FuzzPact

CS 699 : Course Project

by

A.V.S Bharadwaj 193050010
Anant Kumar 193050016
Kaushik Ganorkar 193050078
Team Impact

under the guidance of

Prof. Kavi Arya
Head TA. Diptesh Kanojia

Department of
Computer Science and Engineering
Indian Institute of Technology, Bombay
Mumbai 400 076

Contents

1 Introduction

1.1 Fuzzy Matching

Fuzzy matching allows you to identify non-exact matches of your target item. It is the foundation stone of many search engine frameworks and one of the main reasons why you can get relevant search results even if you have a typo in your query or a different verbal tense.

1.2 Natural Language Processing

Natural language refers to the way we, humans, communicate with each other.Namely, speech and text.We are surrounded by text. Measuring the similarity between words,sentences, paragraphs and documents is an important component in various tasks such as information retrieval, document clustering, word-sense disambiguation, automatic essay scoring, short answer grading,machine translation and text summarization.
Finding similarity between words is a fundamental part of text similarity which is then used as a primary stage for sentence,paragraph and document similarities.Words can be similar in two ways lexically and semantically. Words are similar lexically if they have a similar character sequence.Words are similar semantically if they have the same thing, are opposite of each other, used in the same way, used in the same context and one is a type of another.Lexical similarity is introduced in this survey though different String-Based algorithms, Semantic similarity is introduced through Corpus-Based and Knowledge-Based algorithms. String-Based measures operate on string sequences and character composition. A string metric is a metric that measures similarity or dissimilarity (distance) between two text strings for approximate string matching or comparison.Corpus-Based similarity is a semantic similarity measure that determines the similarity between words according to information gained from large corpora. Knowledge-Based similarity is a semantic similarity measure that determines the degree of similarity between words using information derived from semantic networks.(1)

1.3 String-based distance measures

1.3.1 Levenshtein Distance

Levenshtein distance intends to calculate the dissimilitude between two strings.Hence, the higher the value, the less similar they are. This metric represents the number of operations that need to be applied to one of the strings to make it equal to the other one. Such operations include the following: inserting a character, deleting it or substituting it. It is worth mentioning that each operation has an associated cost, although it's common to set the cost to 1 for all operations.(3)
In approximate string matching, the objective is to find matches for short strings in many longer texts, in situations where a small number of differences are to be expected. The short strings could come from a dictionary, for instance. Here, one of the strings is typically short, while the other is arbitrarily long. This has a wide range of applications, for instance, spell checkers, correction systems for optical character recognition and software to assist natural language translation based on translation memory.(2)
The Levenshtein distance can also be computed between two longer strings. But the cost to compute it, which is roughly proportional to the product of the two string lengths, makes this impractical. Thus, when used to aid in fuzzy string searching in applications such as record linkage, the compared strings are usually short to help improve the speed of comparisons.(2)

1.3.2 Jaccard Distance

The Jaccard index, also known as Intersection over Union and the Jaccard similarity coefficient is a statistic used for gauging the similarity and diversity of sample sets. The Jaccard coefficient measures similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets: The Jaccard distance, which measures dissimilarity between sample sets, is complementary to the Jaccard coefficient and is obtained by subtracting the Jaccard coefficient from 1, or, equivalently, by dividing the difference of the sizes of the union and the intersection of two sets by the size of the union(4)
Plagiarism: Finding plagiarized documents tests our ability to find textual similarity. The plagiarizer may extract only some parts of a document for his own. He may alter a few words and may alter the order in which sentences of the original appear.Yet the resulting document may still contain 50% or more of the original.

No simple process of comparing documents character by character will detect a sophisticated plagiarism.(5)
Mirror Pages : It is common for important or popular Web sites to be duplicated at a number of hosts, in order to share the load. The pages of these mirror sites will be quite similar, but are rarely identical. For instance, they might each contain information associated with their particular host, and they might each have links to the other mirror sites but not to themselves. A related phenomenon is the appropriation of pages from one class to another. These pages might include class notes, assignments, and lecture slides. Similar pages might change the name of the course, year, and make small changes from year to year. It is important to be able to detect similar pages of these kinds, because search engines produce better results if they avoid showing two pages that are nearly identical within the first page of results(5)

