Available online at www.elixirpublishers.com (Elixir International Journal)

Computer Science and Engineering

Elixir Comp. Sci. & Engg. 44 (2012) 7494-7495

ISSN: 2229-712X

Active operation based edit distance for duplicate detection

A.Venkatesh Kumar[1] and G. Rajendran[2]

[1]Department of Science and Humanities,, Kongu Engineering College, Perundurai-638052,Tamilnadu, India.

[2]Department of Mathematics, Kongu Engineering College, Perundurai-638052, Tamilnadu, India.

ARTICLE INFO

Article history:

Received: 9 October 2011;

Received in revised form:

16 March 2012;

Accepted: 24 March 2012;

Keywords

Edit distance,
Duplicate detection,
De-duplication,
Similarity measure,
Character measure,
String measure.

ABSTRACT

The problem of identifying approximately duplicate records in databases is an essential step for data cleaning and data integration processes. In the real world, entities have two or more representations in databases. Errors are introduced as the result of transcription errors, incomplete information, lack of standard formats, or any combination of these factors. Most of the existing approach has been depend on the generic or manual intervention is required to calculate the edit distance. None of the existing approach hasn't focus on how many individual operations required for edit distance calculation, existing relied on total operation. In this paper, we present a thorough analysis of the literature on edit distance for duplicate detection. We cover similarity metrics that are commonly used to detect number of individual operation required to find a duplicate record, which includes the total required edit operation.

Introduction

Database played an important role in today's real-time. Databases frequently contain same field-values and records that refer to the same entity but are not syntactically identical. Variations in representation can arise from typographical errors, misspellings, abbreviations, as well as integration of multiple data sources. Such approximate duplicates should create many effects; many industries and system depend on the accuracy of databases data to carry out decision making operations.

While integrating data from different data based or data sets to implement a data warehouse, organizations should aware of potential systematic differences or conflicts data, as well as they must aware of nature of data. Some previous work has addressed the problem of identifying duplicate records, where it was referred to as record linkage across dataset [4, 5], duplicate detection [3, 6]. Typically, standard string similarity metrics such as edit distance [9] or vector-space cosine similarity [1] are used to determine whether two values or records are alike enough to be duplicates. Some of the recent work [2, 6, 8] has examined the use of pairing functions that combine multiple metrics.

Estimate of similarity between pair of string can vary significantly depending on the domain and specific field under consideration, characteristics of data, traditional similarity measures may fail to estimate string similarity correctly. If we are going for token level comparison, certain token may be more informative than second compared string. For example in some place people written "STREET" as it, when facing a space problem that time the same word scrutinize as "ST". In this scenario at character level, certain character can be consistently replaced by other. Thus, accurate similarity computations require adjusting string similarity metrics for each record of the database with respect to the particular data domain. The character-based similarity metrics are intended to handle typographical errors, which inserting missing character, replacement of character, deletion of character well. In this section, we explained some of the similarity metrics.

Rather than hand-tuning a distance metrics for each field, we propose to use a trainable method for find a similarity measure. Which will produces a total edit operation required to compare the string, as well as it produce the operation types and its counts. The propose method include two steps, fist one is construct a truth table based on the pair of string which will help to represent the character into binary value. Second step is calculate the total number of TRUE (1) and FALSE (0) in term of row wise for identify total require edit operation and then calculate the total number of TRUE (1) and FALSE (0) in term of column wise for identify the individual operation count.

Calculation of edit distance metrics

When the distance between records composed of multiple fields is being calculated, it is necessary to combine similarity estimates for individual fields in a meaningful manner. Because correspondence between overall record similarity and similarity across individual fields can vary greatly, it is necessary to weight fields according to their contribution to the true similarity between records.

Record distance is calculated by the combination of the attribute distances. There are many ways of combining the attribute distances and we discuss truth table way for calculate edit distance in next. We use(C, R) to denote the distance between two records R1 and R2. R represents the row and C represent column.

Every Boolean function can be specified as a table with the value of 0, 1 and function has a "n" argument, then the total possible argument combinations are 2n. In this section to construct the logical representation of two different string tokens truth table. In pair of strings which one have more length Where Ci, i=1…..n represents the i'th character of column. Rj, j=1…..m represents the j'th character of row.

