ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

Duplicate Detection in Hierarchical Data Using Improved Network Pruning Algorithm- (Result Paper)

Bhavana Dhake[1], Dr.S.S.Lomte[2], Prof.Y.R.Nagargoje[3], Prof.R.A.Auti[4], Prof.B.K.Patil[5]

ME Student, Department of CSE, Dr. Seema Quadri Institue of Technology, Aurangabad, India [1]

Professor, Department of CSE, Dr. Seema Quadri Institue of Technology, Aurangabad, India [2]

Assistant Professor, Department of CSE, Dr.Seema Quadri Institue of Technology, Aurangabad, India [3,4,5]

ABSTRACT: Duplicate detection is the problem of determining that different representation of entities in a data source actually represents the same real world entity. It is most critical task in huge database system. In this paper, we proposed an novel method for finding duplicates in semi-structured or hierarchical data like XML, called XMLDup method. This method uses Bayesian network to determine the probability of two XML elements being duplicates. This method considers not only data but also the way of data is structured when finding duplicates. To improve the effectiveness & efficiency, we proposed a novel pruning method that checks for typographical errors & remove white spaces while comparing two XML elements. With the help of experiments, we are able to show how our system gives higher precision & recall values for various datasets mentioned here than the previous one.

KEYWORDS: Duplicate Detection, Bayesian Netwronks, Data Cleaning, XML, Record Linkage.

I. INTRODUCTION

Electronics data plays an important role because of increasing used of World Wide Web. It is the part of numerous business applications, processes & decision. This data may contain noise or it may be an incomplete data. This thing affects the quality of data. It will also affect the result of data mining because the knowledge discovery from such data is more difficult. For examples, in 1992, 100,000 tax refund cheques could not be delivered by postal mail due to errors in address data.

In this paper we are going to focus on such types of errors, called duplicates. Duplicates means representing the same real world entity in more than one form. The challenge in duplicate detection is to detect duplicate representations that are *not exactly equal* due to errors in the data, and that *cannot be identified using a universal identifier* (e.g., the ISBN of a book). Examples for errors are typos and misspellings or the lack of a standard representation for the data, for instance when a date can be represented both in the European format (day.month.year) or the American format (month/day/year). Further errors are missing, outdated, or contradictory data. As ac consequence, duplicate detection cannot be performed just by checking on the equality of object attributes or global identifiers. Instead, more complex algorithms are required, e.g., objects need to be compared pairwisely using a complex similarity measure. Such algorithms are necessary in both data cleaning and data integration scenarios.

Data cleaning consists in correcting errors and inconsistencies in data and is an issue of critical practical importance as it improves overall data quality. High data quality is the prerequisite for meaningful data analysis, required in scenarios such as report generation over data warehouses, customer relationship management, and data mining, to name just a few. When an object has multiple representations, analysis assumes that they are actually multiple different objects and therefore generates wrong results. XML is increasingly popular as data representation, especially for data published on the World Wide Web and data exchanged between organizations. In XML data, it is

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

also true that different types of objects are described within a single schema, however, they are not necessarily described by a fixed set of single valued attributes due to the semistructured nature of XML. Consequently, similarity measures designed to compare equally structured flat tuples are no longer applicable to semi-structured hierarchical XML elements. Therefore, our goal is to devise strategies for XML duplicate detection, which, in addition to considering relationships, also necessitate new techniques for similarity measurement.

A.Schema of XML by example:

XML is used both for large-scale electronic publishing of data, and for the exchange of data on the Web and elsewhere. The two main features of XML are that the data is organized hierarchically, and is semi-structured, mixing content, e.g., text and structure, using so called XML tags. A file conforming to the XML format is called an XML document.

An XML document includes (but is not limited to) a set of nodes in the document (a root node, element nodes, attribute nodes, and value nodes) all having an identity, as well as a set of edges between nodes such that a tree is obtained. The root node has no ancestors, an has a set of children elements that can be element nodes, attribute nodes, and value nodes. XML documents must contain a root element. This element is "the parent" of all other elements. The elements in an XML document form a document tree. The tree starts at the root and branches to the lowest level of the tree. All elements can have sub elements (child elements) as fallows

```
<root>
  <child>
        <subchild>…….</subchild>
  </child>
</root>
```

The terms parent, child, and siblings are used to describe the relationships between elements. Parents' elements have children. Children on the same level are called siblings (brothers or sisters) [7]. An XML element is everything from (including) the element's start tag to (including) the elements end tag. An element contains other element, text, attributes, or a mix of all this things. Consider the following example,

```
<prs1 name="Bhavana"  dob="25-03-1988" > <pob1>India</pob1>
    <cnt1>
        <eml1>bgdhake03@Gmail.com</eml1>
        <add1>43,Vidyanagar,Dombivali(West),Mumbai- 400064</add1>
         <add2>4th st,Gulmohar Colony ,Aurangabad-431006  </add2>
    </cnt1>
</prs1>
```

The root element in the example is <prs> which is element contents**.** The <prs> element has 2 children: <pob>, <cnt> & <add1><add2><eml><pob>have text content because they contain text. The <prs> also has an attribute (name="Bhavana D" dob="25-03-1988").

In This paper, we first present a probabilistic duplicate detection algorithm for hierarchical data called XMLDup.This algorithm consider both the similarity of attribute contents & the relative importance of descendent elements with respect to the overall similarity score. The algorithm presented here extends our previous work by two things 1.Sgnificantly improving efficiency & 2.showing a more extensive set of experiments.

