Phonetic Matching Report

Thomas Gregory

March 20, 2024

1 Summary

This report is a brief analysis of Joseph Whiting's Phonetic Testing script, which aims to match names used throughout a document in order to create a consistent sense of identity. It relies primarily on assessing phonetic similarity to achieve this.

The input.csv file lists some names, each with an associated unique ID number. The script main.py then imports the text, attempts to analyse which names are similar, and produces an output.csv with each row representing a name, listing all of the ID numbers associated with that name. In doing this, the core functions of matching are imported from the module match.py, which in turn depends on the phonetics library.

The script as written passes all 4 of its provided tests upon running pytest.

2 Code Analysis

Overall, the code is capable of distinguishing different versions of the same name in some settings. However, it relies solely on the 'phonetics' package to assess similarity. This is capable of identifying some similarities, but should only be one part of a larger process for a broader name matching goal.

The main.py script seems effective, assuming the input and output format are prescribed. The output is easily read and interpreted as a csv file.

he main.py script works, but could be improved. There is a bug in the match function - the comment signifies the intent to check that the original letters match, but uses the index '1', where indexing starts at '0' in Python. Currently, the second letter is being checked in each word. There are more efficient ways of producing the merged list in the matchlist function than has been used here.

3 Added Tests

3.1 test_possessive ()

When analysing a document, a name may come up in forms other than its stem. In English, this is mostly limited to the possessive case - for example, "John" and "John's". This test was introduced to check that these were being appropriately matched, which they currently aren't. The current code fails this test.

3.2 test_lowercase ()

Whilst this particular casing issue was implicitly checked in test_match_list_tracks_ids (), this test is checking for appropriate ID matching not for letter cases in particular, and so it would not be clear to interpret a failure on the test as being related to a failure to match a word with a lowercase word with a title case word. The current code passes this test.

3.3 test_shortform ()

In name matching, names may come up in separate forms. For example, most readers would naturally identify "Sam" and "Samuel" as names that could refer to the same person, and may use both inter-

changeably throughout a text. Good name matching software should be able to put these together. The current code fails this test.

3.4 test_seans ()

Whilst a basic example of checking different spellings of the same name was tried in test_different_spelling_matches (), some spelling variants look less similar than 'John' and 'Jon'. Testing four distinct spellings of 'Sean' (/'Shawn'/'Shaun'/'Shawne') ensures that more complex names with multiple spellings are picked up. The current code fails this test - in part due to a current bug in the code.

4 Recommendations

- As of Python 3.10, 'match' has in-built functionality. Whilst it isn't a reserved keyword, I would avoid its use as a user-defined function as it may cause confusion to other code contributors, and produce unexpected hard-to-trace errors if accidentally misused.
- Line 15 in match.py is longer than best practice would suggest, and attempts several logical steps in one line. This is hard for other users to follow, and could lead to unexpected bugs when amended. I suggest splitting this out over a few lines.
- For use in name matching, I would recommend checking other features beyond phonetic similarity to ensure similar names are picked up (e.g. use of possessive case as discussed above.
- A more comprehensive README that sets out the goals and features of the script would be very helpful - this should be updated to make the scope of the code, it's approach to the problem and what it intends to achieve clearer to the user.