ISSN:0975-9646

Unsupervised Detection of Duplicates in User Query Results using Blocking

Dr. B. Vijaya Babu, Professor [1], K. Jyotsna Santoshi [2]

[1] *Professor, Dept. of Computer Science,*
K L University, Guntur Dt., AP, India

[2] *M.Tech Student, Dept. of Computer Science,*
K L University, Guntur Dt., AP, India

Abstract— **Unsupervised learning involves exploring the unlabeled data to find some intrinsic or hidden structures. Duplicate detection enables to identify the records that represent the same real world entity. In the field of Data mining, there is an exponential growth in the amount data available. Thus, linking or matching records from various web databases is a major challenge as it involves complexity of comparing, each record in one database with all the records in other databases. Supervised learning methods fail in web database scenario as the records to be matched are query dependent. In the previous work, to handle this context, an online based untrained record linkage method, UDD was suggested. UDD proficiently identifies corresponding record pairs that represent same entity, from multiple web databases but is time consuming. This paper focuses on enhancing the performance of UDD by adding a blocking step. A computationally cheap clustering approach, Canopy Clustering is deployed in blocking step. Thus, prior to classifying records into duplicates and non-duplicates, clustering is performed and blocks of candidate record pairs are generated. Experimental results show that blocking optimizes the working of UDD in web database scenario.**

***Index Terms*— Unsupervised learning, Record linkage, Duplicate Detection, Record Matching, Blocking, Canopy Clustering, Web database, Query result.**

I. INTRODUCTION

Enormous amount of data is available on web, stored in the form of web databases. The information from multiple heterogeneous databases is combined using the process of Data integration. Record Linkage or matching has an important role in data integration as linked data can contain information that is not otherwise available or would entail tedious and exorbitant collection of specific data. Record matching deals with the problem of identifying the records that describes the same real world entity. The Web databases consist of static web and hidden or deep web. The deep web includes well-structured and high quality information that is expected to have a faster growth rate as compared to static web. Most of the web databases produce web pages dynamically in response to user queries. These databases contain various forms of data such as text, images, documents, and links. To access the information within web databases, user provides queries through query interface. As the query is submitted, the web server processes the query and retrieves the corresponding results from the back-end database and displays to the user. Most of the Integration techniques exploit textual similarity of object names in different web databases for extracting useful information. The records that are co-referent are said to be duplicates. They may be exact duplicates or potential duplicates. The records that are exactly the same in all relevant matching fields are *exact* duplicates. The records with minute differences but represent same entity are referred to *Potential* duplicates. Using a straightforward preprocessing step, exact matching, can merge those records that are exactly the same in all relevant matching fields.

The problem of duplicate detection requires comparison of each record in a particular database with every other record in multiple databases. Consider two data sources A and B. Let there be s records in data source A and t records in data source B. Every record of A or B have a set of fields/attributes. Each of the t records in data source B can potentially be a duplicate of each of the s records in data source A. The duplicate identification aims at determining the matching status, i.e., duplicate or non-duplicate, of these $s \times t$ record pairs. Earlier, well trained record matching methods such as PEBL, Christen method, SVM classifiers are incorporated into search engines for mining and eliminating duplicate query results from different databases. In these works, linking records is based upon hand-coded rules by domain experts or matching rules trained by some learning methods using set of training examples. Such perspectives cannot to be applied to web database scenario as matching records is highly query-dependent. Moreover, online queries represent only a partial and biased portion of all the data in the source web databases. Accordingly, hand-coding or offline-learning approaches are ill-suited.

