International Journal of Computer Trends and Technology
ISSN: 2231 – 2803 / https://doi.org/10.14445/22312803/IJCTT-V69I3P101
Volume 69 Issue 3, 1-4, March, 2021
© 2021 Seventh Sense Research Group®

Original Article

Comparison of Apache SOLR Search Spellcheck String Distance Measure – Levenshtein, Jaro Winkler, and N-Gram

Parameswara Rao Kandregula1

[1]*IT Consultant, Cognizant Technology Solutions, Houston, USA*

Received Date: 12 January 2021
Revised Date: 01 March 2021
Accepted Date: 03 March 2021

***Abstract* - String Distance is one of the key metrics for string comparison used in spell correction, and Levenshtein, JaroWinkler, and N-Gram are famous string distance and similarity measuring algorithms. Spelling mistakes are often not more than two or three characters for the normal user when typing on a website search functionality. In this article, in the context of the e-commerce website, we will test and compare the results of spellcheck distance measure implementations provided by apache SOLR search, which are Levenshtein, JaroWinkler, and N-Gram*

***Keywords* - Search engine, SOLR, Natural language processing, String distance, Levenshtein.*

I. INTRODUCTION

String Distance is one of the key metrics in natural language processing for string comparison used in spell correction to identify the correct word for the miss-spelled word. Levenshtien, JaroWinkler, and N-Gram are famous string distance measuring algorithms. But common spelling mistakes [6] are often not more than two or three characters for the normal user when typing on a website search functionality. And e-commerce websites are more keen on selling products related to the input words. Thus, does it really matter if we pick one algorithm over the other for a normal e-commerce website search. In this article, we will test and compare the results of spellcheck distance measure implementations provided by apache SOLR search, which are Levensthein [1], JaroWinkler [2], and N-Gram [3].

II. SOLR AND STRING DISTANCE

Solr is the most famous open-source search engine built on lucene, written in java, and used by several enterprises to implement their search functionality with the capability to handle millions of requests per second. As part of its features, to implement spellcheck, it provides few spell check implementations bundled. It also provides flexibility to extend and implement custom implementations. It has a base interface for string distance called org.apache.lucene.search.spell.StringDistance [4]. It has a method of getDistance(string s1, string s2) with returns value between 0 to 1 based on how similar the parameter strings s1 and s2 are. 0 means not at all similar and 1 being exactly similar. Solr provides four implementations of it using the famous similarity metrics approaches - Levenshtein, JaroWinkler, and N-Gram:

- org.apache.lucene.search.spell.Levenshtein distance
- org.apache.lucene.search.spell.LuceneLevenshteinDistance
- org.apache.lucene.search.spell.JaroWinklerDistance
- org.apache.lucene.search.spell.NGramDistance

The edit distance between two strings s1 and s2, at a high level is a number of letter level edits needs to make one string similar to the other. Edit operations are like insert, delete, substitute and transpose. All above-listed SOLR implementations are implemented along with the edit distance with added weightage and formula related to other characteristics like prefix similarity, length, the distance between letters, etc. LuceneLevenshteinDistance implementation is not an efficient one and is recommended by SOLR only to merge responses from other implementations. Thus for this test, Lucene Levenshtein Distance implementation has been excluded from the test and comparison.

III. SOLR SPELL CHECK COMPARISON

The test were run locally using local solr on MacBook using a dataset from Apache Solr and Kaggle as described below:

A. Hardware and Software versions used

- MacOs Big Sur, Version 11.2.1
- Processor : 2.3 GHz Dua-Core Intel i5
- Memory; 8 GB 2133 MHz LPDDR3

- Solr : 8.8.0
- Java : 12.0.1
- Jetty : 9.4.34.v20201102

B. Index Source Test Dataset

The test data of tech products bundled with solr has only 32 document records. The e-commerce product data set found on Kaggle[5] was downloaded, columns modified, added to tech products solr core, and used as index source for the testing. The eventual test core has 20032 documents.

C. Source Dictionary Data

Spell Check needs a dictionary against which the correctly spelled words can be derived after comparing with input mis-spelled words. In solr, it can be a fixed dictionary file, or the index of source test data itself can be used. We will be using the index of source test data as the objective of the test is to validate results from an e-commerce website perspective.

