# A Non-Word Error Spell Checker for Patient Complaints in Bahasa Indonesia

Chanifah Indah Ratnasari[1], Sri Kusumadewi[2], Linda Rosita[3]
[1,2]Department of Informatics, [3]Medical Education
Universitas Islam Indonesia
Yogyakarta, Indonesia
[1]chanifah.indah@uii.ac.id, [2]cicie@uii.ac.id, [3]linda.rosita@uii.ac.id

***Abstract*—A Non-word error is misspelled word that are caused by a typographical error. Moreover, the text in the form of free-text or narrative text is likely to contain typographical error. Likewise with patient complaints data. The data are stored in the form of free-text data or a medical narrative by the doctor when taking the medical history or conducting the medical interview. Based on data on patient complaints obtained from physicians, this study develops lexicon resource which is used as spell detection, and then using the Levenshtein distance for spell correction. Spell checker that was built in this study has 4 main components, namely pre-process, error detection, error correction, and user feedback. Pre-process stage was performed to generate the required texts for error detection. Error detection checks there is any spelling error or not. Error correction is looking for candidate word as proposed solution to correct the misspelled word. User feedback to ensure to the user whether a proposed word is the word that is intended or not. Using 669 words as the lexicon resource and 224 words as the tested words. The experimental results achieved 97.59% accuracy of error detection and 94.03% accuracy of error correction. The errors are caused by the word is not found in the lexicon resource have distance that exceeds the threshold value and there is minimum similarity value more than one for a misspelled word.**

***Keywords—non-word error; spell checker; Indonesian spell checker; Levenshtein distance; spell checker in Bahasa Indonesia***

## I. INTRODUCTION

Natural language is the language commonly used by humans to communicate. Artificial language is a language used for specific purposes, one of them is computer programming language [1]. In order to make computer can understand natural language, an approach is required to bridge the differences between the languages. That approach is by using Natural Language Processing (NLP) [2].

To perform text-based natural language processing, required text that free from spelling mistakes. Spell checking is important in producing an error-free text. Spell checking is the process of checking the spelling of a word to detect mispelled word and give the candidate of correct words [3]. Spell checking is needed, especially to handle the text in the form of a free-text or a narrative text that allows typographical errors.

In Indonesia, patient complaints that expressed when the patient checks to the doctor - which is part of anamnesis - are stored in Electronic Medical Records (EMR) [4]. Anamnesis, also called as history taking or medical interview, is a term used for the collection of information about past events and current conditions of patient by doctor for the purpose of medical treatment [5]. Anamnesis is the initial stage of a series of patient examination aims to obtain thorough information of the patient in question, which can be used to infer the alleged impaired organ/system, or for make a diagnosis [6].

Recording of anamnesis in EMR using natural language in the form of textual description, medical narrative, or free-text. It is very possible existence of spelling errors or typographical errors. Therefore, it is necessary to have a spelling checker to overcome it.

The rest of this paper is organized as follows. Section II describes about spell checker. Section III describes the data used in this study. Section IV describes the approach used in this study. Section V describes the results of this study. Finally, the conclusions of this work are provided in Section VI.

## II. SPELL CHECKER

Spell checker is an application that can detect and solve a writing error, and then, if necessary, provide suggestions to fix the error [7]. Spell checker is divided into two types, namely non-word error spell checker and real-word error spell checker [3]. Non-word error spell checker focuses on handling misspelled words that are caused by a typographical error. While real-word error spell checker emphasis on handling the placement errors of the word in the sentence.

In non-word error, a word may incorrectly typed because there is extra space, extra character, misspelled word, or other possibilities. In real-word error, the cause is an error in recognizing the grammar of word in sentences. This is one of the problems faced Natural Language Processing (NLP), including ambiguity and out of vocabulary (OOV) [7].

## III. DATA

Language of patient complaints different from the language used in Bahasa Indonesia in general. Patient complaints are more likely on the language used everyday (non-standard language). Also because of location of the data collection in this study is in Yogyakarta, the language of patient complaints used is Bahasa Indonesia mixed with Javanese language, which is the local language used in that area.

The data used in this study were obtained from doctors. There are 926 sentences of patient complaints in which there are 669 unique tokens (there is no similar token between one another).

## IV. Our Approach

Spell checker in this study is one of the preprocessing stages of the research that has been done previously, which is published in [8]. The proposed spell checker that was built has 4 main components as in Fig. 1, namely pre-process, error detection, error correction, and user feedback.

