Detection and Correction of Real-Word Errors in Bangla Language

Conference Paper · September 2018

DOI: 10.1109/ICBSLP.2018.8554502

CITATION
1

READS
586

6 authors, including:

Mashod Rana
Arollo Tech Ltd.
3 PUBLICATIONS **1** CITATION
SEE PROFILE

Mohammad Tipu Sultan
The University of Asia Pacific
3 PUBLICATIONS **1** CITATION
SEE PROFILE

M. Firoz Mridha Ph. D.
Bangladesh University of Business and Technology (BUBT)
58 PUBLICATIONS **107** CITATIONS
SEE PROFILE

Eyaseen Arafat
The University of Asia Pacific
3 PUBLICATIONS **1** CITATION
SEE PROFILE

Detection and Correction of Real-word Errors in Bangla Language

Md. Mashod Rana
Department of Computer Science and Engineering
University of Asia Pacific
Dhaka, Bangladesh
mashod0rana@gmail.com

Mohammad Tipu Sultan
Department of Computer Science and Engineering
University of Asia Pacific
Dhaka, Bangladesh
tipu07u5@gmail.com

M. F. Mridha
Department of Computer Science and Engineering
University of Asia Pacific
Dhaka, Bangladesh
firoz@uap-bd.edu

Md. Eyaseen Arafat Khan
Department of Computer Science and Engineering
University of Asia Pacific
Dhaka, Bangladesh
eyaseenarafatkhan08@gmail.com

Md. Masud Ahmed
Department of Computer Science and Engineering
University of Asia Pacific
Dhaka, Bangladesh
mdmasudrana81uap@gmail.com

Md. Abdul Hamid, *Member, IEEE*
Department of Computer Science and Engineering
University of Asia Pacific
Dhaka, Bangladesh
ahamid@uap-bd.edu

***Abstract*—Detection of spelling error is not so facile in Bangla. To check for real-world error in a sentence, it comes with more difficulties. In this paper, we focus on correcting homophone error in real-word error. We use N-gram Model which is used in many purposes like machine translation, speech recognition, to extract syntactic information etc. We have used a combination of Bi-gram and Tri-gram with candidate word which is going to be detected whether it is a real-word error or not. We have developed corpora which contain: (i) one of them is a collection of sets of homophone (confusing) word, (ii) another two are the collection of bigrams and trigrams using homophone word and (iii) other seven are the test sets. A candidate word extracts the set of homophone words from the corpus. In our proposed method, we create tri-gram and bi-gram using homophone word, then it checks the validity and takes the frequency of bi-gram or tri-gram, and finally calculates the probability for making the final decision about the candidate word. We have used around a million words to inspect our system. Our proposed method achieves more than 96% accuracy in detecting and correcting real-word errors of Bangla Text.**

Keywords—Bangla homophones, NLP, Real-word error, N-gram, Markov model.

I. INTRODUCTION

People communicate with each other through languages. It is almost like all other animals; we are also involved in communicating through verbal, sign or textual representation to express our views to other people.

Textual representation is the most emergent way of communication through which people can express their desire to other people. We may get inspiration of textual representation by Newspaper, Diary, Manuals, Books, Novel, Publications etc. By the textual representation of language, we can keep and sustain the information via media and evolution of the legal system. Bangla is the primary language in Bangladesh and second most spoken language in India. Bangla is one of the most widely spoken language with around 250 million people.

Bangla language comes from Indo-Aryan, Indo-European languages. Bangla language is one of the most critical languages in the world. In Bangla, there are 11 vowels and 39 consonant characters. So, there are total 50 letters overall for maintaining the whole Bangla language. It is not easy to process Bangla language for its complex orthographic rules. There are many critical grammatical rules that are quite so hard to follow in our textual representation. That is why it becomes a common expectation for auto-correction in our text which is known as spelling correction.

A spell checker is an application which detects errors and also provides the suggestions. Generally, a spell checker says about the word where it is misspelled or not. If a word has not any existence on the corpus or dictionary, then it will be an invalid word or misspelled word. There are some common reasons for spelling errors like similar phonetic letters in Bangla, existing of the similar pronounced word, less skill in spelling rules etc. There are many types of errors such as typographical error, cognitive error etc.

