TELKOMNIKA, Vol.16, No.2, April 2018, pp. 827~833
ISSN: 1693-6930, accredited **A** by DIKTI, Decree No: 58/DIKTI/Kep/2013
DOI: 10.12928/TELKOMNIKA.v16i2.6890

Fast and Accurate Spelling Correction Using Trie and Damerau-levenshtein Distance Bigram

Viny Christanti M.*[1], Rudy[2], Dali S. Naga[3]
Facuty of Information Technology, Tarumanagara University, Telp. (021) 5671747 Jl. Letjen S. Parman No. 1, Jakarta, Indonesia
*Corresponding author, e-mail: viny@untar.ac.id, 535120049@fti.untar.ac.id, dalinaga@gmail.com

Abstract

This research was intended to create a fast and accurate spelling correction system with the ability to handle both kind of spelling errors, non-word and real word errors. Existing spelling correction system was analyzed and was then applied some modifications to improve its accuracy and speed. The proposed spelling correction system is then built based on the method and intuition used by existing system along with the modifications made in previous step. The result is a various spelling correction system using different methods. Best result is achieved by the system that uses bigram with Trie and Damerau-Levenshtein distance with the word level accuracy of 84.62% and an average processing speed of 18.89 ms per sentence.

Keywords*: damerau-levenshtein distance, bigram, spelling correction, trie*

1. Introduction

Language is an arbitrary sound symbol system, which was used by society's member to work, interact and self identification [1]. There are two forms of language, written and spoken. Writing has an important role in conveying the written meaning. Errors in the writing may cause the meaning to be unconveyed or even distorted into different meaning.

Spelling correction is a task for detecting and fixing any spelling errors. There are two kinds of spelling errors, a real word error and non-word error [2]. Real word error is an error that causes a certain word to transform into other word with the same or different meaning. While a non-word error is an error that causes a certain word to transform into a meaningless word.

Existing research on spelling correction is generally conducted only on one kind of the spelling errors [3-6]. The methods used to handle each kind of spelling error quite differ from one another. Most spelling correction system tackles each problem separately. The main purpose in this research is to create a fast and accurate spelling correction system with the ability to fix both kinds of spelling errors.

Some of the relevant research used as main reference are done by Setiadi [3] and Verberne [4]. In this research, Setiadi showed an alternative in optimizing the performance of spelling correction in term of speed and accuracy. Setiadi reduced the edit distance computation using a certain word length rule and using simple assumption to add extra value or rank to a certain word correction candidate. Verberne uses trigram to handle the real word error and split sentence into parts in trigram form before checking and correcting any spelling error.

Existing system is then modified in this research to improve its accuracy and speed. Trie data structure is added in Setiadi's system to represent known vocabulary list which is supposed to improve processing speed on word validity check and word correction candidate generation. Smoothing technique is added in Verberne's system, supposedly to reduce data sparsity problem. And a precomputation of edit distance is used to remove the recomputation of previously seen word (on generation of correction candidate word). Bigram is tried as an alternative to trigram in Verberne's system. The proposed system is built based on the method and intuition in the aforementioned system with the modifications made to improve its performance.

Received July 26, 2017; Revised February 12, 2018; Accepted February 28, 2018

2. Research Method

There are some categories of spelling errors, such as substitution, deletion, insertion, transposition and split word [6]. In this research, all categories of spelling errors will be covered, except for split word error. Illustration for the spelling errors categories can be seen in **Error! Reference source not found.**.

