Post-print of: Boiński T., Zimnicki A., Kujawski J., Draszawka K.: Evaluating Asymmetric N-Grams as Spell-Checking Mechanism. 2018 11th International Conference on Human System Interaction (HSI). DOI: : 10.1109/HSI.2018.8431345

Evaluating asymmetric n-grams as spell-checking mechanism

Tomasz Boiński
Department of Computer Architecture
Faculty of Electronics, Telecommunication and Informatics
Gdańsk University of Technology
Gdańsk, Poland
tobo@eti.pg.edu.pl

Julian Kujawski
Department of Computer Architecture
Faculty of Electronics, Telecommunication and Informatics
Gdańsk University of Technology
Gdańsk, Poland
tobo@eti.pg.edu.pl

Artur Zimnicki
Department of Computer Architecture
Faculty of Electronics, Telecommunication and Informatics
Gdańsk University of Technology
Gdańsk, Poland
tobo@eti.pg.edu.pl

Karol Draszawka
Department of Computer Architecture
Faculty of Electronics, Telecommunication and Informatics
Gdańsk University of Technology
Gdańsk, Poland
kadr@eti.pg.edu.pl

***Abstract*—Typical approaches to string comparing marks two strings as either different or equal without taking into account any similarity measures. Being able to judge similarity is however required for spelling error corrections, as we want to find the best match for a given word. In this paper we present a bi2quadro-grams method for spelling errors correction. The method proposed uses different n-grams dimension for the source (checked) and target (from the dictionary) words. For different types of errors proper weights were introduced. This way an increase in the quality and performance of the algorithm can be observed and the method becomes dedicated to the task of spelling errors correction. The results obtained so far suggest that the method is a viable solution competitive to other currently used approaches. The paper presents the proposed method, test suite and experimental results. Some discussion is also presented.**

***Index Terms*—bi2quadro-grams, spell-checking**

I. Introduction

String recognition is a very important element in all tasks coping with information retrieval [1] systems. In computer systems string recognition is usually done by word lookup in given dictionary based on string comparison. This process is usually of binary nature - two literals are either the same or not. Due to that the computers usually consider strings as equal or different, they do not take into account similarity of literals when doing for example database look-ups. Such behavior is in turn necessary for spelling corrections as misspelled word given as input will not be found in the corpora and the algorithm has to find the closest correctly spelled match.

Making mistakes is, in turn, a vital part of our nature. When creating documents however we need to eliminate those mistakes for various reasons such as the professional look of the document or even an ability to understand it. Error elimination also plays a crucial role when performing conversation with a computer. Independently from the aim of the conversation, whether it is for facts extraction [2] and knowledge acquisition [3], where the machine has to be able to understand the query the user provided, or document writing [4]. As a result, spelling errors need to be eliminated in almost all systems and solutions. Furthermore, the common occurrence of spelling errors requires an efficient method for words selection and judging of their correspondence.

In this paper, we propose to use bi2quadro-grams method for imperfect string matching as a method for spellchecking corrections of texts in English language. The algorithm has been compared with Microsoft Word 2013 spellchecker, which is a widely used solution. The tests were done using an application for learning how to write without looking on the keyboard. User activities were recorded and analyzed using the aforementioned algorithms and compared with the words that the users were supposed to enter.

The structure of the paper is as follows. Section II shows existing solutions. The next section (III) describes the bi2quadro-grams algorithm. Section IV shows performance evaluation, the test application, the results gathered from the players and presents the quality evaluation of the results. Finally some conclusions are given.

