International Journal of Scientific Engineering and Technology (ISSN : 2277-1581)
Volume No.2, Issue No.9, pp : 895-899 1 Sept.2013

Record Matching in Web Databases Using Unsupervised Approach

Fouzia Sultana, Manjusha Kalekuri
Department of Computer Science& Engineering, Muffakam Jah College of Engineering & Technology, Banjara-Hills,Hyderabad-500034, INDIA.
Email: fouzia.sultana114@gmail.com, manjushakalekuri@gmail.com

Abstract— ***Record Matching is the problem of combining information from multiple heterogeneous databases. One step of data integration is relating the records that appear in the different databases specifically, determining which sets of records refer to the same real-world entities. Performing record matching solves the duplication detection problems; hence the needs for identifying the suitable record matching technique follow. Most of record matching methods are supervised, which requires the user to provide training data. These methods are not applicable for the Web database scenario, where the records to match are query results dynamically generated. To overcome the problem, a new record matching method named Unsupervised Duplicate Detection (UDD) is proposed which, for a given query, can effectively identify duplicates from the query result records of multiple Web databases and eliminating duplicates among records in dynamic query results. The idea of this paper is to adjust the weights of record fields in calculating similarities among records. Two classifiers namely weight component similarity summing classifier and support vector machine classifier are iteratively employed with UDD to identify duplicates in the query results from multiple Web databases.***

Keywords— **Record Matching, Unsupervised, UDD, Query Results**

I. Introduction

Web Database is a database application that is designed to be managed and accessed through the internet. Such Databases produces the result dynamically in response to the given query. Most Web databases are only accessible via a query interface through which users can submit queries. Several data sets is often required as information from multiple sources needs to be integrated, combined or linked in order to allow more detailed data analysis or mining.

Record matching can be done by supervised learning where training dataset is required beforehand. In the web databases, the result records are obtained through online queries. Most previous work is based on predefined matching rules hand-coded by domain experts or matching rules learned offline by some learning method from a set of training examples. Such approaches work well in a traditional database environment. The representative training set in supervised learning cannot be

applicable for the web results that are generated on-the-fly. For each new query, depending on the results returned, the field weights should probably change too, which makes supervised-learning based methods even less applicable.

Hence, we define an unsupervised technique named Unsupervised Duplicate Detection (UDD) which uses two classifiers for record matching and duplicate detection. This eliminates the user preference problem in supervised learning. In this paper, by employing two classifiers that collaborate in an iterative manner, UDD identifies duplicates based on the dissimilarity among these records, field's weight is set and record matching is done by the first classifier. These results i.e., the matched records form the duplicate or positive set. The second classifier uses both duplicate and the non-duplicate sets to identify the remaining duplicate record pairs.

II. RELATED WORK

Duplicate record detection is the process of identifying different or multiple records that refer to one unique real world entity or object. Typically, the process of duplicate detection is preceded by a data preparation stage during which data entries are stored in a uniform manner in the database, resolving the structural heterogeneity problem. The data preparation stage includes parsing, data transformations, and a standardization step [ii]. The duplicate detection methods can be broadly divided into two categories: Approaches that rely on training data to "learn" how to match the records. This category includes (some) probabilistic approaches and supervised machine learning techniques. Approaches that rely on domain knowledge or on generic distance metrics to match records. This category includes approaches that use declarative languages for matching and approaches that devise distance metrics appropriate for the duplicate detection task [ii].

Supervised learning, consist of two steps training and prediction. In first step a number of labeled examples are usually required for training an initial predictor which is in turn used for exploiting the unlabeled examples in second step. One of the problem with the supervised learning techniques is the requirement for a large number of training examples. While it is easy to create a large number of training pairs that are either clearly nonduplicates or clearly duplicates, it is very difficult to generate ambiguous cases that would help create a highly accurate classifier. However, in many real-world applications there may exist very few labeled training examples, which makes the predictor difficult to generate, and therefore these supervised learning methods cannot be applied.

In Unsupervised learning the users are not required to provide training data. Thus for real time applications for example in this case to solve the problem of record matching from multiple web databases unsupervised learning methods are applicable. This can be used in a web database scenario where the matching results depends could be query dependant. In previous work Christen's method [iv] and PEBL[iii] are the two work, which are used for performing the record matching operation and are highly related to our UDD method.

III. Duplicate Detection in UDD

This paper aims at studying the existing strategies to match the records from multiple web databases for the given query, and ultimately proposes an unsupervised duplicate detection method to identify duplicates. The objective of this paper is to improve overall system performance by addressing the problem of record matching in the Web database scenario, which for a given query can effectively identify duplicates from the query result records of multiple Web databases. UDD is basically designed for web databases where records to match are highly Query dependent and each record is defined with multiple fields.