Kukich [1] classified spelling errors into two types which are the typographical error and cognitive error. A typographical errors occur while typing (‘দোসর’ as ‘দোসরর’) and cognitive errors (‘বাস’ as ‘বাষ’) occur due to lack of knowledge of how to spell the word. Typographical error also includes insertion error, deletion error, substitution error, transposition error. Cognitive error includes phonetic error.

Kukich also introduced with real-word error and non-word error. Non-word error is a word-level error that happens when a word is not a valid word, i.e.; the word which does not exist in the Bangla grammatical rules. So, the system gives a message about this error. Example: “করিম সাহেবের অনেক অবাব” here “অবাব” is a non-real-word error since it is not a correctly spelled word from the dictionary. The real-word error is a sentence level error which occurs when a word is right from the dictionary but it is not that kind of word which is the suitable word for that sentence. Example: “জেলেরা মাছ ধরার জন্য

978-1-5386-8207-4/18/$31.00 ©2018 IEEE

নদীতে জ্বাল ফেলছে", here the "জ্বাল" word is the correct word but it is not appropriate in the context of the sentence.

In this paper, we have proposed an approach for developing an effective Bangla real-word errors detector. We use the N-gram algorithm for solving the real-word error problem in the sentence. We have used a combination of Bi-gram and Tri-gram with candidate word which is going to be detected whether it is a real-word error or not. We have developed corpora which contain (i) one of them is a collection of sets of homophone (confusing) word, (ii) another two are the collection of bigrams and trigrams using homophone word and (iii) other seven are the test sets. To evaluate the proposed method, we have used 333 sets of homophone words those are collected manually. We collected data from blogs, newspapers etc. Performance evaluation demonstrates that more than96% accuracy may be achieved with our proposed approach.

The rest of the paper is organized as follows. In Section II, Related works are discussed. We have presented our proposed approach in Section III. Performance evaluation is presented in section IV. Finally, Section V concludes this paper.

II. RELATED WORK

To correct the real-word error, there are many techniques that are introduced in this research field. Golding, Andrew, Yves [2] proposed a method which is a combination of Trigram and Baye's theorem. Bayes is the basis on features of words and trigram is basis on parts of speech. For different confusing word POS tagging trigram is used and in case of same POS tagging of confused words, Bayes is used. After applying Tribayes method, the appropriate word is extracted by the calculation of probability. Davide Fossati [3] used POS tagging, Hidden Markov Model and Trigram. To tagged the sentence POS tagging is used and tagged sentences with confusing words are compared with the HMM labeled tags for detecting the difference. Differences are the indication of the need for the correction and trigram are used to correcting the error. Mays [4] has given a statistical method to handle the real-word error. He collected data from the different source. He transforms the corrected sentence into misspelled sentences. He calculated the probability of sentences by using the maximum likelihood estimation of probability. Pratip, Bidyut [5] proposed a model which is used to generate confusion set by Levenshtein Distance and calculates probability by bigram and trigram. By using weighted combination score he detected the error and provided the suggestions. Sumit, Swadha [6] introduce with a model which is used trigram and Bayes approach to extract the features to deal with the real-word error. They also used the synonym of corresponding words, if the features of the targeted word are not found in the corpus. But all of them are designed with the English Language.

In Bangla, it is rare to find the work which deals with the real-word error. Nur Hossain [7] have introduced with a model which is check correctness of a word in a sentence using n-gram. In his approach, he used n-gram of characters and there is no strong specification that they can deal with the real-word error. In what follows, there is not much research done for detecting and correcting real-word error for Bangla language. Therefore, we believe that our approach will bring a new research direction and extension for Bangla real-word error detection and correction.

III. METHODOLOGIES

A. Proposed Framework

Identification and correction of written text are known as spell checking. In Bangla Language, it is harder to find significant research that gives attention specifically to real-word error correction. In this paper, we focus on how this problem might be solved by using n-gram. In our method, we will extract a set of homophone word (different spelling but same pronunciation) where the candidate word is belonging from the test corpus. By using N-gram combination we extract context feature and by which we detect the error and produce a suggestion set against the candidate word.

