Auto-correction of English to Bengali Transliteration System using Levenshtein Distance

Conference Paper · June 2019

DOI: 10.1109/ICSCC.2019.8843613

5 authors, including:

Farhan Labib
East West University (Bangladesh)
2 PUBLICATIONS 9 CITATIONS
SEE PROFILE

Amit Kumar Das
East West University (Bangladesh)
39 PUBLICATIONS 290 CITATIONS
SEE PROFILE

Auto-correction of English to Bengali Transliteration System using Levenshtein Distance

Md. Mosabbir Hossain, Md. Farhan Labib, Ahmed Sady Rifat, Amit Kumar Das, Monira Mukta
Department of Computer Science and Engineering
East West University
Dhaka, Bangladesh
e-mail: mosabbirtarek7@gmail.com, farhan.labib4@gmail.com, sadyrifat@gmail.com, amit.csedu@gmail.com, monira.mukta7@gmail.com

Abstract— **The automated transliteration process is a function or software application that checks words against a computerized corpus to ensure that they are correct. A transliteration system is required either during the typing of text or when a user does not know the correct spelling of a word. The main objective of this research is to develop a system that is much better to check the spelling of the transliterated word by calculating Levenshtein distance. To make the mechanism more efficient and accurate unigram method has been implemented. Several techniques have been integrated for data collection to make the system more reliable and flexible. Almost twenty thousand words are included to create a data lexicon for this research work. This system is able to deal with signed or unsigned numeric values and float numbers with 78.13% accuracy.**

Keywords- ***Data Corpus, Data Processing, Levenshtein Distance, N-gram, NLP, Unigram, Transliterate Word.***

I. INTRODUCTION

Since the dawn of civilization, people are using the writing method to express their thoughts and views. To expose one's feelings, it is the best way till now. Hence, to express verbally there is no need of checking spell. But in case of the composed method, it is a crucial issue to check the spell of each word of a sentence. Correct spelling helps the reader to understand the meaning instantly.

Moreover, manual checking of a spell is difficult and time-consuming. So, the necessity of automatic spell checking is beyond description. For that purpose, natural language processing (NLP) is required to recognize the word or speech [1].

Words that the spell checker recognizes as incorrectly spelled are typically featured or underlined. With the assistance of NLP, an automatic spell checker will detect a word as errored or misspelled if that particular word is not matched with the corpus. The system not only detects but also corrects the word. Hence, it also suggests the appropriate word for a sentence. The primary spell checkers were verifiers rather than correctors. Even they did not suggest mistakenly spelled word. There may be some spelling checker for English language but the Bengali language; it is sporadic. Moreover, approximately 189 million people speak in the Bengali language around the world. These people indeed use the transliterated word (Bangla mixed with English) to share their thoughts in writing purpose.

In our research, we have developed a system where the system will distinguish the misspelled Bengali word and will recommend a suitable right word for the sentence calculating Levenshtein distance and using unigram strategy. For instance: if someone writes "amra besay zabo" then the result should show "আমরা বাসায় যাব" instead of "আম্রা বেসায় যাব". Here, the word "besay" is misspelled. So, the goal of our research is to diminish the incorrect word and replace it with the appropriate word written in the correct spell.

The remaining paper ordered as follows- Section II gives a compact description of some previous works related to this research area. Section III gives a simple overview of our working procedure. In Section IV, we briefly describe the methodology of our research work and provides information regarding dataset processing. In Section V, we present our result section by comparing the two methods. We discuss our future directions and conclude the paper with some useful suggestions in Section VI.

II. RELATED WORK

There is some framework that has introduced in this research field to check and correct the misspelling word. There are several techniques to develop this framework like: according to the string, statistical approach, rule-based framework, similarity keyword, probabilistic and so on. A statistical method to handle the errored word was adopted by Mays [2]. In that research, various data were collected, and those were transformed into misspelled sentences. Then calculate the probability of sentences by using maximum likelihood estimation of probability. Another study was done by Bidyut [3] where a confusion set was generated, and probability was calculated by using bigram and trigram technique. The weighted score detected the error. These works were done for the English language which was not applicable for the Bengali language. Hence, a few works have been accomplished on the Bengali language. A research was done on double metaphonic encoding for Bangla language spelling checker [4] where the system was complex and based on the consonant cluster. Moreover, it was not automated. Another work has been done for detecting and correcting words based on social data set by filtering messages [5]. Murthy, Akshatha, Upadhyaya & Kumar have done their research work on

Kannada spell checker with sandhi splitter [6]. Another work has been accomplished by Donghuilee and peng over Chinese language implementing Soundex algorithms [7].