Structure : This paper is organized as follows: Section 2 presents literature survey.Section3 summarizes our baseline algorithm, i.e Bayesian Network Our strategies to accelerate this algorithm are then presented in Section 4,Network pruning algorithm. In Section 5 ,We perform an experimental evaluation of these techniques over artificial and real world data, and discuss the results. Finally, in Section 6 we conclude and present suggestions for future work.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

II. RELATED WORK

In this section, we described survey did for duplicate detection in hierarchical data, which is the focus of this paper. Here we described the previous method related to XML data duplicates detection and there disadvantages. Luis L. et. al. [3] proposed his own unique method for XML data duplication called as XMLDup. The basis of XMLDup is Bayesian network. Bayesian network used to determine the probability of two XML data elements which is used to be compared for duplicate detection. Here authors not only consider the information within the elements but also the way that information is structured. XMLDup mainly uses 2 types of probabilities Prior probability and Conditional probability. It requires little user intervention for the fixing of threshold values. The model is very flexible. XMLDup gives good results in the form of precision and recall. To check the run time efficiency of XMLDup, a network pruning strategy is given. This works on either of the 2 ways that is lossless approach and lossy approach. In lossless approach there is no impact on the final result but in the lossy approach there is slightly impact on the recall parameter.

S. Chaudhuri et. al. [10] develops an algorithm to remove duplicates in dimensional tables in data warehouse called as DELPHI (**D**uplicate **El**imination in the **P**resence of **Hi**erarchies) which reduces the number of false positives without missing out on detecting duplicates. Authors use dimensional hierarchy which consists of a chain of relations linked by key – foreign key dependency to develop high quality duplicate elimination algorithm and then it evaluates on real datasets from an operational data warehouses. Here final duplicate detection function is a weighted voting of the predictions from using co-occurrence similarity function and textual similarity function.

F. Naumann et. al. [4] proposed a framework for both efficiency & effectiveness in duplicate detection called as DogmatiX. DogmatiX stands for **D**uplicate **O**bjects **G**et **MAT**ched **I**n **X**ml. The framework mainly contains three steps: 1) Candidate Definition, 2) Duplicate Definition, and 3) Duplicate Detection. Here first two steps are carried out in offline mode when system setup & last step is carried out in online mode where actual algorithm exists. Candidate definition gives which objects we want choose for duplicate detection. Duplicate definition characterizes which portion of actual data is used for selection to find out duplicates. Duplicate detection performs six various sub-steps on actual data to detect duplicates in duplicate candidates. Here first three prepare the data for comparisons, whereas the remaining three perform the actual duplicate detection. While comparing XML elements, DogmatiX not only consider their data values but also the similarity of the children, parents, structure etc.

M. Weis et. al. [5] proposed a unique method for Fuzzy duplicate detection in semi structured or hierarchical XML data. It not only focuses on the duplicate status of the children nodes but also gives more importance to the probability of descendants being duplicates. Probabilities of two XML elements are efficiently calculated with the help of Bayesian network model. This model derives from the structure of the XML objects that being compared. This algorithm provides great flexibility in its configuration, by allowing the use of different similarity measures for the data values and different conditional probabilities to join the similarity probabilities of the XML elements to be compared. This algorithm gives high precision and recall values while data sets contains higher amount of errors and missing information.

A Kade et. al. [6] proposed scheme where matching of XML documents are carried out in the Highly dynamic applications like web and peer-to-peer system. For highly dynamic application systems requires great effort for document management system. The author takes full advantage of the flexibility of XML documents for matching the similar XML documents. This unique method solves the matching problem of XML documents i. e. the problem of defining which parts of two XML document contains the same information. Matching is the first step of integration process. This approach is unique in the sense; it joins similarity information from the content of the elements with information from the structure of the documents. The total work is divided into three parts to calculate similarity between two XML documents: 1) the node's name, 2) the element's content's, and 3) the node's path.

P. Calado et. al. [13] proposed a unique method which combines two optimization strategies: 1) to select appropriate object to compare and 2) to optimize pair-wise object comparison. Here authors introduce machine learning approach to determine the required parameter. It does not need of user intervention. This method combines a traditional

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

blocking strategy, which allows a object comparison level, and a pair wise optimization strategy which again allows a attribute comparison level. Additionally authors give how to automatically learn the parameter required by both strategies, without taking the specific knowledge of the database.

M. Weis et. al. [21] proposed unique domain independent algorithm that efficiently find outs the duplicate in XML documents. The top down approach is used for traversing the XML tree to identify the duplicates on each level. Threshold similarity function is used for detecting pairs of duplicate elements. Here authors used an appropriate filter function to reduce the number of pair wise element comparison. The similarity measure includes string similarity for pairs of string which is again measured by the edit distance. To increase the efficiency of the method authors avoid the computation of edit distance for pairs of strings using three filtering methods.

S Zadrozny et. al. [8] gives a method to detect coreferent object in XML metadata. Here coreferent object means duplicate objects of same real world object. The detection is based on either data or on both the data and the metadata. This method firstly compares the path from root element to a given element in the schema where each path defines with the help of the location and context of the specified element in the schema. Here path matching was achieved by the comparison of the different steps of which paths are made. The PTV- Possibilistic Truth Values is used for matching steps and it aggregated by the Sugeno integral. This method contains 4 different steps 1) Extraction, 2) Generation of Coreferent path matrix, 3) Mapping of algorithm at path level, 4) Aggregation at path level.

Thander L. et. al. [9] proposed an algorithm to detect duplicate objects in XML data and also used MD5 algorithm to reduce the number of false positive. According to authors there are two types of heterogeneity-structural and lexical. When the fields in the tuples of the database are structured differently in different database then it is called as Structural heterogeneity. And when the tuples have identically structured fields in the database and data in the tuples are different then it is called as Lexical heterogeneity. Authors proposed algorithm works for lexical heterogeneity. The algorithm mainly works with the help of three modules Selector, Preprocessor and Duplicate identifier. It takes input as XML document and candidate definition and produces output as duplicate objects.