In the preceding work, an innate solution to online duplicate detection problem for specific type of records, Unsupervised Duplicate Detection (UDD) was propounded. This methodology befits to web databases from the same domain i.e., web databases that display same type of records in response to user queries. The key features of UDD include adjusting the weights of the record fields accordingly during similarity calculation and consider dissimilarity among the records i.e., their relative distance. It

operates using two classifiers namely WCSS and SVM that co-operate with each other in a continual manner. In record linkage techniques, each record in one dataset potentially has to be compared with all records in second dataset. As such, the number of candidate record pair comparisons to be made multiplies quadratically with the size of datasets. This approach becomes computationally impractical for high dimensional data sets. In such context, blocking technique can be effectual recede the number of record comparisons. It ideally deals with quadratic complexity and efficiently selects a subset of record pairs for subsequent similarity computation by disregarding dissimilar pairs as inapt. The choice of blocking technique is very crucial as an adaptive and scalable blocking method reduces the variance in the quality of the blocking results and renders speed and accuracy resulting in a better performance. Various blocking methods include Standard Blocking [4], Sorted neighbourhood [6],

Bigram indexing [5], and Canopy Clustering with TF-IDF [10]. Blocking methods directly affect sensitivity since, if record pairs of true matches are not in the same block, they will not be compared and can never be matched and indirectly affect specificity as a better reduction ratio of the number of record pair comparisons allows more computationally rigorous comparators to be incorporated[2]. Among these accessible techniques, Canopy clustering effectively demarcates the data into overlapping called *Canopies,* subsets by utilizing a cheap approximate distance measure. This technique provides less computational cost accompanied by no loss in clustering accuracy. Canopy Clustering can be befitting to many other key clustering algorithms as Greedy Agglomerative Clustering, K-means and Expectation Maximization.

II. LITERATURE REVIEW

Every record matching framework composes of two prime steps:

i) Similarity assessment using metrics
ii) Linking records.

The most remarkable works closely associated with UDD are discussed in [19] and [20].

According to [19], Christen method involves unsupervised record pair classification that encompasses two stages namely comparison step and convergence step. Here, highly similar pairs are positive examples, whereas highly dissimilar pairs are negative examples. Initially, high quality representative training data are instinctively opted from the compared record pairs. The samples are used by SVM classifier to learn. The first step involves a Nearest-neighbour classifier which improvises the performance of SVM by providing more no. of samples to training sets. The key disadvantages of this method include:

i. In this approach, adjustment of weights is unvarying unlike UDD where there is a provision for dynamic weight assignment.
ii. At the time of iterations, single classifier is involved.

The PEBL [20] work deals with positive example based learning for classification of web pages. This strategy works us classification problem using two steps, which is termed as Mapping-Convergence algorithm. The first stage applies a weak classifier to derive strong negative and approximated examples from unlabeled data. It the next stage, an internal classifier SVM is trained by the positive examples and negative examples from the previous step. It operates iteratively to detect further duplicates until it converges. This makes use of results extracted by same classifier as retraining examples in the following iteration. Thus, failing in retrieval of better hypothesis.

The paper [10] states the necessity of clustering the high dimensional datasets. It suggests clustering method, canopy clustering to deal with multi-dimensional datasets that can be grouped efficiently with minimal computation cost. It requires a cheap distance metric for apportioning the data into overlapping subsets. This method is proved to enhance performance and speed whilst increasing the accuracy slightly. An interesting feature of this clustering is that it adds all the items belonging to any true cluster to same canopy thereby, ensuring no loss in accuracy. This restricts the comparisons of items within the same canopy only. The canopy approach successfully dealt with reference matching problem associated with bibliographic citations.

A mixed variety of string similarity metrics are available for matching purposes. Several works have conducted to analyse the precision of these string metrics. These metric-categories are edit-distance metrics, numeric similarity metrics, token-based distance metrics, fast heuristic string comparators and hybrid methods [9], [12]. These functions depicts entity pairs in form of vectors rather than strings, so that, they can be applied to entities with non-trivial structures. Based on methodology involved and requisite scalability, any function can be used. Additionally, genuine user interest and domain expertise can also be considered.

A great number of approaches and algorithms are available for approximate duplicate detection [13]. These approaches methods can be widely categorized into two:

i. Approaches that count on training examples. These are exemplified by probabilistic approaches and supervised machine learning techniques.
ii. Approaches that count on domain knowledge or on generic distance metrics. Typically, declarative languages for matching and distance metrics based approaches are under this category.

Major categories of duplicate detection models are:

i. Probabilistic matching models
ii. Supervised and semi-supervised learning models
iii. Active learning based techniques.
iv. Distance based learning.
v. Rule-based approaches
vi. Unsupervised learning.

