The University of Kansas


Technical Report

# XML Classification Using Content and Structure

Swathy Giri, Aravind Chandramouli, and
Susan Gauch

ITTC-FY2007-TR-31020-02

April 2007

Project Sponsor:
National Science Foundation
ITR0225676
(Science Environment for Ecological Knowledge)

# XML Classification Using Content and Structure

Swathy Giri, Aravind Chandramouli & Susan Gauch
University of Kansas
Lawrence, KS 66044
aravindc@ku.edu

**Abstract**

The number of XML documents on the Web has seen a phenomenal increase in the past few years. However, most existing efforts for XML document classification employ standard information retrieval methods based on the document contents alone or the document structure alone. In this paper, we present a system that classifies XML documents based on both their content and structure. In particular, we classify XML documents by a weighted combination of field-wise content similarities and show that this approach outperforms classification that ignores the structure. We also show that weighting the fields differentially outperforms an approach in which each field contributes equally to the classification process. We then present an algorithm that learns the relative importance of each field in order to automatically infer appropriate field weights.

## 1. Introduction

The eXtensible Markup Language (XML) is becoming one of the most convenient ways to represent and transport data on the World Wide Web (WWW). Although most of the documents currently on the Web are in HTML, there has been a migration towards XML for data representation. As the amount of information available in XML format grows, we will need ways to organize that information for efficient and effective browsing, search, and analysis.

A key component of many information processing applications is text classification, the assignment of text documents to previously known categories or classes. In this paper, we describe our system for XML document classification that uses an algorithm that selects weights for fields in an XML document for classification *a priori.* We will also show that this algorithm works significantly better than systems that classify XML documents based on their contents alone, disregarding the document structure.

## 2. Related Work

There has been a tremendous amount of research in text classification over the years. The various approaches differ in how the categories and documents are represented, how the features are extracted and weighted, and how the similarity between the documents and categories is calculated. A wide range of algorithms have been used for text classification (Dasarathy, 1991; Rocchio, 1971; Vapnik, 2000). A simple approach to classification of XML documents will be to strip all the tags in the XML document and apply any of the text classification algorithms to the resulting document. In contrast to flat classifiers, hierarchical text classification (Sun et al., 2003; Wang et al., 2001) exploits the structural relationship between the categories. Hierarchical classification using document contents has been applied to XML documents (Fuhr & Weikum, 2002). In contrast to algorithms that use the content, XML classification algorithms have been developed that make use of the structure of the document alone (Zaki & Aggarwal, 2003). Some XML classification systems make use of both document content and structure (Theobald et al., 2003). Theobald et al. (2003) use expanded textual features extracted from the document

contents as well as features describing the XML paths for the document fields. They show increased accuracy when these features are used together. However, they do not explore technologies to determine which feature combination will be more accurate, rather, they try all combinations and hence, efficiency is a concern. Recent work has focused on the transformation of XML document structure into different formats for classification (Candillier et al., 2005). Our work is most similar to (Theobald et al., 2003) in that we also use weighted combinations of the fields. However, we go further in that we learn a weighting scheme that can be used for different schemas and is efficient.

## 3. XML Classifier

In order to provide high efficiency, the XML classifier is built from a standard Rocchio classifier (Rocchio, 1971) developed as part of the KeyConcept project (Gauch et al., 2004). The Rocchio classifier has been extended to classify XML documents using all or some fields or ignoring the XML structure altogether.

Current XML classification systems either ignore the structure of the document or consider all the fields in the document to be equally important. We hypothesize that some fields are more important than others and weigh fields differently. The key idea behind this approach is that we might obtain higher classifier accuracy when XML documents are classified by considering the content of some fields more than the others. As a baseline system, the complete XML documents are classified using the unmodified Rocchio classifier by removing all tags and ignoring the document structure. We then extend the classifier so that the XML documents are split during the training phase into $N$ sub-documents, where $N$ is the number of fields in a document, in which each of the sub-documents contains the content of a single field extracted from the original XML document. The training phase then trains a classifier for each field separately from the sub-documents. The classifiers trained in this phase are then used to classify documents in the classification phase, after they go through the same splitting process as the training documents and return a weighted list of categories for each field. The field weights for each category are calculated using the formula given in Eq. 1.

$$\text{Combined\_weight}_i = \sum_i W_i * \text{field\_weight}_i \qquad (1)$$

where $W_i$ denote weight given to field $i$.