B. N-gram Approach

Using n-gram in Natural Language Processing (NLP) was first introduced by Shannon [8]. N-gram is a way where sequences of elements are divided into n number of contiguous elements; with the uses, elements would be set of word, character, speech, text etc. A great benefit of n-gram is that it is language independent [9].

In a sentence, n-gram is n number of contiguous word. With one word it is called Unigram when n=1, also for n=2 is Bigram, n=3 is Trigram n=4 is Four gram and finally n-gram.

Example for- "মা আজকে পায়েস রান্না করেছে। "

- set of Unigram} মা ,আজকে ,পায়েস ,রান্না ,করেছে{
- set of Bigram }মা আজকে ,আজকে পায়েস ,পায়েস রান্না ,রান্না করেছে{
- set of Trigram} মা আজকে পায়েস ,আজকে পায়েস রান্না ,পায়েস রান্না করেছে{

C. Building the Approach

Bigram and Trigram are good at extracting the context of its neighbor word. Bigram happens more frequently but has less context feature where trigram has fewer occurrences, but it extracts more features about the context. That is why we are giving first priority to trigram. It is common in other language where trigram or bigram is used for real-word error check. But in Bangla, it is not common and we use n-gram model in our approach. We use the combination of Back n-gram and Front n-gram where n is not greater than 3 and n could be 1 for either Back n-gram or Front n-gram but not less than 2 for both at a time.

We assume that there have no non-word errors in our sentence which will be evaluated. We take a word as CW (candidate word) and search for it in our corpus which is the collection of sets of homophone words. If it can extract a set where it belongs, then we will start our work with this set of homophone word. We generate Back-trigram: CW with previous two words and Front-trigram: CW with next two words. If it fails to generate trigram, then we will generate bigram. CW will be replaced by every word which belongs to the set of homophone words. After that, we will find the frequency of n-gram ($2 \leq n \leq 3$) from that we calculate the

probability by the help of Markov Theory which is the base in our Model.

Markov assumption says that next state is dependent current state only. And by this assumption, we can find the probability of a sentence using the bigram model. But here we are going to use that the occurrence of any word depends on its previous and next words and independent of other words in the sentence which is used by Pratip, Bidyut [7]. But unlike [7], we use the probability of only back trigram or back bigram; back trigram and front bigram; back trigram and front trigram; back bigram and front trigram; back bigram and front bigram, only front bigram or front trigram. This provides us more contextual features.

Let assume we have a sentence consisting of n-numbers of candidates words,

$$\text{Sentence} = \{CW_1, CW_2, \ldots\ldots, CW_n\},$$

and a confusion set (CS) consisting of Z-number of words

$$CS(CW_i^j) = \{CW_i^1, CW_i^2, \ldots\ldots, CW_i^Z\}$$

Now CW_i^j is the j^{th} confusing word from the n-number of words of the sentence which will check for error and replace on i^{th} place to generate n-gram.

If CW_i is in the CS then take the confuse set and for every confuse word from the CS including CW_i find Back and front trigram if trigram is not possible then bigram.

- Back Trigram=$CW_{i-2}\ CW_{i-1}CW_i^j$
- Back Bigram=$CW_{i-1}\ CW_i^j$
- Front Trigram= $CW_i^j CW_{i+1}\ CW_{i+2}$
- Front Bigram=$CW_i^j CW_{i+1}$

Where CW_i^j is the i^{th} word form sentence set and which will be replaced for every j^{th} word for CS and range is $1 \le i \le n$,$1 \le j \le Z$.

After that for CW_i^j find the frequency of Back n-gram and Front n-gram from the n-gram corpus where we collected the data and created bigram and trigram corpus with the frequencies to multiple text file.