D. Solr SpellCheck Configuration Constants

Solr configuration constants related to spellcheck components are as below. Of all, one of the configurations worth noting is "MaxEdits' i.e., the maximum number of edits allowed. It is allowed to be either 1 or 2 only. Any word misspelled with three or more characters would not be producing any suggestions. This is as per guidelines that most spelling mistakes are two in word by the user and also the performance impact since search queries are expected to be very fast. If there is a need to allow suggestions for more than 2 edits, solr allows having custom implementation, plug into solr, and use it. Anyhow for this test case, solr's out-of-the-box implementations have been used. By default minimum prefix needed for spellcheck is 1, i.e., the starting letter has to match. But since we wanted to compare the impact of misspelled the first letter in a word, the configuration value for the minimum prefix has been changed to zero. Also, the collations have been set to false(spellcheck.collate=false) as the test is more on single word spell correction, and collation would anyhow take corrections from individual words and then create a new phrase. Another parameter is called spellcheck. The count has been set to three, and it defines the maximum number of suggestions requested.

```
<!-- minimum accuracy needed to be considered a valid spellcheck suggestion -->
<float name="accuracy">0.5</float>
<!-- the maximum #edits we consider when enumerating terms: can be 1 or 2 -->
<int name="maxEdits">2</int>
<!-- the minimum shared prefix when enumerating terms -->
<int name="minPrefix">0</int>
<!-- maximum number of inspections per result. -->
<int name="maxInspections">5</int>
<!-- minimum length of a query term to be considered for correction -->
<int name="minQueryLength">3</int>
<!-- maximum threshold of documents a query term can appear to be considered for correction -->
<float name="maxQueryFrequency">0.01</float>
```

E. Solr Log Configuration

To capture the response time properly, the log level in solr admin using log4j2 for org.eclipse.jetty.server.* has been set to all levels. Initial test response times were not showing much difference as they were logged in microseconds. Log level timestamp has been changed to capture time at nanosecond level : yyyy-MM-dd HH:mm: ss,nnnnnnnnn.

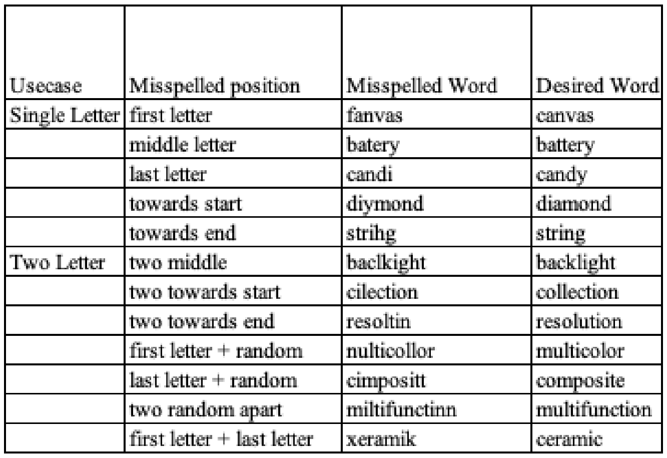
F. Solr SpellCheck Component

Three custom solr spellcheck components have been created in solrconfig.xml. They were named levenh, jarowink, and ngram, each map to solr spell check implementations correspondingly to - LevenshteinDistance, JaroWinklerDistance, and NGramDistance.

G. Spell Check Test Data

Since solr index is used as a dictionary source, the test data has to be selected based on the words in the solr index documents, and else there would be no suggestions post spell check. Thus the words in test documents were examined, and below words picked to use for this test.

Table 1. Input Test Data



Usecase	Misspelled position	Misspelled Word	Desired Word
Single Letter	first letter	fanvas	canvas
	middle letter	batery	battery
	last letter	candi	candy
	towards start	diymond	diamond
	towards end	strihg	string
Two Letter	two middle	baclkight	backlight
	two towards start	cilection	collection
	two towards end	resoltin	resolution
	first letter + random	nulticollor	multicolor
	last letter + random	cimpositt	composite
	two random apart	miltifunctinn	multifunction
	first letter + last letter	xeramik	ceramic

H. Test Cases Execution

For each of the words from the test data, three tests were run on solr spellcheck, one each for LevenshteinDistance(levenh), JaroWinklerDistance(jarowink), and NGramDistance(ngram). For noting response time, the server restarted before each set of executions for lvenh, jarowink. And ngrams. The restart is necessary to clear caches and avoid skewing the response time captured.

IV. OUTPUT AND ANALYSIS

For each of the execution, related output data were captured – i) list of suggestions if any returned ii) is the desired word returned in suggestions iii) string distance measure between misspelled and desired word in the scale of 0 to 1. iv) response time in nanoseconds.