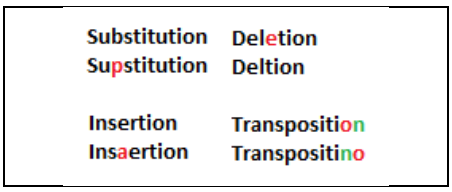

Figure 1. Spelling errors categories

Edit distance method is used in the referenced system, as well as in this research. Edit distance is the operation cost needed to transform a word into another [7]. There are various other string comparison methods such as the Hamming distance, Longest Common Subsequence, Jaro Wrinkler distance, etc. The most generally known edit distance is Levenshtein distance. Levenshtein distance considers the cost for a certain operation such as deletion, insertion and substitution of characters in word. As such, it should be more suitable for spelling correction task. Another variation of Levenshtein distance is the Damerau-Levenshtein distance, which considers another operation cost, which is the transposition of adjacent letters in a word [7]. Based on [7], equation for Levenshtein distance can be seen in (1) and Damerau-Levenshtein distance in (2).

$$d_{a,b}(i,j) = \begin{cases} \max(i,j), \text{ if } min(i,j) = 0, \\ \min \begin{cases} d_{a,b}(i-1,j) + 1 \\ d_{a,b}(i,j-1) + 1 \\ d_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} , else \end{cases} \quad (1)$$

$$d_{a,b}(i,j) = \begin{cases} \max(i,j), \text{ if } \min(i,j) = 0, \\ \min \begin{cases} d_{a,b}(i-1,j) + 1 \\ d_{a,b}(i,j-1) + 1 \\ d_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \\ d_{a,b}(i-2,j-2) + 1 \end{cases} , \text{ if } i,j > 1 \;\&\; \mathrm{a}_i = \mathrm{b}_j - 1 \;\&\; \mathrm{a}_i - 1 = \mathrm{b}_j \\ \min \begin{cases} d_{a,b}(i-1,j) + 1 \\ d_{a,b}(i,j-1) + 1 \\ d_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} , \text{ else} \end{cases} \quad (2)$$

N-gram is a contiguous sequence of N item, which could be words, letters, syllables, or phonemes [2]. For example bigram (2-gram) is sequence that consist of two words sequence such as “salah satu”, “satu kota”, “kota besar”. The probability of a long sentence can be computed by splitting it into smaller parts and use the conditional probability rule to compute the overall probability. N-gram gives the highest accuracy value to all word gram level for similarity detection in stemming process [8]. In a spelling correction task, N-gram is used to set the probability of correction words. Based on [2], the general equation for N-gram probability can be seen in (3).

$$P(w_n | w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1} w_n)}{C(w_{n-N+1}^{n-1})} \quad (3)$$

In computing a probability, it gives zero to unseen events. Smoothing is a process of giving a certain probability value to unseen events by reducing the probability of seen events [2]. Smoothing is used to avoid the zero probability given by the language model. There are various

smoothing methods, such as Laplace, Good-Turing, Kneser-Ney, Add-K, Back-Off, Interpolation, etc. Smoothing method is usually combined to produce better result such as using Good-Turing and Interpolation [2]. In this research, combination of Laplace and Back-off is used as the smoothing method. Based on [2], Equation for Laplace smoothing can be seen in (4) and Back-off in (5).

$$P_{Laplace}(w_n) = \frac{C(w_n)+1}{N+V} \tag{4}$$

$$P_{Back-off}(w_n|w_{n-N+1}^{n-1}) = \begin{cases} P^*(w_n|w_{n-N+1}^{n-1}), \text{ if } C(w_{n-N+1}^{n-1}) > 0 \\ \alpha(w_{n-N+1}^{n-1}) P_{Back-off}(w_{n-N+1}^{n-1}), \text{ else} \end{cases} \tag{5}$$

Perplexity is an intrinsic evaluation method for a language model. Perplexity of a language model is computed by inverting the probability from a test set (sentence) and normalized by the count of words [2]. By inverting the probability, it means that the higher the probability of a given test set, the lower is the perplexity. Low perplexity means that the applied language model is better, although it doesn't guarantee the increase in accuracy for extrinsic evaluation. Based on [2], equation for Perplexity can be seen in (6).

$$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_1 \ldots w_{i-1})}} \tag{6}$$

First step of this research is to build the referenced existing spelling correction system by using the same programming language as the proposed system. This was done in order to accurately compare the referenced spelling correction system against the proposed one as programming languages may vary from one another in term of performance. The referenced spelling correction can be accurately built based on the explanation in the respective paper and with the help of source code (of different language) generously shared by the referenced system's author.

