A Comprehensive Bangla Spelling Checker

Naushad UzZaman and Mumit Khan
Center for Research on Bangla Language Processing, BRAC University, Bangladesh
naushad@bracuniversity.ac.bd, mumit@bracuniversity.ac.bd

Abstract

We present a comprehensive Bangla spelling checker that improves the quality of suggestions for misspelled words. The complex rules for Bangla spelling presents a significant challenge in producing suggestions for a misspelled word when employing the traditional methods; one must take phonetic similarity into account for suggested alternatives to be reasonably accurate. In Bangla there are several algorithms available for spell checking, however, none of these considers the complex orthographic rules of Bangla. As a result, spelling checker application does not perform well. In this paper, we describe the process of checking the spelling of a Bangla document (i.e. detecting misspelled words, generating suggestions for misspelled word, and ranking the suggestions), compare the methodologies with existing solutions available in the literature, and then propose solutions for each step. Finally, we conclude by showing the performance and evaluation of our proposed solution.

1. Introduction

There are more than 200 million native speakers of Bangla, the majority of who live in Bangladesh and in the Indian state of West Bengal [1]. However, there has been very little research effort in the computerization of the Bangla language, leading to a dearth of Bangla natural language processing applications and tools. A Bangla spelling checker, one such application, is an essential component of many of the common desktop applications such as word processors as well as the more exotic applications, such as a machine language translator. One particular challenge facing the development of a usable spelling checker for Bangla is the language's complex orthographic rules, in part a result of the large gap between the spelling and pronunciation of a word [2]. One impact of this complexity can be seen in the observation that two of the most common reasons for misspelling are (i) phonetic similarity of Bangla characters and (ii) the difference between grapheme representation and phonetic utterances [3]. While there has been a sustained effort of late to develop a usable spelling checker, none of the solutions has been able to handle the full orthographic complexity of Bangla [4-9].

In the following sections we will describe the steps in the process of checking the spelling of a word:

a) detect whether it is misspelled or not,
b) generate suggestions if it is misspelled, and
c) rank the suggestions so that the most likely candidate is placed first.

We then propose a solution for each of these steps, and compare our solution with those in the literature. Lastly, we show the performance and evaluation of our proposed solution.

2. Detecting a misspelled word

To give suggestions for a misspelled word, the first step for a spelling checker is to detect the misspelled word. But before detecting a misspelled word, we need to know what a misspelled word is. Misspelled words or errors can be of many types, such as typographical error, cognitive error, etc.

Kukich [10] breaks down human typing errors into two classes, typographical error and cognitive error. Typographical errors (e.g., misspelling 'spell' as 'speel') generally occur due to people's mistakes while typing. Cognitive errors (e.g., misspelling 'separate' as 'seperate') are caused by writers who do not know how to spell the word.

Cognitive errors include phonetic errors (e.g., misspelling 'separate' as 'separate'), substituting a phonetically equivalent sequence of letters and homonym errors (e.g., misspelling 'peace' as 'piece'), happens from typographical errors (insertion, deletion, transposition, substitution), which accidentally produce a real word (e.g., misspelling 'there' as 'their'), or because the writer substituted the wrong spelling of a homophone or near-homophone (e.g., 'dessert' as 'desert', or 'piece' as 'peace', and vice versa).

2.1. Previous work on detecting misspelled word in Bangla

Detecting a misspelled word for a language is trivial for typographical errors and the cognitive phonetic errors. But cognitive homonym errors, which are real word errors[1], cannot be detected easily. We need to consider the context of a word to detect a misspelled word in this case.

For Bangla, approximate string matching algorithms [4] and a direct dictionary look up method [5] have been used so far for the detection of typographical errors and cognitive phonetic errors. In our spelling checker, we used the direct dictionary look up method for detecting a misspelled word. But none of these methods, including our method, deals with homonym errors.

