Dis-Similarity Calculation Using Clustering Mechanism

A.Sumathi
Part–Time Ph.D (Category – B), R&D Centre, Bharathiar University, Coimbatore &
Assistant Professor, Department of Computer Science,
Navarasam Arts and Science College for Women, Erode, Tamil Nadu, India.
A.Venkatesh Kumar
Technology Specialist, Cognizant Technology Solution, Coimbatore, Tamil Nadu, India.

Abstract- This paper delineates evolution of Dis-Similarity percentage calculation using grouping technique. None of the existing algorithm produces the dis-similarity percentage between pair of string. There are two types of evolution model for duplicate detection i.e., duplicates detection without grouping and duplicate detection with grouping. This re-search proved that the duplicate detection with grouping is more powerful and performance wise also it is much better than duplicate detection without grouping. This research introduced new technique which includes merits and features of clustering algorithm and de-duplication algorithm to improve the performance and accuracy of the new technique.

Keywords- Duplicate Detection, De-Duplication, Dis-Similarity, Grouping, Clustering.

1. INTRODUCTION

Duplicate detection is based on entropy algorithm for discovering a finding by using dis-similarity calculation. This is a new solution for duplication problem. It employs entropy formula to calculate the homogeneity of a data for continuous attributes McCallum et al (2000) and Barrodaleand Ericson (1980). Gain is computed to estimate the gain produced by a split over an attribute. The quality of the result and the performance of the algorithm have been compared. The rest of the paper is organized as follows. In Section 2, the related studies rendering the fusion methods with brief review. The characteristics of ENTROPY are described in detail and the proposed method ENTROPY is established in detail way in Section 3. In Section 4, the characteristics of INFORMATION GAIN(IG) is described . In Section5 algorithm description of with grouping and without grouping is analyzed. In section 6, the performance analysis is shown. Finally, a conclusion is given in section 7.

2. RELATED WORKS

Record de-duplication is a growing research topic in database and related fields as digital libraries. Today, this problem arises mainly when data is collected from disparate sources using different information description styles and metadata standards. Other common place for replicas is found in data repositories created from OCR documents. These situations may lead to inconsistencies that may affect many systems such as those that depend on searching and mining tasks. To solve these inconsistencies it is necessary to design a de-duplication function that combines the information available in the data repositories in order to identify whether a pair of record entries refers to the same real-world entity. In the realm of bibliographic citations, for instance, this problem was extensively discussed by Lawrence et.al (1999a,b) They proposed a number of algorithms for matching citations from different sources based on edit-distance, word matching, phrase matching, and subfield extraction. As more strategies for extracting disparate pieces of evidence become available, many works have proposed new distinct approaches to combine and use them. Elmagarmid et al (2007) classify these approaches into the following categories:

Ad–Hoc or Domain Knowledge Approaches - This category includes approaches that usually depend on specific domain knowledge or specific string distance metrics. Techniques that make use of declarative languages Elmagarmid et al (2007) can be also classified in this category;

Training based Approaches - This category includes all approaches that depend on some sort of training supervised or semi supervised in order to identify the replicas. Probabilistic and machine learning approaches fall into this category. Next, we briefly comment on some works based on these two approaches (domain knowledge and training–based), particularly those that exploit the domain knowledge and those that are based on probabilistic and machine learning techniques, which are the ones more related to our work.

Domain Knowledge Approaches - The idea of combining evidence to identify replicas has pushed the data management research community to look for methods that could benefit from domain-specific information found in the actual data as well as for methods based on general similarity metrics that could be adapted to specific domains Elmagarmid et al (2007). As an example of a method that exploits general similarity functions adapted to a specific domain, we can mention Chaudhuri et al (2003). There the authors propose a matching algorithm that, given a record in a file (or repository), looks for another record in a reference file that matches the first record according to a given similarity function. The matched reference records are selected based on a user-defined minimum similarity threshold. Thus, more than one candidate record may be returned. In such cases, the user is required to choose one among them indicating which is the closest one. Records matching on high weight tokens (strings) are more similar than those matching on low weight tokens. The weights are calculated by the well-known IDF weighting method Baeza-Yates (1999). This weighting method is also

used in WHIRL Cohen (2000) a database management system that supports similarity joins among relations that have free text attribute values. In Carvalho and Da Silva (2003), the authors use the vector space model for computing similarity among fields from different sources and evaluate four distinct strategies to assigning weights and combining the similarity scores of each field. As a result of their experiment, they found that using evidence extracted from individual attributes improves the results of the replica identification task.