Blocking can be performed using several available methods [2],[3] and [14]. Entity matching, Clustering and schema mapping algorithms are associated with Pairwise similarity

computations. Since the computational cost of similarity assessment between instances of record grows quadratically with the dataset size, computing similarity between all entity pairs becomes speculative and quixotic for large datasets and complex similarity functions. Blocking alleviate this complication. Well-established and traditional blocking methods are standard blocking and Sorted Neighbourhood method. Recent methods include Bigram indexing, Canopy clustering with TF-IDF. In contrast to former methods, latter methods enable to achieve greater performance speed-ups and preferable accuracy. The standard Blocking (SB) method assembles records with corresponding blocking keys into the same cluster. Usually, it performs $O(n^2/b)$ candidate pair comparisons, where n denotes no. of records in each data source and b is no. of blocks. The Sorted Neighbourhood (SN) method creates a sorting key for each record and sorts them. A window of fixed size is moved progressively over the sorted records and thus, pairing records within same window which creates list of candidate record pairs. The resultant no. of comparisons of the sorted neighbourhood method is given by $O(wn)$, where n is no. of records in each dataset and w is window size. The Bigram Indexing (BI) method, list of bigrams used as blocking key values. A bigram is a sub string of two characters. These bigram lists are organized as inverted index, which will be used for retrieval of record with corresponding number. Like standard blocking, BI methods also results in $O(n^2/b)$ comparisons. Canopy clustering will be further discussed in section 4.

III. UDD APPROACH

UDD methodology yields a potent solution to online duplicate identification and elimination of co-referent record pairs from heterogeneous web databases.

This method works on the basis of certain assumptions and observations. It adopts two influential assumptions:

1. A global schema that exemplifies the specific type of records is pre-defined and user query schema has to be compared to the global schema.
2. Each source database can be accessible via record extractor i.e., wrappers and then, insert the resultant data into a relational database as per global schema

Apart from these assumptions, it utilizes the following two observations:

Apart from these assumptions, it utilizes the following two observations:

1. Same source records consistently reflect the same format.
2. Exact matching method plays a pivotal role in the removal of same data source duplicates.

As a part of pre-processing step, exact matching eliminates the records that are same in all corresponding fields. It productively reduces the duplicate ratio. Duplicate ratio (d) can be defined as the ratio of number of duplicates generated to the number of record pairs taken from a data source.

$$d = \frac{t}{n}$$

Where no. of record pairs, n= m (m-1)/2;

m= no. of records extracted from a data source.

t= no. of duplicate record pairs.

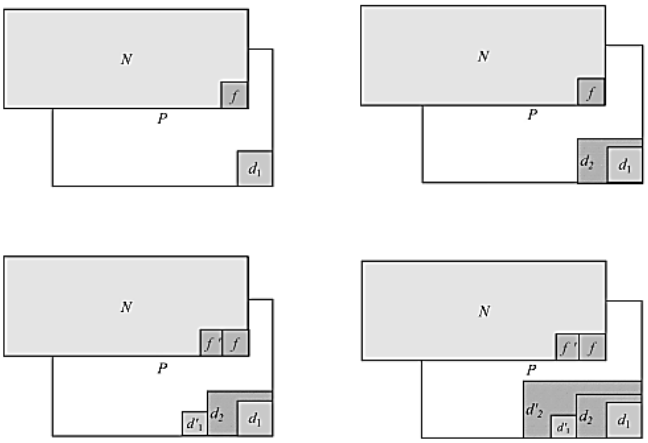
A. Similarity Computation

At the beginning, as a part of pre-processing stage, exact-matching was adapted to clear away exact duplicates. Then, similarity computation was done and similarity vector for each pair of records depicted as $V_{12} = <v_1, v_2, \ldots v_n>$.Here, record pair can be represented as $P_{12}=\{r_1,r_2\}$. The similarity between record pairs can be assessed using one or more similarity functions [7], [8]. In general, similarity score occurs within 0-1. In this approach, TF-IDF was applied which is a transformation-based similarity metric. TF-IDF stands for Term Frequency- Inverse Document Frequency. The frequency or occurrence of a token in a string interpreted as Term frequency and the reciprocal or inverse of number of strings containing fields with tokens interpreted as Inverse Document Frequency. A token's TF is usually 1 and as the user query contains tokens that appear in multiple strings in a field in varying manner, IDF recede the weight assigned to tokens. Consider a set of documents D, a word w, an individual document d ϵ D, [21], weight can be determined as:

$$w_d = f_{w,d} * log(|D| f_{w,D}) \quad (2)$$

In this context, similarity vectors are of two types:

i. Duplicate vector that construes duplicate vector pair similarity
ii. Non-duplicate vector that construes non duplicate pair similarity.



B. UDD Algorithm

Two vector sets forms the basis of this algorithm:

1. Non- Duplicate vector set N that comprises of vectors of record pairs from same data sources.
2. Duplicate Vector set P that comprises vectors of record pairs from heterogeneous data sources.

It employs two classifiers in an iterative and co-operative manner:

i. WCSS ii. SVM

Initially, a classifier is trained using vector set N and the learned classifier along with another co-operating classifier works on vector set P. The operation of UDD resembles to PEBL and Christen's method.

Input: Potential duplicate vector set P
Non-duplicate vector set N
Output: Duplicate vector set D
C_1: a classification algorithm with adjustable parameters W that identifies duplicate vector pairs from P
C_2: a supervised classifier, e.g., SVM

Algorithm:
1. $D = \emptyset$
2. Set the parameters W of C_1 according to N
3. Use C_1 to get a set of duplicate vector pairs d_1 from P
4. Use C_1 to get a set of duplicate vector pairs f from N
5. $P = P - d_1$
6. While $|d_1| \neq 0$
7. $N' = N - f$
8. $D = D + d_1 + f$
9. Train C_2 using D and N'
10. Classify P using C_2 and get a set of newly identified duplicate vector pairs d_2
11. $P = P - d_2$
12. $D = D + d_2$
13. Adjust the parameters W of C_1 according to N' and D
14. Use C_1 to get a new set of duplicate vector pairs d_1 from P
15. Use C_1 to get a new set of duplicate vector pairs f from N
16. $N=N'$
17. Return D

C. Component Weight Assignment

The summation of all component weights comes to 1. A component's weight implies the significance of corresponding field. The assignment intuits two fundamental rules:

i. Duplicate Intuition: The similarity score of duplicate vector pair must be close to 1.
ii. Non-duplicate Intuition: The similarity score of non-duplicate vector pair must be close to 0.

Input: Duplicate vector set D
Non-duplicate vector set N
Weighting scheme co-efficient a
Output: Component Weight W

Algorithm:
1. For i=1 to n
2. p_i=0
3. q_i=0
4. For each vector $V_k=\{v_{k1},\ldots, v_{kn}\}$ in D
5. $p_i = p_i + v_{ki}$
6. $S = \sum_{i=1}^{n} p_i$
7. For i=1 to n
8. $w_{di} = p_i / S$
9. For each vector $V_k=\{v_{k1},\ldots, v_{kn}\}$ in N
10. $q_i = q_i + 1 - v_{ki}$
11. $S = \sum_{i=1}^{n} q_i$
12. For i=1 to n
13. $w_{ni} = p_i / S$
14. $w_i = a \cdot w_{di} + (1-a) w_{ni}$
15. Return $W=\{w_1,\ldots, w_n\}$

1. Duplicate Vector weight assignment scheme:

For all duplicate vectors in vector set D,

$$p_i = \sum_{v \in D} v_i$$

where p_i gives cumulative i^{th} component similarity; $v_i = i^{th}$ field similarity between two records r_1 and r_2.

$$w_{di} = \frac{p_i}{\sum_{j=1}^{n} p_j}$$

where w_{di} = i^{th} component's normalized weight.