Tele:

E-mail addresses: venkateshakumar@gmail.com

Initialize: Row ← 0; Column ← 0
Loop L1: While! Empty (String_1)
C1 ← String_1 [Row]
Row = Row + 1;
Loop L2: While! Empty (String_2)
C2 ← String_2 [Column]
Column = Column + 1
If C1 = C2
Then
Truth [Row, Column] = 1
Else
Truth [Row, Column]=0
Endif
Goto L2
Goto L1

It can be proved that this proposed technique is guaranteed to converge to a local maximum of likelihood of observing the training corpus w(C, R). The trained model can be used for estimating distance between two pair of strings by computing the probability of generating the aligned pair of strings summed across all possible paths as calculated by the no of 0's in the total in terms of row is the total number of edit operation and number of 0's in the column is representing total required operation for replacement. Subtraction of no of 0's in row into no of 0's in column is the total required operation for insertion. A practical problem that may arise in this computation is numerical underflow for long strings, which can be solved by mapping all computations into logarithmic space or by periodic scaling of all values in truth table.

Experiments

For example, the distance between [7] "kitten" and "sitting" is 3, since the following three edits change one into the other, and there is no way to do it with fewer than three edits:
kitten → sitten (substitution of 's' for 'k')
sitten → sittin (substitution of 'i' for 'e')
sittin → sitting (insert 'g' at the end).
Count of 0's in Total® = 3
Count of 1's in the Total® = 4
Count of 0's in Total© = 2
Count of 1's in Total© = 4
Total No of Insertion Operation = Count of 0's in Total® - Count of 0's in Total©
= 3 – 2
= 1
Total No of Replacement Operation = Count of 0's in Total©
= 2

Conclusion

In this paper, we have proposed a method to identify required edit distance between pair of text. First we have introduced truth table construction algorithm for represent the string value into logical representation of strings C and R. Second calculated the total number of 0's in the total, this is the expected output of proposed algorithm. The new method offers more accuracy of result without user feedback at the time of duplicate detection.

Reference

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, New York, 1999.
[2] W.W.Cohen and J. Richman. Learning to match and cluster large high-dimensional data sets for data integration. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2002), Edmonton, Alberta, 2002.
[3] A E. Monge and C. P. Elkan. An efficient domain-independent algorithm for detecting approximately duplicate database records. In Proceedings of the SIGMOD 1997 Workshop on Research Issues on Data Mining and Knowledge Discovery, , May 1997, pages 23–29, Tuscon, AZ.
[4] H. B. Newcombe, J. M. Kennedy, S. J. Axford, and A. P. James. Automatic linkage of vital records. Science, 1959, 130:954–959.
[5] W. E. Winkler. The state of record linkage and current research problems. Technical report, Statistical Research Division, U.S. Bureau of the Census, Wachington, DC, 1999.
[6] S. Sarawagi and A. Bhamidipaty. Interactive deduplication using active learning. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2002), Edmonton, Alberta, 2002.
[7]A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios, "Duplicate record detection: A survey," IEEE Trans. Knowl. Data Eng., 2007, vol. 19, no. 1, pp. 1–16.
[8]S. Tejada, C. A. Knoblock, and S. Minton. Learning domain-independent string transformation weights for high accuracy object identification. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2002), Edmonton,Alberta,2002.

Table I. Truth table constructions for calculate the logical representation for pair of text character.

	C1	C2	…….	Cn	Total©
R1	R1C1	R1C2	…….	R1Cn	0/1
R2	R2C1	R1C2			0/1
.	.				
.	.				
.	.				
Rm	RmC1	RmC2	…….	RmCn	0/1
Total®	0/1	0/1		0/1	

Table II. Truth table constructions for sample data

	s	i	t	t	i	n	g	Total©
k	0	0	0	0	0	0	0	0
i	0	1	0	0	0/1	0	0	1
t	0	0	1	0/1	0	0	0	1
t	0	0	0/1	1	0	0	0	1
e	0	0	0	0	0	0	0	0
n	0	0	0	0	0	1	0	1
Total®	0	1	1	1	0	1	0	