Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 157 (2019) 183–190

Procedia Computer Science

www.elsevier.com/locate/procedia

4th International Conference on Computer Science and Computational Intelligence 2019 (ICCSCI), 12-13 September 2019

Quranic Latin Query Correction as a Search Suggestion

Wildhan Satriady[a,*], Moch Arif Bijaksana[a], Kemas M Lhaksmana[a]

[a]*School of Computing, Telkom University, Bandung, Indonesia 40257*

Abstract

This research proposes a method to help correct typo errors, such as wrong letters, missing letters, and additional letters. To date, the Lafzi search system is an example of an effective application for searching Arabic queries based on sound similarity. The Lafzi application helps correct typos due to the sound of Arabic letters that are almost the same as the pronunciation but not correct errors due to typos. Typos could prevent the search system not to display the desired results. This research proposes a solution by employing auto-complete to equipped missing trigram and the edit distance metric to calculate the differentiation value between the corrected query with the initial query. The way the system works is by separating and sorting trigram tokens from queries (user inputs) based on the verse. Each verse that has a missing trigram token will be equipped and re-transformed into a corrected query. Each corrected query will be compared to the edit distance value against the initial query (input from the user), then a corrected query will be taken which has the smallest edit distance value and will be made as a suggested query. The evaluation shows that the proposed method produces the highest recall value at 93.40% and the highest MAP value at 86%. This outperforms the previous Lafzi system approach which achieves recall at 85.23% and MAP at 79.83%.

Keywords: Edit Distance; Search Suggestion; Typo; Query Correction.

1. Introduction

Most Muslims in Indonesia are not used to writing Arabic so writing with Latin queries is needed. The search system called Lafzi [1] is one example of an effective application for searching Arabic queries based on sound similarity. The Lafzi application helps correct writing errors due to the sound of Arabic letters that are almost the same as the pronunciation but not correct errors due to typos. In the other side, Latin writing will usually cause writing errors or typos such as the ”Bisnillah” query which should be ”Bismillah” (In the name of Allah). Writing errors occur because the position of the letter ”N” on the QWERTY type keyboard is adjacent to the letter ”M”[2]. This writing error is usually ignored by the users and causes the results of the search to not be found. The system for generating

* Corresponding author. Tel.: +62-821-2911-0534.
E-mail address: wildhansat321@gmail.com

10.1016/j.procs.2019.08.156

suggestions is needed to correct errors in Latin queries so that the initial query can be corrected and give the desired search results.

In this research, the Query Correction system will correct errors from the queries entered by the user. The Query Correction system using auto-complete to equipped missing trigram and the edit distance metric to calculate the differentiation value between the suggested query with the initial query. The way the system works is to separate and sort trigram tokens from queries (user input) based on the verse. Each verse that has a missing trigram token will be equipped and re-transformed into a corrected query. Each corrected query will be compared to the edit distance value against the initial query (input from the user), then a corrected query will be taken which has the smallest edit distance value and will be made as a suggested query.

2. Related Work

2.1. Longest Contiguous Subsequence (LCS)

Longest Contiguous Subsequence (LCS) is an algorithm for finding a subsequence from a set of sequences that have a determined gap value between an item. This algorithm is almost the same with the Longest Increasing Subsequence (LIS) algorithm, in that LIS is a subsequence that has a value that monotonically increases[3]. The difference is the LIS algorithm doesn't need a gap between an item in sequences to determine the subsequence. The example of choosing a subsequence with LCS algorithm is shown in Table 1.

Table 1: Choosing longest contiguous subsequence with seven gap value

Sequence	Longest Contiguous Subsequence
[1,2,3,4,5,11,12,13]	[1,2,3,4,5]
[11,20,31,32,33,34]	[31,32,33,34]

2.2. Lafzi

The program accepts input of a piece verse in the form of Latin text such as "Bismillah" (In the name of Allah). After that, the input code will be converted into the "BISMILAH" (In the name of Allah) phonetic code and then it will be tokenized as [BIS, ISM, SMI, MIL, ILA, LAH]. Each trigram that has been typed in will be searched for each inverted index[4] in the existing database as in Table 2.