Using frequency, we determine the probability-

$$P_{B_3}\left(CW_i^j \middle| CW_{i-2}\ CW_{i-1}\right) = \frac{frq(CW_{i-2}\ CW_{i-1}CW_i^j)}{\sum_{j=1}^{j=z} frq(CW_{i-2}\ CW_{i-1}CW_i^j)} \quad (1)$$

$$P_{B_2}\left(CW_i^j \middle| CW_{i-1}\right) = \frac{frq(CW_{i-1}\ CW_i^j)}{\sum_{j=1}^{j=z} frq(CW_{i-1}CW_i^j)} \quad (2)$$

$$P_{F_3}\left(CW_i^j \middle| CW_{i+1}\ CW_{i+2}\right) = \frac{frq(CW_i^j CW_{i+1}\ CW_{i+2})}{\sum_{j=1}^{j=z} frq(CW_i^j CW_{i+1}\ CW_{i+2})} \quad (3)$$

$$P_{F_2}\left(CW_i^j \middle| CW_{i+1}\right) = \frac{frq(CW_i^j CW_{i+1})}{\sum_{j=1}^{j=z} frq(CW_i^j CW_{i+1})} \quad (4)$$

P_{B_3}, P_{B_2}stand for probability of Back trigram and Bigram, respectively.

P_{F_3}, P_{F_2} stand for probability of Front trigram and Bigram respectively.

Equation (1) says about the probability of back trigram , equation (2) says about back bigram ,equation (3) and (4) says about front trigram and bigram respectively .Denominator of equation (1) and (3) is sum of frequency of trigram or bigram generated with replacing the i^{th} word by j^{th} word from the set CS. We combined the probability of equation (1) and (3) to generate score because trigram is our first priority.

$$\text{Score}\left(W_i^j\right) = P_{B_3} + P_{F_3}$$

$$= \left(CW_i^j \middle| CW_{i-2}\ CW_{i-1}\right) + \left(CW_i^j \middle| CW_{i+1}\ CW_{i+2}\right)$$

If P_{B_3}could not be generated then replace by P_{B_2} and for P_{F_3} by P_{F_2}.

Here, $0 \le \text{Score}\left(W_i^j\right) \le 2$.

The flow chart is shown in Fig. 1 to describe the view of how the propose model works.

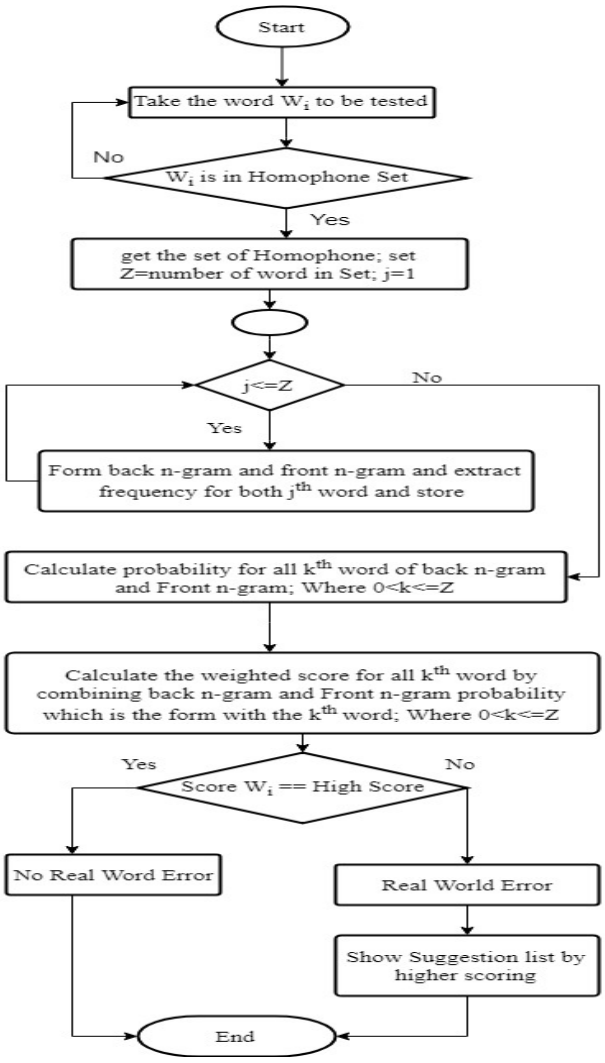

Fig. 1. Flowchart of Proposed Approach

IV. PERFORMANCE EVALUATION

For evaluating the proposed model, we use a corpus which is a collection of confusing words. We have collected so

many articles from the web and from that we have created seven corpora. From those, first, we have generated the trigram and bigram and used our confuse corpus to collect the neighbor words which are used to extract where confuse word occurs.