3. Generating suggestions for misspelled words

After detecting the misspelled word we need to generate the suggestions for it. Before going in to the details of suggestion generation, we will discuss the error patterns in usual typing and also the phonetic error patterns found in Bangla language.

3.1. Error pattern of typographical error

Damerau [11] finds that 80% of all misspelled words (non-word errors) in a sample of human keypunched text were caused by single error misspellings, i.e., any of the following errors:

1. Insertion. For example: mistyping *the* as *ther*
2. Deletion. For example: mistyping *the* as *th*
3. Substitution. For example: mistyping *the* as *thw*
4. Transposition. For example: mistyping *the* as *the*

Damerau's [11] report was for English and although the case for Bangla is not the same, it is similar to Damerau [11].

B.B. Choudhury [4] finds that 41.36% of all misspelled words, out of 15,162,317 words, were caused by single error misspellings (which he termed as error zone length = 1) and 32.94% with error zone length = 2.

It is clear from the discussion above that we can generate good suggestions for typographical errors in Bangla if we consider the words for errors up to 2-edit distance[2]. Edit distance is not the same as error zone in [4] - error zone is a subset of edit distance. So, if we consider 2-edit distance, then 2-error zone is also automatically considered.

3.1.1. Previous work on typographical error. Almost all the major Bangla spelling checkers handle up to 2-edit distance, which includes more than 70% of the errors [4]. B.B. Choudhury [4] handles it using error zone length; Abdullah and Rahman [5] handle it using their unique recursive simulation method.

3.1.2. Our proposal for generating suggestion for typographical error. It is clear that other methods handle typographical errors up to 2-edit distance. Their technique can be used but we preferred our own effective way of handling this case. B.B. Choudhury's method [4] needs twice the amount of memory for the reverse dictionary. Abdullah and Rahman's [5] recursive simulation, on the other hand, trades off time for space, requiring more than m^(2*n+1) dictionary lookups for an 'n' length word, where 'm' is the average number of letters in their circular list. The value of 'm' is an integer, which varies generally from 1-5 and is usually more than 2 or 3.

In our case, for a particular misspelled word, we define a subset of the lexicon that is then used to produce the list of suggestions. This subset, called the "short list", consists of the words whose lengths are within +/- 2 units of the length of the misspelled word, as shown below.

```
Length of short-listed words = words
with length of misspelled word OR
length of misspelled word + 1) OR
length of misspelled word – 1) OR
length of misspelled word + 2) OR
length of misspelled word – 2)
                                    (1)
```

From the short-listed words, we find the words with edit distance of 2 from the misspelled word. Note that only the words in the short list will have a maximum edit distance of 2 from the misspelled word, which obviates the need for computing the edit distances of the entire lexicon from the misspelled word.

[1] By 'real word error' we mean a correctly-spelled word but not the intended word in the sentence, thus making the sentence syntactically or semantically ill-formed or incorrect.

[2] Edit distance [12] is defined as the number of insertions, deletions, and substitutions required changing on string into another. B.B. Choudhury [4] uses a technique to find the position in the word where the error occurred. This error length is the error zone length.

Typographical suggestion list = Words having Edit Distance[3] (misspelled word, each word of short-list words) less than and equals to 2.

(2)

3.2. Error pattern for cognitive phonetic error

Bangla has complex orthographical rules. One reason behind the existence of these rules is a large number of words in Bangla are from Sanskrit, an ancestral predecessor of Bangla. However, these words have either been modified in terms of pronunciation or both in terms of spelling and pronunciation. Thus there exists a gap between spelling and pronunciation requiring complex orthographical rules.

Below we will discuss the challenges for generating suggestions for phonetic error, which we face because of complex orthographical rule described above.