Probabilistic Approaches - Bayesian et al were the first ones to address the record duplication problem as a Bayesian inference problem (a probabilistic problem) and proposed the first approach to automatically handle replicas. However, their approach was considered empirical by Elmagarmid et al (2007) since it lacks a more elaborated statistical ground. After Newcomb et al work, Fellegi (1969 a,b) proposed a more elaborated statistical approach to deal with the problem of combining evidence. Their method relies on the definition of two boundary values that are used to classify a pair of records as being replicas or not. Tools that implement this method (Freely Extensible Biomedical Record Linkag [6]), usually work with two boundaries as follows:

Positive Identification Boundary – if the similarity value lies above this boundary, the records are considered as replicas.Negative Identification Boundary – if the similarity value lies below this boundary, the records are considered as not being replicas.

For the situation in which similarity values stand between the two boundaries, the records are classified as "possible matches" and, in this case, a human judgment is necessary. Usually, most of the existing approaches to replica identification depend on several choices to set their parameters, and they may not be always optimal. Setting these parameters requires the accomplishment of the following tasks:

Choosing the best evidence to use the more evidence, the more time is required to find the replicas, since more processing is needed to calculate the similarity among the attributes.

Finding how to combine the best evidence some evidence may be more effective for replica identification than others.

Finding the best boundary values to be used bad boundaries may increase the number of identification errors (e.g., false positives and false negatives), nullifying the whole process.

Machine Learning Approaches - The proposals that are more related to our work are those that apply machine learning techniques for deriving record-level similarity functions that combine field-level similarity functions, including the proper assignment of weights (Bilenko et al (2003) and Cohen and Richman (2002) and Tejada et al (2001). These proposals use a small portion of the available data for training. This training data set is assumed to have similar characteristics to those of the test dataset, which makes feasible to the machine learning techniques to generalize their solutions to unseen data. The good results usually obtained with these techniques have empirically demonstrated that those assumptions are valid. In Bilenko et al (2003), the authors use a machine learning technique to improve both the similarity functions that are applied.

An approach distinct from the previous ones is presented in Guha et al (2004). The main idea is to generate individual rankings for each field based on generated similarity scores. The distance between these rankings is calculated by using the well-known Footrule metric, which is minimized by a modified version of the Hungarian Algorithm specifically tailored to this problem by the authors. Then, a merge algorithm based on a score scheme is applied to the resulting rankings. At the end of this process, the top records in this global ranking are considered to be the most similar to the input record. It may be noticed that this approach requires no training. Unfortunately, the experiments conducted do not evaluate the quality of the global ranking with respect to the record matching effectiveness. In Carvalho et al (2006), we propose a GP-based approach to improve results produced by the Fellegi and Sunters method ("Freely Extensible Bio-medical Record Linkage," [6]). Particularly, we use GP to balance the weight vectors produced by that statistical method, in order to generate a better evidence combination than the simple summation used by it. Our experimental results with real datasets show improvements of 7% in precision with respect to the traditional Fellegi and Sunters method. In comparison with our previous results in De Carvalho et al (2008) this article presents a more general and improved GP-based approach for deduplication, which is able to automatically generate effective deduplication functions even when a suitable similarity function for each record attribute is not provided in advance. In addition, it also adapts the suggested functions to changes on the replica identification boundary values used to classify a pair of records as replicas or not. These two characteristics are extremely important since they free the user from the burden of having to select the similarity function to use with each attribute required for the deduplication task and tune the replica identification boundary accordingly.