S. Intakosum et. al. [14] presents a method called as PathMatch which helps to find out semantic similarity rate by cost matrix model between the two XML paths with the help of edit distance algorithm. PathMatch similarity function is an idea of path matching that improves the PathSim algorithm. The steps for the PathMatch algorithm is as follows

1. Create a matrix M with *m* rows and *n* columns where *m* is equal to or less than *n.*
2. Fill dissimilarity rate of every row until finish by compare with each column.(1 – Sim(A[i], B[j])).
3. Sum the minimum number of dissimilarity rate of each column.
4. Calculate PathMatch similarity rate.

III. PROPOSE SYSTEM

Proposed system will simulate hierarchical or semi structured data environment and perform duplicate detection on XML data. And it will process only one XML files as input for data duplication which contains combination of original records and some duplicate records. We are given input four different XML data set to our project. Out of four dataset, three data set are standard data set named as *Cora.* Country ,CD data sets.

A.Problem Defination :

To design & implement the method which find out duplicates in the XML data elements. And also check for its typographical errors when comparing two XML elements.

B. Objective :

To find out duplicate objects stored in a complex hierarchical or semi structure data like XML data. To find out multiple representation of same real world objects. Duplication detection will be based on what actual data stored in parent node & also at child node. To achieve high precision & recall in various datasets. To remove white spaces &

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

then do duplicate detection. We will also extend this for Typographical Errors such as Sudipta = Shudipta , Ashok = Asoke etc.

C.*Bayesian Network formation :*

XMLDup uses Bayesian network to determine the probability of two XML elements being duplicates. In this section we described how this Bayesian Network formed & how this will used to calculate similarity score for two XML objects. The two objects are duplicates iff its similarity score is above the threshold.

The Bayesian network provides brief idea of a joint probability distribution. The BN model seen as a directed acyclic graph(DAG) in which the nodes represents random variables & dependencies between those variables are represented by the edges.The main idea of two XML objects being duplicates is "The fact that two XML nodes are duplicates depends only on the fact that their values are duplicates and that their children nodes are duplicates".The two XML trees are duplicates only when their nodes are duplicates.This can be described by considering following example:

Let us consider the two XML elements that represent the same persons which are shown as trees in Figure 3.1. Here both represent object person named as '*prs*'. These elements have two attributes that are *name* & *dob* (date of birth) & also two child node as *pob* (place of birth) & *cnt* (contact). Again at next level contact contains add_1 (correspondence address), add_2 (permanent address) & eml_1 (email). Here leaf elements have a text node which contains the data. For example name has text node consists of text as a "Bhavana D" as its value. In this example the ultimate goal of our method is to detect the both the XML elements as duplicate with respects to their different data values. To carry out this we compare first value of attributes of both tree with each other i.e. *name* & *dob* and then check for its children node i. e. *pob* & *cnt*. Furthermore, *pob* node are duplicate depending on whether or not their values are duplicates, and the *cnt* node are duplicate depending on whether or not their children nodes i. e. *eml* and *add* are duplicates. This process carried out until the leaf nodes are reached.

Figure 3.2 represents the Bayesian Network to compute the similarity between 2 XML objects that represent same elements which is shown in figure 1. In this type of network the node prs_{11} represent the possibility of node prs_1 in the XML *TreeU* with respect to the duplicate of node prs_1 in the XML *TreeU'*. This root node prs_{11} has two child node V_{prs11} and C_{prs11} where node V_{prs11} which represent the possibility of attribute values in the *prs* nodes being duplicates and node C_{prs11} which represent the possibility of children nodes of the *prs* nodes being duplicates. In detailed, node V_{prs11} contains value of attributes as prs_{11}*[name]* and prs_{11}*[dob]* as shown in rectangle which checks for its duplication. Node C_{prs11} contains node pob_{11} and cnt_{11} where node pob_{11} which represent the possibility of node pob_1 in the XML *TreeU* with respect to the duplicate of node pob_1 in the XML *TreeU'*; node cnt_{11} represent the possibility of node cnt_1 in the XML *TreeU* with respect to the duplicate of node cnt_1 in the XML *TreeU'*.

There is slightly different procedure is to be fallowed when there is same type of multiple nodes are present given XML element just like as a add field in our example of figure 3.3. So in this case, we want to compare the full set of nodes, instead of each node independently, Exemplar based Inpainting technique is used for inpainting of text regions, which takes structure synthesis and texture synthesis together. The inpainting is done in such a manner, that it