Table 2: Inverted index for every trigram

Trigram	Postings
BIS	{"1":[1], "27":[101], "30":[46], "49":[8], "52":[10], ...}
ISM	{"1":[2], "92":[94], "132":[96], "134":[44], ...}

Once found in the database, the verse data will be stored in a variable that holds a verse along with the order in which the trigrams appear in the verse. This command will continue to be executed until all of the typed trigrams are searched in the database, and entered into a variable that holds the verse along with the order in which the trigrams appear on the database. After getting several candidates who have been accommodated in the variable, the candidate will be calculated with LCS method.

From these results it was found that there were five trigrams that were suitable in paragraphs one and four of the trigrams that matched verse 21. Next, the difference between the order of occurrence of the trigrams and the inverse function in Equation 1 will be calculated[1].

$$c = \frac{1}{n-1} \sum_{i=1}^{n-1} \frac{1}{s_{i+1} - s_i} \tag{1}$$

As in the Equation 1, n is count of matched subsequence, s_i is current subsequence in a looping and c is the result of the inverse function. The result is between [0..1]. After that to determine the score of each variable value is the result of the inverse value with the number of trigrams that match as in the Equation 2.

$$score = c \cdot countLCS \tag{2}$$

2.3. Arabic User Search Query Correction and Expansion

This research describes the correction and expansion technique of a multilingual search query submitted to the Arabic centered search engine.The query expansion mechanisms also use Arabic words roots and thesauri built automatically by a categorization engine, in order to achieve better recall[5]. This research provides the search system with Arabic form query input and correction.

3. Query Correction System

3.1. System Overview

This research will produce a system that can make a correction to a query that has a typo. Most people do not understand about the structure and writing of Arabic vocabulary so it is necessary to tolerate these errors[6]. By using trigrams we can tolerate if there are queries that have only half of the Arabic words. If the query has a typo, the query correction system needs to adjust it to make a relevance result between the typo query. To find out that a query has typo it can be known by two conditions, namely: the search results are not found and the highest score of the ranking results in the search system is below 0.96. This number is obtained from the results of experiments on the test set that has been made. From a few queries that have a typo, the highest ranking score is 0.96 namely only one character typo. There are several main processes in this system of query correction. First, is get a matched trigram from the previous searching, second, re-ranking the matched trigram based on the highest count of LCS, third, completing a missing sequence of a trigram and the result as corrected query, and ranking the corrected query based using edit distance metric between the input query. The system architecture can be seen in Figure 1.

