International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 – 5161

Available at http://inpressco.com/category/ijcet

Research Article

Record Normalization by Eliminating Duplicate entries from Multiple Sources

Kalyani Ashok Sankpal and Kalpana V. Metre

Department of Computer Engineering MET's BKC Institute Of Engineering, Adgaon, Nashik, Maharashtra

Received 10 Nov 2020, Accepted 10 Dec 2020, Available online 01 Feb 2021, **Special Issue-8 (Feb 2021)**

Abstract

A bulk data is generated from various sources. The sources may provide duplicate data with some representative changes. To mine such big data and create a representative data is a challenging task. The data importance increases when it is linked with similar resources and similar data is fused in one source. Lot of research work has been done to provide a single representative data of all real world entities by removing the duplicate records. This task is called as record normalization. This technique focuses on precision of record normalization as compared with the existing strategies. For record normalization it uses record level, field level and value level normalization technique. The precision of unique representation of record is increases in each level. Along with unique representation, the data is linked with similar resources by comparing the similar record field values. The system is tested on citation record based dataset and its accuracy and execution time is compared.

Keywords: *Record normalization, data clustering, data fusion, data linking, data integration*

Introduction

The bulk data is generated in the world wide web. Based on the user search parameter the data is collected from various sources. The structured data contents are stored in web warehouses containing web databases and web tables. The relevant data collection is done from various warehouses likes Google, Bing Shopping. Google Scholar is important mining domain. It is known as web data integration. In web data integration, the structured data should be matched automatically coming from various web warehouses. A data containing similar records, records that point to the same entity should be grouped together as a standard record set.

The result set generated after searching a query on search engine generates the redundant results, showing multiple entries of same record coming from various sources. This record representation contains duplicate and unnecessary entries. Such result set is inconvenient to the end user for analysis.

Record normalization is important is variety of domains. For example, in case of research publication domain Citeseer or Google Scholar are important integrator websites that collects data from various sources from automatic data collection technique. The data is displayed to the user based on the user query. The data should be clear and in normalized form. The search result should be:

1. Best match search
2. Data should be de-duplicated

If ad-hoc approaches for data matching is followed or all the matched records are displayed to the end user then it will very frustrating for end user to sort and extract useful information from the generated result set. Ad-hoc extraction of records may lead to record with missing value or incorrect data representation.

The record normalization is a challenging problem because various resources provide same data in various formats. There is conflict in data which is collected from various sources due to erroneous data, incomplete data, different data representation or missing some attribute values.

Consider an example: User fire a search query as: "Data integration: the teenage years", based on the title matching various records are fetched like:

Table I. Publication records

Sr. No.	Author	Title	Venue	Date	Pages
1.	Halevy, A.; Rajaraman A.; Ordille, J.	Data integration: the teenage years	in proc 32nd int conf on Very large data bases	2006	
2.	A. Halevy, A. Rajaraman, J. Ordille	Data integration: the teenage years	in VLDB	2006	9-16
3.	A. Halevy,	Data	in proc	2006	pp.916

	A. Rajaraman, J. Ordille	integration: the teenage years	32nd conf on Very large data bases		
4.	A. Halevy, A. Rajaraman, J. Ordille	Data integration: the teenage years		2006	9-16

In the above table, the same author name representation is in the various form. Venue and pages contain some missing value or variation in representation of same data. By analyzing all the records the normal record should be generated as:

Table II. Normalized Records

Sr. No	Author	Title	Venue	Date	Pages
1	A. Halevy, A. Rajaraman, J. Ordille	Data integration: the teenage years	in proc 32nd int conf on Very large data bases	2006	pp.916

For normalized record generation record level duplication should be removed. With the record level comparison, field level comparison should be done. In the above example author, title, venue data and pages are various fields in a record. For more precision the values in a field should be normalized. In the following section literature survey is discussed followed by problem formulation. Based on the analyzed problem a new system is proposed in section IV. Implementation details are discussed in section V followed by the conclusion.

Literature Survey

Culotta et al. proposes a record normalization at the very first time. The normalization technique is also called as Canonicalization. This is a process of converting the data in one standard canonical form by analyzing various parameters. In this paper author proposes a technique for the record normalization on database. For normalization 3 type of solutions are provided. The solution is in terms of field values. These solutions are enlisted as follows:

1. String edit distance to find most relevant central record
2. Optimize the edit distance parameter
3. Feature-based solution to improve performance of Canonicalization.

This paper does not consider the value component level normalization and hence the normalized record database contains many instances of repetitive data and unnecessary normalized records [2].