2. Non-duplicate Vector weight assignment scheme:

For all non-duplicate vectors in vector set N,

$$q_i = \sum_{v \in N} (1 - v_i)$$

where q_i gives cumulative i^{th} component dissimilarity; $v_i = i^{th}$ field similarity between two records r_1 and r_2.

$$w_{ni} = \frac{q_i}{\sum_{j=1}^{n} q_j}$$

where w_{ni} = i^{th} component's normalized weight.
Combining these two intuitions, a more rational weighting strategy was framed as:

$$w_i = a.w_{di} + (1-a)w_{ni}$$

where a ϵ [0,1] emphasizes the significance of duplicates and non duplicate vectors.Initially, 'a' value is set to 0. It is incremented by 0.5 for consecutive iterations.

D. Classifiers:

1)Weighted Component Summing Classifier(WCSS):

WCSS filters out duplicate vectors even in the absence of positive examples. Collaborating with the other classifier, it further identifies the duplicates in subsequent iterations. It utilizes the field similarity and weight of each component to evalaute the similaity between given records r_1 and r_2.

$$\text{Sim}(r_1, r_2) = \sum_{i=1}^{n} (W_i . F_i)$$

where $w_i \epsilon$ [0,1]

$$\sum_{i=1}^{n} (W_i) = 1$$

$\text{Sim}(r_1, r_2)$ occurs within the range of 0-1. A threshold value T_{sim} is set up.
When the record pair $\text{Sim}(r_1, r_2) \geq T_{sim}$, threshold value, it is said to be Duplicate.

2. Support Vector Machine (SVM):

The duplicate vectors with scores greater than threshold are identifed as Positive examples by WCSS. These examples are usd to train SVM classifier. This classifier furthers identifies duplicates from vector sets P and N'. the identified duplicate vectors of D set are *Positive examples*. The remnant non-duplicate vectors in set N' are *Negative*

examples. Support Vector Machine is very fitting in this approach as it exhibits certain features:

i. SVM is inconsiderate towards respective sizes of positive and negative examples, critically at the first step.
ii. Limited no .of training examples should not effect the performance and final result.

IV. Optimizing UDD With Blocking

Data preprocessing cleans and standardizes the real-world data that is otherwise incomplete, inconsistent, noisy and unformatted. Usually, after data cleaning and standardization, comparison functions are used to compute similarity scores. Thereafter, record pair comparison and classification follow. If the large databases are being linked or deduplicated, the process may become incommodious and error-prone leading to complexity. In the current work, blocking has been subsumed as a part of record linkage process. Blocking is a type of indexing technique that aims at reducing the number of potential record pair comparisons by excluding some pairs as non-matches. This technique segregates the database records into non-overlapping blocks, as such only records within a particular block are compared against each other.

In this section, we describe the metrics used for similarity vector calculation in 4.1, steps involved in clustering records into canopies in 4.2 and then, overall methodology.

A. Similarity Calculation:

As aforementioned, any similarity can be implemented with UDD. Since Canopy Clustering has been adopted for blocking, two similarity measures, namely, TF-IDF and Jaccard coefficient [22] are used for similarity calculation in this paper. These metrics considers tokens that may be characters, words or q-grams. Jaccard similarity between any pair of records is given by

$$S_J = \frac{|\text{token}(r_1) \cap \text{token}(r_2)|}{|\text{token}(r_1) \cup \text{token}(r_2)|}$$

where $s_J \epsilon [0, 1]$.

Since TF-IDF demands for additional information to be evaluated and stored, S_J has better average precision as it neither requires frequency information nor normalization. Thus, TF-IDF metric is computationally more expensive.