The key concepts of our method are: we focus on weight assignment scheme to assign the weights to each field of a record , which later is used to calculate the similarity between each candidate pair of records to determine whether they are duplicate or not. We have collected the sample of universal data consisting of record pairs from different sources as negative training dataset.

Duplicate Detection Process –

The main aim of UDD is to identify the matching status of each of these records retrieved from multiple web data sources as duplicate and nonduplicate This is also related to classification problem of each record using only a single class of training example i.e. negative. UDD consist of following steps

- Generation of Similarity Vectors
- Computation of Potential duplicate vector set P and Non duplicate vector set N
- Component Weight Allocation
- Similarity Score calculation
- Initial Duplicate Identification using WCSS Classifier C1
- Identifying remaining duplicates from P using SVM Classifier C2
- Identifying actual duplicate vector set D

A. Generation of Similarity Vectors

Similarity vectors are used to represent each field's similarity value between two records. Suppose for given query we have retrieved two r1 and r2,independent of whether they have received from the same source or different sources their similarity vector can be represented as V12= <v1,v2,….,vn>, in which vi represent the ith field similarity between r1 and r2. We call a similarity vector formed by a duplicate record pair a duplicate vector and a similarity vector formed by a nonduplicate record pair a nonduplicate vector.

B. Computation of Potential duplicate vector set P and Non-duplicate vector set N

A nonduplicate vector set N consist of all similarity vectors formed by any two different records from the same data source. A potential duplicate vector set P consists of all similarity vectors formed by any two records from different data sources.

C. Component Weight Allocation

Weights are assigned to each component of a record to indicate the importance of its corresponding field. The Dynamic allocation of weights to different fields in each record is performed by the Dynamic Weight Allocation Algorithm. It considers both Duplicate Vector D and Non-duplicate vector N[i]. The weights are assigned in the following manner for both Duplicates and Non-duplicate vectors: For duplicate vectors we design the following weight assignment scheme considering all duplicate vectors in D:

$$pi = \sum_{v \in D} vi \quad (1)$$

and

$$wdi = pi \ / \sum_{j=1}^{n} pj \quad (2)$$

in which pi is the accumulated ith component similarity value for all duplicate vectors in D and wdi is the normalized weight for the ith component.

According to the nonduplicate intuition, we use the following weight assignment scheme considering all non-duplicate vectors in N:

$$qi = \sum_{v \in N} (1 - vi) \quad (3)$$

and

$$wni = qi \ / \sum_{j=1}^{n} qj \quad (4)$$

in which, q_i is the accumulated ith component dissimilarity value for all nonduplicate vectors in N and w_{ni} is the normalized weight for the ith component .

D. Similarity Score Calculation

Any similarity functions can be employed in UDD approach [vii], [viii]. Initially weight vector is initialized in such a way that sum of the weight components do not exceed 1. The similarity Sim (v1,v2) is calculated for each candidate pair of the records. A similarity threshold value is set to identify initial potential duplicate vector and nonduplicate vector set. Two

records r1 and r2 are duplicates if their similarity value is equal to or greater than a similarity threshold.

E. Duplicate Identification using WCSS Classifier C1

Classifiers that need class information to train, such as decision tree and Naive Bayes, cannot be used here to identify duplicate because no duplicate vectors are available initially. WCSS Classifier is used to detect initial duplicate from both non-duplicate vector set N and Potential duplicate vector set P especially when there are no positive examples are available. Identified duplicates from P and N are placed into actual duplicate vector set D.

F. Support Vector Machine Classifier C2

Support Vector Machines, a promising new method for the classification of both linear and nonlinear data. It uses a nonlinear mapping to transform the original training data into a higher dimension. Within this new dimension, it searches for the linear optimal separating hyperplane (that is, a "decision boundary separating the tuples of one class from another). The SVM finds this hyperplane using *support vectors* ("essential" training tuples) and *margins* (defined by the support vectors). In our proposed method classification is performed on Potential duplicate vector (P) and new actual duplicates are identified. The Classifier is trained using actual duplicate vector (D) and Non-duplicate vector. Support Vector Machine [v], is insensitive to the number of training examples. As our algorithm will be used for online duplicate detection, we use a linear kernel, which is the fastest, as the kernel function in our experiments.

Duplicate Vector Detection Algorithm:

Algorithm:
Step 1: Set the parameters W of C1 according to N
Step 2: Use C1 to get a set of duplicate vectors d1 from P and a set of duplicate vectors f from N
Step 3: Remove the identified duplicates from P and N and place into actual duplicate vector set D.
Step 4: Train C2 using D and N'
Step 5: Classify P using C2 and get a set of newly identified duplicate vector pairs
Step 6: Adjust the parameters W of C1 according to N´ and D
Step 7: Use C1 to get a set of duplicate vectors d1´ from P and a set of duplicate vectors f´ from N
Step 8: Return D, repeat the process until no new duplicates are identified by C1.