After considering the above cases we resolved to build up the Bengali spell checker with more effectiveness and all more precisely in a simple way using unigram strategy. To develop Bengali spell checker, we deployed the Levenshtein algorithm to predict the correctly spelled word.

III. Proposed Model

Our framework is able to take an input of any transliterate word string which means Bangla interspersed with English. For instance: "Ami bhat khai". Here, the sentence is written in transliterate word. This transliterates word sentence is be converted into Bangla sentence with the assistance of this system. For anticipating anything, the machine needs lots of data. So, a Bangla data corpus is added to the system where all the Bangla words are written incorrectly spelled. At the point when input is taken, it will be compared to the data corpus. For making a comparison, Levenshtein algorithm has been utilized. This algorithm calculates the distance between two sequences.

Our proposed system will compare the input string with all of the similar data which are included in the data corpus. The minimum distance will be considered as the predicted result. This framework won't just anticipate the alphabetic spelling yet, also, think about the numeric esteem. For instance: If an input is taken as "dhatuti 500 degree Celsius-e uttopto ache" which implies the metal is heated in 500 degree Celsius will be resulted in the form of "ধাতুটি ৫০০ ডিগ্রী সেলসিয়াসে উত্তপ্ত আছে". Here, the numeric value will not be scattered or dissipated. It will also be incorporated into the system. Again, in case of floating point for numeric value, there will be no correction for those values [8]. For instance:

"চালের দাম প্রতি কেজিতে ১০. ৫০ টাকা বাড়িয়েছে।"

Here, the program does not consider "." as a punctuation mark. Rather it considers as floating point and returns that as it is. When the system converts transliterate word, it converts "." to "।" as the meaning of "." is the ending point of the sentence. In Bangla, the ending point is indicated by "।". Even it can deal with "+" and "-" symbol such as:

"কানাডায় আজকের তাপমাত্রা -৪ ডিগ্রি সেলসিয়াস।"

Here, "-8" is a numeric value with special character "-". This will not be edited by the system. However, if a user gives input of some sentences with punctuation mark like "।", "?", "!", ",", ":", ";" then the system omits these marks for better calculation. Finally, when the correction process is done the system replaces the punctuation mark as the user has given in the input sentence.

In brief, the system takes transliterate word as input and converts it into Bangla. Hence, split the sentence into words. Those words are compared to the data corpus computing distance. If the distance is above 65 percent or equal, then the word is added for suggesting. Else the system skips the word and will keep the same word.

The following flow chart demonstrates the working procedure of our system:

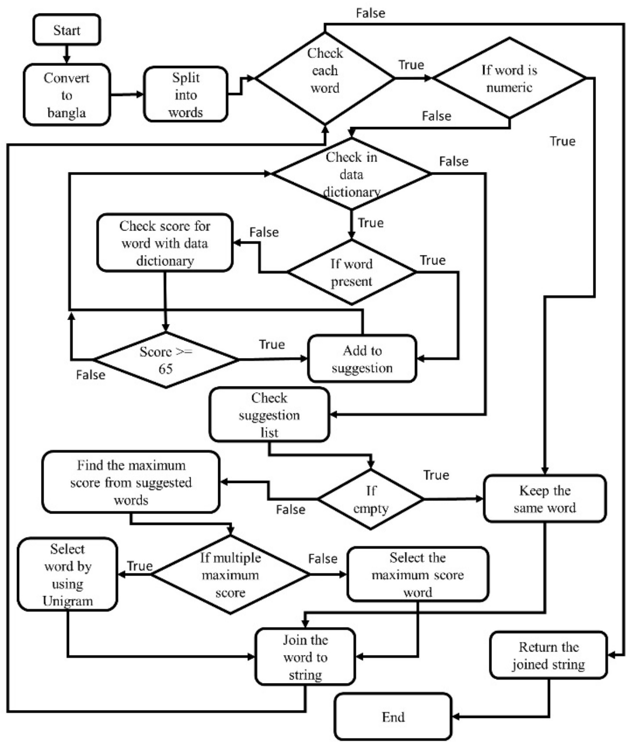

Figure 1. Working procedure of spell checker

In Figure.1 the proposed system starts with converting the transliterated sentence into Bangla sentence using a text parser which alters Bangla written in the Roman inscription to its phonetic comparable in Bangla. After that, the Bangla sentence is split into individual words. Each word is checked with the data corpus. At the point when a word is found which is a numeric value, it is straightforwardly added to the output string. Let's take a sentence like:

"চালের দাম প্রতি কেজিতে ১০ টাকা বাড়িয়েছে।"

Here the word "১০" is a numeric value. When this type of numeric value appeared, then there will be no correction. However, in the case of non-numeric value, the word is sent to the data corpus.

Subsequently, the program checks whether the word is available in the corpus or not. At that point, if the word is available, then program adds that word to the suggestion list. But if the word is not available in the corpus, then the program checks its data words and makes a correlation. As a consequence of correlation, if the score value is greater than or equal to 65 percent then that word from the data corpus appends to the suggestion list. Otherwise, it goes for the following word for making a comparison from the data corpus. After finishing the calculations with all words from the data corpus, it checks the suggestions list whether it is empty or not. If it is empty, then it keeps the same word as the user has given in the input data.

On the contrary, if the suggestion list is not empty then it finds the maximum score from the suggested words. If the same scores appear for multiple words then it corrects the word by using unigram method. Otherwise, it simply selects the word having maximum score value and then it joins the word to the output string. Then it goes for the next input word. When the calculation is finished for all input words then the joined string will be replaced as predicted correct sentence [9].

IV. Methodology

A. *Levenshtein Distance*

The system is developed based on Levenshtein algorithms. It calculates the distance between two strings. For calculating the distances following equation was considered:

$$Lev_{a,b}(m,n)\left\{\begin{matrix} max(m,n) \\ min\left\{\begin{matrix} Lev_{a,b}(m-1,n)+1 \\ Lev_{a,b}(m,n-1,)+1 \\ Lev_{a,b}(m-1,n-1)+1_{(a_m\neq b_n)} \end{matrix}\right\} \end{matrix}\right\} \quad (1)$$

Here the function (am ≠ bn) denotes to 0 when am ≠ bn and equal to 1 otherwise, and lev$_{a,b}$ (m, n) is the distance between the first m characters of a and the first n characters of b. It has to be noted that the rows on the minimum above lead to deletion, an insertion, and a substitution respectively. Levenshtein algorithms consider the minimum distance to suggest correctly spelled word. The higher the distances, the different the strings are. This technique also calculates the similarity ratio between two words using the following formula:

$$\frac{(|a|+|b|)-Lev_{a,b}(m,n)}{|a|+|b|} \quad (2)$$

Where |a| and |b| represents the length of the sequence of a, b respectively.

If the two strings are 100% similar then it requires no change, the only substitution occurs. But if any string is found which do not match with the data corpus, then it will add the string to the corpus by erasing the misspelled word which refers to insertion and deletion. For example:

ব	ি	ক	া	শ	ে	র
ব	ি	ক	া	স	ে	র

Edit distance = 1(বিকাশের, বিকাসের)

ক	ে	ন	ো
ক	ে	ন	

Edit distance = 1(কেন, কেনো)

প		ন	্	য		
প	ু	ন	্	য	ে	র

Edit distance = 3 (পন্য, পুন্যের)

Insertion
Substitution
Deletion

The above scenario represents three methods of calculating edit distance. For the first case of substitution "বিকাসের" will be replaced by "বিকাশের" where edit distance is 1.