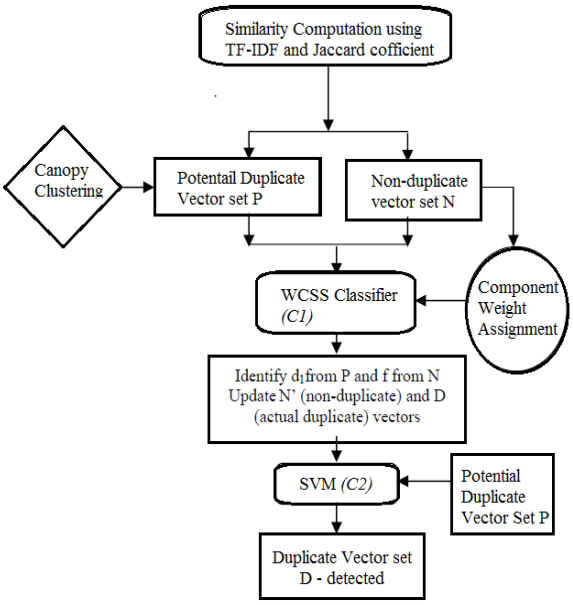
Metric	MaxF1	AvgPrec
SFS	0.528	0.357
TFIDF	0.518	0.369
Jaccard	0.567	0.402
L2 Jaro-Winkler	0.746	0.770
SoftTFIDF	0.685	0.782
Jaro-Winkler	0.648	0.703
Jaro	0.687	0.731
NaiveAvgOverlap	0.697	0.731
AvgOverlap	0.701	0.736
Levenstein	0.832	0.901

B. Canopy Clustering:

Once similarity calculation is done, records are clustered using this technique. This method is applied to generate high dimensional overlapping clusters called as *Canopies.* Canopies constitute candidate record pairs. It proceeds by partitioning the data into overlapping subsets and then, performs more substantial clustering within each group.

STEPS:

i. A record r_c is selected randomly from a pool of records and this becomes the centroid of cluster.
ii. The distance of r_c is approximated to all other records in the pool.
iii. All the records that are near to distance threshold of r_c are added to that cluster. Thus, clusters are generated.



V. Experiments And Results

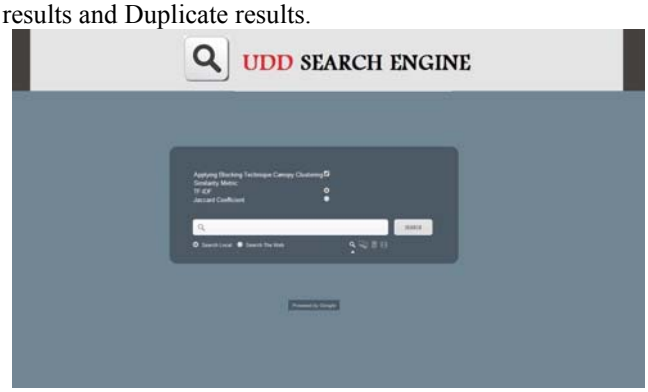
A. Application

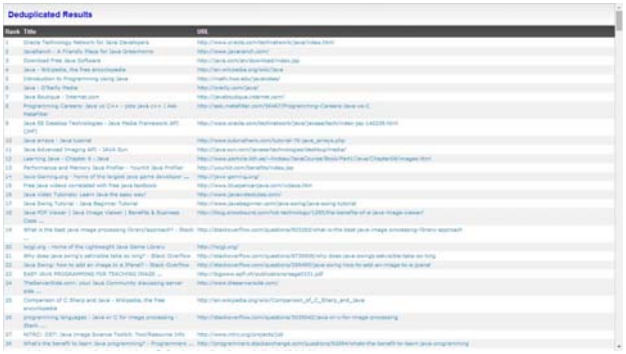
In this section, we describe the efficiency of UDD searching process, without and with application of canopy clustering to the relevant data sets within commercial process execution. For efficient execution of this step, in search engine management applications, domain knowledge about research process, relevant searching terminology with creative data bases is prerequisite. We developed an efficient and commercial user interface for retrieving relevant results by comparing to a user given query with all the available data records. Two search log data sets named as UDD and UDD2 were created and loaded onto MySQL. The algorithm code was implemented in java language on a system with Intel core i5 processor, running on Windows operating system. Various instances of time complexity are observed as under:

UDD	UDD with Canopy Clustering
1.2564	0.89654
0.987	0.6274
0.7856	0.02564
0.2563	0.1452
0.01452	0.00452

For example, when we search for a term 'java' in search engine, the query interface checks the similar words across the datasets available. In this context, every word follows

processing steps in real time database applications. The interface has provision to opt any one of the two metrics, TF-IDF and Jaccard. The search can be operated only with UDD and also applying canopy clustering to UDD. The results are displayed as separate sections as Deduplicated results and Duplicate results.