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

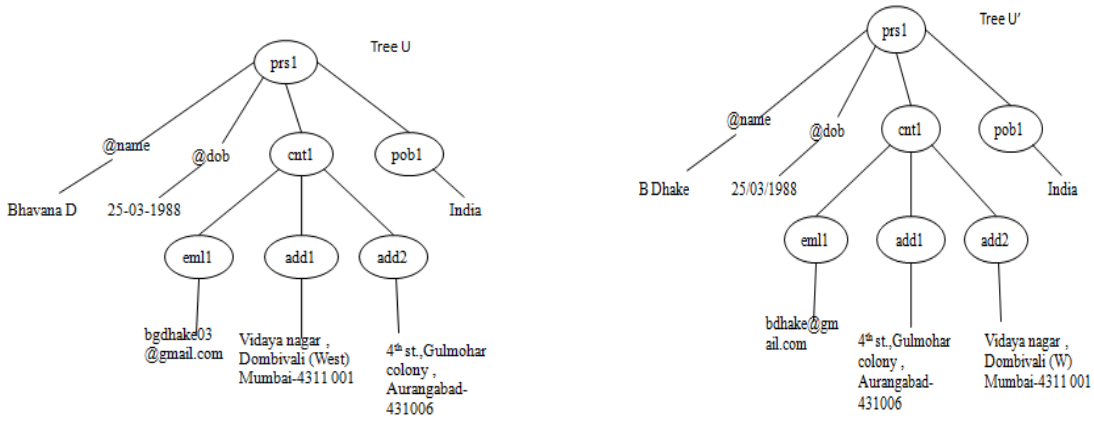

Figure 3.1 Two XML elements that represents the same person

fills the damaged region or holes in an image, with surrounding colour and texture. The algorithm is based on patch based filling procedure. First find target region using mask image and then find boundary of target region. For all the boundary points it defined patch and find the priority of these patches. It starts filling the target region from the highest priority patch by finding the best match patch. This procedure is repeated until entire target region is inpainted.

The algorithm automatically generates mask image without user interaction that contains only text regions to be inpainted. for this reason we create node add_{**} under that node we compared add_{11}, add_{12}, add_{21}, add_{22} of *TreeU* and *TreeU'*.

In accord with our assumption, the probability of the two XML nodes being duplicates depends on 1) whether or not their values are duplicates, and 2) whether or not their children are duplicates. Thus, node prs11 in the BN has two parent nodes, as shown in Figure. 3.2. Node Vprs11 represents the possibility of the values in the prs nodes being duplicates. Node Cprs11 represents the possibility of the children of the prs nodes being duplicates. As before, a binary random variable, that can be active or inactive, is assigned to these nodes, representing the fact that the values and children nodes are duplicates or nonduplicates, respectively.

D. Computing Probabilities:

There are two types probabilities are used in this method which are: Prior Probability & Conditional Probability. Prior Probabilities are the probabilities of values being duplicates with respect to their parent XML node,i.e. , $P(prs_{11}[name])$, $P(prs_{11}[dob])$, $P(pob_{11}[value])$, $P(eml_{11}[value])$, and $P(add_{ij}[value])$. This type of probability can be defined with the help of similarity function as *sim(.)* whose values are ranges from the 0 to 1.
There are four types Conditional Probabilities are as follows:

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

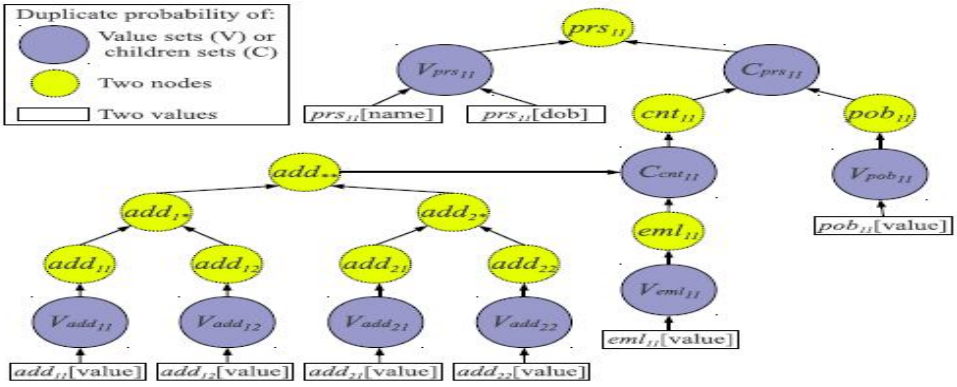

Figure 3.2 BN to compute the similarity of the trees in Figure 3.1

Conditional probability 1 (CP1): The probability of the values of the nodes being duplicates, given that each individual pair of values contains duplicates. Intuitively, 1) if all attribute values are duplicates, we consider the XML node values as duplicates; 2) if none of the attribute values are duplicates, we consider the XML node values as non duplicates; 3) if some of the attribute values are duplicates, we determine that the probability of the XML nodes being duplicates. This value represents the importance of the corresponding attribute a in determining if the nodes are duplicates.

Conditional probability 2 (CP2): The probability of the children nodes being duplicates, given that each individual pair of children are duplicates. Intuitively, it makes sense to say that two nodes are duplicates only if all of their child nodes are also duplicates. However, it may be the case that the XML tree is incomplete, or contains erroneous information. Thus, we relax this assumption and state that the more child nodes in both trees are duplicates, the higher the probability that the parent nodes are duplicates.

Conditional probability 3 (CP3): The probability of two nodes being duplicates given that their values and their children are duplicates. Essentially, we consider the nodes as duplicates if both their values and their children are duplicates.

Conditional probability 4 (CP4): The probability of a set of nodes of the same type being duplicates given that each pair of individual nodes in the set is duplicates.

Final probability will be calculated once all prior and conditional probabilities are defined; the BN can be used to compute the probability of two XML trees being duplicates

E.*Proposed Algorithm:*

In this section, we discussed the actual algorithm which we follow to find out duplicates in XML data. In order to improve the BN evaluation time, we propose a lossless pruning strategy. This strategy is lossless in the sense that no duplicate objects are lost. Only object pairs incapable of reaching a given duplicate probability threshold are discarded.

As stated before, network evaluation is performed by doing a propagation of the prior probabilities, in a bottom up fashion, until reaching the topmost node. The prior probabilities are obtained by applying a similarity measure to the pair of values represented by the content of the leaf nodes. Computing such similarities is the most expensive operation in the network evaluation and in the duplicate detection process in general. Therefore, the idea behind our pruning proposal lies in avoiding the calculation of prior probabilities, unless they are strictly necessary.