3. EVOLUTION OF DIS-SIMILARITY CALCULATION

We have so far discussed two techniques in the dis-similarity calculation. The two techniques perform individual task but final output is same. The Duplicate Detection without grouping generates the dis-similarity value, but its number of comparison is huge and each record must want to compare with all other records in the data set. So the number of iteration for each data set is n*n and turnaround time for this method is more.

Algorithm: Duplicate Detection without grouping

Step 1: Extract the pair of record from given data set and Initialize the pair of attribute C1, C2.
Step 2: Apply Decision making algorithm to construct Truth Table.
Step 3: Apply the Entropy calculation logic "Entropy (P,N)"
Step 4: Apply the Gain calculation logic "Gain (G)"
Step 5: Return Dis-Similarity Percentage.
Step 6: Repeat step 1 to 5 until it reaches end of record for each iterations.

Due to this performance and number of iteration reason, we introduced Duplicate Detection with grouping. This method will generate a dis-similarity value, and the number of comparison is less. Each record does not want to compare with all other records in the data set. It will compare the record within the group alone and so the total number of iteration for each group of data is G(n*n) and turnaround time for this method is less. We have proved and compared with the executed result.

Algorithm: Duplicate Detection with grouping

Step 1: Extract the pair of record from given data set. Initialize the self-sized vector ←VEC [N].

Step 2: Validate the extraction record if it is valid Send the record to further process. If it is not valid Reject the record and skip further process.

Step3: Apply Pre-condition of ADTree for first level clustering If clustering index is already available Append the new record into the existing cluster P_VECTOR Else Create a new cluster P_VECTOR. Apply condition of ADTree for second level clustering if clustering index is already available Append the new record into the existing cluster C_ VECTOR. Else create a new cluster C_VECTOR.

Step 4: Extract each cluster C_VECTOR one by one.

Step5: Extract the pair of record from the given cluster. Initialize the pair of attribute C1, C2.

Step 6: Apply Decision making algorithm to construct Truth Table.

Step 7: Apply the Entropy calculation logic “Entropy (P, N)”

Step 8: Apply the Gain calculation logic “Gain (G)”

Step 9: Return Dis-Similarity Percentage.

Step 10: Repeat step 1 to 9 until it reaches end of record for every iteration within the cluster record.

4. PREDICTIVE ACCURACY

The proposed algorithm is coded in java programming language on windows platform with Intel CORE i3 and 1GB RAM. This implementation produced accurate result within a short period.

Extracted data from the data base grouping are not random selections. It will be observed that there are distinct patterns in the observed relations and that a particular entity is more likely to co-occur in a group with a specific subset of all entities rather than a random one. If we assume the entities belong to groups or cliques then the entities in any observed link will come from the same group in most cases. We may allow each entity to belong to multiple groups at the same time. For example consider the query given below for group detection while extracting data from the data base. If short name is already available in the data base, this query will retrieve the data as per our expected grouping output. And also this query is doing the basic cleansing work before grouping the data.

```
Query = "SELECT BORROWER_NAME, SHORT_NAME, ACCOUNT_ID FROM BORROWER_BRANCH_DETAILS_LOAD WHERE SHORT_NAME NOT LIKE\'%&%\'";

Query = Query +"AND SHORT_NAME NOT LIKE\'%-%\'"; Query = Query +"AND SHORT_NAME NOT LIKE\'%1%\'"; Query = Query +"AND SHORT_NAME NOT LIKE\'%2%\'";

Query = Query +"AND SHORT_NAME NOT LIKE\'%3%\'"; Query = Query +"AND SHORT_NAME NOT LIKE\'%4%\'"; Query = Query +"AND SHORT_NAME NOT LIKE\'%5%\'";

Query = Query +"AND SHORT_NAME NOT LIKE\'%6%\'"; Query = Query +"AND SHORT_NAME NOT LIKE\'%7%\'"; Query = Query +"AND SHORT_NAME NOT LIKE\'%8%\'";

Query = Query +"AND SHORT_NAME NOT LIKE\'%9%\'"; Query = Query +"AND SHORT_NAME NOT LIKE\'%0%\'";

Query = Query +"ORDER BY SHORT_NAME ASC, LENGTH (SHORT_NAME) ASC
```