II. existing solutions

Imperfect string matching is usually based on distance metrics. One of the first and the most intuitive is the Hamming distance [5] denoting the number of differences between two words of the same length. This, unfortunately, is not sufficient for spelling errors correction as it takes into account only letter change and omits other types of errors. The improved algorithm has been proposed by Levenshtein [6]. This measure is based on calculating number of operations needed to convert

978-1-5386-5024-0/18/$31.00

one string into another. Optimized version of this algorithm has very good performance and is used in many solutions [7]. This algorithm, however, does not cope well with the most common type of errors - swapping letters [8]. A modification was done to mitigate this problem (called Damerau-Levenshtein method [8]) but it influences the performance of the algorithm to a great extent. To improve the performance of the algorithms dedicated indexes [9] or hashing [10], [11] can be used. Unfortunately existing techniques of performance improvements introduce accuracy problems. For example, solutions based on Locality Sensitive Hashing [9], may decrease precision and/or recall of dictionary entries retrieval, as usually the quality of the obtained results is correlated with the similarity function.

Some solutions try to bypass this problem. One of them is based on a type of neural networks called denoising encoders [12]. This solution provides high quality results, the O(1) complexity and the possibility to tune the model for user preferences. Also, the representation of the dictionary is much more compact and saves memory. This method requires however fine tuning during learning of the neural network and may be corpora dependent. The learning process is also long as the network has to be taught using distorted strings.

Other type of alternative approaches are n-gram based [13], [14], [15]. In this approaches, words from the dictionary and the word being verified are divided into n-grams (usually bi-grams). Later on, n-grams are matched, and the similarity measure is based on the number of successful matches. Such methods compensate the problems found in editing distance based measures by extending the strings representation. As such the imperfect string matching based on n-grams representation usually gives better results than standard distance metrics [16].

III. BI2QUADRO-GRAMS

The our method, proposed by Department of Computer Architecture at Faculty of Electronics, Telecommunication and Informatics, Gdańsk University of Technology, is an extension to widely used n-grams methods. In our approach, we use different string representation than usually used with n-grams. Our experiments show that using quadro-grams as a representation for target words (that is the dictionary word) and bi-grams for source words (the one being assessed) [4], where typical solutions in both cases use bi-grams, can increase the quality and performance of the algorithm. Example of dividing the string on bi/tri/quadro-grams is presented in Table III. To be able to map n-grams in different dimensions we add additional blank letters to the representation with bigger n. Then the bi-grams are mapped onto fragments of quadro-grams.

The final string similarity is related to the number of matched bi-grams from the source word. The distance between two strings is calculated as $distance = 1 - similarity$, where normalized similarity is calculated as in Formula 1 [4].

$$similarity = \frac{number\_of\_matched\_bi\text{-}grams}{total\_number\_of\_bi\text{-}grams} \quad (1)$$

TABLE I
EXAMPLE OF DIVIDING THE STRING ON BI/TRI/QUADRO-GRAMS.

string	n-grams						
keyboard	ke	ey	yb	bo	oa	ar	rd
keyboard	key	eyb	ybo	boa	oar	ard	rd_
keyboard	_key	keyb	eybo	yboa	boar	oard	ard_

TABLE II
WEIGHTS FOR DIFFERENT TYPES OF MATCHINGS.

Type of matching	Symbol	Weight
exact match	ab	1.0
shifted match	bc, da	0.8
commutative match	ac, db	0.8
reverse match	ba, cb, ad, ca, bd	0.1
adding additional letter	x	0.7

The n-grams matching algorithms needs to take into account different types of spelling errors that can be introduced to the text. During tests we identified the following types of matching:

- exact match: $ab \rightarrow abcd$.
- shifted match $ab \rightarrow acbd$.
- commutative match $ab \rightarrow cabd$.
- reverse match.: $ab \rightarrow bacd$.
- adding additional letter $ab \rightarrow abcx$

By differentiating between types of matching we can introduce weights to the similarity measure making it suitable for specialized task which is spelling errors correction. Without the weights the results can be misleading, e.g. the distance between „crdle" and „cradle" would be 0 which is obviously not true.

The weights' values were based on the ratio between a number of matches for different types of errors to the number of exact matches for 2281 incorrectly spelled words within Wikipedia [17] and further tweaked during the testing of the algorithm. The final values are presented in Table III.