Data in this research is an online news article taken from Kompas (kompas.com). Data is taken by crawling the site using a crawler built with the help of an external software library. The amount of data crawled and used in this research is 5000 online news articles. The crawled news articles are from 2nd to 12th March 2016. The crawled news category count can be seen in Table 1.

Table 1. News Article Category Distribution

Category	Amount
Regional	867
Megapolitan	830
Nasional	714
Entertainment	393
Bola	365
Ekonomi	355
International	294
Automotive	228
Travel	204
Property	184
Sports	142
Techno	140
Health	136
Female	99
Science	34
Education	15
TOTAL	5000

The characteristics of Setiadi's system that yielded the best result in term of performance is that the edit distance computation is pruned or reduced using a simple rule [3]. The rule used is dismissing computing edit distance if the correction candidate word has a

length of less or more than 1 compared against the mistyped word. This method helps to speed up processing time. An alternative for more efficient computation is by using a Trie data structure to represents the known words. Computing edit distance on words represented in a Trie data structure can greatly improve the processing speed because Trie data structure groups words based on its prefix [8]. Normally, edit distance is computed on every words with the same prefix. By using a Trie data structure, words with the same prefix will only be computed once. Validation of a word should also be faster since the time required for a string matching is O(n) for n as the longest string length [8]-[10]. Illustration for Trie data structure can be seen in Figure 2.

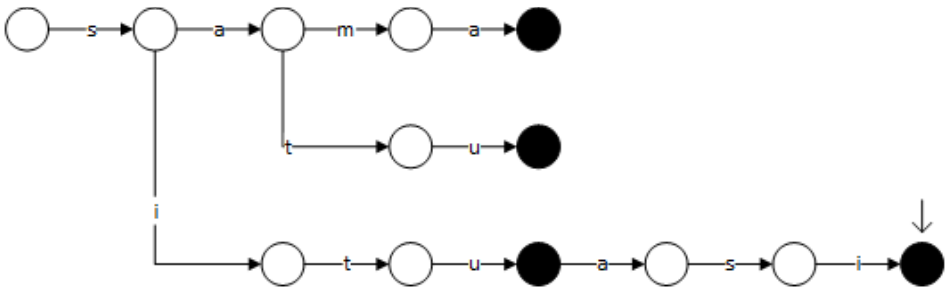

Figure 2. Trie Data Structure Illustration for Word: “sama”, “satu”, “situasi”

Setiadi also used a bayes theorem to make a simple assumption to add an extra value or rank to a certain correction candidate word. The assumption is that a word with longer length is more likely to have the need to correct its mistyped word when compared with another correction candidate word of shorter length. Although, this simple assumption yielded the best result than when without using it in Setiadi’s research, we’ll try and see if it also yields a good result when using our data.

In Verberne’s system, the generation of correction candidate words was conducted everytime it detected a real word error. A word might be detected as an error multiple times in a sentence, hence the system will regenerate (recompute edit distance) correction candidate words of the same words multiple times. We can precompute this process to speed up processing time since the word is limited to only known words in case of a real word error. Bigram is experimented as an alternative to Trigram in Verberne’s system to see its impact on the performance.

The proposed system is built by combining the referenced system along with its modification to improve the overall performance. Setiadi’s system accepts single word as input, while Verberne system accepts a whole sentence but splits them into trigram parts and corrects it separately. The proposed system will accept sentence as input then splits them into n-gram parts, corrects it separately and combines it back into a whole sentence using the chain rule probability.

The language model of the data is evaluated by computing the perplexity of each model (unigram, bigram, and trigram). Categories of the test made against the system is the accuracy rate, overall processing speed, and false-positive rate. Test data for the testing of the perplexity is a different news article taken from a random source, which is different from the ones used as the main data for the system. Perplexity is computed as an average of overall perplexity across all sentences in the article. Test data for the other test is a 50 sentences (475 words) data taken from a news article of random source which is modified to contain a real word error (58 error words), a non-word error (50 error words), and a combination of both (104 error words) (3 sets).

