2.4.2 The INEX case

Our experiments compare the use of different feature sets. As such, they result in different pairwise document similarity values. Thus, it is clear that estimating the feature sets based on inherent internal quality measures would not make any sense. Therefore, we must use external quality measures. Nevertheless, any external quality measure relies on an existing manual classification, and at the time of the experiments, the only classification existing for the INEX collection were the year and journal volume in which an article was published. For more consistency, we have used the journals as our classes. We have also made another class of the 125 volume descriptions, which contain a listing of the articles published in the corresponding volume.

Unfortunately, this classification has a number of problems. The main issue is that these classes form a partition of the document collection, that is, the classes are disjoint. This property is rather inappropriate for document collections, as there exist no such strict border between two articles as there may be with other data types. The fact that an article was published in a given journal rarely means that it could not have been published in another one. Hence, the journal title classification is probably too strict.

In fact, a good classification for evaluating document clustering is typically a manual assessment of the answers to a set of queries. By using the topics of an IR evaluation initiative (e.g., TREC or INEX) as classes, and the corresponding documents as the elements of the class, researchers have often found a satisfying way to evaluate the quality of clustering methods. These classes offer a more trustable human-expert classification, that furthermore allows a document to belong to many classes or none. Therefore, we plan in further work to use the manual assessments of the INEX evaluation,

originally aimed at information retrieval systems, so as to evaluate the relevance consistency of documents clusters.

Finally, the clusterings have been evaluated according to the 18 journals where the documents were published, plus the additional volume class. The 12,232 documents of the INEX collection have thus been mapped to 19 classes.

Of course, in order to keep the experiments fair, we pruned all document elements containing the name of the journal where the document was published. In practice, this means the elements <doi>, <fno> and <hdr> and their content were ignored.

3. RESULTS

We have implemented and experimented the techniques described above on the INEX collection, using the publicly available clustering tool implemented by George Karypis, University of Minnesota[2].

We have run k-means with k $\in$ {5, 10, 15, 20, 25, 35} for text-only, tags-only and tags&text. We have then computed entropy and purity using the journal titles as classes.

The results of our experiments for 5, 15, 20 and 35 clusters are shown respectively in tables 1,2,3, and 4. The runs were computed on a 1333 Mhz desktop with 1 gigabyte of memory.

Table 1: Results of k-way clustering for k=5

Features	Text	Tags	Text + Tags
Entropy	0.711	0.836	0.812
Purity	0.301	0.211	0.216
Clustering Time	150s.	4s.	160s.

Table 2: Results of k-way clustering for k=15

Features	Text	Tags	Text + Tags
Entropy	0.633	0.798	0.678
Purity	0.379	0.228	0.372
Clustering Time	754s.	11s.	837s.

Table 3: Results of k-way clustering for k=20

Features	Text	Tags	Text + Tags
Entropy	0.598	0.775	0.677
Purity	0.413	0.237	0.332
Clustering Time	1101s.	15s.	1191s.

3.1 Including all tags decreases quality!

A clear observation is that, for any desired number of clusters k, the best quality is obtained with the text features. Tag features as a stand-alone perform much worse, and when they are combined to the text features, the worsening is just averaged.

[2]CLUTO, http://www-users.cs.umn.edu/~karypis/cluto/

Table 4: Results of k-way clustering for k=35

Features	Text	Tags	Text + Tags
Entropy	0.568	0.758	0.612
Purity	0.454	0.254	0.385
Clustering Time	2016s.	25s.	2215s.

However, one may expect that adding an extra piece of information about documents would improve their description, and subsequently their pairwise similarity measures, and should finally result in a better clustering quality.

There are various reasons for this quality worsening following the inclusion of tag features. First, the quality evaluation issue must be recalled; For example, using the tag features, a few clusters have terms like 'tmath' or 'math' as their most descriptive features. They mainly gather articles from the journals "Transactions on Computers" and "Transactions on Parallel & Distributed Systems". This predominance of two different classes implies low external quality measures. It is however impossible to claim as a consequence that those clusters are not valuable. On the other hand, some clusters are doubtlessly negatively affected by the addition of the tag features. Document clusters dominated by style features (e.g., 'b' or 'tt') are rather grouping documents based on their authors' writing style. As an illustration, those clusters are almost equally distributed amid the classes.