Hence, for insertion "কেন" is inserted instead of "কেনো" and for deletion "পন্য" was generated from the word "পুন্যের".

This technique was deployed to measure the edit distance between two words.

B. *Data Processing*

To establish this framework various data are needed to make a data corpus for predicting or recommending any word. These data are collected automatically from various online Bangla newspapers by using ParseHub. As correctly spelled Bangla words are challenging to collect, so the source of data collection is chosen from online Bangla newspapers. ParseHub is a tool where lots of data is possible to extract from the various website within a short time. But all of those data are not pure data as English letter or some special character (like:" : ; \ / @ # $ % & ? | ! + - _ and so on) can be stirred up there. So, data filtering is needed to extract pure data from raw or garbage data. At the time of data filtering any brackets, English characters, special characters are removed [10]. For instance: "নোভা থ্রি আই (NOVA 3i) স্মার্টফোনের দাম কমিয়েছে। স্পেনের বার্সেলোনায় অনুষ্ঠিত এ বছরের মেলায় বড় চমক ছিল ফোল্ডিং (Folding) বা ভাঁজ করা স্মার্টফোনের পুনরায় আবির্ভাব। এ ক্ষেত্রে বড় দুই মুঠোফোন নির্মাতা স্যামসাং (Samsung) ও হুয়াওয়ে. (Huawei) ছিল আলোচনার কেন্দ্রবিন্দুতে। তাদের দেখানো গ্যালাক্সি ফোল্ড (Galaxy Fold) এবং মেট এক্স প্রতিযোগিতায় প্রতিষ্ঠান দুটিকে এক ধাপ এগিয়ে নিল।".

The above sentences will be converted into: "নোভা থ্রি আই স্মার্টফোনের দাম কমিয়েছে। স্পেনের বার্সেলোনায় অনুষ্ঠিত এ বছরের মেলায় বড় চমক ছিল ফোল্ডিং বা ভাঁজ করা স্মার্টফোনের পুনরায় আবির্ভাব। এ ক্ষেত্রে বড় দুই মুঠোফোন নির্মাতা স্যামসাং ও হুয়াওয়ে ছিল আলোচনার কেন্দ্রবিন্দুতে। তাদের দেখানো গ্যালাক্সি ফোল্ড এবং মেট এক্স প্রতিযোগিতায় প্রতিষ্ঠান দুটিকে এক ধাপ এগিয়ে নিল।".

After getting pure Bangla data in a paragraph format, the next step is to split the data into sentence format following the punctuation mark. For instance, the above paragraph is converted into:

- নোভা থ্রি আই স্মার্টফোনের দাম কমিয়েছে।
- স্পেনের বার্সেলোনায় অনুষ্ঠিত এ বছরের মেলায় বড় চমক ছিল ফোল্ডিং বা ভাঁজ করা স্মার্টফোনের পুনরায় আবির্ভাব।

Hence, these data are again split into words to make the data corpus. For data corpus, unique words will be selected. Let's consider the above sentences:

- নোভা থ্রি আই স্মার্টফোনের দাম কমিয়েছে ।
- চালের দাম প্রতি কেজিতে ১০ টাকা বাড়িয়েছে।
- আইফোন ৫ এর দাম কমিয়েছে।

Here the word "দাম" is appeared three times in all sentences and the word "কমিয়েছে" appeared two times in the first and last sentence. But only one time the word is appended to the data corpus so that each and every word remains different from each other. Thus, the data corpus will be created by selecting unique words. We use corpus rather than corpus because of the simplicity of sentence as Bangla language has complicated grammar.

