Levenshtein Distance Tutorial

Abstract – This is an introductory tutorial on the Levenshtein Edit Distance (LED). This is the number of insertions, deletions, and substitutions need to change a sequence into a different one.

Keywords: levenshtein, distance, edits, edit distance, insertions, deletions, substitutions, sequence analysis
Published: 02-20-2015; Updated: 10-06-2016

Introduction

This tutorial was written as a companion for our Levenshtein Distance Calculator (Garcia, 2016a). The Levenshtein Distance, also known as the Edit Distance and Levenshtein Edit Distance (LED), is a metric named after his inventor, Vladimir I. Levenshtein.

Dr. Levenshtein is a pioneer in the theory of error correcting codes and known as the father of coding theory in Russia. He is an IEEE Fellow and the recipient of the 2006 Richard W. Hamming Medal and a member of the Moscow Mathematical Society and a research professor at the Keldysh Institute for Applied Mathematics at the Russian Academy of Sciences in Moscow (IEEE, 2015).

Dr. Levenshtein's work and theories have widespread use across technologies and industries. The distance that goes by his name is at the heart of today's spell-checking, sequence analysis, and text mining software, including commercial search engines.

About the Levenshtein Distance

The Levenshtein Distance is the number of edits needed to convert a sequence A into another sequence B.

Edits are insertions, deletions, and substitutions with an operational cost usually set to 1 per edit. Implementations for the computation of Levenshtein distances are available in several programming flavors (Gilleland, 2015). A variant that includes transpositions is the Damerau-Levenshtein Distance (Wikipedia, 2015; Wikibooks, 2015; Levenshtein.net, 2015).

Our implementation is a visual and interactive one. Figure 1 depicts its output for the transformation of *kitten* into *sitting*.

		k	i	t	t	e	n
	0	1	2	3	4	5	6
s	1	1	2	3	4	5	6
i	2	2	1	2	3	4	5
t	3	3	2	1	2	3	4
t	4	4	3	2	1	2	3
i	5	5	4	3	2	2	3
n	6	6	5	4	3	3	2
g	7	7	6	5	4	4	3

Figure 1. Output for the transformation of *kitten* into *sitting*.

These results were obtained by the following procedure:

1. A matrix (table) was initialized by measuring in the (i,j)-cell the Levenshtein distance between the i-character prefix of one sequence with the j-prefix of the other sequence.
2. The table was filled from the upper left to the lower right corner. Each "jump" horizontally or vertically corresponds to an edit operation.
3. The diagonal "jump" costs either one, if the two characters in the row and column do not match or 0, if they do.

The number in the lower right corner of the table is the Levenshtein distance between both words.

Levenshtein Distance-Similarity Transformations

As discussed in one of our previous tutorials, distances and similarities are used in data mining and information retrieval as association measures (Garcia, 2016b).

As a distance metric, a Levenshtein Distance is not a measure of similarity, but of the lack of resemblance between sequences. The greater this distance, the more dissimilar the sequences are.

To convert a distance D into a similarity score S, we can use

$$S = 1/(1 + D) \quad (1)$$

as described by Lin (1998). Our tool does this for you, so you can use it to reproduce Lin's results for the conversion of *grandiloquent*.

Practical Applications

The Levenshtein Distance has been used:

1. in the implementation of a matrix distance calculator for reverse compliment DNA code design.
2. for the perceptive evaluation of dialect distance measurements.
3. for automatic marking of musical dictations.
4. for regular expressions approximate matching.
5. to identify if two genetic sequences have similar functions.
6. to filter blocks of email lists (candidate spam addresses) within a LED threshold value.
7. as the ultimate baby name explorer.
8. to name products and services like domains, brands, etc.
9. to conduct fuzzy search matches in EXCEL or your preferred environment.
10. for spamdexing search engines - by randomly converting text into gibberish.
11. for spam stemming search engines - by systematically appending edits to valid stems.
12. as part of a spell checker routine.
13. to identify duplicated content and plagiarism.
14. as a sorting criterion.

Conclusion

We have briefly discussed a distance metric, the Levenshtein Distance as a measure of the lack of similarity between sequences. This metric is a valuable concept for those conducting sequence analysis and text mining.

Sequences can be names, words, phrases, sentences, paragraphs, articles, entire corpuses, and so forth. All sort of variants and implementations have been documented.

References

Garcia, E. (2016). Levenshtein Edit Distance Calculator. Retrieved from http://www.minerazzi.com/tools/levenshtein/levenshtein-distance-calculator.php

Garcia, E. (2016b). A Tutorial on Distance and Similarity. Retrieved from http://www.minerazzi.com/tutorials/distance-similarity-tutorial.pdf

Gilleland, M. (2015). Levenshtein Distance, in Three Flavors. Retrieved from http://people.cs.pitt.edu/~kirk/cs1501/Pruhs/Fall2006/Assignments/editdistance/Levenshtein%20Distance.htm

IEEE (2015). Vladimir Levenshtein. Retrieved from http://www.ieeeghn.org/wiki/index.php/Vladimir_I._Levenshtein

Lin, D. (1998). An Information-Theoretic Definition of Similarity. ICML '98 Proceedings of the Fifteenth International Conference on Machine Learning. pp. 296-304. Retrieved from http://www.cs.ualberta.ca/~lindek/papers/sim.pdf

Levenshtein.net (2015). Levenshtein Implementation. Retrieved from http://www.levenshtein.net/levenshtein_implementation.htm

Wikibooks (2015). Algorithm Implementation/Strings/Levenshtein distance. Retrieved from http://en.wikibooks.org/wiki/Algorithm_Implementation/Strings/Levenshtein_distance

Wikipedia (2015). Damerau-Levenshtein Distance. Retrieved from http://en.wikipedia.org/wiki/Damerau%E2%80%93Levenshtein_distance