B. Evaluation Metric:

The operation of UDD approach and its enhancement are assessed by the following:

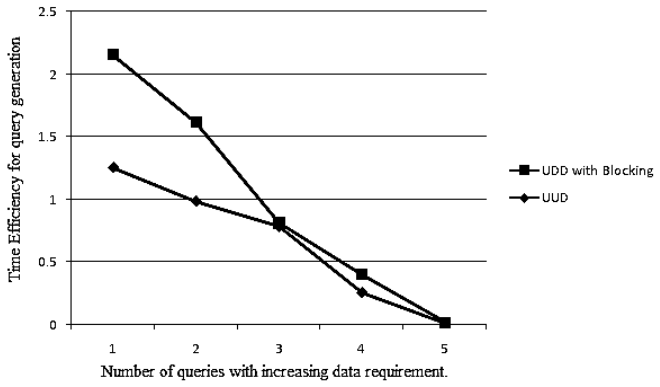
i. Precision
ii. Recall
iii. F-measure

Precision = # of Correctly Identified Duplicate Pairs / # of All Identified Duplicate Pairs

Recall = # of Correctly Identified Duplicate Pairs / # of True Duplicate Pairs

F-measure = 2 * Precision * Recall / (Precision+Recall)

The graph below depicts the overall performance of UDD and UDD approach with blocking, at particular intervals of time:



The application of canopy clustering for creating blocks in UDD approach makes the UDD to outperform the existing approach.

CONCLUSION

Unsupervised Duplicate Detection algorithm can serve as foundation for developing applications that involve mining heterogeneous web databases. Using an additional step of blocking can result in outperformance of UDD. Duplicate detection is an important problem that needs more attention and has some advantages over offline/supervised learning methods. When compared to traditional databases, in Web-based retrieval system, records to match are greatly query-dependent, a pre-trained approach is not appropriate as the set of records in response to a query is a biased subset of the full data set. UDD algorithm is an unsupervised, online approach for detecting duplicates and a suitable solution, when query results are fetched from multiple Web databases. The main advantage of UDD algorithm relies on using WCSS and SVM classifiers to assign weights and classify data. The performance of UDD algorithm has been enhanced by the application of scalable and computationally cheaper clustering technique. Partitioning the data into blocks reduces search space and matching complexity, maintaining linkage accuracy at the same time. UDD integrated with Canopy clustering is advantageous as compared other approaches for duplicate detection, especially in web database scenario.

REFERENCES

[1] Weifeng Su, Jiying Wang, Frederick H. Lochovsky, "Record Matching over Query Results from Multiple Web Databases" IEEE Transactions On Knowledge And Data Engineering, Vol. 22, No. 4, April 2010.

[2] B. R. Baxter, P. Christen, and T. Churches, "A Comparison of Fast Blocking Methods for Record Linkage," Proc. KDD Workshop Data Cleaning, Record Linkage, and Object Consolidation, pp. 25-27, 2003.

[3] Peter Christen, "A Survey of Indexing Techniques for Scalable Record Linkage and Deduplication", IEEE Transactions On Knowledge And Data Engineering, Vol. 24, No. 9, September 2012

[4] M. A. Jaro. Advances in Record Linkage Methodology as Applied to Matching the 1985 Census of Tampa, Florida. JASA, 84(406):414–420, 1989.

[5] P. Christen, T. Churches. Febrl: Freely extensible biomedical record linkage Manual, V0.2, http://datamining.anu.edu.au/linkage.html, April 2003.

[6] M. M. Hernandez and S. Stolfo. Real-world data is dirty: data cleansing and the merge/purge problem. J.DMKD, 1(2), 1998.

[7] 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.

[8] S. Chaudhuri, K. Ganjam, V. Ganti, and R. Motwani, "Robust and Efficient Fuzzy Match for Online Data Cleaning," Proc. ACM SIGMOD, pp. 313-324, 2003.