3.2 "Tags only" permits very fast clustering

The clustering based solely on tag features is computed much faster. This is no surprise as the number of items is then 183, when it is 188,417 for text only. What is surprising is how good the tags-only results are, considering that the whole process runs in seconds on a standard desktop.

In applications involving a huge number of documents, and requiring fast clustering (e.g., "prior to query" document clustering for IR), the trade-off between quality and efficiency may advantage the tags-only option.

It is however difficult to tell, besides the raw quality scores, how well the tag features clustering are matching the "text only" clustering. A good question to ask is how close are these clustering solutions ? We know that the tags-only clustering performs reasonably well wih respect to its computational ef-

Table 5: Results of k-way clustering for the 'volume' cluster with k=15

Features	Text	Tags	Text + Tags
Precision	28%	99%	100%
Recall	100%	100%	100%
Entropy	0.722	0.016	0
Purity	0.295	0.992	1
Internal Similarity	0.094	0.900	0.912
Clustering Time	754s.	11s.	837s.

ficiency, but how close is this good answer to the better answer issued from the "text-only" clustering ? Unfortunately, this is still in the list of future work!

3.3 Exception: the 'volume' class

For each clustering, most of the 125 volume.xml files, compiling entity references to the articles of a given volume of a journal, are found within the same cluster. In contradiction with the general observation that text features give higher quality clustering, we have found that for this specific cluster, "text+tags" and "tags only" give the best performance. In table 5, we have computed values for recall and precision for this 'volume' cluster. Precision is the number of 'volume' documents found in the volume cluster (true positives), divided by the size of that cluster. Recall is the number of 'volume' documents found in the volume cluster, divided by the total number of volume documents (i.e., 125).

This result is due to the very specific structure of the volume files. They contain the list of the titles of all articles published in the corresponding journal volume. This type of documents totally misleads the text features approach, as in this case the most specific features are not article titles, but various publishing details (month of publication for example). We have computed the most descriptive terms for the volume cluster, for each of the three feature sets in table 6. The descriptivity measure for a feature within a cluster is the percentage of internal similarity that is due to this particular feature. When the feature set contains element labels (i.e., tags), they tend to dominate the text features, as a consequence of a more discriminant distribution.

3.4 Best clustering method, with respect to journal title classes

Following these results, we foresaw a better clustering method, based on the principle to use the tag features clustering as a preprocessing. The idea is to pre-detect those documents which are structurally different. This is harmless from a computational point of view, as the tag features are so few, and since their extraction is done in linear time.

Even though the "text+tags" performs slightly better, the efficiency/quality trade-off obviously plaids for prefering the tag-feature clustering as the preprocessing for the simple clustering method we present right below.

- Step 1: k-means clustering of the full document collection based on the "tags-only" representation. The n clusters with an average internal similarity above a threshold (say 0.9) are kept.
- Step 2: A $(k-n)$-means clustering is then led, based on the remaining documents (those that do not belong to preselected clusters).

With the INEX collection and k=15, only the volume cluster is preselected. The results are shown in table 7 and confirm the clustering quality improvement. However, we are aware that such a method can not be claimed to be superior, before further experiments are made (particularly using different collections).

Table 7: Text features based clustering with and without tag features pre-clustering, for k=15

	Text features	Same, but with pre-clustering
Entropy	0.633	0.630
Purity	0.379	0.394
Time	754s.	11+742s.

Anyhow, we believe that the general idea to use structure-based clustering as a preprocessing of standard clustering must permit to improve the clustering quality. But to extend the application of this principle to the general case, we are willing to consider more general and sophisticated structural similarity measures. A recent work has notably provided an edit tree distance between the structure trees of XML documents [7].

4. DISCUSSION AND CONCLUSION

We adressed the problem of clustering homogenous structured document collections. We experimented a common partitional clustering algorithm with various sets of features. As the current evaluation system is not yet reliable, the results we found can not be considered as definitive, but should rather be seen as hints.