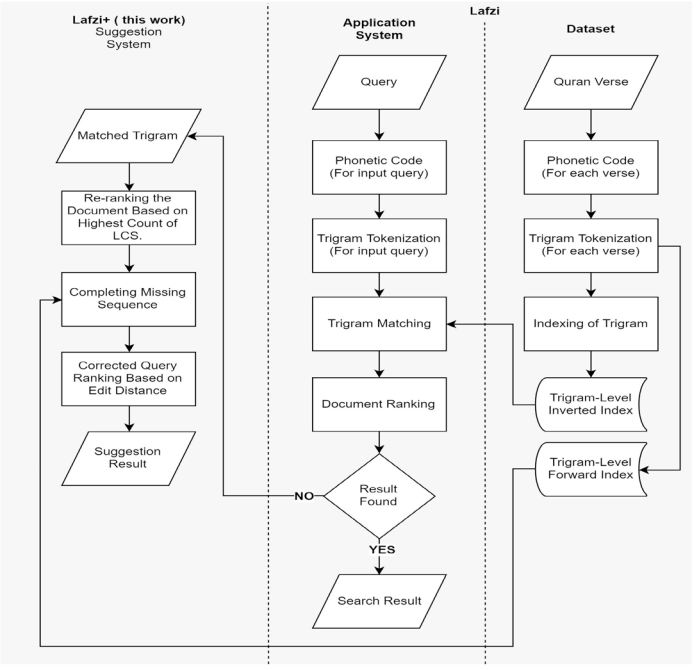

Fig. 1: Flow chart Lafzi and Lafzi+

3.2. Dataset

Lafzi Indexer is a program to generate a dataset that will be used for the search system and query correction system. All of the Quran verses are transform into Latin and converted to phonetic code and it will be converted into an inverted index. The dataset consists of the whole Quran as well as the translation, inverted index data trigram vocal and non-vocal[1]. For this research, we also generate a Forward Index as a reference document to help auto-complete methods. The format of Forward Index is shown as in Table 3. The trigram column is an array of a collection of trigrams in a specific verse.

Table 3: Forward Index for helping auto-complete method is the tokenization results for each verse taken from the Latin database query.

Verses	Trigram
1	[BIS, ISM, SMI, MIL, ILA, LAH,..., HIM]
21	[WAX, AXI, XIZ, IZA, ZAL,...., XUN]

3.3. Matched Trigram Data and Re-ranking Document

The query correction system accepts a document that has a matched trigram token from the user input. The document was selected as many as five documents based on the highest number of matched trigram tokens and have a sequential gap between the other index is a maximum of four trigram tokens. The documents are illustrated as in Table 4.

Table 4: The example of a document candidate to be processed in the auto-complete algorithm.

Verse	Sequence	LCS (Longest Contiguous Subsequence)	Count LCS
1	[1,5,6,11,13]	[1,5,6]	3
21	[11,20,31,34]	[31,34]	2

3.4. auto-complete Missing Sequence

To complete the missing sequence index, we can equip the missing trigrams but, the result of the query correction system can be irrelevance. The typos may occur in the middle, beginning or at the end of the query[2,7,8]. Moreover, errors such as deletion and insertion character to the query also need to be tolerated[9]. we don't know where is the wrong, missing, and additional character. So, we collect all the possibilities and make that as an algorithm. The algorithm for auto-completing missing sequences will be done in several options. Every option generates one corrected query candidate to be compared with the other corrected query. The algorithm options showed as follows:

- Additions in the middle of the sequence (Add mid).
- Additions in the middle and right side (Add mid and right).
- Additions in the middle and left side (Add mid and left).
- Additions in the middle, left side, and right side (Fair ADD).
- Additions in the middle, right and reduction on the left side (Add mid,right decrease left).
- Additions in the middle, left and reduction on the right side (Add mid,left decrease right).
- Reduction on the left side and addition in the middle (Add mid decrease left).
- Reduction on the right side and addition in the middle (Add mid decrease right).
- Reduction on the right, left, and additions in the middle (Fair decrease).

Table 4 shows some document which has an imperfect trigram index sequence. The typos query makes the imperfect trigram sequence because there is some token which is not matched or missing in the current verse. The several candidates will be processed with the auto-complete method.

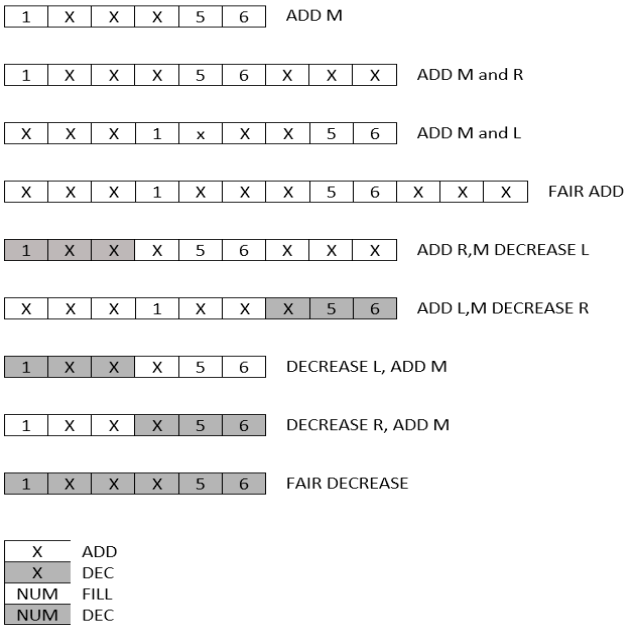

Fig. 2: The options for auto completing the missing trigram sequence. Add is for adding the missing sequence of a trigram, dec is to remove the sequence, and fill is for filling the trigram in a specific index of sequence in trigram.