2 Motivation

Natural Language Processing is an extremely vast and promising field that builds up on progress made in seemingly unrelated fields such as linguistics, machine learning, information technology etc. It can be as challenging as it is rewarding. Therefore, with the guidance of an expert, we have found a problem area within this field that deserves attention - fuzzy matching of documents. so, we wish to come up with a tool that helps researchers working on the frontiers of this field with fuzzy matching of documents at a non-semantic level.
What string distance to use depends on the situation. If we want to compensate for typos then the variations of the Levenshtein distances are of good use, because those are taking into account the three or four usual types of typos. The metric could be improved f.x. by factoring the keyboard layout into the calculation. On an english keyboard the distance between "test" and "rest" would then be smaller than the difference between "test" and "best" for obvious reasons. This would be a top-down-assessment of a string metric. The bottom-up counterpart would be by trying to quantify the question "What would a human being (me) assume as similar?" and its answer. This is naturally tough to compute(6)
So from a top down perspective a good string metric would consider two strings very close if the first and last letter are matching and the letters in between are just permuted. The developed algorithms won't perform exceptionally well and the one's that do are probably just immune to permutations on a whole but its interesting to see how metrics respond to permutations. Okay one further aspect – given that even though human reading seems to be unimpressed by framed permutations ambiguous cases might arise – "ecxept"/"except" and "expcet"/"expect" – then the hamming distance would perhaps determine the interpretation(6)

3 Deployment

Using this module is as simple as downloading it from our github repository and using it as :

```
from fuzzPact import *
```

The readme.md on the repository's landing sheds more light on the workings of various aspects of the solution.

It can be accessed here

More documentation can be accessed via the docs folder. The documentation explaining the code structure has been prepared using sphinx.

4 Project Outcomes

4.1 User Oriented

The end users that this solution is designed to cater to are either NLP researchers or people in the Tech community.Our project presents an layered model , which helps the user set different parameter weights which help them assign certain amount of priority to certain distance metrics. The blending functionality of the module lets user consider a weighted average of various distance scores. This in turn should give the users sufficient flexibility to consider the the proximity of 2 text entities on a case to case basis.

They may now pick the best aspects of different distance measures and manage the variations in the sentences without actually changing the distance computing methodology drastically midway.This helps the user overcome the trade-offs that occur when we use different metrics individually. For instance, the users may now choose an measure which is an hybrid both of Jaccard and Levenshtein.

The way of coming up with a fuzzy match puts no restriction upon the levels and boundaries under consideration. Which in turn should make it highly scalable.

4.2 Developer Oriented

There is something especially exciting for budding developers that our soultion has to offer.

- How many different distance metrics can be implemented in the same style we have implemented ours?
- How many blend variation can be made. And how well will they work in different scenarios?
- How will our fuzzy matching get affected if we were to consider paragraph separation as a new level?

By exploring such possibilities, users can up with new libraries enabling more researcher in the NLP field. Some examples of the different areas in research in which the module developed in this project may be able to contibutes are:

1. Phonetic variations: Kohl's versus Coles

2. Typographical errors: Microsoft vs. Microsft
3. Contextual differences: Company vs. Organization
4. Reordered terms: Sam Hopkins vs. Hopkins Sam
5. Prefixes and suffixes: AJO Technology Company Limited vs. AJO tech Private Co., Ltd.
6. Abbreviations, nicknames, and initials: AJ Wilson vs. Alex Jane Wilson
7. Cross-language comparative analyses: Private limited vs.
8. Transliteration differences: Traditional Chinese vs. PinYin
9. Truncated letters and missing or extra spaces: Chu Kong Transport Company vs. ChuKong Transport Co. Ltd.

References

[1] Wael H. Gomaa,Aly A. Fahmy *A Survey of Text Similarity Approaches* . `https://pdfs.semanticscholar.org/5b5c/a878c534aee3882a038ef9e82f46e102131b.pdf`

[2] Nikhil Babar *The Levenshtein Distance Algorithm* `https://dzone.com/articles/the-levenshtein-algorithm-1`

[3] *Fuzzy matching with Levenshtein and PostgreSQL* `https://towardsdatascience.com/fuzzy-matching-with-levenshtein-and-postgresql-ed6`

[4] *Jaccard index* `https://en.wikipedia.org/wiki/Jaccard`$_i$*ndex*

[5] Ullman *Chapter 3 :Finding Similar Items.* . `http://infolab.stanford.edu/ ullman/mmds/ch3.pdf`

[6] Raffael Vogler *Comparison of String Distance Algorithms.* . `https://www.joyofdata.de/blog/comparison-of-string-distance-algorithms/`