3. Results and Analysis

3.1. Perplexity

Perplexity result for language model using Equation (6), can be seen in Table 2. The lowest result is achieved by using unigram. This shows that there is a high rate of data sparsity in the used data set. Trigram yield the highest value, which shows that the spelling correction

system that uses trigram might not perform well than that using bigram. To reduce the data sparsity, a smoothing technique is added to the spelling correction system.

Table 2. Perplexity Test Result

N-Gram	Average Perplexity
Unigram	21834.7362
Bigram	170415.4672
Trigram	84060875.6503

3.2. Referenced System

The test result for the referenced system can be seen in Table 3 and Table 4.

Table 3. Setiadi's System Test Result Comparison

Test Categories	Setiadi	Setiadi (Modified)
Sentence Accuracy (%)	70	78
Word Accuracy (%)	76	84
False-Positive (%)	1.6	1.6
Time Spent (s)	24.35	0.49
Average Time Spent (ms)	487.01	9.71

Table 4. Verberne's System Test Result Comparison

Test Categories	Verberne	Verberne(Modified)
Sentence Accuracy (%)	6	26
Word Accuracy (%)	5.17	29.31
False-Positive (%)	0	1.09
Time Spent (s)	1.35	0.37
Average Time Spent (ms)	26.99	7.46

Based on the test result of Setiadi's system, there is a significant change in term of processing speed when using a Trie data structure. The achieved accuracy is also increased although it's not that significant. This shows that the use of simple assumption as mentioned previously doesn't perform well on our data. Based on the test result of Verberne's system, the processing speed increases when the edit distance is precomputed beforehand. Accuracy is also increased when a smoothing technique is added into it. Although the achieved accuracy is still considered low, it shows that the smoothing technique is able to reduce the data sparsity which in turn improves the accuracy. The smoothing technique used is the Add-1 Laplace smoothing (4) in combination with Stupid Back-off (uses 0.4 as α) (5).

3.2. Proposed System

The proposed system was built by combining the method in both of the referenced systems with the previously mentioned modifications. The proposed method uses a Trie data structure, a precomputation on edit distance, and a smoothing technique. Bigram and trigram is tried along with Damerau-Levenshtein (DLD) and Levenshtein distance. Different sizes of data are also tried on Bigram and Trigram.

Table 5. Proposed System (Trie+DLD) Test Result Comparison for Bigram

Test Categories	1000 Data	2000 Data	3000 Data	4000 Data	5000 Data
Sentence Accuracy (%)	42	48	46	50	50
Word Accuracy (%)	80.77	82.69	84.62	85.58	84.62
False-Positive (%)	7.2	6.5	7.8	6.5	6.5
Time Spent (s)	0.8	1.8	0.8	0.9	1.0
Average Time Spent (ms)	15.1	36.8	15.3	17.4	18.9

Table 6. Proposed System (Trie+DLD) Test Result Comparison for Trigram

Test Categories	1000 Data	2000 Data	3000 Data	4000 Data	5000 Data
Sentence Accuracy (%)	6	8	10	12	16
Word Accuracy (%)	16.35	24.04	27.89	30.77	34.62
False-Positive (%)	0.94	0.94	0.94	0.94	0.94
Time Spent (s)	0.51	0.62	0.88	0.91	1.04
Average Time Spent (ms)	10.27	12.23	17.56	18.06	20.72

Based on the comparasion of the test results of the proposed system for different sizes of data, the accuracy rate and the time spent increase as more data is used. The overall false-positive rate remains the same across all sizes of data. The main difference between bigram and trigram is that the test result for trigram is more consistent. In the test result for bigram, the accuracy falls from 85.58% to 84.62% when 5000 data is used. The newly added information (new word list, change in word frequency) affects the spelling correction result. Some of the previously correct words become incorrect and vice versa.

