International Journal of Computer Trends and Technology- volume4Issue3- 2013

An Efficient Classification Approach for the XML Documents

Navya sree.Yarramsetti[#1] , G.Siva Nageswara Rao[#2]

[#1]M.Tech(CSE) Student
Department of CSE, K L University
Guntur, Andhra Pradesh, India

[#2] Associate Professor
Department of CSE, K L University
Guntur, Andhra Pradesh, India

***Abstract*- Extensible Markup Language (XML) has been used as standard format for a data representation over the internet. An XML document is usually organized by a set of textual data according to a predefined logical structure. Due to the presence of inherent structure in the XML documents, conventional text classification methods cannot be used to classify XML documents directly. In this paper, we propose the learning issues with XML documents from three major research areas. First, a knowledge representation method, which is based on typed higher order logic formalism. Here, the main focus is how to represent an XML document using higher order logic terms where both its contents and structures are captured. Second-symbolic machine learning. Here, a new decision-tree learning algorithm determined by precision/recall breakeven point (PRDT) for the XML document classification problem. Precision/recall heuristic is considered in xml document classification is that the xml documents have strong connections with text documents. Finally, we had a semi-supervised learning algorithm which is based on the PRDT algorithm and the co-training framework. By producing comprehensible theories, the tentative results exhibit that our framework is capable to attain good performance in both the machine learning techniques.**

Keywords: precision/recall, Co-training, machine learning, knowledge representation, semi-supervised learning.

1. INTRODUCTION:

Along with dramatic growth of xml documents on the internet, it becomes more and more challenging for users to retrieve their desirable information. The demand of efficient search tools has led to extensive research in the area of information retrieval. XML documents are semistructured, including the logical structure and textual data. Structural information plays an important role in information retrieval.

Classifying XML documents can basically be done in three ways:
(1) using exclusively the textual contents of XML documents as usually done in traditional text categorization systems,
(2) using exclusively the structure of XML documents, and
(3) using both, the contents, and the structure, in a hybrid manner.

First, Knowledge representation (KR) is usually the first step toward intelligent reasoning. Generally speaking, KR involves using some formal symbols to represent the knowledge around us. The knowledge representation must be faithful to the original information in order to allow the system to capture all the information. In the case of XML, which has a semistructured data model, the KR method must be able to represent both the structure and the content. However, traditional inductive learning uses an attribute-value representation method. An individual is represented as a sequence of attributes. Though the relative simplicity of this method allows the implementation of efficient learning systems, often data may contain complex structures that cannot be captured by the values of a predetermined set of attributes. This would typically be the case for XML documents.

Second, in many cases comprehensible learning theories are traded-off for high accurate but incomprehensible theories. The advantage of comprehensibility of the output learning theory has been recognized by machine learning researchers. A comprehensible theory could provide insight to an expert. For example, through an understandable theory medical researchers could find interesting clues to a disease from a medical database; social scientists could understand the courses of some social behaviour from a social science data set; Businessmen could discover business critical or security critical knowledge from a financial data set. A human-understandable output theory could also enhance the user's trust of the intelligent system. For example, a user may want to understand the predictions made by his email classification system in order to trust its behaviour. Symbolic machine learning algorithms have the power to output human comprehensible theories. The decision-tree algorithm that is used in this paper is a type of symbolic learning algorithm.

Third, learning with minimal labeled training examples is desirable for web intelligence. Supervised

learning often requires a large number of labeled examples in order to learn accurately. However, usually it is the user who labels the training examples for the web agent systems. In contrast, unlabeled examples are usually available in large quantities and easily obtained for web intelligence. Using a few labelled examples and a large number of unlabeled examples as training examples is of important practical significance to web learning. This type of machine learning is called Semi supervised learning. The key point is that unlabeled data provides information about joint probability distribution of features.

2. RELATED WORK:

The related work is reviewed from three aspects: knowledge representation of XML documents, learning with XML documents, and semi-supervised learning.

