Author-title similarity scores

Jonathan Hooper, Leeds. May 2022

Contents

Similarity scores

We calculate author- and title-similarity scores for each set of author data we download from a source. These can be used in the export scripts to decide whether that set of data is a likely-enough match to the reading list citations.

These are saved as percentages (100 = perfect match, 0 = no similarity at all)

The export scripts multiply author- and title-similarities together (and divide by 100) to get a single similarity score - currently, anything 80 or over is included in the short report given to tutors.

This 80% cut-off was chosen by the Library on the basis of a set of actual results from a live reading list, balancing the risk of including false matches against the risk of missing genuine matches.

A looser set of matches (60% and over) is included in the long report, and all matches regardless of similarity are in the raw JSON data for each module.

*NB the export scripts **always** include matches retrieved by DOI, regardless of how low the similarity score is. This is because we can safely assume a DOI search will always return the correct citation - a low similarity score may reflect issues in our data; different forms used for names/titles; or situations where we hold a single author but the external source holds multiple.*

Calculation

Similarity for both authors and titles is essentially calculated as:

Similarity = 100 x Levenshtein (list-field, source-field) / Max-length (list-field, source-field)

The Levenshtein function gives the number of byte changes (additions, insertions, changes) needed to turn one string into another.

Refinements

There are a number of modifications to this calculation:

- Where strings are too long for PHP's Levenshtein function, PHP's similar_text() function is used instead
- Strings are normalised before comparison - duplicate/redundant whitespace is removed, punctuation is removed, accents are removed from characters (e.g. **é** is turned into **e**), and strings are converted to all-lower-case (so that comparison is case-insensitive).

Notes on non-Roman and multi-byte characters

- *Although punctuation is removed before comparison, non-Roman characters e.g. Chinese characters are **retained unchanged** for comparison.*
- *Similarity is typically calculated using any versions of author and title present in the reading list and the external source - in some cases, comparison of the Chinese-language versions produces a genuinely higher score than the Romanised versions, because of differences in the Romanisation.*
- *Because Levenshtein distance and string length are based on **bytes** and byte changes, **not characters** and character changes, in some situations the resulting similarity may be higher or lower than it intuitively should be.*

Source code

utils.php::similarity($string1, $string2, $type, $crop, $alphabeticise)

Comparison types

$type parameter to similarity()

The code supports various string comparison algorithms - options are:
Levenshtein | similar_text | metaphone | lcms [longest matching substring]

In practice, following trial-and-error with real citations, only Levenshtein is used (although the code falls back on similar_text when either string is too long for the Levenshtein algorithm i.e. over 255 characters).

Multiple forms of title and author

Generally there is more than one "author" and more than one "title" field in both the reading list and the external source. In general, the calling code calculates the score for each combination, and keeps the highest score found.

Modifications to basic calculation

There are a number of optional refinements that calling code can use:

- Words in each field can optionally be sorted into alphabetical order before comparison (e.g. "perkins karl" => "karl perkins")
  $alphabeticise = TRUE in similarity()

- Where the lengths of the two fields differ, and the longer looks like it contains surname + forenames, and the shorter surname + initials, the longer one can be rewritten as surname + initials
  $crop = initials in similarity()
- Where the lengths of the two fields differ, and the longer contains a colon, this longer one can be split at the colon and either the leading or the trailing portion compared *i.e. allow a plain title to match either the title or the sub-title*
  $crop = colon | post-colon in similarity()

Use in practice in the integrations

Generally for each citation, multiple attempts are made to calculate scores, and the best score is retained.

The alphabeticise and initials options are often used in author comparisons. Titles are often compared as-is, and also splitting at colons (taking both leading and traling portion).

As a result there is a risk that falsely high scores will be given to some matches, but this is offset by the cases where these modifications result in the correct match being found.