Swoosh treats the data duplication problem as entity relationship problem. The problem is like a black box function. This back box matches and merges the records. The ER algorithm is defined to invoke these functions. The system generates de-duplicate records but not generate the normalized records. It increases the complexity of record matching problem [3].

Wick et al. proposes a technique for data integration using schema matching. It also focuses on co-reference resolution, record canonicalization. For implementation it uses discriminatively-trained model. Due to combined objective, the system complexity increases. The paper only deal with field level record matching and not at the value level and hence the system do no generate the complete normalization records.[4]

Tejada et al. proposes a technique for database record normalization called as object normalization. The system collects the data from various web sources and saves collectively in a database. At the time of search these database object are normalized with duplication removal. The system uses attribute ranking as well as string ranking in attribute, based on the user's confidence score. [5]

Wang et al. works on shopping dataset. The dataset is normalized in terms of records. It works on data integration and data cleaning. It works on record marching and replacing the missing values with the most relevant values. It also corrects the data which is best suitable to the record by comparing the other dataset record entries. It do not work on value level and working globally on field level normalization.[6]

Chaturvedi et al. works on pattern discovery in the records. This technique do not focus on data normalization and removal of duplicate records but it extracts patterns form duplicate record and find the most important and prevalent patterns in the dataset. This approach can be applicable for data normalization.[7]

Dragut et al. works on automatic labeling called as Label normalization. The label normalization is used for record normalization and assigning meaningful labels to the elements of an integrated query interface. It works on field level labeling and assign labels to each attribute within the global interface. [8]

S. Raunich et. al. proposes an ATOM system. The Atom system works on Ontology merging which is nothing but a record normalization. But in merging phase user involvement is required. The approach should be automated with less involvement of end user [9].

Yongquan Dong et. al. works on automatic record normalization. The normalization is performed at three levels: record level, field level and value level. The normalization accuracy increases at each level of data pruning. The duplicate records are removed. A single entry is created by analyzing the duplicate entries. The related entries are not clubbed together. A single representation of record is created. For more informative data representation data should be normalized and linked together [1].

Problem Formulation

Let E1 be the real world entity. Re is set of records collected from various sources representing the same entity E1. Re= {R1,R2,..Rp}. This record is the collection of various fields. In each field various string values are present. Let FS be the set of fields FS = {f1, f2, …, fq} and ri[fi] is the value in the field fi. There is need to define the problem as record normalization and linking problem. From the set of Re, generate a new customized record that represent the entity E1 more accurately in a very descriptive manner. The records from other entities like E1 should be linked together by matching the field and value level components.

Proposed Methodology

A. Preliminaries:

1. Frequency Ranker:

The frequency ranker ranks the mostly occurred unit u in the list of distinct units.
FR(U)= [u1,u2,..up]
Where, FR(U) is a sorted list in the descending order of units based on the occurrence frequency.

2. Length Ranker

The length ranker ranks the length of unit u in the list of distinct units.
LR(U)= [u1,u2,..up]

Where, LR(U) is a sorted list in the descending order of units based on the number of characters present in the unit.

3. Centroid Ranker

This gives the ordered list of distinct units. It initially calculates the similarity score among unit and finds the centroid. The centroid is calculated as:

$$\mathrm{UCS(u)} = \frac{1}{|U|^2} \sum_{u \varepsilon U'} \mathrm{AuAvSM(u,v)}$$

Where,
U = bag of units
U' = distinct units in U
Au and Av: occurrence frequency of u and v.

4. Edit-distance based Similarity measure:

The number of edit required to transform one string to another. Edit distance based similarity between two string a and b is given as:

$$\mathrm{Sim-ed(a,b)} = \frac{\mathrm{ed(a,b)}}{\max(|a|,|b|)}$$

|a| and |b| is lengths of a and b respectively.

5. bigram similarity measure:

This distance is based on 2- character substring present in string. The similarity measure between string a and b is given as:

$$\text{Sim-bigram(a,b)} = \frac{2*(|\text{bigram(a)} \cap \text{bigram(b)}|)}{(|\text{bigram(a)}| + |\text{bigram(b)}|)}$$

Bigram(a) and bigram(b) are 2-grams of a and b respectively.

6. Feature-based rankers:

Feature based rankers are divided in 2 sections:

a. Strategy feature:

This is binary indicator that indicates the unit is representative unit ranked by some ranking criteria. b. Text Feature:

This feature examines the property of string. It checks the string is acronyms or abbreviations of certain representative string or not. For example: conf is abbreviation of conference whereas VLDB is acronym for Vary Large Databases.

7. Collocation:

Collocation is sequence of consecutive terms with the inverse term document frequency (idf) value less than the given threshold. N-collocation defines the consecutive n terms.