A. List of suggestions

List of suggestions(if any) for each of the word in test data from spell check using each of the string distance implementation of solr is as below:

Table 2. List of Suggestions

Misspelled Word	Desired Word	Suggestions - Levenh	Suggestions - JaroWinkler	Suggestions - NGram
fanvas	canvas	canvas,kanvas,fantasy	fans,fantasy,canvas	canvas,kanvas,fantasy
batery	battery	battery,bakery,batter	battery,battrey,batter	battery,bakery,batters
candi	candy	candy,candid,handi	candid,candies,candied	candy,candid,candle
diymond	diamond	diamond,diamonds,diamong	diamond,dimonds,daimond	diamond,diamonds,diamong
strihg	string	string,strong,strict	stright,string,strings	string,strict,stripe
baclkight	backlight	backlight,blacklight	backlight,blacklight	backlight,blacklight
cilection	collection	collection	collection	collection
resoltin	resolution	resolution,resulting,resoluti	resolution,resulting,resoluti	resolution,resulting,resultion
nulticollor	multicolor	NO RESULTS	NO RESULTS	NO RESULTS
cimpositt	composite	composite	composite	composite
miltifunctinn	multifunction	multifunction	multifunction	multifunction
xeramik	ceramic	NO RESULTS	NO RESULTS	NO RESULTS

B. The desired word returned (yes/no)

If the desired word was returned for each of the words from test data against each of the string distance implementation of solr is as below:

Table 3. Desired Word Returned (Yes/No)

		Desired Word Returned(Yes/No)		
Misspelled W	Desired Word	Levenh	JaroWin	NGram
fanvas	canvas	Yes	Yes	Yes
batery	battery	Yes	Yes	Yes
candi	candy	Yes	Yes	Yes
diymond	diamond	Yes	Yes	Yes
strihg	string	Yes	Yes	Yes
baclkight	backlight	Yes	Yes	Yes
cilection	collection	Yes	Yes	Yes
resoltin	resolution	Yes	Yes	Yes
nulticollor	multicolor	NO	NO	NO
cimpositt	composite	Yes	Yes	Yes
miltifunctinn	multifunction	Yes	Yes	Yes
xeramik	ceramic	NO	NO	NO

C. String Distance Measurer (0 to 1)

Sting distance was measured using the strdist function provided with solr, which takes three parameters: two parameters as strings for comparison and the third being the distance measure to use to compare the two strings. String Distance Measure for each of the word from test data against each of the string distance implementation of solr is as below:

Table 4. String Distance Measure (0 to 1)

Misspelled Word	Desired Word	StringDistance - Levenh	StringDistance - JaroWinkler	StringDistance - Ngram
fanvas	canvas	0.8333333	0.88888884	0.75
batery	battery	0.85714287	0.9666667	0.85714287
candi	candy	0.8	0.91999996	0.9
diymond	diamond	0.85714287	0.9238096	0.85714287
strihg	string	0.8333333	0.9333333	0.8333333
baclkight	backlight	0.7777778	0.97407407	0.7777778
cilection	collection	0.8	0.9066667	0.8
resoltin	resolution	0.8	0.9666667	0.7
nulticollor	multicolor	0.8181818	0.90606064	0.77272725
cimpositt	composite	0.7777778	0.82380956	0.8333333
miltifunctinn	multifunction	0.84615386	0.8773534	0.84615386
xeramik	ceramic	0.71428573	0.8095238	0.71428573

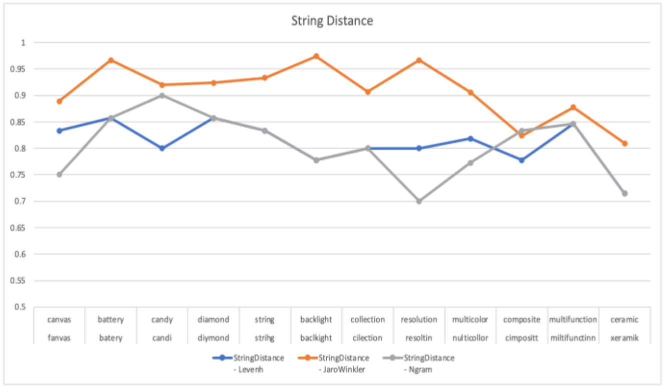


Fig. 1 String Distance Measurer (0 to 1)

D. Response Time

Response Time for each of the word from test data against each of the string distance implementation of solr is as below:

Table 5. Response Time

Misspelled Word	Desired Word	Response Time (nanoseconds) - Levenh	Response Time (nanoseconds) - JaroWinkler	Response Time (nanoseconds) - Ngram
fanvas	canvas	27056000	31297000	36946000
batery	battery	33762000	33131000	22423000
candi	candy	29138000	26148000	30209000
diymond	diamond	37564000	28699000	25353000
strihg	string	25216000	30609000	15394000
baclkight	backlight	24901000	20504000	19358000
cilection	collection	23364000	15454000	15328000
resoltin	resolution	24888000	19056000	22755000
nulticollor	multicolor	23831000	14564000	14286000
cimpositt	composite	19524000	17750000	16576000
miltifunctinn	multifunction	14614000	15398000	19071000
xeramik	ceramic	16663000	15473000	11765000

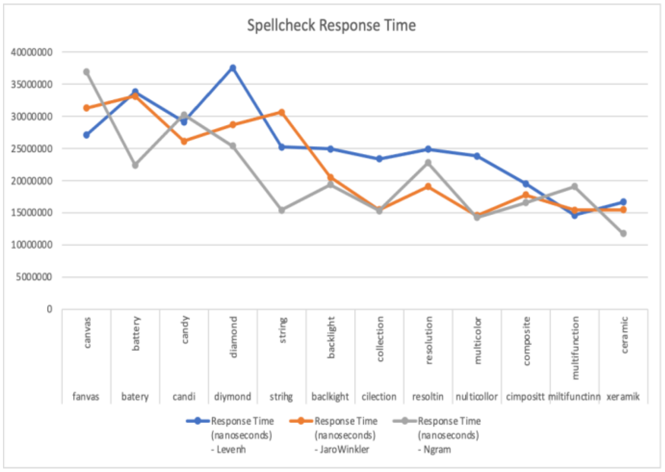


Fig. 2 Response Time

E. Analysis and Derivations

All the three (LevenshteinDistance, JaroWinklerDistance, and NGramDistance) returned an almost similar set of suggestions. Levenshtein distance and NGramDistance were identical

All had the desired word in the returned suggestions except for “multicolor” and “ceramic.” Interestingly all three had negative results for the same two words.

All three had almost similar string distance. LevenstheinDistance and NGramDistance are almost identical, and JaroWinklerDistance had a small difference from the other two in string distance. Even the words(“multicolor” and “ceramic”) which did not have the desired word in the returned list of suggestions have decent string distance similarity. There might have been other limitations like the number for comparisons limit, which would have made it not result from those two words as suggestions.

Even from performance wise also there isn’t much to choose between the three. Levenshtein distance had negligible increased response time in few cases.

VI. CONCLUSION

All the three solr string distance implementations behaved similarly in the test conditions. But the context of the testing is for e-commerce site data. Thus linguistic accuracy need not be a hundred percent achieved, and understanding user intent and user retention on a website are more important. The user-desired word might not be sold by the website; thus, the dictionary for the website should always be picked from website records to avoid showing word suggestions that do not have any products sold by the website. As long as the closest desired word is returned, the results can be tuned to the benefit of business and user, like boosting the desired products or adding contextual weightage to the search request. Even if the first suggested word is not what the user is looking for, websites can easily use the “did you mean” feature and show the remaining suggestions. In addition to spellcheck, Solr also provides query intent and training data to apply machine learning to return results as a closet to the user’s intent and to serve business needs at the same time.

If the same is used for scientific analysis like DNA comparison, then more analysis is needed to check if these string distance implementations are apt for them. All the three give added weightage if the starting of string matches, thus in use cases in which all the positions of string to compare are equally important, then these might not give the desired results. Anyhow, such cases are restricted to very few fields like medical, etc., and for most of the rest, these would equally work with great results and performance and options to tune.

ACKNOWLEDGMENT

Would like to thank the apache foundation for making solr, a powerful search engine as open source and all the volunteer contributors from worldwide who regularly maintain solr, document it, add features and keep it to the highest standards

REFERENCES

[1] Levenshtein distance. [Online]. Available: https://en.wikipedia.org/wiki/Levenshtein_distance
[2] Jaro Winkler distance. [Online]. Available: https://en.wikipedia.org/wiki/Jaro%E2%80%93Winkler_distance
[3] N-Gram distance. [Online]. Available: https://lucene.apache.org/core/8_0_0/suggest/org/apache/lucene/search/spell/NGramDistance.html
[4] Solr StringDistance [Online]. Available: https://lucene.apache.org/core/8_0_0/suggest/org/apache/lucene/search/spell/StringDistance.html
[5] Kaggle Flipkart Products Data [Online]. Available: https://www.kaggle.com/PromptCloudHQ/flipkart-products
[6] Common spelling mistakes [Online]. Available: https://www.lexico.com/grammar/common-misspellings