IV. Results and Tables

The proposed method can be applied to various dataset to detect the duplicates. This is best suitable for web databases like Movie, Book and Hotel Booking etc.

To show the applicability of our algorithm for detecting duplicate query results collected from Web databases, we have collected some real Web data from Web databases in Book domain. We collected records with Title, Author, Price, and ISBN fields. In our experiments, we run the UDD algorithm on the full data set with all four fields. Our system is successful in detecting most of duplicate book records obtain from multiple web databases.

The concept of this paper is implemented and different results are shown below, The proposed paper is implemented in Java technology on a Pentium-IV PC with 20 GB hard-disk and 1GB RAM. The propose paper's concepts shows efficient results and has been efficiently tested on different online Datasets.

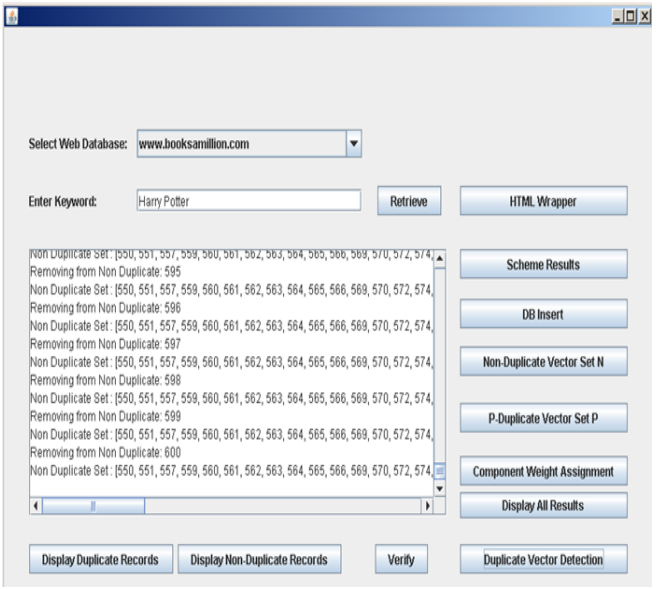

Figure1- Duplicate vectors Detection

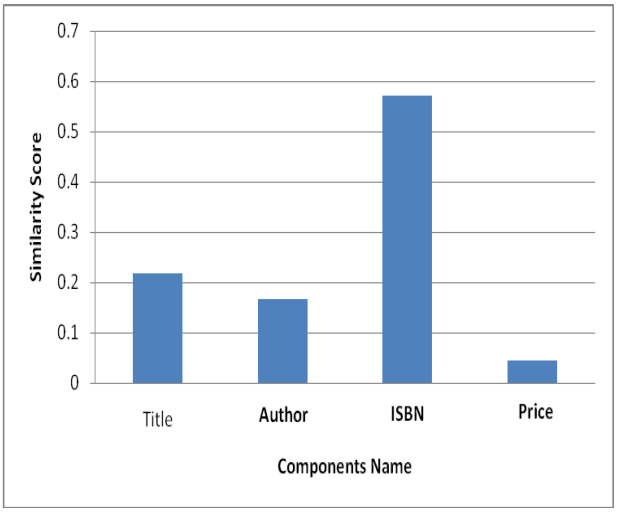

Figure 2. Component Weight Allocation

Figure 2 showss weights calculated for each fields of a record. From the figure we can say that ISBN having the more similarity score compare to remaining fiels similarity score,it means that ISBN has more importance compared to remaining fields.

Evaluation Metrics

To measure the effectiveness of the system three ratios are used: precision, recall and F-measure which are defined as follows:

$$\text{Precision} = \frac{\#\text{of Correctly Identified Duplicate Pairs}}{\#\text{of All Identified Duplicate Pairs}}$$

$$\text{Recall} = \frac{\#\text{of Correctly Identified Duplicate Pairs}}{\#\text{of True Duplicate Pairs}}$$

These commonly used accuracy measures are not very suitable for assessing the quality of record matching, due to the usually imbalanced distribution of matches and nonmatches in the weight vector set [vi]. Thus, we also use the F -measure, which is the harmonic mean of precision and recall, to evaluate the classification quality [ix]. This is given aS

$$F - \text{measure} = \frac{2 * \text{Precision} * \text{recall}}{(\text{precision} + \text{recall})}$$

Experimental Results

We have computed the Precision, Recall and F-measure for different Query terms to see the performance of our system. The table below contains the detail. The same calculations are also shown by plotting the bar graph.