The strategy follows the premise that, before comparing two objects, all the similarities are assumed to be 1 (i.e., the maximum possible score). The idea is to, at every step of the process; maintain an upper bound on the final probability value. At each step, whenever a new similarity is computed, the final probability is estimated taking into

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

consideration the already known similarities and the unknown similarities that we assume to be 1. When we verify that the network root node probability can no longer achieve a score higher than the defined duplicate threshold, the object pair is discarded and, thus, the remaining calculations are avoided.

First of all the algorithm takes an input as (N,T) where N stands for nodes and T stands for predefined threshold value upon which we can say that the given node is duplicate or not. Then it takes the list of all parent nodes of N and assumes that their probability score is 1.

Algorithm: ImprovedNetworkDupDetect(N, T)

Require: The node N, for which we intend to compute the probability score; threshold value T, below which the XML nodes are considered non-duplicates

Ensure: Duplicate probability of the XML nodes represented by N

1. Find out list "L" which is ordered list of parents.
2. Assign parentScore[n] ← 1& currentScore ← 0
3. for each node n in L do
4. if "n" is a value node then {which is selected at the run time}
5. Eliminate BLANK/WHITE SPACE if any
6. score ← getSimilarityScore(n)
7. else
8. newThreshold ← getNewThreshold (T,parentScore)
9. score ← NetworkDupDetect(n, newThreshold)
10. end if
11. Assign parentScore[n] ← score, currentScore ← computeProbability(parentScore)
12. if currentScore < T then End network evaluation
13. end if
14. end for
15. return currentScore

Then in nest step the actual probability value of each node of the parents of N are computed. If a node *n* is a value node then we compute the probability score by finding the similarity of the values it represents. Note that selection of which nodes & attributes are chosen will be depending on the user which will be given at run time. On the other side if a node *n* is a not value node means it is child node then we compute recursively until the leaf node not found with updated threshold. value with respect to that particular node. Ones the score for node *n* is calculated then our algorithm compares the total score of for *N* with threshold value and then decides the whether to continue or stop the algorithm. Here function *computeProbability* consists by applying one of the conditional probability which is discussed above.

IV. EXPERIMENTAL RESULTS

A.Data Sets :

To perform the Duplicate Detection we are using four different types of data sets which represent different data domains. These data sets are Employee, Country, Cora, CD 2. These data sets are a real database so we want to add some artificially polluted data related to the data objects. And also make sure that it contains different types of errors such as typographical error, missing data, and duplicate erroneous data. The size of the data sets are varying from 200 objects (Country) through 150 objects (Employee). For all experimental results we will be used threshold value as 0.7. If possible we will be varying the threshold value to achieve high accuracy in the results.

The data sets i. e. Cora, CD 2, Country is easily available at the Hasso Plattner Institute website. I prepared the Employee data set for this project. The attributes used for data set Employee is Employee id & Name and sub node are – Hire date, Add1, and Add2. The attributes used for data set Country is Contry Name and sub node are – Car Code, Government, and City. For the data set Cora, the attributes are Publication_id and sub nodes are Author, Title, Volume and Date. And For the data set CD, the attributes are Disc_id and sub nodes are Artist, Disc title, Category and Tracks.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

B.Experimental Setup :

To know the effectiveness of our method, we applied commonly used *Precision* and *Recall*. *Precision* measures the percentages of correctly identified duplicates, over the total set of objects determined as the duplicates by the system. *Recall* measures the percentage of duplicates correctly identified by the system, over the total set of duplicate objects.

There is set of the all documents with in which there is relevant document set and retrieved document set. Intersection of relevant and retrieved document gives the duplicate objects in the all document.

Formula :

Precision = A / A+C * 100% Recall = A / A+B * 100%

where A = The number of relevant records retrieved,

B = The number of relevant records not retrieved, and C = The number of irrelevant records retrieved.

Assume the following example to calculate Precision and Recall .A database contains 80 records on a particular topic A search was conducted on that topic and 60 records were retrieved. Of the 60 records retrieved, 45 were relevant. In this example A = 45, B = 35 *(80-45)* and C = 15 *(60-45).Precision* = (45 / (45 + 15)) * 100% = 45/60 * 100% = >75%.*Recall* = (45 / (45 + 35)) * 100% =45/80 * 100% = >56%.

C.Reults :

For Employee Dataset:

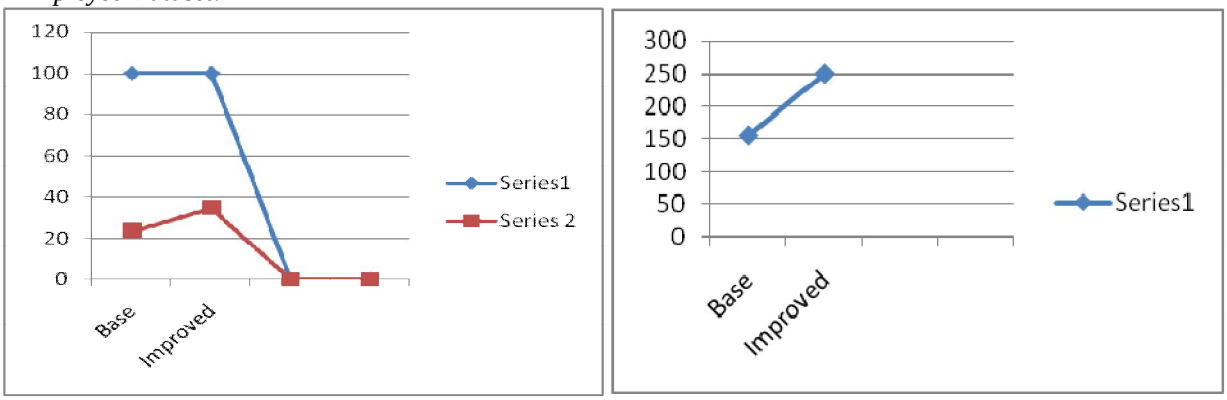

Figure 4.2(a) Precision & Recall for Employee Dataset (b) Time Complexity for Employee Dataset