8. Sub-collocation

Is the substring of n-collocation string with k consecutive terms. For example "in the conference" is the sub-collocation of " in the conference of VLDB".

9. Template collocation:

An n- collocation terms is called as template collocation if its inverse term document frequency (idf) is greater than the given threshold.

10. Twin template collocation:

The terms tc1 and tc2 are twin collocation if it satisfies the following conditions:
P(tc1, tc2) > p(tc1, tc), for all tc Ɛ TC and tc1 <> tc2
(p(tc1,tc2))/(p(tc2))>threshold

B. System Architecture

Redundant record Set is input to the system. After processing, system generates Non-redundant normalized record set along with the data linking. The data processing is mainly categorized in 5 sections:

1. Data preprocessing
2. Record Level Normalization
3. Filed Level Normalization and
4. Value Level Normalization.
5. Filed Based Clusters

Following figure shows the architecture of the system.

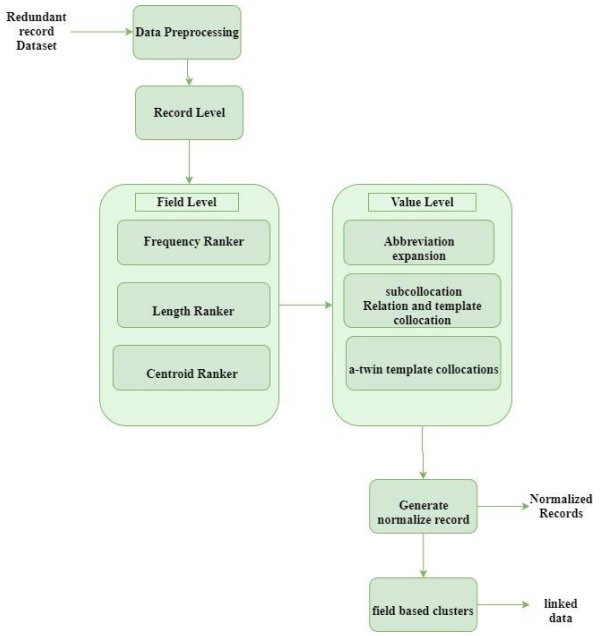

Fig. 1. System Architecture

C. System Description:

1. Pre-processing step: Initially from the given data each record is separated and from each record various fields are extracted. For Example: Consider the following citation: A. Halevy, A. Rajaraman, J. Ordille, "Data integration: the teenage years", in proc 32nd int conf on Very large data bases,2006, pp.9-16 In this citation the following fields can be separated as: Author: A. Halevy, A. Rajaraman, J. Ordille Title: Data integration: the teenage years Venue: in proc 32nd int conf on Very large data bases Date: 2006 Pages: pp.9-16 All the comma separated values are extracted and added in the respective fields. 2. Record selection: The record is generated with the combination of various fields. There should be all values present in each field so that a complete informatory citation can be generated as a representative of all redundant data. This is a selection criterion for record level data filtering. The selected records are further processed using field and value level. 3. Field Selection: The normalized record is generated by combining the most descriptive features of all fields. From all the records each field data is normalized and then a new record is generated. For record normalization frequency ranker, length ranker, centroid rankers and feature based ranker are used. 4. Value Selection: The values of each field are extracted. The abbreviation and acronyms are replaced by Mining Abbreviation-Definition Pairs algorithm. Afterwards its collocation, sub collocation cPGCON 2020 (Post Graduate Conference for Computer Engineering) - ` and twin-collocation is identified using Mining TemplateCollocation-SubCollocation Pairs (MTS)algorithm. And normalized record is generated at the value level. 5. Field based Clusters: Based on the normalized value extracted for each field in the record, relevant records are linked as per the field value details.

D. Algorithms

1. Mining Abbreviation-Definition Pairs
Input : collection of all values of the field fi
Tlen, Tidf, Tpos : Threshold Values Output: AWP: a set of abbreviation-word pairs Processing:
2. cwords = EMPTY; AWP = EMPTY;
3. pwords = tokenize data in fi
4. uwords = find unique words in pwords;
5. for each uword in uwords do
6. if len(uword) >= T_{len} and idf(uword,Re) ≤T_{idf} then
7. insert uword into cwords
8. end if
9. end for
10. for each cword in cwords do
11. pa-words = Find words in same Context(cword, uwords, , T_{pos})
12. if pa-words <> EMPTY then
13. abbreviations = find Abbreviations(cword, pa-words)
14. end if
15. if abbreviations <> EMPTY then
16. for each abbreviation in abbreviations do
17. insert (abbreviation, cword) into AWP;
18. end for
19. end if
20. end for
21. Algorithms: Mining TemplateCollocation-SubCollocationPairs (MTS)
Input: CVal(f) – abbreviations in val(f).
T_{idf} .: Threshold value
Output: TCSP: A list of template collation Tc and its subcollations Stc pair Processing:

1. Initialize TCSP = EMPTY;
2. m =getMaxWordCount(CVal(f));1-collocs = Find One Word Collocations(CVal(f));
3. if 1-collocs <> EMPTY then
4. for each 1-colloc Ɛ 1-collocs do
5. add (1-colloc, NULL) to TCSP
6. end for
7. ews = Find Candidate Expand Words(CVal(f)))
8. for n = 2 to m do
9. n-collocs = Find NCollocations(CVal(f), n, Tidf);
10. if n-collocs == EMPTY then
11. break
12. end if
13. Y = EMPTY
14. for each n-colloc Ɛ n-collocs do
15. cspairs = Find Expanded Subcollocation Pairs(ncolloc, ews, TCSP)
16. if cspairs <> EMPTY then
17. for each cspair Ɛ cspairs do
18. X = {c} ∪ Sc;
19. insert (n-colloc, X) into TCSP
20. add cspair to Y
21. end for

22. end if
23. end for
24. TCSP = TCSP − Y
25. end for
26. remove the pairs of the form (c, NULL) from TCSP
27. End if
28. return TCSP

Result and Discussions

The system is implemented on windows system with 8gb ram and i3 processor. For programming java development kit- jdk1.8 is used.

A. Dataset:

PVCD[10] dataset is used. This dataset is a publication dataset. It contains publication venue information. It contains 3,683 publications and 100 distinct publication records. The dataset contains acronyms, abbreviations, and misspellings.

B. Performance Measures:

1. Accuracy:

The fivefold cross validation is performed and accuracy is measured in terms of correctly normalized units at record and field level with respect to the predicted normalized units. The accuracy is measured for record level, filed level and value level normalization.

2. Processing Time:

The processing time for each level processing is measured.

C. Implementation Status:

The system implemented partially. The frequency and length rankers are applied on dataset.

The dataset contains venue information. Initially for frequency ranker, distinct venue fields are extracted from dataset with its occurrence frequency. The list is sorted in descending order of frequency count.

For length ranker, length of characters in a field is calculated and field list is sorted in descending order of length value.

Following table shows the time required for processing frequency and length ranker.

Table III : Time Evaluation

Number of records	Frequency ranker(time in Seconds)	Length Ranker(time in Seconds)
500	0.91	0.71
1000	1.58	1.22
1500	2.12	1.72
2000	2.37	2.16

Conclusions

The proposed system generates Normalized records by removing duplicate entries that points to the same entity. For data normalization processing is applied at tree levels: record level, Field level and value level. The precision of deduplication increases from record level to value level. Along with the duplication removal similar entities are grouped together using field and value level data comparison. The grouped data is linked together to generate more representative data. In future system can be extended to handle numeric and more complex values.

References

[1] Yongquan Dong, Eduard C. Dragut and Weiyi Meng, "Normalization of Duplicate Records from Multiple Sources", in IEEE Transactions on

Knowledge and Data Engineering, Vol. 31 , Issuue 4 , April 2019, pp. 769

– 782

[2] A. Culotta, M. Wick, R. Hall, M. Marzilli, and A. McCallum, "Canonicalization of database records using adaptive similarity measures,"in SIGKDD, 2007, pp. 201–209.

[3] O. Benjelloun, H. Garcia-Molina, D. Menestrina, Q. Su, S. E. Whang, and J. Widom, "Swoosh: A generic approach to entity resolution,"VLDBJ, vol. 18, no. 1, pp. 255–276, 2009.

[4] M. L. Wick, K. Rohanimanesh, K. Schultz, and A. McCallum, "A unified approach for schema matching, coreference and canonicalization,"in SIGKDD, 2008, pp. 722–730.

[5] S. Tejada, C. A. Knoblock, and S. Minton, "Learning object identification rules for information integration,"Inf. Sys., vol. 26, no. 8, pp. 607–633, 2001.

[6] L. Wang, R. Zhang, C. Sha, X. He, and A. Zhou, "A hybrid framework for product normalization in online shopping,"in DASFAA, vol. 7826, 2013, pp. 370–384.

[7] S. Chaturvedi and et al., "Automating pattern discovery for rule based data standardization systems,"in ICDE, 2013, pp. 1231-1241.

[8] E. C. Dragut, C. Yu, and W. Meng, "Meaningful labeling of integrated query interfaces,"in VLDB, 2006, pp. 679-690.

[9] S. Raunich and E. Rahm, "Atom: Automatic target-driven ontology merging,"in ICDE, 2011, pp. 1276-1279.