Name of the Book	Harry Potter	Kite Runner	Life of Pi	Twilight	Jane Eyre
Precision	0.89	0.95	0.91	0.93	0.95
Recall	0.79	0.87	0.86	0.76	0.95
F-measure	0.84	0.90	0.88	0.84	0.95

Table1-Computation of Evaluation Metrics for different Keywords

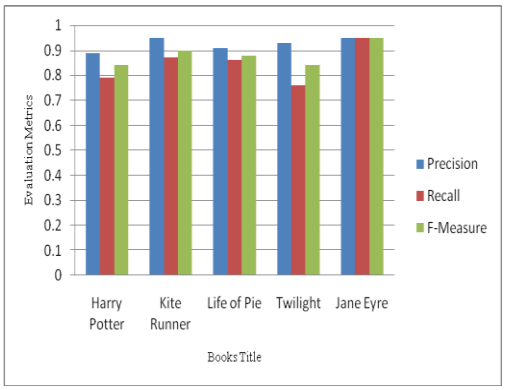

Figure3- Precision, Recall and F-measure graph

Comparison with Other Classification Methods

Method	Precision	Recall	F-measure
UDD	0.926	0.846	0.884
PEBL	0.902	0.803	0.851
Christen	0.886	0.837	0.876

Table 2- Performance Comparison between UDD and Other Learning Method

Table 2 compares the performance of UDD with the existing methods i.e PEBL and Christens method. From the table-2 we can conclude that UDD outperforms both PEBL and Christen's method on both precision and recall. This is because of the two classifiers in UDD alternately cooperating inside the iterations while in both PEBL and Christen's method; there is only one classifier in the iteration.

IV. Conclusion

In this paper, we propose an effective approach to improve the performance of the record matching to solve duplication detection. To overcome these problems, a better approach for an existing unsupervised, online approach, for detecting duplicates over the query results of multiple Web databases has been discussed i.e., an unsupervised online approach called Unsupervised Duplicate Detection (UDD) is presented. It uses two classifiers namely WCSS and SVM does not require any pre-labeled training examples. It detects the duplicate records from the results generated on-the-fly. It does not suffer from user preference problems.

Acknowledgement

First and foremost, I praise and thank ALMIGHTY GOD whose blessings have bestowed in me the will power and confidence to carry out my work. We also thank Dr. A. A. Qyser, the Head of the Department of Computer Science & Engineering, for his continuous support in providing the facilities for conducting the Research. We also express our thanks to the Principal, Dr. Basheer Ahmed for encouraging the staff & students to conduct research and publish papers.

References

[i] *Weifeng Su, Jiying Wang, and Frederick H. Lochovsky, —Record Matching over Query Results from Multiple Web Databases,‖ IEEE Transaction Knhe opowledge and Data Engineering, April 2010 (vol. 22 no. 4) pp. 578-589.*

[ii] A.K. Elmagarmid, P.G. Ipeirotis, and V. S. Verykios. —Duplicate Record Detection: A Survey‖, IEEE TKDE, 19(1):1-16, 2007.

[iii] H. Yu, J. Han, and C.C. Chang, ―PEBL: Web Page Classification without Negative Examples,‖ IEEE Trans. Knowledge and Data Eng., vol. 16, no. 1, pp. 70-81, Jan. 2004D. Kunder, "Multi-resolution Digital Watermarking Algorithms and Implications for Multimedia Signals", Ph.D. thesis, university of Toronto, Canada, 2001.

[iv] P. Christen, ―Automatic Record Linkage Using Seeded Nearest Neighbour and Support Vector Machine Classification,‖ Proc.ACM SIGKDD, pp. 151-159, 2008

[v] V. Vapnik, The Nature of Statistical Learning Theory, second ed.Springer, 2000.

[vi] P. Christen and K. Goiser, "Quality and Complexity Measures for Data Linkage and Deduplication," Quality Measures in Data Mining, F. Guillet and H. Hamilton, eds., vol. 43, pp. 127-151,Springer, 2007.

[vii] N. Koudas, S. Sarawagi, and D. Srivastava, "Record Linkage: Similarity Measures and Algorithms (Tutorial)," Proc. ACM SIGMOD, pp. 802-803, 2006

[viii] L. Gravano, P.G. Ipeirotis, H.V. Jagadish, N. Koudas, S. Muthukrishnan, and D. Srivastava, "Approximate String Joins in a Database (Almost) for Free," Proc. 27th Int'l Conf. Very Large Data Bases, pp. 491-500, 2001.

[ix] R. Baeza-Yates and B. Ribeiro-Neto, Modern Information Retrieval. ACM Press, 1999.

[x] L.M. Manevitz and M. Yousef, "One-Class SVMs for Document Classification," J. Machine Learning Research, vol. 2, pp. 139-154, 2001.