## 4 Experimental Procedure

### 4.1 Data Sets

We created two data sets, each containing 160 XML documents. One of these sets is used for experimentation and the other for validation purposes. The documents in each set belong to four different categories, with forty documents per category. 30 documents per category randomly selected have been used for training and the remaining 10 documents from each category have been used for testing.

Data set 1 (DS1) contains news articles collected from two Websites, www.bbc.com and www.rediff.com. News, Business, Science and Health are the categories from which we downloaded between January 2004 to March 2004. These documents were then manually annotated with the tags from the schema that were appropriate for the documents.

Data set 2 (DS2) contains information about four different categories of companies on the WWW, i.e., Hardware, Technology, Advertising and Marketing, and Cosmetics. As in the case of DS1, the Web pages collected about companies in these categories were manually annotated with XML tags. We used DS2 for validation of the algorithm we developed based upon experiments with DS1.

### 4.2 Evaluation Metric

We evaluated the classifier accuracy by comparing the classifier results for each test document with 'truth'. The classifier produces a list of category id's and weights, sorted in decreasing order of weights. The evaluation algorithm compares the truth value for each document to the classifier result and the evaluation metric calculates the percentage of test documents for which the classifier has the truth value as the top match and in 2nd, 3rd and 4th position. These values are represented cumulatively, reporting 100% for the 4th place (assuming all documents are classified).

### 4.3 Evaluation experiment with DS 1

The goal of this experiment was to evaluate the effect of different XML fields on classification accuracy. Hence, the baseline for our experiments was a classifier trained and tested on the full text documents that ignored the XML markup altogether. We performed a detailed analysis of the content of fields in DS1 to identify characteristics of different fields that affected classification results and designed an algorithm based on the characteristics.

**Setting up a baseline**

The classifier as trained using the training documents from DS1 (30 per category) after which test documents from DS1 (10 per category) were classified. Full-text documents were used for training and classification for the baseline. Table 1 summarizes the results.

| Fields | ALL | Individual Fields | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | date | copyright | creator | link | title | language | description | details |
| % matches at #1 | 90 | 25 | 25 | 27.5 | 27.5 | 55 | 65 | 70 | 92.5 |
| % matches at #2 | 97.5 | 65 | 52.5 | 50 | 52.5 | 72.5 | 65 | 90 | 97.5 |
| %matches at # 3 | 97.5 | 77.5 | 72.5 | 75 | 72.5 | 77.5 | 67.5 | 97.5 | 97.5 |
| %matches at # 4 | 100 | 100 | 97.5 | 100 | 97.5 | 80 | 100 | 100 | 100 |

**Table 1: Classification Accuracy on DS 1**

## Discussion

The baseline system produced a classification accuracy of 90%. For classification performed with individual fields, the details field, which contains the most tokens, produced the highest

classification accuracy of 92.5%. Note that fields such as date that contain dates and/or numbers did not perform well. Also, fields such as copyright and creator that contain almost identical content in each document also produce low classification accuracy. Hence, we can identify the following characteristics for identifying useful fields for classification: fields that have a large number of tokens and fields with higher variability in their content across documents should be highly weighted, while fields that primarily have dates and/or numbers in their content should contribute little to the classification decision.

### 4.4 Classification Algorithm

We next wanted to design an algorithm that could automatically determine field weights for an XML document based on features extracted from the field contents. In order to identify features correlated with the highly-performing fields shown in Table 1, we analyzed the content of fields in DS1 (see Table 2) using three features: the number of tokens, variability in content, and the percentage of numbers.

