ISSN 2349-9842, Volume 1, Issue 1, March 2014

International Journal of Innovative Trends and Emerging Technologies

RECORD LINKAGE AND DEDUPLICATION USING FEBRL FRAMEWORK AND BLOCK, SORTING, BIGRAM INDEXING TECHNIQUES

S.THILAGAVATHI
Assistant Professor, Department of Computer science and Engineering, Aksheyaa College of engineering
E-mail: thilaga.cse001@gmail.com

Abstract--**Record linkage of millions of individual health records for ethically-approved research purposes is a computationally costlier task. Blocking methods are used in record linkage systems to reduce the number of candidate record comparison pairs to a feasible number to maintain linkage accuracy. Various blocking methods have been implemented recently using high-dimensional indexing or clustering algorithms. Compare two new blocking methods, bigram indexing and canopy clustering with TFIDF (Term Frequency/Inverse Document Frequency), with two older methods of standard traditional blocking and sorted neighbourhood blocking. The results show that recently blocking methods such as bigram indexing and canopy clustering provide scalable blocking methods while maintaining or improving upon record linkage accuracy. There is a potential for large performance speed-ups and better accuracy to be achieved by these new blocking methods in FEBRL (Freely Extensible Biomedical Record Linkage) Framework.**

1. INTRODUCTION

A. Motivation

Record linkage techniques are used to link together records which relate to the same entity (e.g. patient or customer) in one or more data sets where a unique identifier is not available. Record linkage is an important initial step in many research and data mining projects in the biomedical and other fields, where it is used to improve data accuracy and to assemble longitudinal or other data sets which would not otherwise be available. The processing view of a standard record linkage system architecture as implemented in TAILOR, Febrl [1] or AutoMatch . The major challenges in record linkage are computational complexity and linkage accuracy. Linking data sets with millions of records can take from hours to days on modern computing systems. Recent developments in information retrieval, database systems, machine learning and data mining have the potential to improve the efficiency and accuracy of record linkage system components. These developments include efficient blocking methods, adaptive distance metrics for evaluation of record pair similarity and learning methods for the classification

Task of deciding whether a record pair is a match, nonmatch or possible match. As potentially each record in one data set has to be compared to all records in a second data set, the number of record pair comparisons grows quadratically with the number of records to be matched. This approach is computation ally infeasible for large data sets. To reduce the huge number of possible record pair comparisons, traditional record linkage techniques work in a blocking fashion, i.e. they use a record attribute (or sub-set of attributes) to split the data sets into blocks. Record pairs are then generated for all the records in the same block (i.e. records with the same value in a blocking attribute). Such detailed comparison functions include approximate string comparisons for names and addresses, and date or age comparisons (e.g. for date of birth).

B. Objective

Comparing the speed and accuracy of new blocking methods with established blocking method implementations. The performance bottleneck in a record linkage system is usually the evaluation of a similarity measure between pairs of records. The choice of a good blocking method can greatly reduce the number of record pair evaluations to be performed and so achieve significant performance speed-ups. Contribution is to empirically compare the speed-up and accuracy (sensitivity and specificity) performance of these blocking methods. Blocking methods directly affect sensitivity (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 intensive comparators to be employed).

C. Overview

Compare Blocking, the Sorted Neighbourhood method and Bigram Indexing. This paper's contribution is to empirically compare the speed-up and accuracy (sensitivity and specificity) performance of these blocking methods in FEBRL framework. Blocking methods directly affect sensitivity (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 intensive comparators to be employed)**2** *Related Work*

The Recent researches are using the discrete wavelet transform (DWT) is applied in image compression format (JPEG) 2000 and Motion

photographic group (MPEG)-4. Chen.P et al., [1] have proposed secrete message is embedded in the high frequency co-efficient of the wavelet transform by leaving the low frequency co-efficient sub-band unaltered.

Raja.K.B et al., [2] have proposed a novel image steganographic technique in integer wavelet transform domain. Babita Ahuja, et aI., [4] proposed for more hiding capacity achieved by Filter Based scheme in Steganography. Jan Kodovsky and Jessica Fridrich [3] worked out the specific design principles in Steganographic scheme for the JPEG format and their security.

Mohamed Ali Bani Younes, et. aI., [5] proposed a steganographic approach for hiding. This approach hides the least significant bits insertion to hide the data within encrypted image data. Chang-Chu Chen, et al., [6] have proposed that data hiding scheme was a modification of the LSB based Steganography using the rule of reflected gray code.