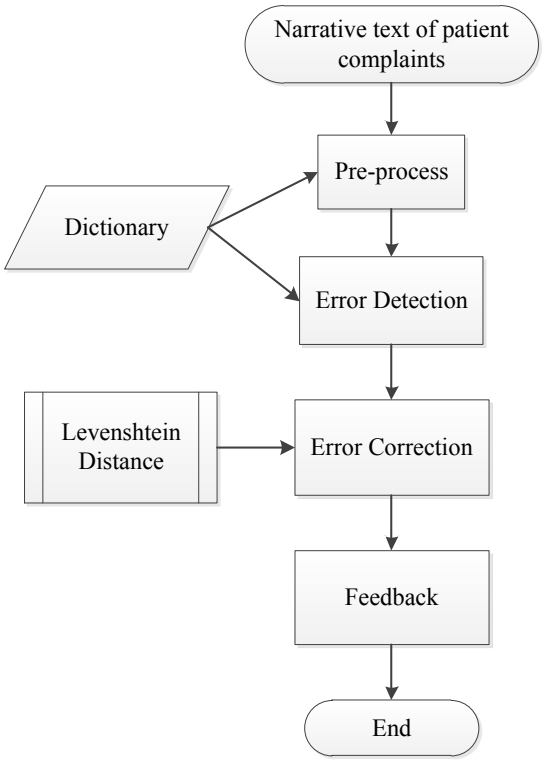


Fig. 1. Spell checker components

### A. Pre-process

Pre-process is the first phase which is done before error checking. This phase includes conversion narrative text to lowercase, sentence detection, tokenization, and identification of abbreviations.

- Conversion narrative text to lowercase. The Input in the form narrative text of patient complaints is converted to lowercase letters to make it easier later in the further process.
- Sentence detection. Narrative texts of patient complaints that have been converted into lowercase letters, split into separate sentences. This is done because it is possible there are more than one sentence in the narrative text. The splitting is based on a dot (.) character or new line.
- Tokenization. Sentences that have been obtained from the previous stage, split into tokens (tokenization). Tokenization is the process of splitting a set of characters into tokens as basic units before further processing [9]. Approach for this tokenization is by using a space character or punctuation as a separator. In this phase also performed some special handling in breakdown into tokens, which are the handling of units of time, repeated word, negation, punctuation should be considered as a token, and handling of numbers.
- Identification of abbreviations. Tokens that have been obtained, further to identify the abbreviations, namely the process of identifying the presence or absence the abbreviations of tokens, considering recording of patient complaint anamnesis is inseparable from the use of non-standard abbreviations. For example, the word "sll" for "selalu". The approach used to identify abbreviations is using a lexicon / dictionary.

### B. Error Detection

Error detection checks on each word obtained from the previous stage, whether there is a spelling error or not. In this stage, the method used is a dictionary lookup method. Dictionary is lexicon resource that was built from data obtained from doctors. Each token is searched on the dictionary. A word that is not found in the dictionary is regarded as a misspelled word, so it will do error correction.

### C. Error Correction

Error correction handles word that considered as misspelled word. Error correction is looking for candidate word as proposed solution to correct the misspelled word. The method used is the Levenshtein distance.

The Levenshtein distance is used to measure the similarity between two words. This method is looking for candidate word to be proposed based on the minimum number of characters that have to be replaced, inserted, or deleted to transform the word from *string1* into *string2*. In this case, *string1* is misspelled word and *string2* is the word contained in the dictionary. The complexity of the algorithm is $O(m*n)$, where $m$ and $n$ are the length of *string1* and *string2*.

For example, the following is an example of the transformation of *string1* into *string2*. In the example used, it is assumed M is a misspelled word and D is a word in the dictionary.

- Character replacement

  For example M = nyuri and D = nyeri. String M is transformed into D by the character replacement at the third position, that is the character 'u' in M is replaced with 'e' in D.

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| *M* = | *n* | *y* | *u* | *r* | *i* |
| *D* = | *n* | *y* | *e* | *r* | *i* |

- Characters insertion

  For example M = plek and D = pilek. The character insertion is done by inserting the character "i" at the second position in M shown as follows.

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| *M =* | *p* | - | *l* | *e* | *k* |
| *D =* | *p* | *i* | *l* | *e* | *k* |

- Character deletion

  For example M = gatalk and D = gatal. Deletion is done for the character "k" at the sixth postion in M. The deletion shows the transformation from M to D that are illustrated as follows.

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| *M =* | *g* | *a* | *t* | *a* | *l* | *k* |
| *D =* | *g* | *a* | *t* | *a* | *l* | - |

The Levenshtein distance produces a similarity score between misspelled word with every word in the dictionary. The word with the lowest score is deemed the best match. The formula used to calculate Levenshtein distance [10] is shown in (1).

$$f(i, j) = min[(f(i - 1, j) + 1, f(i, j - 1) + 1, f(i - 1, j - 1) + d(q_i , l_j))] \quad (1)$$

where $d(q_i , l_j) = 0$ if $q_i = l_j$
else $d(q_i , l_j) = 1$

A function $f(i, j)$ is calculated for all misspelled word letters and all dictionary-word letters, iteratively counting the string difference between the misspelled word $q_1, q_2, \ldots, q_i$ and the dictionary word $l_1, l_2, \ldots, l_j$. Each insertion, deletion, or substitution is awarded a score of 1.

In this study, the threshold was used as the maximum value of the Levenshtein distance result. It aims to the proposed word has a distance that is not too far, so it can give the correct of proposed word.

In this study did not use a result ranking of Levenshtein distance. Therefore if there are minimum similarity value more than one, then the value of minimum similarity that found the earliest proposed as a candidate word.