| Characteristics | **Date** | **copyright** | **creator** | **link** | **title** | **language** | **description** | **details** |
|---|---|---|---|---|---|---|---|---|
| # of tokens | 242 | 26 | 241 | 311 | 1554 | 5496 | 6319 | 135929 |
| Normalized Score # of tokens | 0.0016 | 0.0002 | 0.0016 | 0.0021 | 0.0104 | 0.0366 | 0.0421 | 0.9055 |
| Variability | 0.128 | 0.077 | 0.008 | 0.013 | 0.714 | 0.075 | 0.378 | 0.172 |
| Non-numbers | 152 | 26 | 241 | 311 | 1544 | 4496 | 6280 | 13522 |
| **# tokens Score** | 0.01 | 0.00 | 0.01 | 0.02 | 0.08 | 0.29 | 0.34 | 7.24 |
| **Variability Score** | 0.654 | 0.393 | 0.002 | 0.006 | 3.650 | 0.383 | 1.932 | 0.879 |
| **Non numbers Score** | 0.676 | 1.077 | 1.077 | 1.077 | 1.069 | 0.881 | 1.069 | 1.070 |
| **Total Score** | 1.35 | 1.47 | 1.099 | 1.123 | 4.879 | 1.844 | 3.681 | 16.429 |
| **Weights** | 0 | 0 | 0 | 0 | **0.2** | 0 | 0 | **0.8** |

**Table 2: Analysis of characteristics of fields in DS1**

We determined the characteristics by using the following formulae:
# of tokens in $T_i$ = Total number of words in field $T_i$ for all documents

$$\text{Variability for } T_i = \frac{\text{Number of unique words in } T_i \text{ for all documents}}{\text{\# of tokens in } T_i}$$

Non-numbers in $T_i$ =Total number of non-numbers in $T_i$

Once we obtained these values for all the fields in the collection, we assigned a score to each characteristic for every field in the range of 1-8 (the number of tags). The scores were assigned as follows:
**#** of tokens score **-** We calculate the % of tokens that occur in a given field and multiply that by the number of fields.

$$\% \text{ of tokens } T_i = \frac{\text{\# of tokens for field } T_i}{\sum_{i=1}^{n} \text{\# of tokens } T_i}$$

After we obtained the % of token score, we multiplied it by the number of fields in DS1, to obtain the overall score for # of tokens, that is, # of tokens score for $T_i$ = % of tokens $T_i$ * 8.

Variability score and non-numbers score is obtained similarly. Total score is calculated using the following formula.

Total Score = 2 * # of tokens score + Variability score + non numbers score.

Since the results from experiment 1(Table 1) on DS1 showed that the best results occurred when the details field was used for classification and details field has the maximum number of tokens, we have weighted the # of tokens score 2 times more than the other fields. The threshold value is the average of all the scores. We consider the tags above the threshold. Weights are assigned to the tags above the threshold using the following formula.

$$\text{Weight for a tag} = \frac{\text{Score of the tag}}{\text{Sum of scores of tags above threshold}}$$

Using the above formula, we calculated the weights for the tags as shown in Table 2. The Details tag was given a weight of 0.8 and Title tag was given a weight of 0.2. All other tags were ignored. The combination predicted by our algorithm yielded an accuracy of 92.5 %. Surprisingly, this is the same accuracy as that when the Details field is used alone, which ties for highest accuracy (and is more accurate then the 90% produced when the tag structure is ignored and the 82.5% when all tags are used and weighted equally).

### 4.5 Validating the Classification Algorithm

To see how well our automatically extracted features and learned field weights performed on a completely different XML schema and document collection, we validated the algorithm described in Section 4.4 using DS 2. The detailed analysis of the characteristics of DS2 is shown in Table 3. The full-text baseline system using DS2 was set up using the same technique described for DS1. The combination produced by our algorithm yielded an accuracy of 75% whereas our full-text baseline system yielded an accuracy of 65% for DS2 and weighting all fields equally produced an accuracy of 52.5%. Thus we achieve an improvement of 15.3% (10% absolute) in our XML classifier over the full text classifier.