The figure 4.2(a) shows the comparisons of the Base paper which implements XMLDup method with our proposed novel method named as Improved Network Pruning Algorithm. On the X-axis, we have show the Precision and Recall values for base algorithm and our improved algorithm. On the other hand i. e. on the Y-axis, we had plot the range of Precision and Recall values in percentage which starts from 0 to 100. The blue color shows the result of Precision values while red color shows the result of recall values. The *Employee* dataset contains over all 150 records. Here Base algorithm detects only 36 records as duplicates while Improved algorithm detects 52 records as duplicates. The precision values are near about same for both Base and Improved algorithm i. e. 100%. But the recall value for Improved Network Pruning Algorithm is better as compared to the base algorithm by 44.58%.

The figure 4.2(b) shows the comparisons of the Base paper which implements XMLDup method with our proposed novel method name as Improved Network Pruning Algorithm for time complexity. On the X-axis, we have shows the values for base algorithm and our improved algorithm. On the other hand i. e. on the Y-axis, we had plots the time (in milli-second) taken to execute the both algorithm which ranges from 0 to 300. The blue color shows the result of Base algorithm while red color shows the Improved algorithm. The *Employee* dataset contains over all 150 records. Here Base algorithm takes 156ms to detect records as duplicates while Improved algorithm takes 249ms to detect records as duplicates. The time taken for executing Improved network pruning algorithm compared to the base algorithm by 59.62%.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

For Country Datasets:

The figure 4.3(a) shows the comparisons of the Base paper which implements XMLDup method with our proposed novel method named as Improved Network Pruning Algorithm. On the X-axis, we have show the Precision and Recall values for base algorithm and our improved algorithm. On the other hand i. e. on the Y-axis, we had plot the range of Precision and Recall values in percentage which starts from 0 to 100. The blue color shows the result of Precision values while red color shows the result of recall values. The *Country* dataset contains over all 200 records. Here Base algorithm detects only 74 records as duplicates while Improved algorithm detects 94 records as duplicates. The precision values are near about same for both Base and Improved algorithm i. e. 100%. But the recall value for Improved Network Pruning Algorithm is better as compared to the base algorithm by 25.68%.

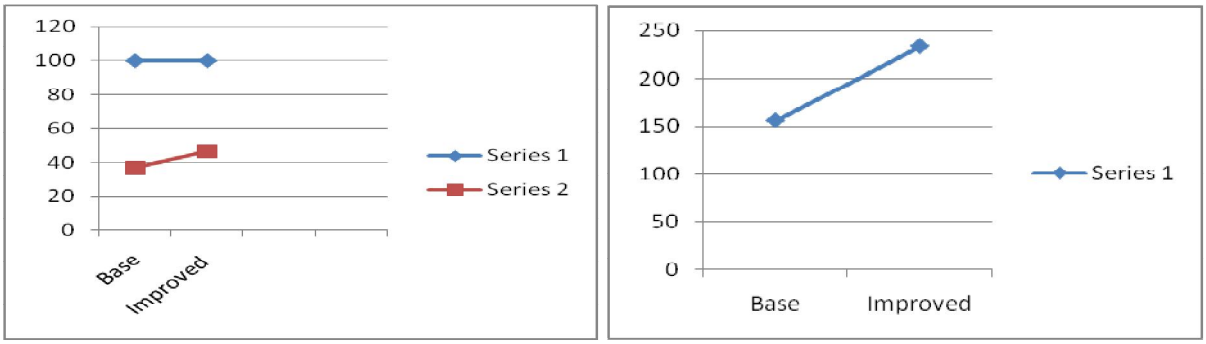

Figure 4.3(a) Precision & Recall for Country Dataset (b)Time Complexity for Country Dataset

The figure 4.3(b) shows the comparisons of the Base paper which implements XMLDup method with our proposed novel method name as Improved Network Pruning Algorithm for time complexity. On the X-axis, we have shows the values for base algorithm and our improved algorithm. On the other hand i. e. on the Y-axis, we had plots the time (in milli-second) taken to execute the both algorithm which ranges from 0 to 250. The blue color shows the result of Base algorithm while red color shows the Improved algorithm. The *Country* dataset contains over all 200 records. Here Base algorithm takes 156ms to detect records as duplicates while Improved algorithm takes 234ms to detect records as duplicates. The time taken for executing Improved network pruning algorithm as compared to the base algorithm by 50.00%.

For Cora Datasets :

The figure 4.4(a) shows the comparisons of the Base paper which implements XMLDup method with our proposed novel method named as Improved Network Pruning Algorithm. On the X-axis, we have show the Precision and Recall values for base algorithm and our improved algorithm. On the other hand i. e. on the Y-axis, we had plot the range of Precision and Recall values in percentage which starts from 0 to 100. The blue color shows the result of Precision values while red color shows the result of recall values. The *Cora* dataset contains over all 189 records. Here Base algorithm detects only 18 records as duplicates while Improved algorithm detects 80 records as duplicates. The precision values are near about same for both Base and Improved algorithm i. e. 100%. But the recall value for Improved Network Pruning Algorithm is better as compared to the base algorithm by 69.20%.