C. Unigram Approach

Unigram approach deals with a single item from a sequence. It originates from the concept of the N-gram model. In N-gram model prediction occurs considering two or more than two (N= 2,3,4….) subsequent words. If the N-gram model chooses "N=2" then it is Bigram, and for "N=3" it is Trigram. Bigram approach considers two consecutive words for its prediction. For instance:

"গতকাল সারাদিন অনেক গরম পড়েছিল"

Bigram: {("গতকাল, সারাদিন"), ("সারাদিন, অনেক"), ("অনেক গরম"), ("গরম পড়েছিল")}

In Trigram, approach system thinks about three consecutive words for its prediction. For instance:

"গতকাল সারাদিন অনেক গরম পড়েছিল"

Trigram: {("গতকাল সারাদিন অনেক"), ("সারাদিন অনেক গরম"), ("অনেক গরম পড়েছিল")}

Words are the primary substance for sentences. The group of words provides more advantages to express the meaning of a sentence. From this perspective Bigram or Trigram approach seems better than the unigram approach as they deal with sentences. But the objective of this system is not to manage sentences rather than words. So, Bigram and Trigram approach is not suitable for this proposed system. The system concerns about individual word for making a suggestion from data corpus which is incorporated in the unigram approach. Thus, the unigram approach has been chosen to increase the performance of this framework.

In the unigram approach, the system will consider only one single word (N=1) which has been used most frequent time in the data corpus. For instance:

"আমি ভাত কাই"

The above sentence is the set of words {আমি, ভাত, কাই}.

The word "কাই" incorrect with that particular sentence. At this point, it is necessary to choose an appropriate word from the data corpus to make the sentence accurate. So, the unigram approach is implemented here to resolve the error. Hence, Unigram will find all of the words from the corpus which are related to the word "কাই". For instance:

{খাই, নাই, যাই, পাই, গাই, ভাই}

Here the score value of the word "কাই" with all the suggested words are the same. After finding the same score value unigram considers that which word has been used most frequently or maximum times for the similar type of sentence. The highest score of utilization of that particular word is selected as a correctly spelled word. For instance:

- খাই = [20 times]
- নাই = [15 times]
- যাই = [12 times]
- পাই= [07 times]
- গাই= [13 times]
- ভাই= [05 times]

Here, "খাই" is the word that appeared maximum times among all of the similar words. So, the unigram will choose this word for that sentence. This is how unigram filters the words. To generate unigram around 10,000 sentences has been utilized. Those sentences consist of more than 1,26000 words. And the unigram set build for around 18,000 words.

For applying the unigram approach, the following formula has been adopted:

$$Y(w_i) = \frac{X(w_i)}{X(w)} \qquad (3)$$

Here, "Y(wi)" represents the probability of the recommended word and it is a specific word. "X(wi)" is some times that the word occurred in the data corpus and "X(w)" stands for a total number of words in the data corpus for all words.

V. Results and Analysis

To suggest any word from the data corpus, the system finds out the percentage of the similarity between two Bangla words. If the similarity is at least 65% or above then it considers that particular word for the suggestion. Either it adds the word in the corpus. The accuracy of this proposed system is calculated automatically. To estimate the accuracy, some functions were considered. Table. I represent the overall performance considering the unigram approach for this framework and Table. I represent the performance without considering the unigram approach. To establish the automatic function, the following formula has been considered:

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \qquad (4)$$

Here, TP = True positive
TN = True Negative
FP = False Positive
TN = False Negative

Table I: Performance Evaluation

Exp No.	No. of input Words	Correct Words		Error Words		Accuracy	
		With Unigram	Without Unigram	with Unigram	Without Unigram	With Unigram (%)	Without Unigram (%)
1	25	22	20	3	5	88.00	80.00
2	20	12	10	8	10	60.00	50.00
3	15	12	10	3	5	80.00	66.67
4	15	11	13	4	2	73.33	86.67
5	15	13	11	2	4	86.67	73.33
6	15	10	12	5	3	66.67	80.00
7	15	12	9	3	6	80.00	60.00
8	15	11	11	4	4	73.33	73.33
9	15	14	10	1	5	93.33	66.67
10	15	12	09	3	6	80.00	60.00
Total	165	129	115	36	50	Avg. Accuracy = 78.13	Avg. Accuracy = 69.67

In the evaluation scheme, more than 18,000 unique words are considered as training data. In Table. I 165 words are considered as test data where 129 are correct words based on the user's context. On the contrary, in Table. I 165 words are considered as test data where 115 are correct words based on the user's context. After executing the system for those ten experiments, the total error is found 36 words for Table. I and 50 words for Table. I. However, most of the words are also correct words but incorrect based on the user's context.