| Characteristics | **name** | **url** | **hq location** | **br location** | **product** | **service** | **date visited** | **creator** | **hq phone** | **br phone** |
|---|---|---|---|---|---|---|---|---|---|---|
| # of tokens | 385 | 550 | 914 | 99 | 2134 | 759 | 360 | 240 | 338 | 34 |
| Normalized Score # of tokens | 0.0662 | 0.0946 | 0.1572 | 0.0170 | 0.3671 | 0.1306 | 0.0619 | 0.0413 | 0.0581 | 0.0058 |
| Variability | 0.618 | 0.305 | 0.639 | 0.919 | 0.568 | 0.623 | 0.058 | 0.008 | 0.763 | 1.000 |
| Non-numbers | 384 | 549 | 689 | 80 | 2124 | 758 | 200 | 240 | 3 | 0 |
| **# tokens Score** | 0.53 | 0.76 | 1.26 | 0.14 | 2.94 | 1.04 | 0.50 | 0.33 | 0.47 | 0.05 |
| **Variability Score** | 1.123 | 0.554 | 1.161 | 1.671 | 1.032 | 1.132 | 0.105 | 0.104 | 1.387 | 1.818 |
| **Non-numbers Score** | 0.763 | 1.092 | 1.371 | 0.159 | 4.225 | 1.508 | 0.398 | 0.477 | 0.006 | 0 |
| **Total Score** | 2.946 | 3.166 | 5.052 | 2.11 | 11.137 | 4.72 | 1.503 | 1.241 | 2.333 | 1.918 |
| **Weights** | 0 | 0 | **0.24** | 0 | **0.53** | **0.23** | 0 | 0 | 0 | 0 |

**Table 3: Analysis of Characteristics of fields of DS 2**

## 5. Conclusion and Future Work

We have presented the idea of classification of XML documents based on the fields and its contents and have shown that, due to the characteristics of the content of fields, some can be used to improve classification over that achievable with full-text. We identified characteristics of the most useful fields and developed an algorithm to predict useful fields and their weights for new XML data sets. We have shown an improvement in classification accuracy on a previously unseen data-set using this algorithm when compared to a bag-of-words approach.

The system works on a single XML schema at a time and we are currently extending this to work on collection of documents from multiple schemas. One possible approach is to normalize all the participating schemas to a generic schema and then perform classification. Automating the field selection is another improvement that can be made to this system. We presented our preliminary results on a small data set. A larger data set and comparison with other XML classification systems form part of the current/future work.

## 6. Acknowledgements

This work was partially supported by NSF ITR 0225676 (SEEK).

## References

Candillier, L., Tellier, I. & Torre, F. (2005) Transforming XML Trees for efficient classification and clustering. In *INEX 2005 Workshop on Mining XML documents*, (pp. 469--480), November.

Dasarathy, B.V. (1991) *Nearest Neighbor (NN) norms: NN Pattern Classification Techniques*. Las Alamitos, California: IEEE Computer Society Press.

Fuhr, N. & Weikum, G. (2002) Classification and Intelligent Search on Information in XML. *Bulletin of the IEEE Technical Committee on Data Engineering*, 25(1), 51--58.

Gauch ,S., Madrid, J.M., Induri, S., Ravindran, D. & Chadalavada, S (2004). KeyConcept: A Conceptual Search Engine, Information and Telecommunication Technology Center. Technical Report: ITTC-FY2004-TR-8646-37, University of Kansas.

Rocchio, J. (1971) Relevant feedback in information retrieval. In Salton, G (Ed.), *The smart retrieval system - experiments in automatic document processing*. NJ: Englewood Cliffs.

Sun, A., Lim, E. & Ng, W. (2003) Performance Measurement Framework for Hierarchical Text Classification. *Journal of the American Society for Information Science and Technology*, 54(11), 1014--1028.

Theobald, M., Schenkel, R. & Weikum, G. (2003) Exploiting Structure, Annotation, and Ontological Knowledge for Automatic Classification of XML Data. In *WebDB Workshop*,1--6.

Vapnik, V. (2000). The Nature of Statistical Learning Theory. (2nd ed.): Springer New York.

Wang, K., Zhou, S. & He, Y. (2001). Hierarchical classification of real life documents. In *Proceedings of the 1st SIAM International Conference on Data Mining*, Chicago, US.

Zaki, M.J. & Aggarwal, C. (2003). XRULES: An Effective Structural Classifier for XML Data. In *9th International Conference on Knowledge Discovery and Data Mining*, 316--325, Washington,DC, August.