With the introduction of weights the results of the algorithm improved. Taking weights into account gives more intuitive results. The example for a word with a removed letter is presented in Table III. In the case from the example the distance for quadro-grams can be calculated as $dist = 1 - (1 \cdot 1 + 1 \cdot 0.8 + 2 \cdot 0.8)/7 = 0.15$.

The modifications regarding matching types weights al-

TABLE III
LETTER REMOVAL – EXAMPLE INCLUDING WEIGHTS.

n-gram type	string	n-grams				matched n-grams
source word	crdle	cr	rd	dl	le	4
bi-grams	cradle	cr ab	ra -	ad -	dl -	1
tri-grams	cradle	cra ab	rad ac	adl bc	dle bc	4
quadro-grams	cradle	_cra ab	crad ac	radl bc	adle bc	4

TABLE IV
AVERAGE NUMBER OF WORDS PER SECOND THAT CAN BE CORRECTED BY EACH ALGORITHM

Metric	words/s	relation to the fastest (%)
bi2quadro-grams	98	54.02
Hamming	182	100.00
Levenshtein	68	37.62
Damerau-Levenshtein	30	16.49
n-grams	110	60.73

lowed the algorithm to recognize the misspelled words. Without the modifications, the n-grams approach allowed precise comparison of only identical words. After the modification the bi2quadro-grams algorithm remains language independent - the user only needs to change the corpora so it will contain words in given language. This is an important characteristic of all n-grams based methods.

IV. EVALUATION

A. Performance evaluation

To test the performance of the algorithm we did some synthetic tests using Wikipedia, Aspell and Trec corpora containing in total 21413 words [4]. During the tests, performed on standard Windows based PC, we measured the time needed to compare every source word with every word in the given dictionary, and the time needed to correct the source word. The results are presented in Table IV. The table shows the average number of corrected words per second for selected, widely used algorithms.

Our proposed algorithm is twice slower than the fastest algorithm, and three and a half times faster than the slowest one. Regarding performance, it achieves good results and from this point of view can be usable as even the slowest algorithm is widely used.

B. Quality evaluation

Preliminary tests were done during our previous research [4]. In this paper, we show that the solution can be used in every day use and compare it with other widely used solution - a spellchecker implemented in Microsoft Word 2013.

To gather data from different users and test the quality of the algorithm in real life scenarios, we implemented a dedicated application described in detail in the next subsection. The tool supports users in learning how to write without looking at the keyboard. During the process some statistics are gathered about the style, speed, and accuracy of writing. It also stores and compares the suggestions for misspelled words that bi2quadro-grams algorithm provided and compares them with the words that the users were supposed to write.

C. Test application

As test application, we developed a system for learning how to write text using a computer without looking on the keyboard. The application, written using .NET framework, is a set of games/tasks that are aim at helping to learn how to write properly. The application contains elements that should encourage users to use the application - it records the time taken to solve the task as well as the accuracy and speed of writing. Based on that parameters the user is awarded points and the application contains an achievement system that marks certain milestones.

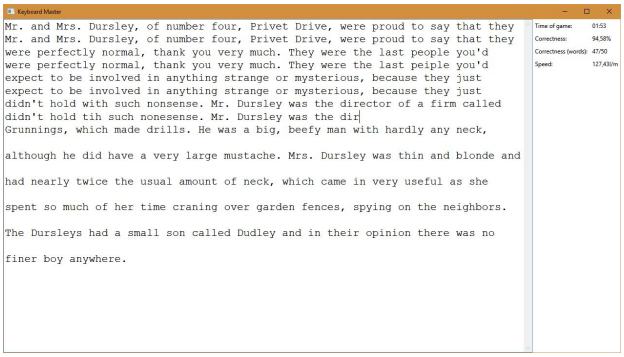

Fig. 1. The text game, the text comes from Harry Potter book

Fig. 2. The image game, the hidden image comes from desicomments.com

Currently, the application has two mini games implemented - text rewriting and image discovery. In the first game (the text game, Fig. 1) the player is tasked with rewriting the available text. Only the last entered letter can be changed. The player is rewarded for speed and accuracy. The second game (the picture game, Fig. 2) tasks the player with revealing the previously covered image. It shows random words to the player in a random section of the screen. To reveal the part of the image the player needs to rewrite the word without any errors - in this game the player cannot correct even a single character. Once again the game awards players based on their speed and accuracy.