Table 4.1: Characteristics of Sample Cluster Bucket

S.No	Name	Short Name	Account Type	Account No	Region	Cluster No
1	Venkatesh	V	SB	SB9922	North	1,2
2	Venkatesh Kumar	VK	SB	SB28109r	South	1
3	Venkatesh Kumer	VK	CR	CR723759	South	1
4	Venketesh Kumar	VK	CD	CD7279	South	1
5	Venkatesh Kumar	VK	DB	DB729875	South	1
6	Venketesh	V	CR	CR124188	North	1,2
7	Venkatesh Arumugam	VA	SB	SB639386	East	2
8	Venketesh Arumugam	VA	CR	CR644226	East	2
9	Venkates Arumugam	VA	CD	CD01274	East	2
11	Venkatesh Arumugum	VA	DB	DB9212221	East	2
12	Kumar	K	SB	SB65113	West	3,4
13	Kumar Muru gesh	KM	SB	SB124574	South	3
14	Kumer Murugesh	KM	CR	CR536124	South	3
15	Kumar Murugash	KM	CD	CD912094	South	3
16	Kumar Muruge	KM	DB	DB65187	South	3
17	Kumer	K	CR	CR7218921	West	3,4
18	Kumar D	KD	CD	CD7572121	South	4
19	Kumer D	KD	DB	DB673256	South	4

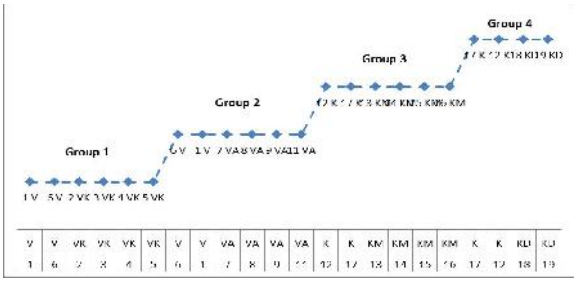
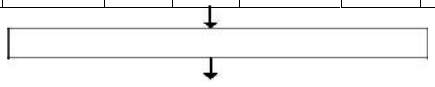


Figure 4.1: Characteristics of Sample Cluster Bucket

Table 4.1 and Figure 4.1 illustrate the predictive accuracy and simplicity of the grouping algorithm. It also clearly shows accurate grouping occur from given set. For the banking domain, for example, we can imagine the entities in a group to be short name with a different account that collaborate. A name can have multiple accounts but all names of any account share common information and same name. We would like to discover these groups given in our group data.

Figure 4.2 shows how De-duplication processes identify duplicates by doing an internal matching using proposed algorithm with configurable set of data element like name. It groups the identified records and assigns the Bucket-ID and Key value for each of the records identified in the group. De-duplication is the process of grouping the similar records and bring those records under one group id. The status of Primary or Secondary will be given within the group after following the algorithmic procedure (which is based on marks for the right match).This process can be applied in lakhs and millions of records which are owned by any concern (such as banks). The status of the record is based on the score value .The one having superior scoring or first one first serve base or which one has bigger length will get the Primary status ; the other below scored records will get Secondary status within the group. The record which does not have match will be the only record in that group which is called unique record. The probability matrix for scoring records will be decided on the basis of thorough analysis of real customer's records. Here we used greater than 80 as match record. The fields which will be given an input for de-duplication process are configurable based on the customer's requirement. Each record may have n number of fields. The scoring will be given for each field. Eventually the records of Primary and Secondary will be decided.