The figure 4.4(b) shows the comparisons of the Base paper which implements XMLDup method with our proposed novel method name as Improved Network Pruning Algorithm for time complexity. On the X-axis, we have shows the values for base algorithm and our improved algorithm. On the other hand i. e. on the Y-axis, we had plots the time (in milli-second) taken to execute the both algorithm which ranges from 0 to 250. The blue color shows the result of Base algorithm while red color shows the Improved algorithm. The *Cora* dataset contains over all 200 records. Here Base algorithm takes 172ms to detect records as duplicates while Improved algorithm takes 232ms to detect records as duplicates. The time taken for executing Improved network pruning algorithm as compared to the base algorithm by 34.88%.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

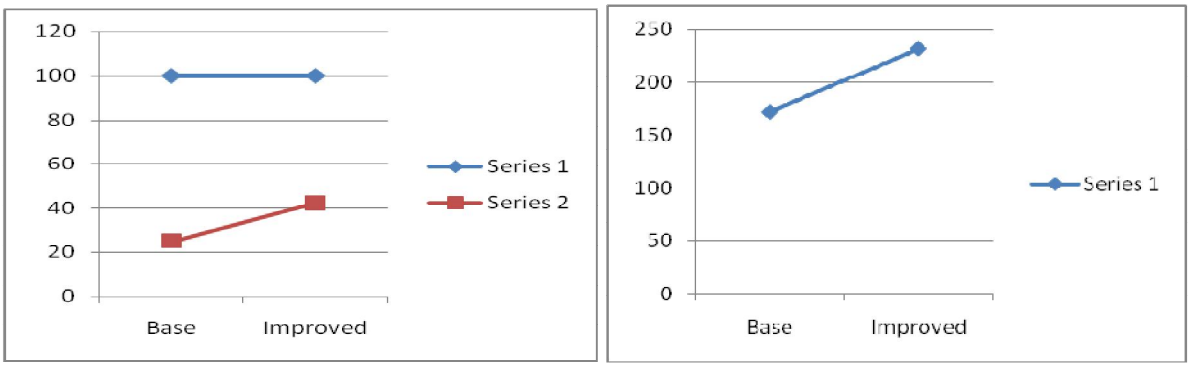

Figure 4.4(a) Precision & Recall for Cora Dataset(b) Time Complexity for Cora Dataset

For CD Dataset:

The figure 4.5(a) shows the comparisons of the Base paper which implements XMLDup method with our proposed novel method named as Improved Network Pruning Algorithm. On the X-axis, we have show the Precision and Recall values for base algorithm and our improved algorithm. On the other hand i. e. on the Y-axis, we had plot the range of Precision and Recall values in percentage which starts from 0 to 100. The blue color shows the result of Precision values while red color shows the result of recall values. The *CD* dataset contains over all 200 records. Here Base algorithm detects only 25 records as duplicates while Improved algorithm detects 27 records as duplicates. The precision values are near about same for both Base and Improved algorithm i. e. 100%. But the recall value for Improved Network Pruning Algorithm is better as compared to the base algorithm by 8.00%. We are still working for this dataset to find the issues related to the less than expected results as compared to the other datasets.

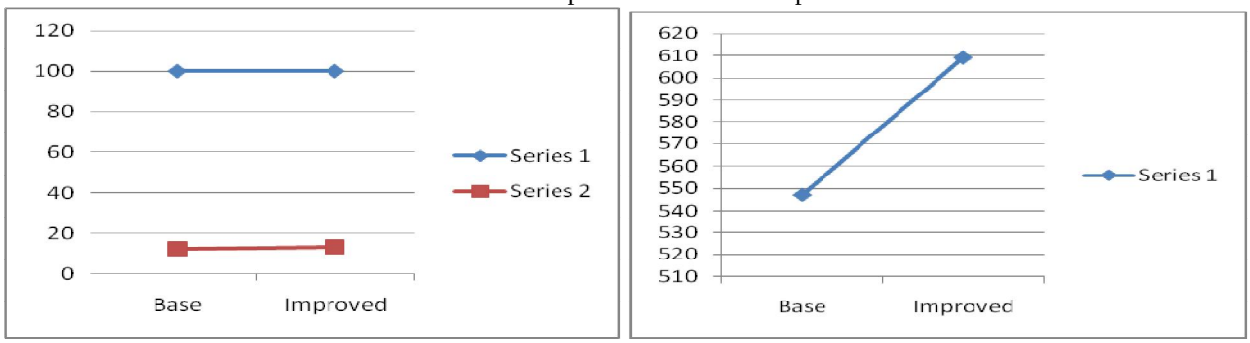

Figure 4.5(a) Precision & Recall for CD Dataset(b) Time Complexity for CD Dataset

The figure 4.5(b) shows the comparisons of the Base paper which implements XMLDup method with our proposed novel method name as Improved Network Pruning Algorithm for time complexity. On the X-axis, we have shows the values for base algorithm and our improved algorithm. On the other hand i. e. on the Y-axis, we had plots the time (in milli-second) taken to execute the both algorithm which ranges from 0 to 650. The blue color shows the result of Base algorithm while red color shows the Improved algorithm. The *CD* dataset contains over all 200 records. Here Base algorithm takes 547ms to detect records as duplicates while Improved algorithm takes 609ms to detect records as duplicates. The time taken for executing Improved network pruning algorithm as compared to the base algorithm by 11.35%. We are still working for this dataset to find the issues related to the less than expected results as compared to the other datasets.