It is worth noting that in the first game the text is selected from fragments of popular books. In this case, there is a risk, that given word will not be present in the corpora used by bi2quadro-grams algorithm. In the second game, the words that the player has to write are taken directly from the corpora so it is guaranteed to be found. As such in the first case we can test both the quality of the algorithm and the completeness of the corpora. In the second case, only the quality of the algorithm is being tested.

In both cases, the game can be paused by pressing escape button. As time is stopped, the screen is hidden with statistics about current play-through (time, speed in letters per minute, accuracy as the percentage of correctly entered letters and a relative number of correctly entered words). After finishing each of the games (either normally or by forcing the end from the pause menu), the user can view complete statistics and download a file containing all entered words with their correct forms and suggestions from the bi2quadro-grams algorithm (Fig. 3).

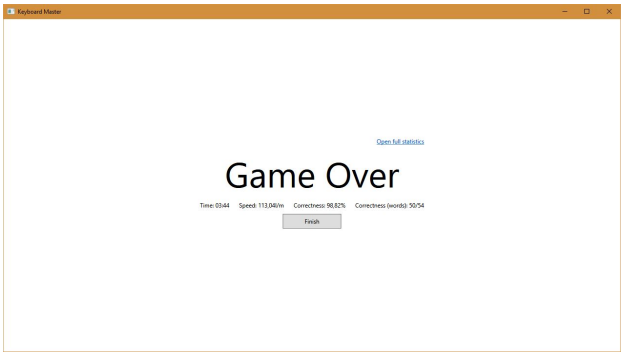

Fig. 3. The statistics screen

D. The results

The tests were performed in friends and family model - the designers of the tool asked their friends to work with the application to gather necessary data. 20 people, mainly students of the Department, responded. The results gathered during the tests were compared with the results obtained from Microsoft Word 2013. For that purpose, a simple console application has been developed. As input it took the statistics files generated by the program. For each file, two additional files were generated. The first file contained user entered word, correct word and all suggestions provided by both bi2quadro-gram algorithm and Microsoft Word (as in Table V). The second file contains the user entered word, correct word and true/false information if the bi2quadro-grams and Microsoft Word algorithm suggested a correct word (as in Table VI).

TABLE V
SUGGESTIONS FROM BI2QUADRO-GRAMS AND MICROSOFT WORD 2013

Input Word	Correct Word		
hysicotherapy	physiotherapy		
bi2quadro-grams	physiotherapy		
Microsoft Word	physiotherapy		
substitutuve	substitutive		
bi2quadro-grams	substitutive		
Microsoft Word	substitutive		
patriniser	patroniser		
bi2quadro-grams	patroniser		
Microsoft Word	patronize	patronized	patronizes

Results from the picture game, where the quality of the algorithm was tested are presented in Table VII. During the tests, the users entered 414 words. The bi2quadro-grams algorithm managed to suggest 402 (97,1%) correctly. Microsoft Word 2013 correctly suggested 363 words (87,7%). Out of the 414 words 306 were entered correctly. For the incorrectly entered 108 words, the bi2quadro-grams algorithm managed to correctly suggest 93 words (86,1%) whereas Microsoft Word 2013 correctly suggested 81 words (75%).