S.No	Name	Short Name	Account Type	Account No	Region	Cluster No
1	Venkatesh	V	SB	SB9922	North	1,2
2	Venkatesh Kumar	VK	SB	SB28109	South	1
3	Venkatesh Kumer	VK	CR	CR723759	South	1
4	Venketesh Kumar	VK	CD	CD7279	South	1
5	Venkatesh Kumar	VK	DB	DB729875	South	1
6	Venketesh	V	CR	CR124188	North	1,2



COMPANY NAME	BUCKET-ID	KEY-VALUE	Account No
Venkatesh Kumar	1	PRIMARY	SB28109
Venkatesh Kumar	1	DUPLCATE	DB729875
Venkatesh Kumer	1	DUPLCATE	CR723759
Venketesh Kumar	1	DUPLCATE	CD7279
Venkatesh	2	PRIMARY	SB9922
Venketesh	2	DUPLCATE	CR124188

Figure 4.2: Duplicate Detection and Bucket Formation

5. PERFORMANCE ANALYSIS

A turnaround measure is significant between the duplicate detection with grouping and duplicate detection without grouping. It clearly shows duplicate detection with grouping is much better than duplicate detection without grouping.

Table 5.1 : Comparison of elapsed time in minute between Single attribute and Single Token

Record Volume	Time(min)	Without Grouping	Cluster-Time Taken	DD Time Taken	Total-Time Taken
5K	10	1.06	0.18	0.31	0.49
10K	20	1.52	0.26	1.02	1.28
35K	30	8.42	0.39	1.46	2.25
50K	40	12.58	0.53	2.59	3.52
70K	50	18.49	1.21	5.07	6.28
80K	60	20.31	1.43	6.27	8.1
100K	70	24.23	2.01	9.03	11.04

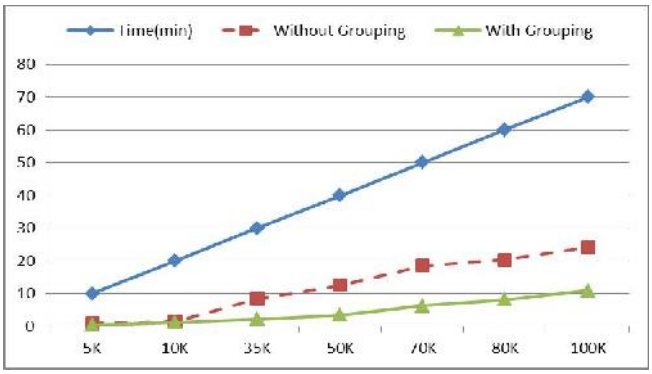


Figure 5.1: Comparison of elapsed time in minute between Single attribute and Single Token

Table 5.1 and Figure 5.1 clearly show duplicate detection with grouping has taken less time for Dis-Similarity calculation with Single attribute and single Token with respect to any volume of data. Our executed result clearly shows for calculating dis-similarity value it takes maximum of around Ten minutes for 100k of data set volumes. If we consider duplicate detection without grouping for 100K volumes it takes around 25 minute.

The overall result indicates admirable performance of reducing more than 150% of turnaround time for calculating dis-similarity value through grouping with respect to single attribute and single token. If we increase the dataset volume the performance will increase for with grouping and decrease for without grouping and there is no relevance for any number of attribute and token.

Table 5.2: Comparison of elapsed time in minute between Single attribute and Multiple Token

Record Volume	Time (min)	Without Grouping	With Grouping		
			Cluster-Time Taken	Gain-Time Taken	Total-Time Taken
5K	10	3.05	0.18	0.43	1.01
10K	20	8.59	0.26	1.56	2.24
35K	30	12.01	0.39	3.01	3.4
50K	40	22.58	0.53	4.09	5.02
70K	50	35.29	1.21	6.53	8.14
80K	60	39.03	1.43	9.08	10.51
100K	70	56.12	2.01	12.47	14.48