With the increase in XML documents, researchers are exploring many methods for their classification. While some approaches extend the traditional information retrieval methods, some other are based on tree edit distance method where XML documents are represented as labeled trees and the distance between the documents is defined as the edit distance between the labeled trees. Some of the research works on classification on XML documents are briefly summarized. The approach proposed by Abdel hamid discovers the structural and content characteristics shared by XML documents of the same class. This approach based on k nearest neighborhood algorithm relies on edit distance measures which consider both the content and structure of XML trees with structure bearing more weight than content. A bottom up approach for XML classification gives more weights to content of the XML documents. It is a similarity based method which begins with the content represented as leaf nodes and then the structural information is embedded. A methodology for indexing and retrieval from XML document is proposed where the structure information is represented using attributes so that the structure of the document can be expanded further. During indexing or retrieval, the attributes are converted into elements for which the existing systems can be used and indexing is made hierarchically. During classification, the rules relevant to a test document are identified and the statistics of the matched rules are used to predict the category of that document.

Content and structure is presented. Here, XML documents are classified by a weighted combination of field wise content similarities. Here the algorithm automatically determines the field weights for an XML document based on features extracted from the field contents. The characteristics used for identifying useful fields for classification are the fields that have a large number of tokens and the fields with higher variability in their content across documents. This technology takes full advantage of the rich information and semantics embedded in XML documents. A report on the XML mining Track at INEX 2005 and 2006 discusses the nine different models used for clustering and classification of XML documents using structure only and structure and content. This report highlights the fact that the structure only task is quite easy among other methods and simple models work very well with this task. In the formal model based on Bayesian classification developed by P.F. Marteau and etal, a structural context of occurrence for unstructured data is defined and a recursive formulation is derived in which parameters are used to weigh the contribution of structural element relatively to the others. The tree structure of the XML document is approximated as a set of nodes from which the path to the root is attached.

3. KNOWLEDGE REPRESENTATION FOR XML DOCUMENTS

In this section, we describe how to represent XML documents using the higher order logic formalism. XML documents have nesting structures which could be complex. Structures add more information to the contents of an XML document. The nesting structures of an XML document provide a way of information aggression and correlation. Both structures and contents play important roles in XML document classification. This requires that the knowledge representation formalism must be able to capture both structures and contents of the data. The higher-order logic is used in two essential ways here. First, individuals are represented as higher-order logic terms; second, predicates are constructed by composing transformations.

3.1 The structure of an XML document

An XML document has a well-nested structure. Each entity is enclosed by a start tag and an end tag. This structure is best described by an example. A simple but complete XML document is given in Figure 1. The first line of the document is the XML version declaration. The second line is the declaration of its DTD (Document Type Definition). Next comes the root element *bib* which has two subelements *book* describing information about two books. The first *book* has four further subelements *title*, *author*, *publisher* and *price*, which give the basic information about this book. The first *book* also has an attribute *year* with a value of *2003*. The *author* further contains two subelements *last* and *first* which contains the last name and the first name of the author, respectively. The second *book* element is similar to the first one, but contains three element authors.

3.2 Representation of individuals

A well-formed XML document is represented as a six tuple.

type XML = XMLDecl *Misclist * DTD * Misclist * Element *Misclist

Here, *XMLDecl* represents the XML declaration; *Misclist* represents a list of miscellaneous items such as comments, processing instructions, and spaces; *DTD* represents the document type declaration; and *Element* represents the root element of the document. All these six components are non-atomic type values and could be further defined. As an example, we give the representation of *Element* below.

```
<? xml version =”1.0”>

<!DOCTYPE  streamsub SYSTEM
“streamsub.dtd”>

<!- -Described about two streams subjects- -!>

<streamsub>

    <stream name=”computer science and
engineering”>

        <subjects>

        <first>software engineering</first>

        <second>principles of programming
languages</second>

        <third>operating systems</third>

        <four>database management systems</four>

        <five>computer organization</five>

          </subjects>

    </stream>

  <stream name=”computer networks and security”>

        <subjects>

        <first>Advanced Computer
networks</first>

        <second>network programming</second>

        <third>network routing</third>

        <four>adhoc networks</four>

        <five>network management</five>

        </subjects>
```

Fig. 1 .An example XML document

The formal representation for an element is

data Element = Elem TagName Attributelist Contents
type TagName = String
type Attributelist = [*Attribute*]
type Contents = [*Content*]

Here, type *Elem* is defined as a data constructor which has three arguments representing the element name, the attributes and the element contents, respectively. *TagName* is a synonym of type *String*. *Attributelist* and *Contents* is a list of attributes and a list of content, respectively.

An attribute is composed of the attribute name and attribute value, and is represented using a tuple as follows.

type Attribute = AttName X AttValue

where *AttName* and *AttValue* are both synonym of *String*, written as follows.

type AttName = String
type AttValue = String

It is the element content that makes the element and the XML document nested and hierarchical. The content of an element could be another element, a piece of text, a reference to some entities, a CDATA section, a processing instruction or a comment.