### *D. Feedback*

In this study, the system will propose the correct word of the misspelled word to the user. It aims to ensure to the user whether a proposed word is the word that is intended or not.

## V. RESULT

In this section, we describe the experiments that we conducted to evaluate the performance of the spelling checker.

### *A. Method*

The experiment consists of 55 patient complaints sentences which contain misspelled words. The sentences consist of 224 words. Each misspelled word is categorized into deletion error, insertion error, and replacement error.

Performed calculating the value of accuracy in spell checker testing results. We define accuracy as the proportion of correctly error detection and error correction.

### *B. Result*

The result of our experiment is shown on Table I. There are 4 mistakes found in misspelled words detection and 11 mistakes in error correction. The accuracy of error detection is 97.59% and the accuracy of error correction is 94.03%.

### *C. Analysis*

Based on the experiment, we observe two cases that cause of incorrect detection and incorrect correction. First, words in input sentence which are not exist in dictionary have distance that exceeds the threshold value. This causes the word in question is considered not to have spelling error. Second, if there are minimum similarity value more than one for a misspelled word, then the value of the minimum similarity that found the earliest will be proposed as a candidate word. This may cause mistake in the error correction.

The proposed solution by authors to address these problems are as follows.

*1) Enrich data which stored in the dictionary:* It is expected by the increasing number of data in dictionary, increasing the likelihood of misspelled word distance value does not exceeds the threshold value.

*2) Build a result ranking of Levenshtein distance:* It aims to sort the list of candidate words by its relevance to the correct word.

*3) The entire word that has minimum similarity value is offered to user:* This is the continuation of the proposed solution in number 2. If there are minimum similarity value more than one, then the whole word that has the minimum value is offered to user. Therefore, the user can choose the most appropriate of proposed words.

## VI. CONCLUSION

This study provide solution to non-word error, which is implemented for the case in the medical field, namely to handles patient complaints text. Dictionary lookup as an error detection method has accuracy 97.59%. The Levenshtein distance as an error correction method has accuracy 94.03%. Dictionary has a very important role. The more words stored in the dictionary, it can make the output system more accurate. Future research will conducted to enrich the data that are stored in the dictionary and build a result ranking of Levenshtein distance in order to produce the correct proposed word.

TABLE I. EXPERIMENT RESULT

| Types of Error | Number of Sentences | Number of Words | Error Detected | False Detection | Error Corrected | False Correction | Accuracy | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | Detection | Correction |
| Insertion | 24 | 111 | 24 | 0 | 24 | 1 | 100% | 99.10% |
| Replacement | 19 | 65 | 21 | 2 | 21 | 7 | 96.92% | 89.23% |
| Deletion | 12 | 48 | 14 | 2 | 14 | 3 | 95.83% | 93.75% |
| Total or Average | 55 | 224 | 59 | 4 | 59 | 11 | 97.59% | 94.03% |
| | Total | | | | | | Average | |

## REFERENCES

[1] I. K. E. Purnama and A. Zaini, "Pengembangan Agent Antarmua Cerdas Berbasis Bahasa Alami untuk Bahasa Indonesia yang Diterapkan Pada Game Edukasi Kecakapan Hidup (Life Skill)," ITS Digital Repository, 2009.

[2] G. Chowdhury, "Natural Language Processing," *Annu. Rev. Inf. Sci. Technol.*, vol. 37, no. 1, pp. 51–89, 2003.

[3] M. Y. Soleh and A. Purwarianti, "A Non Word Error Spell Checker for Indonesian using Morphologically Analyzer and HMM," in *International Conference on Electrical Engineering and Informatics*, 2011, pp. 1–6.

[4] PERMENKES, *Peraturan Menteri Kesehatan Republik Indonesia Nomor 269/MENKES/PER/III/2008*. Indonesia, 2008.

[5] J. R. Moehr and Hannover, "Computer Assisted Medical History," in *Informatics and Medicine*, 3rd ed., vol. 3, P. L. Reichertz and G. Goos, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1977, pp. 460–578.

[6] H. M. S. Markum and D. Widodo, *Penuntun Anamnesis dan Pemeriksaan Fisis*. Pusat Penerbitan Departemen Ilmu Penyakit Dalam Fakultas Kedokteran Universitas Indonesia, 2000.

[7] R. N. Aqsath, M. Kamayani, R. Reinanda, S. Simbolon, M. Y. Soleh, and A. Purwarianti, "Application of document spelling checker for Bahasa Indonesia," in *International Conference on Advanced Computer Science and Information System (ICACSIS)*, 2011, pp. 249–252.

[8] C. I. Ratnasari, S. Kusumadewi, and L. Rosita, "Natural Language Parsing of Patient Complaints in Indonesian Language," in *The International Conference on Science and Technology (TICST)*, 2015, pp. 292–297.

[9] J. Asian, "Effective Techniques for Indonesian Text Retrieval," RMIT University, 2007.

[10] V. J. Hodge and J. Austin, "A Comparison of Standard Spell Checking Algorithms and A Novel Binary Neural Approach," *IEEE Trans. Knowl. Data Eng.*, vol. 15, no. 5, pp. 1073–1081, Sep. 2003.