TABLE VI
RESULTS FROM BI2QUADRO-GRAMS AND MICROSOFT WORD 2013 ALGORITHMS

Input Word	Correct Word	Corrected by bi2quadro-grams	Corrected by Microsoft Word 2013
hysicotherapy	physiotherapy	True	True
substitutuve	substitutive	True	True
patriniser	patroniser	True	False

TABLE VII
RESULTS FOR WORDS FROM CORPORA (THE PICTURE GAME)

	bi2quadro-grams	%	Microsoft Word 2013	%
All words (414)	402	97,1	363	87,7
Miss-spelled words (108)	93	86,1	81	75

For the text game, we gather 897 words. The bi2quadro-grams algorithm correctly suggested 751 words (83,7%) and Microsoft Words 2013 correctly suggested 846 words (94,3%). In the set, only 81 words were incorrectly spelled of which bi2quadro-grams corrected only 36 words (44,4%). Microsoft Word 2013 corrected 60 words (74,1%). Results can be seen in Table VIII.

TABLE VIII
RESULTS FOR WORDS FROM BOOK FRAGMENTS (THE TEXT GAME)

	bi2quadro-grams	%	Microsoft Word 2013	%
All words (897)	751	83,7	846	94,3
Miss-spelled words (81)	36	44,4	60	74,1

In this game, we also checked false negatives, that is cases where given algorithm did not provide correct word when the user entered the word correctly (Table IX). In most cases, they were proper names, but in few cases also common names were problematic. For 816 words correctly entered by the user, the bi2quadro-grams algorithm in 101 cases (12,4%) did not suggest a correctly word. Microsoft Word 2013 did not suggest correctly entered word in 30 cases (3,7%). All the 30 words not suggested correctly were proper names and also were not suggested by the bi2quadro-grams. Remaining 71 words were not in the corpus used by the bi2quadro-grams algorithm.

Combined results for both games can be seen in Table X.

TABLE IX
FALSE NEGATIVES (THE TEXT GAME)

Algorithm	False negatives	%
bi2quadro-grams	101	12,4
Microsoft Word 2013	30	3,7

TABLE X
COMBINED RESULTS (THE TEXT AND THE PICTURE GAME)

	bi2quadro-grams	%	Microsoft Word 2013	%
All words (1311)	1153	87,9	1209	92,2
Miss-spelled words (189)	129	68,3	141	74,6

E. Results evaluation

In general the bi2quadro-grams algorithm behaves correctly and is able to match words if they are within the corpora. Its quality and efficiency is similar to other solutions. Our previous research [4] also shows that the performance of the algorithm is very good as it outperforms other widely used distance methods and allows faster dictionary lookup than Microsoft Word [4].

The main problem lies within the corpora size - the bi2quadro-grams algorithm did not find many even correctly entered words because they were missing from the corpora. Microsoft Word did not produce false negatives (except for the proper names). Further analysis of some of the cases found during the testing phase shows that the proper corpora can increase the quality of the results. Sometimes, however, bigger corpora can hinder the algorithm. For example in one of the tests, the player should enter the word "waterspout" (a rotating column of water and spray formed by a whirlwind occurring over the sea or another body of water). The player entered the word "watersport" which is wrong from the point of view of the game but is a correct word. Microsoft Word 2013, has that word in its corpora and as such did not point it as wrong. The bi2quadro-grams corpora did not contain the word "watersport". It found the nearest matching "waterspout". If the corpora would be bigger, than the bi2quadro-grams algorithm would misbehave just as the Microsoft Word 2013.

Our observation also shows that Microsoft Word 2013 usually gives more than one suggestion for given misspelled word. Bi2quadro-grams algorithm usually gave only one suggestion. More than one suggestion was given only in the cases where the source word could not be found in the corpora.

V. CONCLUSION

The proposed method of bi2quadro-grams achieves good results, usually not worse than other existing solutions in terms of quality and better in terms of performance. Bi2quadro-grams also usually produces only one word as a result whereas most of the widely-used spellcheckers present lengthy lists of words for the user to choose from. This allows the usage of the proposed metric in any word lookup and correction applications.