These methods are illustrated in Figure 2. For the addition and reduction options on the left and right side, the reduction or addition will be done as long as the index on the left side is not less than zero and the index on the right side is no more than the maximum index in the database[10]. Moreover, the number of additions and reductions is the difference between the number of trigrams tokens in the user input and the number of the matched trigram. The difference value is obtained by Equation 3.

$$diff = |CountInputNgram - CountMatchedNgram| \tag{3}$$

The addition of indexes on the left and right side is also done gradually, while the reduction of the index on the left or right side is done by gradually too[11]. There is an exception for the Fair Add option. In Fair Add option we just fill all the index with trigram from Forward Index, because the addition and reduction are represented in other options.

To complete the incomplete trigram, the trigram is taken from the Forward Index data based on the index obtained from the options above. The results of the equiped trigram will be re-transformed to a corrected query. For example [BIS, ISM, SMI, MIL, ILA, LAH] become ”BISMILAH” (In the name of Allah). The result of the re-transformed trigram into a query will be a corrected query candidate and stored in a variable.

3.5. Ranking Using Edit Distance Metric

Each candidate suggestion will be calculated by the edit distance algorithm using Levenshtein Distance and it will be compared to the initial query[12]. Finally, the candidate which has the lowest edit distance value will be shown as a suggestion. Every wrong, missing, and additional character has a value one[7]. We also use different weights for adjacent characters. We use 0.5 weight for every adjacent character such as ”M” substitute with ”N” in QWERTY keyboard. The purpose is, to give more toleration for substitution typo and for missing and additional character the weight is still one. In this research, we will compare the result between weighted and non-weighted edit distance value.

4. Evaluation

4.1. Test Set

To evaluate this system, the author uses recall and MAP (Mean Average Precision). Lafzi+ (this work) is the system that has been changed and added the query correction system. To ensure the performance of a search system based on the sound similarity between the Lafzi and Lafzi+ systems, the author re-examined the test set from previous Lafzi research to this research [1]. This test set called Normal Query Set. The format of the testing dataset looks like in Table 5. Another test set is made by taking 100 queries randomly from several verses in the Quran that have been converted

Table 5: The example of normal query test set

No	Arabic Text	Latin Text	Appearance in Quran (surah:verse)
1	أُولُوا ٱلْأَلْبَـٰب (Those understanding)	ULUL ALBAAB	[2:269, 3:7, 13:19, 14:52, 38:29, 39:9, 39:18]
2	ٱلْحَمْدُ لِلَّهِ (All praises and thanks to Allah)	ALHAMDULILLAH	[34:1, 6:1, 14:39, 18:1, 1:2, 35:1, 23:28, 29:63, 31:25, 27:59, 39:75, 27:15, 40:65, 6:45, 16:75, 37:182, 39:29, 35:34, 10:10]

into Latin letters. This test set purpose is to measure the query correction system performance. There's two scheme of this test set. First is to test the query correction system for typos query in term of substitutes character, called Typo Query Set (Subtitution), and the second one is for testing the typos query in term of insertion and deletion character in the query, called Typo Query Set (Insertion and Deletion). The example of the testset are shown in Table 6.