In this paper we presents a new method of data hiding in the discrete wavelet transform coefficients of the cover image to maximize the hiding capacity to overcome the drawback. The Arnold transformation is performed to scramble the secret image to hide into the wavelet coefficients in the low frequency to increase the system security.

In chapter three we discuss about the proposed method, DWT and IDWT, Arnold transformation and implementation of steganography model, Noise Attacks, Chapter four describes the experimental results and analysis for the proposed steganography method and noise attacks. In Chapter five the conclusion of the paper and suggests for the future improvements of the system.

II. *The Record Linkage Process*

The first step in any record linkage or deduplication [4] project is data cleaning [3] and standardization . Here the conversion of the raw input data into well defined, consistent forms. The next step is indexing step that generates pairs of candidate records that are compared in the comparison step using a variety of comparison functions appropriate to the content of the record fields (attributes).Then record pairs are generated with similarity values.

Using these similarity values, the record linkage process is carried out to compare candidate record pairs into non-matches, matches, and possible matches. If record pairs are classified into possible matches, a review process is required where these pairs are manually assessed and classified into matches or nonmatches. This is usually a time-consuming and errorneous process, especially when large databases are being linked or deduplicated. By evaluating the quality and complexity of a record linkage project is a final step in the record linkage process [9].

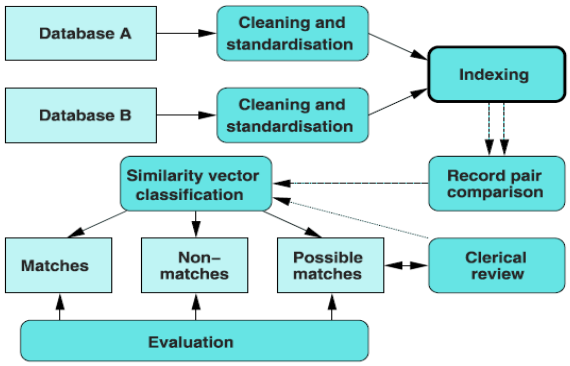


Fig. 1. Outline of the general record linkage process.The indexing step (the topic of this survey) generates candidate record pairs, while the output of the comparison step is vectors containing numerical similarity values.

3. EXISTING SYSTEM

Conceptually, the indexing step of the record linkage process can be split into the following two phases:

1. ***Build***.

All records in the database was read and their respective BKV was generated and the records are inserted in respective data structures. The BKVs become the keys of the inverted index and the records having the same key values were inserted into the same inverted index list. When linking two databases, either a separate index data structure is built for each database, with common key values. In second case with each record identifier flag is generated to indicate from which database the record originates.The efficient access to single random records can be achieved by using an apprppriately indexed database or hash table.

2. ***Retrieve***

For each block, a list of record identifiers are retrieved from the inverted index and candidate record pairs are generated. For a record linkage, all records in a block from one database will paired with all records from the block with the same BKV from the other database, while during deduplication each record in a block will be paired with all other records in the same block. The resulting vector which is generated from the comparison step was given as a input to classifier. Here we concentrate on how different indexing techniques, using the same blocking key to index records from data sets with

different behaviors and in combination with various parameter, affects the number and quality of the candidate record pairs generated.

All indexing methods have the following attributes which can (some need) to be given as arguments when an indexing method is initialised.

- **Name**
  A name for an indexing method. This should be a short string.
- **Dataset**
  A reference to the data set the index is built for (i.e. when the index is built it is assumed that records are having the fields as defined in this data set).
- **Block_definition**
  The definition of how the index and its blocks should be built. Blocks are defined using a list of lists each containing tuples of the form (field_name, method, parameters). The given field names must be available in the defined data set. Methods and parameters are explained in more details in the following subsections.
- **Soundex**
  A possible first parameter is the maximal length of the encoding (in characters), and a second parameter can be the word 'reverse'. When given, the values in the field are reversed before they are encoded. The default value for the maximal length is 4.
- **Nysiis**
  The *NYSIIS* phonetic encoding algorithm. The same parameters maximal length and 'reverse' as with Soundex can be given.
- **Dmetaphone**
  The *Double-metaphone* phonetic encoding algorithm. The same parameters maximal length and 'reverse' as with Soundex can be given.
- **Truncate**
  A string truncation method, where as additional parameter the length must be given (i.e. strings longer than the given length are truncated).

In Existing there are several indexing techniques are adopted for record linkage and deduplication namely

1. Traditional blocking
2. Sorted Neighborhood Indexing
3. Q-Gram-Based Indexing
4. Suffix Array-Based Indexing
5. Canopy Clustering
6. String-Map-Based Indexing

A. Traditional blocking

All records that have the same BKV are grouped into the same block and the records which is in the same block was compared to each other And then each record is inserted into one block (assuming a single blocking key definition).