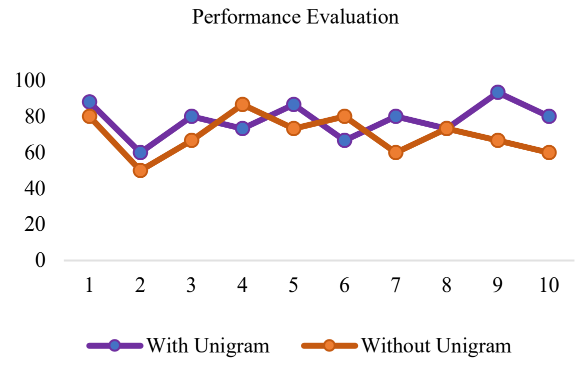

Figure 2. Performance Evaluation

After analyzing the performance of the system, it is found that the rate of average accuracy for Table. I give 78.13% utilizing the unigram approach and for Table. I without unigram it is 69.67% which is less than the unigram approach.

Figure 2. demonstrates the performance evaluation of the unigram approach and without the unigram approach. Consequently, the Unigram Approach increases the accuracy level and improves this proposed system efficiently with Levenshtein distance. The main factors that lead to the error in our result are- for regular use, a user generally use the lowercase letter for typing but in some Unicode mapping used Uppercase letter and on the other hand, user try to use shortcut form of a regular word.

VI. CONCLUSION

This research aimed to build up a framework that would almost certainly distinguish misspelled transliterated words and would recommend the correctly spelled word naturally. This spell checker is additionally ready to look the words alphabetically to lessen the proposal time. Since a massive amount of data are added in the lexicon to make suggestion and unigram approach extracts the appropriate word from those suggestion lists, so the accuracy level stands in a standard structure. By this way, the system corrects automatically.

REFERENCES

[1] G. Chowdhury, "Natural language processing," Annual Review of Information Science and Technology, vol. 37, no. 1, 2005, pp. 51-89.

[2] E. Mays, F. Damerau and R. Mercer, "Context-based spelling correction," Information Processing & Management, vol. 27, no. 5, 1991, pp. 517-522.

[3] P. Samanta and B.B. Chaudhuri, “A simple real-word error detection and correction using local word bigram and trigram,” ROCLING, 2013, pp 211-220.

[4] N. UzZaman and M. Khan, "A Double Metaphone encoding for Bangla and its application in spelling checker," 2005 International Conference on Natural Language Processing and Knowledge Engineering, Wuhan, China, 2005, pp. 705-710.

[5] Z. Z. Wint, T. Ducros and M. Aritsugi, "Spell corrector to social media datasets in message filtering systems," 2017 Twelfth International Conference on Digital Information Management (ICDIM), Fukuoka, 2017, pp. 209-215.

[6] S. R. Murthy, A. N. Akshatha, C. G. Upadhyaya and P. R. Kumar, "Kannada spell checker with sandhi splitter," 2017 International Conference on Advances in Computing, Communications, and Informatics (ICACCI), Udupi, 2017, pp. 950-956.

[7] D. Li and D. Peng, "Spelling Correction for Chinese Language Based on Pinyin-Soundex Algorithm," 2011 International Conference on Internet Technology and Applications, Wuhan, 2011, pp. 1-3.

[8] J. Islam, M. Mubassira, M. R. Islam and A. K. Das, "A Speech Recognition System for Bengali Language using Recurrent Neural Network," 2019 4th International Conference on Computer and Communication Systems (ICCCS), Singapore, 2019.

[9] R. A. Tuhin, B. K. Paul, F. Nawrine, M. Akter and A. K. Das, " An Automated System of Sentiment Analysis from Bangla Text using Supervised Learning Techniques," 2019 4th International Conference on Computer and Communication Systems (ICCCS), Singapore, 2019.

[10] A. K. Das, T. Adhikary, M. A. Razzaque and C. S. Hong, “An intelligent approach for virtual machine and QoS provisioning in cloud computing,” The International Conference on Information Networking 2013 (ICOIN), Bangkok, 2013, pp. 462-467.