Table 6: The example of typo query test set

No	Arabic Text	Typo Latin Text	Appearance in Quran (surah:verse)	Type of Typo
1	ٱلْحَمْدُ لِلَّه (All praises and thanks to Allah)	AL*J*AMDULILAH	[1:2,34:1...]	Subtitution typo
2	أَلَّا تَطْغَوْا فِى ٱلْمِيزَانِ (That you not transgress within the balance.)	ALATATGA FILMIZAN	[55:8]	Deletion typo
3	فِى سِدْرٍ مَّخْضُودٍ ([They will be] among lote trees with thorns removed)	FISIDRIMA*D*HDUD	[56:28]	Insertion typo

4.2. Experiment Result

In Table 7, the highest value from recall and MAP has been obtained by using a combination of deletion and insertion typos query. The combination of that method produces a relevant corrected query. By testing fifty typo query sets (substitution) and fifty typo query sets (Addition and deletion), as many as 81 queries were successfully corrected and produce a recall value one, seven queries produce a recall value greater than zero, and twelve queries produce a

Table 7: Evaluation's Result

	Lafzi+ (this work)		Lafzi	
	Recall	MAP	Recall	MAP
Normal Query	16.21%	77.60%	16.21%	77.60%
Typo Query (Subtitution) + Weighted Edit Distance Metric	77.60%	76.00%	30.21%	19.67%
Typo Query (Subtitution) + Non-Weighted Edit Distance Metric	77.63%	76.00%	30.21%	19.67%
Typo Query (Insertion and Deletion) + Weighted Edit Distance Metric	93.22%	86.00%	85.23%	79.83%
Typo Query (Insertion and Deletion) + Non-Weighted Edit Distance Metric	**93.40%**	**86.00%**	85.23%	79.83%

zero recall value. From the test results, the system in this research produces the highest recall value of 93.40% and the highest MAP value is 86%. This is better than the previous Lafzi system approach, namely recall at 85.23% and MAP at 79.83%.

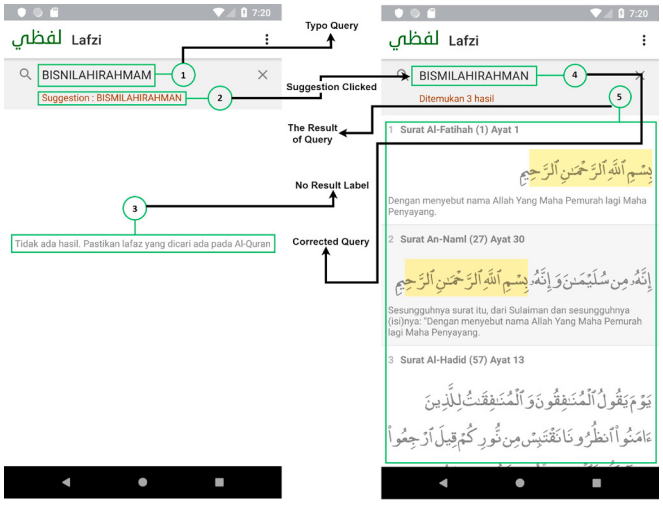

Fig. 3: The example of typo query result

The result of the suggestion application system is shown in Figure 3. In this case the user input query "BISNILAHIRAHMAM" (typo) there is any typo in the query. The Lafzi search system not shown the desired result. The query correction system gives a recommendation/correction query. The correction is shown at point 2 in Figure 3. The correction would be "BISMILAHIRAHMAN" (In the name of Allah the Most Gracious) by the query correction system as shown in point 4 in Figure 3.

4.3. Discussion

The experiment result showed that the query correction method with Non-Weighted Edit Distance is better. In this case, it happens because of the Non-Weighted Edit Distance may produce a lower score to the same size of count character corrected query between query input or the corrected query have different size but the half of corrected query is the same with typo query. For example, the typo query "alhandu" and the gold standard query is "alhamdu" will be calculated with the corrected query "balhamdul" and "algandu". With the Weighted Edit Distance, "algandu" will be better than "balhamdul" but with the Non-Weighted Edit Distance, "balhamdul" will be have same result. The system will choose the first lowest founded corrected query. If we search "balhamdul" the result will be more relevant because we have more characters that intersect with input queries and we use the trigram as a dataset. The query

correction system is quite effective to resolve the typo query. It's proven by the recall and MAP metric result in Table 7. The result for insertion and deletion query set shows a better result than the substitution query set. It happened because due to the trigram piece of the verse to be processed have a trigram piece which is commonly found in other verses. Besides that, it can make the suggestion irrelevance query as shows as in Table 8.