.B. Sorted Neighborhood Indexing

Window positions	BKVs (Surname)	Identifiers
1	Millar	R6
2	Miller	R2
3	Miller	R8
4	Myler	R4
5	Peters	R3
6	Smith	R1
7	Smyth	R5
8	Smyth	R7

Window range	Candidate record pairs
1 – 3	(R6,R2), (R6,R8), (R2,R8)
2 – 4	(R2,R8), (R2,R4), (R8,R4)
3 – 5	(R8,R4), (R8,R3), (R4,R3)
4 – 6	(R4,R3), (R4,R1), (R3,R1)
5 – 7	(R3,R1), (R3,R5), (R1,R5)
6 – 8	(R1,R5), (R1,R7), (R5,R7)

Fig.3 **Sorted Neighborhood Indexing**

The idea behind here is to sort the database(s) based on the BKVs generated, and to move the window sequencely over the fixed number of records sw (w > 1) over the sorted values of database. Then Candidate record pairs are generated from the records within a current window.

Example sorted neighborhood technique based on a sorted array, with BKVs being the surname values from Fig. 3 (and the corresponding record identifiers), and a window size w = 3.

C. Q-Gram-Based Indexing

To create variations for each BKV using q-grams (substrings of lengths q), and to insert record identifiers into more than one block. Each BKV is converted into a list of q-grams, and sublist combinations of these q-gram lists are then generated down to a minimum length, which is determined by a user as threshold value t (t< 1).

Identifiers	BKVs (Surname)	Bigram sub-lists	Index key values
R1	Smith	[sm,mi,it,th], [mi,it,th], [sm,it,th], [sm,mi,th], [sm,mi,it]	**smmiitth**, miitth, smitth, smmith, smmiit
R2	Smithy	[sm,mi,it,th,hy], [mi,it,th,hy], [sm,it,th,hy], [sm,mi,th,hy], [sm,mi,it,hy], [sm,mi,it,th]	smmiitthhy, miitthhy, smitthhy, smmithhy, smmiithy, **smmiitth**
R3	Smithe	[sm,mi,it,th,he], [mi,it,th,he], [sm,it,th,he], [sm,mi,th,he], [sm,mi,it,he], [sm,mi,it,th]	smmiitthhe, miitthhe, smitthhe, smmithhe, smmiithe, **smmiitth**

Fig. 4. Q-gram-based indexing with surnames used as BKVs, index key values based on bigrams (q = 2), and calculated using a threshold set to t =0.8. The right-hand side shows three of the resulting inverted index lists (blocks), with the common BKV highlighted in bold in the index key value column.

D. Suffix Array-Based Indexing

Suffix	Identifiers
atherina	R2,R3
atherine	R1
atrina	R4,R5
catherina	R3
catherine	R1
catrina	R4
erina	R2,R3
erine	R1
herina	R2,R3

Suffix	Identifiers
herine	R1
katherina	R2
katrina	R5
~~rina~~	~~R2,R3,R4,R5~~
rine	R1
therina	R2,R3
therine	R1
trina	R4,R5

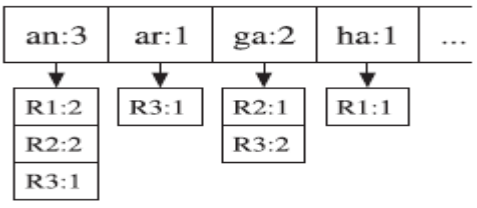
Eg.1 **Suffix Array-Based Indexing**

A suffix array contains alphabetically sorted values based on the suffix of each BKV's.

E. Canopy Clustering

Create high dimensional overlapping clusters from generated candidate record pairs. And the Clusters are created by calculating the similarities between BKVs using measures such as Jaccard or TF-IDF/cosine. Both of these measures are based on tokens which can be characters, q-grams or words.

Identifiers	BKVs (Surname)	Sorted bigram lists
R1	Hanlan	[(an,2), (ha,1), (la,1), (nl,1)]
R2	Gansan	[(an,2), (ga,1), (ns,1), (sa,1)]
R3	Gargan	[(an,1), (ar,1), (ga,2), (rg,1),]



Eg.2. Canopy **Clustering**

F. String-Map-Based Indexing

This indexing technique is based on mapping BKVs (assumed to be strings) to objects in a Multidimensional Euclidean space (i.e) the distances between pairs of strings. The string similarity measure is a distance function (such as edit-distance) that can be used in the mapping process.