The content is formally represented as follows.

data Content = El Element | Tx CharData | Ref Reference | CD CDSect | ContentPI PI | ContentCom Comment

3.3 Representation of features

Here, features refer to predicates on the type of individuals. These predicates are constructed incrementally by composition of *transformations* using *predicate rewrite systems*. A *t*ransformation *f* is a function having a signature of the form

$f : (Q_1 \to \Omega) \to \cdots \to (Q_k \to \Omega) \to \mu \to \sigma$ where $Q_1, \ldots . Q_k$ σ and μ are all types and k≥0.

Transformations on XML are classified into two categories: *generic transformations* and *data-specific transformations*. *Generic transformations* come straight from the XML document representation and are applicable to all well-formed XML documents. Some examples of generic transformations are given next.

projRootElement : XML → Element
projTagName : Element →TagName
projAttributes : Element → Attributes
projContents : Element → Contents

Here, *projRootElement* projects onto the root element of an XML document. *projTagName*, *projAttributes* and *projContents* project an element onto its tag name, attributes and contents, respectively.

4. A PRECISION/RECALL-DRIVEN DECISION TREE LEARNING ALGORITHM

This section describes a novel decision-tree learning algorithm based on the precision and recall criterion for XML document classification.

4.1 The Precision/Recall Heuristic

Precision and recall were originally two statistical measures widely used in information retrieval. For the classification problem, precision is defined as the percentage of the documents correctly classified to the positive class among all documents being classified to the positive class and recall as the percentage of the documents correctly assigned to the positive class among all documents of the positive class. Precision measures the "soundness" of the classifier, and recall measures the "completeness" of it.

The precision Pr and recall Rc can be defined using TP (True Positive), FP (False Positive), FN (False Negative), and TN (True Negative).

$$P_r = \frac{TP}{TP + FP} \qquad R_c = \frac{TP}{TP + FN'}$$

where TP is defined as the number of documents correctly assigned to the positive class; FP, the number of documents incorrectly assigned to the positive class; FN, the number of documents incorrectly assigned to the negative class; and TN, the number of documents correctly assigned to the negative class. When precision and recall are equal (or very close), this point is called the precision/recall-breakeven point (BEP) of the system.

4.2 The Decision Tree Algorithm

The goal of this algorithm is to find a tree that can produce the best BEP value. Starting from a single node which is composed of the training data, the algorithm works toward two goals at the same time: looking for the point where the global precision recall are equal, and improving the F1 measure. The first goal is achieved by selecting the node which can most balance the precision and recall, which is done by a novel node selection algorithm. The second goal is achieved by finding a predicate to split this node which can best improve the F1 value of the tree, which is done by a predicate selection algorithm. The PRDT(Precision/recall-driven Decision Tree) algorithm is modified using this smoothing method to get the *SmoothPRDT* algorithm which output probabilistic decisions.

function *SmoothPRDT*($\varepsilon, \mapsto$); returns: a probabilistic decision tree;
inputs: ε, a set of examples;
$\mapsto$, a predicate rewrite system;

T := *PRDT*($\varepsilon, \mapsto$);
transfer T to an intermediate tree T by computing the node probability and leaf probability for each node in T;
transfer T to the final tree T by combining the leaf probability with the node probability for each node on the path from the root down to the leaf;
return T ;

Fig. 2. The decision-tree algorithm by smoothing the PRDT tree.

5. THE CO-TRAINING FRAMEWORK

Co-training, invented by Blum and Mitchell, is a new strategy for using unlabeled data which has two separate and redundant views. For example, with a web page classification problem, one view is the words in the web page itself, and the other is the words on the hyperlinks pointing to this page. Two classifiers induced from the two views are built incrementally through an iteration. Each classifier is initialised with its corresponding feature set of the labeled data only. In each iteration, each classifier labels the unlabeled data and add the most confidently labeled data into the training set. The two classifiers are rebuilt in each iteration with the updated training set. The basic idea behind the co-training framework is to exploit the compatibility between different views on an example. In the traditional supervised learning.