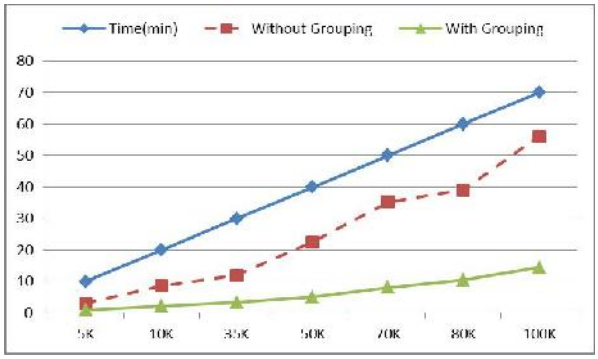


Figure 5.2: Comparison of elapsed time in minute between Single attribute and Multiple Token

Table 5.2 and Figure 5.2 clearly show duplicate detection with grouping has taken less time for Dis-Similarity calculation with Single attribute and Multiple Token with respect to any volumes of data. Our executed result clearly shows for calculating dis-similarity value with grouping it takes maximum of around 15 minutes for 100k of data set volumes. If we consider duplicate detection without grouping for 100K volumes it takes around 56 minute. The overall result indicates admirable performance of reducing more than 300% of turnaround time for calculating dis-similarity value through grouping with respect to single attribute and multiple token. If we increase the dataset volume the performance will increase for with grouping and decrease for without grouping and there is no relevance for any number of attribute and token.

Table 5.3: Predictive Accuracy of the different sector data

Sector Name	Execution Type	Accuracy Of Output
Banking	Single Attribute with Single Token	95.06 ± 98.96
	Single Attribute with Multiple Token	89.31 ± 93.71
Hospital	Single Attribute with Single Token	91.19 ± 96.40
	Single Attribute with Multiple Token	86.26 ± 92.09
Manufacture	Single Attribute with Single Token	93.06 ± 97.84
	Single Attribute with Multiple Token	84.51 ± 90.89
Sales	Single Attribute with Single Token	90.44 ± 94.61
	Single Attribute with Multiple Token	81.73 ± 86.19

Table 5.4 shows the accuracy of the output value has been evaluated for various kinds of sector, namely Banking, Hospital, Manufacture and Sales. As per tabulated result the predictive accuracy is more for banking data. The reason behind this is that the data is already in a standardized manner. The results of all the four sectors show that single attribute and single token produce more accuracy than single attribute and multiple attribute. The average of accuracy is around 90 percentages for both the combination.

Table 5.4: Total number of Iteration between with grouping and without grouping

Record Volume	No Of Iteration (Crore)	Without Grouping	With Grouping
5K	100	2.5	0.16
10K	300	10	0.37
35K	500	122.5	4.56
50K	700	250	9.30
70K	900	490	18.23
80K	1100	640	23.80
100K	1300	1000	37.19

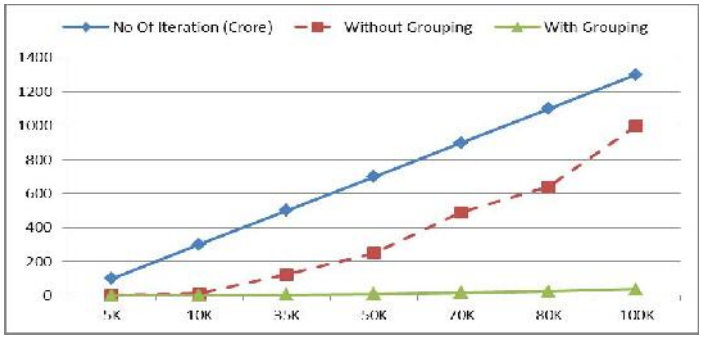


Figure 5.3: Total number of Iteration between with grouping and without grouping

Table 5.4 and Figure 5.3 show duplicate detection without grouping exponentially increase the number of iteration. Due to this large size of data, volume approach is not fit. And also there is a chance to face an out of memory issue and its control is out of our hand. But duplicate detection with grouping gradually increases the number of iteration even if we increase the data size. Grouping method may be able to control the out of memory issue. At the same time we can handle only one group for duplicate detection, as this control is in our hand.

