Comparing Various String Distance Algorithms in the Task of Name Matching

THE UNIVERSITY OF UTAH

Symposium on Data Science and Statistics (SDSS) 2019

ALEKSANDRA ZABA (aleksandra.zaba@utah.edu)

UNIVERSITY OF UTAH

www.researchgate.net/profile/Aleksandra_Zaba
https://www.linkedin.com/in/alexandra-z-89245513b

Introduction

Automated systems have been documented based on measures such as recall (TP/(TP+FN)), precision (TP/(TP+FP)), and the f-measure (2*(recall*precision/recall+precision))

Example: String distance algorithms

Used/researched in, for ex.: DNA matching (Kucherov et al. 2014), machine translation (Przybocki et al. 2006), plagiarism research (Irving 2004), language production (Vitevitch & Sommers 2003; Zaba & Schmidt 2011), language acquisition (Storkel 2004), spell-checking (Norvig 2007), record or phonetic matching (including name matching tasks) (Fellegi & Sunter 1969; Zobel & Dart 1996; Bilenko & Mooney 2003; Cohen et al. 2003; Cheatham & Hitzler 2013; Del Pilar Angeles & Espino-Gamez 2015)

The current pilot study reports recall, precision, and thresholds for highest f-measures for three groups of string distance algorithms contained in R's 'stringdist' package (van der Loo 2014; R Journal 6(1) pp 111-122)

Name Matching Task

US-American female first name (n = 100) is matched in R to itself ('same') and to two of the following: Its foreign version ('same'), its male version in English ('different'), and one of its different, female versions in English ('different'); if 'different' is in the range of algorithm matches (based on threshold) → FP, if 'same' is not → FN

Ex. (of a difficult instance): With threshold 1, *Christina* ('same' & 'match' = TP), *Cristina* ('different' & 'match' = FP), *Christian* ('different' & 'nonmatch' = TN), *Krystyna* ('same' & 'nonmatch' = FN)

Algorithms

Ex. R code (to be adjusted by algorithm):
stringdist(stringdata$A, stringdata$B, method=c(''), useBytes = FALSE, weight = c(d = 1, s = 1, t = 1), q=1, p=0, bt=0, nthread=getOption("sd_num_thread"))

Edit based

Levenshtein (L. 1966): Weighted number of insertions, deletions, and substitutions

Hamming (H. 1950): Character substitutions

(Full) Levenshtein-Damerau (Lowrance & Wagner 1975): Insertions, deletions or substitutions of a single character, or transposition of two adjacent characters

Longest common substring (Needleman & Wunsch 1970): Deletions and insertions

N-Gram based

Jaccard: 1-((intersection)/(union)); 0-1

q-gram: counts n-grams not shared

Cosine: Cosine of angle between two vectors; 0-1

Heuristic

Jaro (J. 1978): Matching characters btw. two strings not too far apart; penalty for matching characters transposed

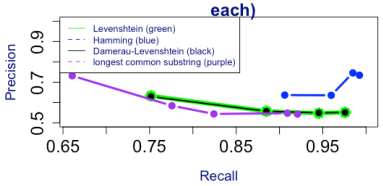
Jaro-Winkler: Adds penalty to character mismatches in the first four characters

F-Measure Results

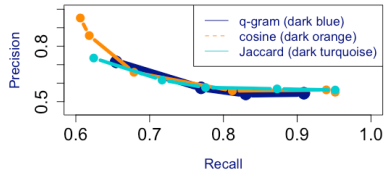
Algorithm	Thresholds with highest f-measure
Edit	Threshold 4 Levenshtein (.7), 4 Levenshtein-Damerau (.7), 3 Hamming (.85), 1 longest common substring (.69)
N-gram	.5 Jaccard (.71), 1 n-gram (.68), .03 cosine (.74)
Heuristic	.05 Jaro (.74), .03 Jaro-Winkler (.74)

Recall & Precision Results

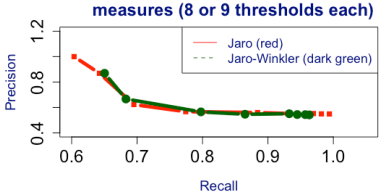
Precision-recall curves for 4 edit based distance measures (4 or 5 thresholds each)



Precision-recall curves for 3 n-gram based measures (4, 5, or 6 thresholds each)



Precision-recall curves for 2 heuristic measures (8 or 9 thresholds each)



Summary & Conclusions

Thresholds with highest f-measure vary by algorithm (see, also, e.g., Cohen et al. 2003 for results that vary by algorithm)

Higher f-measures in heuristic and n-gram groups vs. in edit based group

Summary & Conclusions

In all three algorithm groups, precision is highest at early thresholds, recall highest at later thresholds

Findings may be of interest to many applied fields; ex.: record matching in political surveys

More names needed to confirm results and more detailed thresholds for heuristic and n-gram groups

Different packages in R will be investigated

Acknowledgments

Thank you to Michelle Cheatham, Pascal Hitzler, Mark van der Loo, and Maria del Pilar Angeles for responses to queries regarding their own studies and regarding name-matching tasks in general and/or for details on the 'stringdist' package in R. Any mistakes are my own.

Selected References

Bilenko, M. & R.J. Mooney, (2003) "Employing trainable string similarity metrics for information integration", *Proceedings of the IJCAI-2003 Workshop*, pp67-72.

Cheatham, M. & P. Hitzler, (2013) "String similarity metrics for ontology alignment", *The Semantic Web ISWC part II, LNCS 8219*, pp294-309.

Cohen, W.W., P. Ravikumar, S.E. Fienberg, (2003) "A comparison of string distance metrics for name-matching tasks", *AAAI*.

Del Pilar-Angeles & Espino-Gamez, (2015) "Comparison of methods Hamming distance, Jaro, and Monge-Elkan", *DBKDA* 7.

Fellege, I.P. & A.B. Sunter, (1969) "A theory for record linkage", *Journal of the ASA* 64, pp1183-1210.

Irving, R.W., (2004) "Plagiarism and collusion detection using the Smith-Waterman algorithm".

Kucherov , G., K. Salikhov, & D. Tsur, (2014) "Approximate string matching using a bidirectional index", *Theor. Comp. Sci.*

Przybocki, M., G. Sanders & A. Le (2006) "Edit distance: A metric for machine translation evaluation".

Vitevitch M.S. & M.S. Sommers, (2003) "The facilitative influence of phonological similarity and neighbourhood frequency in speech production in younger and older adults", *Mem. & Cogn.* 31, pp491-504.

Zaba, A. & T. Schmidt, (2011) https://journals.linguisticsociety.org/proceedings/index.php/ExtendedAbs/article/view/549.

Zobel, J. & P. Dart, (1996) "Phonetic string matching: Lessons from info. retrieval", *SIGIR '96 Proceedings of the 19th annual intern. ACM SIGIR conf. on res. and dev. in info. retrieval*, pp166-172.