Week 5: Comparing Strings

Nate Bachmeier

TIM-7241:Constructive Research Design

May 23, 2021

Northcentral University

# Comparing Strings

Micheal Jordan, M. Jordan, and M.J. are three names for the same person. Humans can quickly examine these variations and conclude this fact. Meanwhile, computers need specialized fuzzy string comparison algorithms to make the same deductions. These entity name comparison algorithms focus on either name variations or misspellings. Gong, Wang, and Oard (2009) propose a hybrid solution that addresses both aspects (see Figure 1).

Figure 1: High-level Process

Their process imports and tokenizes names into “names” and “name formats.” Next, they use a directed graph to represent all permutations. Weighted edges between the nodes quantify the similarity to the original text. This configuration enables Dijkstra’s shortest path algorithm to find the right combination for fuzzy comparisons. Finally, the comparison determines if the match candidates are equivalent (Boolean value).

## Success Criterion

The process’s core use-case determines if two names are similar, regardless of spelling or variations (e.g., initials). Therefore, Gong, Wang, and Oard (2009) must maximize the solution’s recall and precision by minimizing false positives or negatives. Additionally, the system must be efficient and performant. Each imported name produces dozens of potential permutations, expanding the total data volume.

## Measuring Correctness

The authors assess multiple machine learning algorithms for determining the appropriate edge weights. Similar to previous publications, they use an F-measurement to assess the accuracy of each algorithm. Data scientists use F-Score as a “way of combining the precision and recall of the model, and [defines] the harmonic mean of the model’s precision and recall (Wood, n.d.).” Researchers can trade-off in their solution to optimize this value for their specific scenario. For instance, a regulated industry might enforce higher penalties on false positives over negatives.

## Measuring Efficiency

The name comparison process must store and retrieve numerous tokens during its assessment. According to the authors, their DBLP data set contains 227 million possible pairs, but only 0.7% are helpful (pg. 1878). This situation requires efficient data structures that can scale across large data sets. The researchers do not measure their efficiency and predominately rely on a NoSQL Data Management System (called Flamingo). They claim that initializing and comparing the most extensive test set completes within five minutes.

## Test Cases

The researchers collected the author’s names and citations from three academic publication sources (see Table 1). Next, they hand-labeled ground truths for the learning algorithms (Soft-TFIDF, Jaro-Winkler, recursive JW, Monge-Elkan, and recursive M.E.). Finally, the fuzzy comparison process ran across all combinations of data sets and methods.

Table 1: Data sources

|  |  |  |  |
| --- | --- | --- | --- |
|  | CiteSeer | arXiv | DBLP |
| References | 2892 | 58515 | 58399 |
| Authors | 1165 | 9200 | 21688 |

These test cases provide coverage over a vast corpus of English names. This situation could miss standard name formats for African or European nations. However, that limitation is outside the project’s scope.

# Create a different string algorithm