The tests done so far suggests that the direct key to improve the proposed algorithm as a spell checking solution is the creation of the proper and complete corpora. We plan on creating such corpora for both English and Polish language. Increased corpora size can influence a bit quality of the system, as mentioned in the previous section in regards to "waterspout" and "watersport" words. We will try to eliminate this issue in the future versions of the algorithm.

The test application presented here is the first stage of a more complex system that is currently developed at the Department of Computer Architecture. We plan on converting it into a web based multi user application. The users will have the option to comment on the results given by the algorithm and show its strengths and weaknesses allowing further improvement of its quality. We also plan on extending the application with a game focused on writing from hearing. In this case, the original text might not be known beforehand as the audio files should be selected at random from the Internet. In such case, the need arises for a fast and accurate algorithm that will be able to check user inputs during writing as we will lack the original text to compare with. The results obtained during our tests in this and previous research show that this can be achieved using bi2quadro-grams algorithm.

REFERENCES

[1] C. D. Manning, P. Raghavan, and H. Schtze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[2] T. Boiński and A. Chojnowski, "Towards facts extraction from text in Polish language," in *INnovations in Intelligent SysTems and Applications (INISTA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 13–17.

[3] J. Szymański and W. Duch, "Semantic memory knowledge acquisition through active dialogues," in *Neural Networks, 2007. IJCNN 2007. International Joint Conference on*. IEEE, 2007, pp. 536–541.

[4] J. Szymański and T. Boiński, "Improvement of Imperfect String Matching Based on Asymmetric n-Grams," in *Computational Collective Intelligence. Technologies and Applications*. Springer, 2013, pp. 306–315.

[5] R. Hamming, "Error detecting and error correcting codes," *Bell System technical journal*, vol. 29, no. 2, pp. 147–160, 1950.

[6] V. I. Lcvenshtcin, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet Physics-Doklady*, vol. 10, no. 8, 1966.

[7] C. Sulzberger, "Efficient Implementation of the Levenshtein-Algorithm," http://www.levenshtein.net/, 2009, [Online: 27.02.2018].

[8] F. J. Damerau, "A technique for computer detection and correction of spelling errors," *Commun. ACM*, vol. 7, pp. 171–176, March 1964. [Online]. Available: \url{http://doi.acm.org/10.1145/363958.363994}

[9] A. Boguszewski, J. Szymański, and K. Draszawka, "Towards increasing f-measure of approximate string matching in o (1) complexity," in *Computer Science and Information Systems (FedCSIS), 2016 Federated Conference on*. IEEE, 2016, pp. 527–532.

[10] S. L. Hantler, M. M. Laker, J. Lenchner, and D. Milch, "Methods and apparatus for performing spelling corrections using one or more variant hash tables," 2017, uS Patent 9,552,349.

[11] R. Udupa and S. Kumar, "Hashing-based approaches to spelling correction of personal names," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2010, pp. 1256–1265.

[12] K. Draszawka and J. Szymański, "Analysis of denoising autoencoder properties through misspelling correction task," in *Conference on Computational Collective Intelligence Technologies and Applications*. Springer, 2017, pp. 438–447.

[13] A. M. Robertson and P. Willett, "Applications of n-grams in textual information systems," *Journal of Documentation*, vol. 54, no. 1, pp. 48–67, 1998.

[14] P. Majumder, M. Mitra, and B. Chaudhuri, "N-gram: a language independent approach to ir and nlp," in *International conference on universal knowledge and language*, 2002.
[15] K. Atkinson, "GNU Aspell," http://aspell.net/, 2011, [Online: 28.02.2018].
[16] G. Navarro, R. Baeza-Yates, E. Sutinen, and J. Tarhio, "Indexing methods for approximate string matching," *IEEE Data Engineering Bulletin*, vol. 24, no. 4, pp. 19–27, 2001.
[17] Wikipedia, "Wikipedia:Lists of common misspellings," http://en.wikipedia.org/wiki/Wikipedia:List_of_common_misspellings, 2012, [Online: 29.03.2012].