Issues in Existing system

It does not set efficient parameter settings depend both upon the quality and characteristics of the data to be linked or deduplicated.

IV. PROPOSED SYSTEM

The aim of indexing is to reduce the potentially huge number of comparisons (every record in one data set with all records in another data set) by eliminating comparisons between records that obviously are not matches. In other words, indexing reduces the large search space by forming groups of records that are very likely to be matches. Indexing can also be seen as a clustering method that brings together records that are similar, so only these records need to be compared using the more expensive (i.e. compute intensive) field comparisons functions.

Currently the **Febrl[1](**(Freely Extensible Biomedical Record Linkage)system contains several indexing methods, including the *traditional* blocking method used in many record linkage systems. These indexing methods are implemented in the module indexing. Indexes are normally built while a data set is being standardised. After an index is built a *compacting* has to be done which builds index data structures that can return the blocks more efficiently.

A. Linkage or Deduplication Process

For the deduplication method, the following arguments need to be defined.

- **Input_dataset**
  A reference to a data set which contains the (raw) input data. This data must be initialised in read access mode.
- **Tmp_dataset**
  A reference to a direct random access data set (initialised in access mode readwrite) that will hold the cleaned and standardised records before they are deduplicated.

- **Rec_comparator**
  A reference to a record comparator that contains field comparison functions that compare record pairs from the temporary data set field by field and that produces a weight vector for each record pair, which is given to the classifier. See on how to define a record comparator.
- **Blocking_index**
  A reference to an indexing object defined on the temporary data set.
- **Classifier**
  A reference to a classifier that classifies weight vectors.
- **First_record**
  The record number of the first record in the input data set to be processed. If this argument is not given (or set to None), then it will be set to the first record number (i.e. record with number 0).
- **Number_records**
  The number of records from the input data set that should be processed. If this argument is not given (or set to none), it will be set to the total number of records in the data set.
- **Weight_vector_file**
  By setting this argument to a string the raw weight vectors will be saved into a CSV (comma separated values) text file. An existing file with the given name will be erased first. If set to None no weight vector file will be written. A header line will be written with the column names being the names of the field comparison functions.
- **weight_vector_rec_field**
  This attribute can either be set to None (the default) in which case the first two columns in the weight vector file (if defined) will be the (internal) record numbers for the two records being compared (resulting in a weight vector).

For a record linkage process, similar arguments are needed.

V. CONCLUSION

This article presented a improvement of the efficiency of the indexing techniques for record linkage and deduplication by implementing in FEBRL (Extensible Biomedical Record Linkage) framework. Thereby we proven that 1) all comparisons between records within a block will have a certain minimum similarity with each other, and 2) the similarity between records in different blocks is below this minimum similarity.

ACKNOWLEDGEMENT

I would like to thank Mr. P. C. Senthil Mahesh (Professor) Department of Computer Science and Engineering, Mrs. Simi Margarat (Assiatant Professor) Department of Computer Science and Engineering, Dhaanish Ahmed College of Engineering affliated to Anna University for their valuable technical discussions for this paper. I would like to thank the associate editor and anonymous reviewers for insightful comments and helpful suggestions to improve the quality, which have been incorporated in this paper.

REFERENCES

1) P. Christen, "*Febrl: An Open Source Data Cleaning, Deduplication 11111and Record Linkage System With a Graphical User Interface*," Proc. 14th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD '08), pp. 1065-1068, 2008.
2) Koudas, A. Marathe, and D. Srivastava, "*Flexible String Matching against Large Databases in Practice*," Proc. 13th Int'l Conf. Very Large Data Bases (VLDB '04), pp. 1086-1094, 2004.
3) E. Rahm and H.H. Do, "*Data Cleaning: Problems and Curren Approaches*," IEEE Technical Committee Data Eng. Bull., vol. 23,no. 4, pp. 3-13, Dec. 2000.
4) P. Christen, "*Febrl: An Open Source Data Cleaning, Deduplication and Record Linkage System With a Graphical UserInterface*,"Proc. 14th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD '08), pp. 1065-1068, 2008.
5) W.W. Cohen, P. Ravikumar, and S. Fienberg, "*A Comparison of String Distance Metrics for Name-Matching Tasks*," Proc. Workshop Information Integration on the Web (IJCAI '03), 2003.

S.Thilagavathi received her Bachelor of engineering degree in Computer science and Engineering from Anna University, Master of Engineering degree in Computer Science and Engineering from Anna University. Currently she is serving as Assistant Professor in Aksheyaa college of Engineering and her major fields of interest are biomedical clustering and linkage process.