8. CONCLUSION

This paper enlightened evaluation of dis-similarity calculation using group detection algorithm for dis-similarity calculation without grouping and dis-similarity calculation with grouping based on single attribute and single token, single attribute and multiple token. It is engaged simultaneously to find group detection solution as well as dis-similarity calculation among each group. It produces more accurate result when data set has a number of mixed token in the same attribute. Dis-similarity calculation with grouping approach reduces the elapsed time as well as produces more accurate and simple to make a good decision. As the various combination values were defined to the parameter attribute, several interesting phenomena were observed. From the observation, it is known that number of group, number of iteration and accuracy of outcome data are fully depended on the characteristics of the data set. The scalability of algorithm was measured by feeding different volumes of dataset. The results clearly indicated that the algorithm produced a more accurate result for a large data set.

REFERENCES

1. Barrodale .I , Ericson.R, “An algorithm for least-squares linear prediction and maximum entropy spectral analysis, Part-1: theory and part:2 fortron program Geophys, 45:420-466,1980.
2. McCallum A,Freitag D, and F.C.N.Pereira, “Maximum Entropy Markov Models for Information Extraction and Segmentation,” Proc. 17th Int Conf. Machine Learning (ICML ‟00), pp. 591-598, 2000.
3. Lawerwnce .S, Giles.C.L, Bollacker.K.D, “Autonomous citation matching in proceedings of the third international conference on Autonomous agents, pp.392-393,1999.
4. Lawrence.S, Giles.L and Bollacker.K,” Digital libraries and autonomous citation indexing”, IEEE computer, vol.32,no.6,pp.67-71,1999.
5. Elmagarmid.A,K.IpeirotisP.G and Verykio.V.S “Duplicate record detection: A survey” ,IEEE transactions on knowledge and Data Engineering, Vol.19,no.1,pp.1-16,2007.
6. Baeza-Yates, R. A. and Ribeiro-Neto, B. A. Modern Information Retrieval. ACM press/Addison-Wesley, 1999.
7. Cohen,W.W. “Data integration using similarity joins and a word-based information representation language,” ACM Transactions on Information Systems,vol.18, no.3, pp 288 -321, 2000.
8. Fellegi, I. P. and Sunter, A. B. “A theory for record linkage,” Journal of American Statistical Association, vol. 66, no. 1, pp. 1183–1210, 1969.
9. Bilenko,M. Mooney,R.J. Cohen,W.W. Ravikumar,P. and Fienberg, S.E.“Adaptive Name Matching in Information Integration,” IEEE Intelligent Systems,vol.18, no. 5, pp. 16-23, Sept./Oct. 2003.
10. Elmagarmid,A. K. Ipeirotis,P.G. and Verykios,V. S. “Duplicate record detection : A survey,” IEEE Transactions on Knowledge and Data Engineering, vol. 19, no. 1, pp. 1–16, 2007.
11. Bilenko.M and Mooney,R.J. “Adaptive duplicate detection using learnable string similarity measures,” in Proceedings of the Ninth ACM SIGKDD, International Conference on Knowledge Discovery and Data Mining, pp.39–48.2003.
12. Cohen.W.W, and Richman.J,” Learning to match and cluster large high – dimensional data sets for integration ,” in proceedings of ninth ACM SIGKDD International conference of knowledge discovery and data mining, pp.475-480, 2002.
13. Tejada.S, Knoblock.C.A and Minton.S, “Learning object identification for information integration, “Information systems, vol.26, no.8,pp.607-633, 2001.
14. Guha.S, Koudas.N, Marathe.A, and Srivastava.D,”Merging the results of approximate match operations, “in proceedings of the thirteenth international conference on very large data bases,pp.636-647, 2004.
15. Caryalho.J.C.P, and Da.Silva, A.S.Finding similar identities among objects from multiple web sources, “ in proceedings of the fifth ACM International workshop on web information and data management, pp.90-93, 2003.
16. De carvalho, M.G. Laender, A.H.F Goncalves. M.A, and DaSilva, A.S,” Replica identification using genetic programming, “in proceedings of the 23rd Annual ACM symposium on Applied Computing-SAC, 2008.