We have used 333 confuse word sets, 36,687 bigrams with frequency, 46,753 trigrams with frequency to evaluate our algorithm. In order to evaluate we use a self-made program to replace correct the word with the confusing word for generating misspelled sentences with real-word errors which are known as a test corpus. We replace 2400 to 10,200 words with the confusing word. Then we run the model through the seven test corpora which contain misspelled sentences. By detecting homophone or confusing word, we calculate the probability of each of the confusing word from the set of the confusing word. We sorted the confusing words decreasingly. If the confusing word with the highest probability is not the candidate word, then the error is declared and confusing words set is provided as a suggestion list. The word with higher probability will be on the top of the suggestion list. The corpus with minimum real-word error, which contains 2475 errors, gives 97.25% accuracy with detection of 2407 errors and suggesting. The corpus with maximum error consisting of 10673 gives 96.31% accuracy with detection of 10279 errors and provides the suggestion. Performance against the entire corpus is listed below in Table I.

TABLE I. PERFORMANCE EVALUATION

Source	**No. of Word in Corpus**	**No. of error words**	**No. of detected words as error**	**Accuracy**
Corpus 1	60,464	2475	2407	97.25%
Corpus 2	117,680	6018	5749	95.53%
Corpus 3	104,614	4906	4751	96.84%
Corpus 4	102,981	4451	4288	96.34%
Corpus 5	114,368	5939	5738	96.62%
Corpus 6	58,143	2951	2816	95.43%
Corpus 7	196,629	10673	10279	96.31%
Total	755,061	37413	36028	96.30%

In an average, it gives 96.30% accuracy, which would be better with the more enriched collection of neighbor words. In other words, the more resourceful feature of the corpus will give better accuracy. Though we gain a good accuracy, it can be increased by reducing some facts like two or more real-word errors that are happened one after another in a sentence.

V. CONCLUSION

Real-word error in writing is a common occurrence. In this research, we have proposed a model which can deal with it. By using our method, we can detect an error and provide a suggestion list against the error. A very good accuracy level is reached. However, closer to 100% accuracy is may be achieved through extending the model. We have a plan to dig out this issue in our future research works. We strongly believe that, though we achieve some success in this approach, it is not the finish line. Because it is not a complete spell checker and the collections of bigram and trigram features are not so rich. That is why we will extend our model in a complete spell checker with space and time efficiency with the Deep Learning in our future endeavor.

ACKNOWLEDGMENT

This research is partially supported by IEERD (The Institute for Energy, Environment, Research and Development) of the University of Asia Pacific, Dhaka, Bangladesh.

REFERENCES

[1]. K. Kukich, “Techniques for automatically correcting words in text,” ACM Computing Surveys, 24 (4), page 377 - 439, 1992.

[2]. Golding and Andrew, "A Bayesian hybrid method for context-sensitive spelling correction," arXiv preprint cmp-lg/9606001, pp 1-15 (1996).

[3]. Fossati, Davide and B. Eugenio, "I saw TREE trees in the park: How to Correct Real-Word Spelling Mistakes," LREC, pp 896-901 (2008).

[4]. Mays, Eric, F. J. Damerau and R. L. Mercer, "Context based spelling correction," Information Processing & Management 27.5, pp517-522 (1991).

[5]. P. Samanta and B. B. Chaudhuri, "A simple real-word error detection and correction using local word bigram and trigram," Published 2013 in ROCLING.

[6]. S. Sharmaa and S. Gupta, “A Correction Model for Real-word Errors,” Procedia Computer Science Volume 70, 2015, Pages 99-106.

[7]. N. H. Khan, G. C. Saha, B. Sarker and M. H. Rahman, “Checking the correctness of Bangla words using n-gram,” International Journal of Computer Application, vol. 89, no. 11, 2014.

[8]. C. E. Shannon, “Prediction and entropy of printed English,” Bell system technical journal, vol. 30, no. 1, pp. 50–64, 1951.

[9]. F. Ahmed, E. W. D. Luca and A. Nürnberger, “Revised n-gram based automatic spelling correction tool to improve retrieval effectiveness,” Polibits, no. 40, pp. 39–48,20.