Table 8: The example of irrelevant output suggestion

Gold Standar	Typo	Suggestion
HATTAYANFADU (Until they disband)	HARAYAMFADU	ARATUFADU

5. Conclusion

Based on the results of outcomes achieved from the analysis of this research data, can be drawn a conclusion as follows. The typos query makes the sequence of trigram have a missing index and a lot of gap between the other index. That's why the previous Lafzi approach cannot bring up the search result. With the auto-complete method and calculate Edit Distance the corrected query between the input query that not explored in Lafzi research, this combination can resolve the typo problem. By selecting a candidate which has the highest count of trigrams in a verse using LCS, then the auto-complete method will generate several sets of trigrams and be re-transformed into a corrected query and then the corrected query will be ranked based on the edit distance value of the input query. The test results of the proposed method can correct typos in queries that are inputted by the user. From the test results, the system in this research produces the highest recall value of 93.40% and the highest MAP value is 86%. This is better than the previous Lafzi system approach, namely recall 85.23% and MAP 79.83%. The result of this research can be added in Lafzi applications system to improve the result of the search system.

References

1 M. A. Istiadi, Sistem Pencarian Ayat Al-quran Berbasis Kemiripan Fonetis, Bachelor's Thesis, Institut Pertanian Bogor (2012).
2 C. Senger, J. Kaltschmidt, S. P. Schmitt, M. G. Pruszydlo, W. E. Haefeli, Misspellings in drug information system queries: characteristics of drug name spelling errors and strategies for their prevention, International journal of medical informatics 79 (12) (2010) 832–839.
3 D. Liben-Nowell, E. Vee, A. Zhu, Finding longest increasing and common subsequences in streaming data, Journal of Combinatorial Optimization 11 (2) (2006) 155–175.
4 P. R. Christopher D. Manning, H. Schütze, Introduction to Information Retrieval, (Stanford University, Yahoo! Research, and University of Stuttgart) Cambridge: Cambridge University Press, 2008, xxi+ 482 pp; hardbound, ISBN 978-0-521-86571-5, 2009.
5 T. Rachidi, M. Bouzoubaa, L. ElMortaji, B. Boussouab, A. Bensaid, Arabic user search query correction and expansion, Proc. of COPSTIC 3 (2003) 11–13.
6 N. S. Hidayat, Problematika pembelajaran bahasa arab, An-Nida' 37 (1) (2012) 82–88.
7 V. I. Levenshtein, Binary codes capable of correcting deletions, insertions, and reversals, in: Soviet physics doklady, Vol. 10, 1966, pp. 707–710.
8 E. Tanaka, T. Kasai, Synchronization and substitution error-correcting codes for the levenshtein metric, IEEE Transactions on Information Theory 22 (2) (1976) 156–162. doi:10.1109/TIT.1976.1055532.
9 Y. Liang, K. Iwano, K. Shinoda, An efficient error correction interface for speech recognition on mobile touchscreen devices, in: 2014 IEEE Spoken Language Technology Workshop (SLT), IEEE, 2014, pp. 454–459.
10 P. Ferragina, R. Venturini, The compressed permuterm index, ACM Transactions on Algorithms (TALG) 7 (1) (2010) 10.
11 C. Schensted, Longest increasing and decreasing subsequences, Canadian Journal of Mathematics 13 (1961) 179–191.
12 N.-L. Tsao, D. Wible, A method for unsupervised broad-coverage lexical error detection and correction, in: Proceedings of the Fourth Workshop on Innovative Use of NLP for Building Educational Applications, Association for Computational Linguistics, 2009, pp. 51–54.