Our results have then hinted that simply adding tag labels to the feature set does not improve the clustering quality. However, our conjecture that a way to exploit the structure exists is still standing. What the first results emphasize is that the solution is not straightforward, and that combining

Table 6: 3 most descriptive features within the 'volume' cluster for k=15

Text only	january (19%)	society (13%)	publish (6%)
Tags only	<entity>(63%)	<title>(20%)	<sec1>(14%)
Text+tags	<entity>(63%)	<title>(20%)	<sec1>(15%)

structural similarities to content similarity indeed permits to improve the clustering quality.

It seems like computing tf-idf measures of tag labels is insufficient, and we are now considering more sophisticated measures for structural similarity between documents. Instead of using the frequency of the elements, options are to weight the documents with the total size of the elements (or with their average size). It would still remain to be decided whether the size of an element should be defined locally or as the total size of its sub-elements, in which case normalization issues would emerge.

So far, this work has been difficult due to the lack of a very large text-centric structured document collection. The INEX initiative has already provided such a collection, but meaningful classes were not yet available at the time of our experiments. The manual assessments of the INEX topics will permit us to further evaluate the various structured document clustering approaches. In this regard, we will also need more document collections, so as to make sure that the results we get are not "statistical accidents", due to specificities of the INEX collection.

There are various possible research directions. One is to develop feature selection methods for tag labels. Some simple ways might be to replace words by their full path expressions, or by their local path expressions. It would also make sense to develop ways to detect different classes of tag labels. The distinct nature of some of these classes would then call for different processing techniques. It is clear, for example, that the tags 'tfmath', 'sgmlmath', and 'math' have much in common and that they may probably be merged to a single 'meta-math' class. For the least, we must try to account for the fact that 'tfmath' and 'sgmlmath' are more similar than 'sgmlmath' and 'ss1' (subsection of level 1).

Another interesting problem emerges, following the work in the INEX initiative: multi-level clustering. The idea is to compute representations of document sub-elements together with the documents, and give as a result clusters containing items of different granularities. This idea is clearly derived from the IR problem posed by INEX, of retrieving the best matching elements, rather than full documents exclusively.

5. REFERENCES

[1] B. Croft. *Organizing and searching large files of document descriptions.* PhD thesis, University of Cambridge, 1978.

[2] D. Guillaume and F. Murtaugh. Clustering of XML Documents. *Computer Physics Communications*, 127:215–227, 2000.

[3] N. Jardine and C. van Rijsbergen. The use of hierarchic clustering in information retrieval. *Information Storage and Retrieval*, 7:217–240, 1971.

[4] K. S. Jones. A Statistical Interpretation of Term Specificity and Its Application in Retrieval. *Journal of Documentation*, 28(1):11–21, 1972.

[5] B. Larsen and C. Aone. Fast and Effective Text Mining Using Linear-time Document Clustering. In *Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, California*, pages 16–22, 1999.

[6] H. P. Luhn. A statistical approach to mechanical encoding and searching of literary information. *IBM Journal of Research and Development*, 1(4):309–317, 1957.

[7] A. Nierman and H. Jagadish. Evaluating Structural Similarity in XML. In *Fifth International Workshop on the Web and Databases (WebDB 2002), Madison, Wisconsin*, 2002.

[8] M. Porter. An algorithm for suffix stripping. *Program*, 14:130–137, 1980.

[9] C. E. Shannon. A mathematical theory of communication. *Bell System Tech*, 27:379–423, 623–656, 1948.

[10] M. Steinbach, G. Karypis, and V. Kumar. A Comparison of Document Clustering Techniques. In *Proceedings of KDD 2000, Workshop on Text Mining*, 2000.

[11] A. Tombros. *The effectiveness of hierarchic query-based clustering of documents for information retrieval.* PhD thesis, University of Glasgow, 2002.

[12] P. Willett. Recent trends in hierarchic document clustering: a critical review. *In Information Processing and Management*, 24(5):577–597, 1988.

[13] J. Yi and N. Sundaresan. A classifier for semi-structured documents. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, Boston, Massachusetts*, pages 340–344, 2000.

[14] Y. Zhao and G. Karypis. Criterion functions for document clustering. Technical report, Department of Computer Science and Engineering, University of Minnesota Twin Cities, 2001.