[9] Peter Christen, "A Comparison of Personal Name Matching: Techniques and Practical Issues", September 2006, Joint Computer Science Technical Report Series, The Australian National University, https://cs.anu.edu.au/techreports/2006/TR-CS-06-02.pdf

[10] A. McCallum, K. Nigam, and L. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In ACM SIGKDD, pages 169–178, 2000

[11] Peter Christen, "Towards Parameter-free Blocking for Scalable Record Linkage", August 2007, Joint Computer Science Technical

Report Series, The Australian National University, http://datamining.anu.edu.au/publications/2007/tr-cs-07-03.pdf
[12] William W. Cohen. Pradeep Ravikumar. Stephen E. Fienberg, "A comparison of string metrics for matching names and records", https://www.cs.cmu.edu/afs/cs/Web/People/wcohen/postscript/kdd-2003-match-ws.pdf.
[13] A.K. Elmagarmid, P.G. Ipeirotis, and V.S. Verykios, "Duplicate Record Detection: A Survey," IEEE Trans. Knowledge and Data Eng.,vol. 19, no. 1, pp. 1-16, Jan. 2007.
[14] M Bilenko, B Kamath, Raymond J. Mooney," Adaptive Blocking: Learning to Scale Up Record Linkage", University of Texas, Austin, http://wwwconference.org/www2006/iiweb2006.cs.uiuc.edu/20.pdf.
[15] W.W. Cohen and J. Richman, "Learning to Match and Cluster Large High-Dimensional Datasets for Data Integration," Proc. ACM SIGKDD, pp. 475-480, 2002.
[16] N. Koudas, S. Sarawagi, and D. Srivastava, "Record Linkage: Similarity Measures and Algorithms (Tutorial)," Proc. ACM SIGMOD, pp. 802-803, 2006.
[17] L.M. Manevitz and M. Yousef, "One-Class SVMs for Document Classification," J. Machine Learning Research, vol. 2, pp. 139-154,2001.
[18] S. Sarawagi and A. Bhamidipaty, "Interactive Deduplication Using Active Learning," Proc. ACM SIGKDD, pp. 269-278, 2002.
[19] P. Christen, "Automatic Record Linkage Using Seeded Nearest Neighbour and Support Vector Machine Classification," Proc. ACM SIGKDD, pp. 151-159, 2008.
[20] 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. 2004.
[21] Juan Ramos, "Using TF-IDF to Determine Word Relevance in Document Queries", Rutgers University, Piscataway, NJ http://www.cs.rutgers.edu/~mlittman/courses/ml03/iCML03/papers/ramos.pdf
[22] Suphakit Niwattanakul, Jatsada Singthongchai, Ekkachai Naenudorn and Supachanun Wanapu, "Using of Jaccard Coefficient for Keywords Similarity", Proceedings of the International MultiConference of Engineers and Computer Scientists 2013 Vol I.
[23] R. Baeza-Yates and B. Ribeiro-Neto, Modern Information Retrieval. ACM Press, 1999.
[24] 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.
[25] Aye Chan Mon, Faculty of Information and Communication Technology, University of Technology "Effective Blocking for Combining Multiple Entity Resolution Systems", IJCSE,Vol. 2 No.04 July 2013, http://www.ijcse.net/docs/IJCSE13-02-04-034.pdf
[26] S. Tejada, C.A. Knoblock, and S. Minton, "Learning Domain-Independent String Transformation Weights for High Accuracy Object Identification," Proc. ACM SIGKDD, pp. 350-359, 2002.

AUTHOR PROFILES

Dr B.Vijaya Babu is currently working as Professor in CSE department of K L University, Vaddeswaram, and Guntur Dt., Andhra Pradesh, INDIA. He has obtained B.Tech.,(ECE) degree from JNTU College of Engineering, KAKINADA, M.Tech.,(CSE)degree from JNTU College of Engineering ,ANANTAPUR and PhD degree from Andhra University, VISAKHAPATNAM .He has published several research papers in various International journals and attended International Conferences conducted in India .

Ms. K. Jyotsna Santoshi, received the B.E degree in Bioinformatics from AVIT college, Chennai, India in 2011. At present, she is pursuing M.Tech degree in CSE from K L Univeristy, Vaddeswaram, Guntur Dt., Andhra Pradesh, India.