V. CONCLUSION

We will propose the novel method which can find out duplicates in the XML data elements with the help of Bayesian network to determine the probability of two XML elements being duplicates. The Bayesian Network is used

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

to form the structure of elements that are being compared with each other by not only computing their probabilities but also the way they are structured in XML documents. In addition to improve the efficiency and effectiveness, we also checks for its one of the type of typographical errors in which the two XML elements are compared by removing its blank or white spaces. We extend this method for one of the other type of typographical error where spelling of any words is differing as per the pronunciation of that individual. It requires little user intervention, because user only needs to give the attribute list to be compared and their threshold value.

We have successfully improved the Recall values by Improved Network Pruning Algorithm which ranges from 25.68% (minimum i. e. for *Country* dataset) to 69.20% (maximum i.e. for *Cora* dataset). The time complexity for Improved Network Pruning Algorithm ranges from 11.35 %(minimum i. e. for *CD* dataset) to 59.62% (maximum i. e. for *Employee* dataset).

Future Work :

As the Future Work, we will evaluate the results of recall values for those datasets which gives false positive results. And we will also extend this algorithm to compare two XML objects with object different structure.

ACKNOWLEDGMENT

I would like to thank my guide Prof.Y.R.Nagargoje ,their guidance & feedback during the course of project. We would also like to thank CSE dept for giving me the resources & the freedom to pursue this project.

REFERENCES

[1] Abraham Silberschatz, Henry F. Korth, S. Sudharshan, "Database System Concepts," Mcgraw-hill International Publication, 5th Edition, 2006.
[2] Erhard Rahm and Hong Hai Do, "Data Cleaning: Problems and Current Approaches," IEEE Data Eng. Bull., vol. 23, no. 4, pp. 3-13, Dec. 2000.
[3] Luis Leitão, Pável Calado and Melanie herschel, "Efficient and Effective Duplicate Detection in Hierarchical Data," IEEE Transaction On Knowledge and Data Engineering, Vol. 25, N0 5, pp. 1028-1040, 2013.
[4] Melanie Weis and Felix Naumann, "Dogmatix Tracks Down Duplicates in XML," Proc. ACM SIGMOD Conf. Management of Data, pp. 431-442, 2005.
[5] Luis Leitao, Pavel Calado, and Melanie Weis, "Structure-Based Inference of XML Similarity for Fuzzy Duplicate Detection," Proc. 16th ACM Int'l Conf. Information and Knowledge Management, pp. 293-302, 2007.
[6] Adrovane M. Kade, and Carlos A. Heuser, "Matching XML Documents in Highly Dynamic Applications," Proc. ACM Symp. Document Eng. (DocEng), pp. 191-198, 2008.
[7] Diego Milano, Monica Scannapieco and Tiziana Catarci, "Structure Aware XML Object Identification," Proc. VLDB Workshop Clean Databases (CleanDB), 2006.
[8] Marcin Szymczak, Sławomir Zadrożny, Guy De Tré, "Coreference detection in XML metadata," IEEE International Conference on IFSA World Congress and NAFIPS Annual Meeting, pp. 1354-1359, 2013.
[9] Thandar Lwin and Thi Thi Soe Nyunt, "An Efficient Duplicate Detection System for XML Documents," IEEE International Conference on Computer Engineering and Applications (ICCEA) , pp. 178-182,2010.
[10] Rohit Ananthakrishna, Surajit Chaudhuri, and Venkatesh Ganti, "Eliminating Fuzzy Duplicates in Data Warehouses," Proc. Conf. Very Large Databases (VLDB), pp. 586-597, 2002.
[11] Mauricio A. Hernández and Salvatore J. Stolfo, "The Merge/Purge Problem for Large Databases," Proc. ACM SIGMOD Conf. Management of Data, pp. 127-138, 1995.
[12] Luis Leitão, and Pável Calado, "Duplicate Detection through Structure Optimization," Proc. 20th ACM Int'l Conf. Information and Knowledge Management, pp. 443-452, 2011.
[13] Luis Leitão, and Pável Calado, "Efficient XML Duplicate Detection Using an Adaptive Two-level Optimization," Proc. 28th Annual ACM Symposium on Applied Computing, pp. 832-837, 202.
[14] Sutheetutt Vacharaskunee, Sarun Intakosum, "XML Path Matching for Different Hierarchy Order of Elements in XML Documents," 11th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, pp. 82-86, 2010.
[15] Dmitri V. Kalashnikov and Sharad Mehrotra, "Domain-Independent Data Cleaning via Analysis of Entity-Relationship Graph." ACM Trans. Database Systems, vol. 31, no. 2, pp. 716-767, 2006.
[16] Joe Fawcett, Liam R. E. Quin, Danny Iyers, "Beginning XML," John & Wile & Sons Publications, 5Th Edition,2012.
[17] Harris Drucker, Chris J.C. Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik, "Support Vector Regression Machines," Proc. Advances in Neural Information Processing Systems (NIPS), vol. 9, pp. 155-161, 1996.
[18] Joyce C. P. Carvalho and Altigran S. da Silva, "Finding Similar Identities among Objects from Multiple Web Sources," Proc. CIKM Workshop Web Information and Data Management (WIDM), pp. 90-93, 2003.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 8, August 2015

[19] Melanie Weis and Felix Naumann, "Detecting Duplicate Objects in XML Documents," ACM International Workshop on Information Quality in Information System (IQIS-04), pp. 10-19, 2004.
[20] Z. Nie, Y. Zhang, J.-R. Wen, and W.-Y. Ma, "Object-Level Ranking: Bringing Order to Web Objects," Proc. Int'l Conf. World Wide Web (WWW), pp. 567-574, 2005
[21] L. Chen, L. Zhang, F. Jing, K.-F. Deng, and W.-Y. Ma, "Ranking Web Objects from Multiple Communities," Proc. 15th ACM Int'l Conf. Information and Knowledge Management, pp. 377-386, 2006.