function *CotrainPRDT*($D^l, D^u, \mapsto_1, \mapsto_2$); **returns:** two decision trees;
inputs: D^l, a set of labeled examples;
D^u, a set of unlabeled examples;
$\mapsto_1$ and $\mapsto_2$, two independent predicate rewrite systems;
Create a smaller pool *Du* by randomly select a certain number of unlabeled
examples from D^u;
while $D^u \neq \emptyset$ do
$T1$:= *SmoothPRDT*($D^l, \mapsto_1$);
$T2$:= *SmoothPRDT*($D^l, \mapsto_2$);
use $T1$ to label the unlabeled data in D^u;
use $T2$ to label the unlabeled data in D^u;
select p positive and n negative most confidently labeled data by $T1$ and
$T2$ and add the non-contradictable ones to D^l;
refill *Du* by random data from D^u;
return two decision trees $T1$ and $T2$ whose predications are combined when classifying new data;

Fig. 3. The CotrainPRDT algortihm

The intuition behind the Co-training algorithm is that it may be easier for one learner to identify an example and this example may provide useful information to the other learner. Therefore, the basic idea of the Co-training algorithm is one learner inclementy trains on the other learner's classification of unlabelled examples. The Co-training algorithm works in an iterative manner.

6. CONCLUSION:

This paper presents a novel inductive learning system that aims to produce comprehensible hypothesis for XML document classification. The knowledge representation

method is based on a higher-order logic formalism which is suitable for representing individuals with complex structures. The learning algorithm of our system is a decision-tree learning algorithm driven by precision and recall. The algorithm works towards two goals at the same time: decreasing the difference between the precision and recall and improve the *F*1 value. In this paper an algorithm exists which combines the Co-training strategy with a decision-tree algorithm driven by precision/recall breakeven point. The two views of the Co-training are created using two different predicate rewrite systems that are built on a novel higher-order logic representation method for XML documents. By using this algorithm we can show that the CotrainPRDT algorithm can be used to successfully classification XML documents using a few labelled examples together with a large number of unlabeled example, provided that there are two sufficient views with the documents.

ACKNOWLEDGMENT

We would like to thank Dr.V.Srikanth, Head of the Department, Computer Science & Engineering, K L university, Vaddeswaram for his encouragement and motivation to write this paper. Also we are grateful to G.Siva Nageswara Rao, (CSE), K L University, Vaddeswaram for guiding us in writing this paper.

REFERENCES

[1] S. Giri, A. Chandramouli, and S. Gauch, "XML Classification Using Content and Structure," Technical Report ITTC-FY2007-TR- 31020-02, 2007.

[2] J.X. Wu and J. Zhang, "Knowledge Representation and Learning for Semistructured Data," Technical Report, CSIRO ICT Centre, 2009.

[3] Bouchachia.A, Hassler.M, "Classification of XML Documents",2007

[4] Xiaobing Jemma Wu, XML Document Classification with Co-training, CSIRO ICT Centre, 2009

[5] Qingjiu Zhang, "Shiliang sun, "Evolutionary classifier ensembles for semi-supervised learning",2010

[6] Yuanyuan Guo, Xiaoda Niu ; Zhang.H "An Extensive Empirical Study on Semi-supervised Learning",2010

[7] X. Zhu and A.B. Goldberg, "Introduction to Semi-Supervised Learning", 2009

[8] Jemma Wu, A Framework for Learning Comprehensible Theories in XML Document Classification, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 24, NO. 1, JANUARY 2012

[9] Classification Tree Embedded XML Document Structure Design for Enhanced Web Document Utilization, Sixth International Conference on Advanced Language Processing and Web Information Technology, 2007,pages: 542-547.

[10] Applications of Data Mining in the Education Resource Based on XML, International Conference on advanced Computer Theory and Engineering, 2008, ICACTE'08. Pages: 943-946.

[11] Graph-based Semi-supervised Learning Algorithm for Web Page Classification, Sixth International Conference on Intelligent Systems Design and Applications,2006. Pages:856-860.

[12] Research on Multi-View Semi-Supervised Learning Algorithm Based on Co-Learning, International Conference on Machine Learning and Cybernatics, 2006. Xing-Qi wang.

[13] Xiaobing Jemma Wu, An Inductive Learning System for XML Documents,2010

[14] A Passive-Aggressive Algorithm for Semi-supervised Learning, International Conference on Technologies and Applications of Artificial Intelligence, 2010. Chien-chung Chang. Pages: 335-341.

[15] A new semi-supervised support vector machine learning algorithm based on active learning , International Conference on Future Computer and Communication (ICFCC), 2010. Li Cunhe. Vol 3, May 2010.