Table 7. Proposed System Test Results for Edit Distance Comparison

Test Categories	Bigram & Levenshtein	Trigram & Levenshtein	Bigram & Damerau-Levenshtein	Trigram & Damerau-Levenshtein
Sentence Accuracy (%)	36	10	50	16
Word Accuracy (%)	73.08	29.92	84.62	34.62
False-Positive (%)	6.54	0.94	6.54	0.94
Time Spent (s)	0.79	0.82	0.95	1.04
Average Time Spent (ms)	15.72	16.32	18.89	20.72

Based on the test result of proposed system for edit distance comparison, the accuracy of the system is improved at the cost of increased processing time. Using Damerau-Levenshtein distance yields a better result without significant change in processing time. The use of bigram or trigram also affects all aspects of the test. Overall, bigram has a higher accuracy, a higher false-positive rate and a lower processing time than those of trigram. The difference in accuracy shows that there is a high rate of data sparsity in trigram, and this is also proved by the previously mentioned perplexity test. The false-positive rate of trigram is higher than that of bigram. Naturally, a trigram contains a more contextual information than a bigram because of the difference in word pair length. While another reason for the low false-positive rate is because of the high rate of data sparsity which in turn lowers the amount of possible correction candidate.

4. Conclusion

The result of the research in creating a fast and accurate spelling correction system with the ability to fix real word error and non-word error ends with a moderately better result. Modifications made to the referenced system is able to increase its performance in term of accuracy and processing speed. The best result is achieved by the proposed system that uses a bigram with Damerau-Levenshtein distance with a sentence level accuracy of 50%, a word level accuracy of 84.62% and an average processing time per sentence of 18.89 ms.

Other things that can be mentioned based on the conducted research are as follows:

a. The processing speed is increased significantly by using a Trie and precomputed edit distance.
b. The use of a smoothing technique (Add-1 Laplace and Stupid Back-off) is able to reduce the high rate of data sparsity problem.
c. The use of bigram yields a better result than using trigram, which is caused by the difference in rates of data sparsity.
d. Using Damerau-Levenshtein distance yields a better result than using Levenshtein distance because it is able reach more variations of spelling errors, and the insignificant differences in processing time can be justified for the increase in accuracy.

Suggestions for further improvement in this research is to reduce memory usage of Trie data structure by compressing it and adding the use of database to store n-gram word list.

Another suggestions is the experiment on other smoothing technique to reduce data sparsity and improve the accuracy such as the Good Turing or Kneser-Ney smoothing. Another alternative is to use other data of different sources such as from the books.

References

[1] Pusat Bahasa Departemen Nasional. Kamus Besar Bahasa Indonesia. Jakarta: Pusat Bahasa. 2008: 119.
[2] Daniel Jurafsky, James H Martin. Speech and Language Processing. 2nd Edition. Upper Saddle River: Prentice Hall. 2008.
[3] Iskandar S. Damerau-Levenshtein Algorithm and Bayes Theorem for Spell Checker Optimization. Bandung Institute of Technology. Report number: X. 2013.
[4] Suzan V. Context-Sensitive Spell Checking Based on Word Trigram Probabilities. Nijmegen. University of Nijmegen. 2002.
[5] D Fossati, B Eugenio. *A Mixed Trigrams Approach for Context Sensitive Spell Checking*. Proceedings of the 8th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing). Mexico City. 2007; 4394: 622-633.
[6] L Bahari, B QasemiZadeh. *CloniZER Spell Checker Adaptive, Language Independent Spell Checker.* ICGST International Conference on Artificial Intelligence and Machine Learning. Cairo. 2005; 5: 65-71.
[7] Frederick J Damerau. A technique for computer detection and correction of spelling errors. Communications of the ACM. 1964; 7(3): 171-176.
[8] Mardiana T, Adji TB, Hidayah I. Stemming Influence on Similarity Detection of Abstract Written in Indonesia. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*. 2016; 14(1):219-227.
[9] Peter Brass. Advanced Data Structure. New York: Cambridge University Press. 2008.
[10] H Shang, TH Merrettall. Tries for approximate string matching. *IEEE Transactions on Knowledge and Data Engineering.* 1996; 8(4): 540-547.