Chapter 11
Basic Data Operators for Entity Resolution

ABSTRACT
This chapter focuses on the basic data operators for entity resolution, which include similarity search, similarity join, and clustering on sets or strings. These three problems are of increasing complexity, and the solution of simpler problems is the building blocks for the harder problem. The authors first introduce the solution of similarity search, covering gram-based algorithms and sketch-based algorithms. Then the chapter turns to the solution of similarity join, covering both exact and approximate algorithms. At last, the authors deal with the problem of clustering similar strings in a set, which can be applied to duplicate detection in databases.

INTRODUCTION
As discussed in previous chapters, entity resolution has many models. Most of entity resolution methods focus on the effectiveness of entity resolution on various data types. With the increasing of data size, efficiency is another issue that is noticed by the literature. To perform entity resolution on big data, many algorithms have been developed. However, most of these algorithms are memory-based algorithms without efficiency assurance. To solve the efficiency problem, some methods have been proposed in database literature.

To solve the problem in the aspect of database, some operators for entity resolution are extracted and efficient algorithms are designed for these operators. With efficient implemented operators, entity resolution could be performed efficiently.

Since entity resolution is to identify different representations for the same real-world entity. With the consideration of errors in data, to identify different representations, one basic operator is to find descriptions that are similar to a given description and the other operator is to find pairs of similar descriptions. The first one is defined as similarity search and the second is defined as similarity join. This chapter focuses these two basic operators. We use an example to illustrate these two operators. Consider a query “stick” on a set $S =$ {stuck, schick, trick, chunk} with threshold = 1, the result is “stuck” since the edit distance between “stick” and “stuck” is 1 while larger than 1 with other strings. For similarity self-join with threshold 2 on the set $S$ is { (stuck, trick) } since the threshold of these two strings is equal or smaller than 2.
Basic Data Operators for Entity Resolution

As the base of the similarity search and similarity join, similarity function should be defined at first for different data type. For numerical data types, the similarity search and similarity join can be performed directly with SQL language on database. For example, consider a relation $R$ with numerical attribute $A$. For the similarity search on $A$ with query value $v$ and the threshold $r$, the query is “select $A$ from $R$ where $A <= v + r$ AND $A >= v - r$”.

For complex data types such as string and set, things are more complex since the similarity comparison process could not be described as SQL expression and thus hardly performed with database engine. As a result, researchers show more interests in the similarity operator for complex data types. Many methods are proposed. Thus, in this chapter, we propose the algorithms for these three operators for common-used data types of string and set.

Additionally, with the result of similarity search and similarity join, the final results of group-wise entity resolution require clustering with the result. Thus clustering is another important operator for entity resolution.

**SIMILARITY SEARCH**

The goal of similarity search is to find similar strings for a given query string. As the readers might have noticed, this problem exists ubiquitously entity resolution, and solution of this problem serves as a powerful weapon for tackling more complicated problems, such as similarity join and clustering.

**Problem Formulation and Related Definitions**

Find strings similar to a given string: $\text{dist}(Q, D) \leq \delta$

Example: Find strings similar to “hadjelethierno”

**Similarity Measures and Distances (Xiao, Wang, Lin, Yu & Wang, 2011)**

- **Jaccard Similarity** is defined as
  
  $$J(x, y) = \frac{|x \cap y|}{|x \cup y|}$$

- **Cosine similarity** is defined as
  
  $$C(x, y) = \frac{x \cdot y}{\|x\| \cdot \|y\|}$$

- **Overlap similarity** is defined as
  
  $$O(x, y) = |x \cap y|$$

- **Hamming distance** between $x$ and $y$ is defined as the size of their symmetric difference:
  
  $$H(x, y) = |(x - y) \cap (y - x)|$$

- **Edit distance**, also known as Levenshtein distance, measures the minimum number of edit operations needed to make two strings identical.
  
  $$H(x, y) = |(x - y) \cap (y - x)|$$

**Q-Gram**

Let $\Sigma$ be an alphabet. For a string $s$ of the characters in $\Sigma$, we use “$|s|$” to denote the length of $s$, “$s[i]$” to denote the $i$-th character of $s$ (starting from 1), and “$s[i, j]$” to denote the substring from its $i$-th character to its $j$-th character.

We define $q$ continues characters as q-gram. Formally, given a string $s$ and a positive integer $q$, a positional q-gram of $s$ is a pair $(i, g)$, where $g$ is the q-gram of $s$ starting at the $i$-th character, i.e., $g = s[i, i + q - 1]$. The set of positional q-grams of $s$, denoted by $G(s, q)$, is obtained by sliding a window of length $q$ over the characters of string $s$. There are $|s| - q + 1$ positional q-grams in $G(s, q)$. For instance, suppose $q = 3$, and $s =$ university, then $G(s, q) = \{(1, uni), (2, niv), (3, ive), (4, ver), (5, ers), (6, rsi), (7, sit), (8, ity)\}$. This example is illustrated in Figure 1.

Since for q-gram, $k$ operations could affect $k \times q$ grams, if $\text{ed}(s, s') \leq k$, then their number of common grams $\geq (|s| - q + 1) - k \times q$. With this...
Related Content

Development of Data Warehouse Conceptual Models: Method Engineering Approach
www.irma-international.org/chapter/development-data-warehouse-conceptual-models/28159/

Privacy Protection in Association Rule Mining
www.irma-international.org/chapter/privacy-protection-association-rule-mining/10728/

Why General Outlier Detection Techniques Do Not Suffice for Wireless Sensor Networks
www.irma-international.org/chapter/general-outlier-detection-techniques-not/39544/

Semantic Data Mining
www.irma-international.org/chapter/semantic-data-mining/11057/

Profit Mining
www.irma-international.org/chapter/profit-mining/10729/