1. There are groups of phonetically similar characters in Bangla; for example, NA (ন) and NNA (ণ); SA (স) and SHA (শ) and SSA (ষ), etc. The contrast between long and short vowels in the script is also in the modern version of the spoken language.
2. Bangla has many consonant clusters or conjuncts with unusual pronunciations (i.e., ক্ষ, ষ্ম, etc.): let us consider ক্ষ. ক্ষ = ক+্+ষ; ক্ষত [KA HASANT SSA TA] /k^{h}ɔ̰to/ is pronounced as খত [KHA TA] /k^{h}ɔ̰to/, where ষ does not have any sound.
3. Bangla has different uses of *Phalaa's,* the cluster final form of the semi-vowels in Bangla (BA, MA, YA, RA and LA), which are represented using a distinct sign-form. BA *phalaa* for example has a distinct pronunciation from a BA in any other position in a cluster or in a standalone configuration.
4. Different pronunciation of letters or conjuncts in different contexts: consider again ক্ষ. At the beginning of word, it is pronounced as খ /k^{h}/. (ক্ষত → খত /k^{h}ɔ̰to/); in the middle or at the end of a word, it is pronounced as কখ /kk^{h}/, (দক্ষ → দকখ /d̰okk^{h}o/).
5. Multiple pronunciations of some letters in the same context, such as হ with ব: According to Bangla phonological rules, হ should be pronounced as ও or উ and ব should be pronounced as **ভ:** আহ্বান → আওভান /aovan/. However, most native speakers pronounce these words the same way as it is written. For example, আহ্বান is usually pronounced as আহভান /ahob^{h}an/. Both pronunciations are considered correct.

3.2.1. Previous work on phonetic error. Phonetic error for Bangla has been noticed by few researchers before but none of them did an in depth analysis of this error.

B.B. Choudhury [4] mentions the phonetic problem and solved this by representing phonetically similar vowels and consonants by a single code; however, this solves only the first problem mentioned above, and it does not deal with other problems that have been mentioned.

Abdullah and Rahman [5] mention the phonetic problem as well and solved this by their own circular list mechanism; however, this too deals with only the first problem mentioned above. Even though Abdullah and Rahman [5] discuss the third problem mentioned above, they do not consider the full phonetic complexity of Bangla orthographic rules.

Haque and Kaykobad [6] propose a phonetic encoding [13] based on Soundex [14] for spelling checking of Bangla, which is also limited in that it handled the first problem and the trivial cases of the third one.

UzZaman and Khan [7] propose a phonetic encoding also based on Soundex, with the same limitations as above. In addition, their encoding is more fine-grained than Haque and Kaykobad's [6], and it handled some trivial cases of Bangla consonant clusters or *jukhtakhors*.

3.2.2. Our proposal for generating suggestion for phonetic error. None of these mechanisms was good enough to face the challenges of phonetic errors described earlier in this paper. We used the phonetic encoding approaches used for Western languages such as English to detect and correct the phonetic errors in Bangla. Before proceeding to our phonetic encoding, we will discuss briefly the English phonetic encoding.

3.2.2.1. Phonetic encoding in English. Phonetic encoding codes a word based on how it is pronounced. For this reason similar sounding words have same phonetic code. So, if phonetic encoding can represent its pronunciation properly then we can easily solve the problem of phonetic error. In the case of applications

[3] Edit Distance (string s1, string s2) returns an integer, which is the edit distance [12] between two strings.

using the phonetic encoding, we will only check the codes not the words.

Back in 1918, Odell and Russell proposed Soundex, the first phonetic encoding for English to use in the US census. Soundex partitions the set of letters in to seven disjoint sets, assuming that the letters in the same set have similar sound. Each of these sets is given a unique key, except for the set containing the vowels and the letters h, w, and y, which is considered to be silent and is not considered during encoding. For example, both *realize* and *realise* has been coded to '642' in Soundex encoding, which works well for the trivial cases but fails to give same code to words where letters change its pronunciation in different contexts. For example, *knight*, *night* and *nite* are similar sounding words but Soundex does not give the same code to these words.

It is clear that to give a phonetic code in English we also need to consider the context of letters. For example, in the word *knight*, by analyzing the language we can find that the 'k' at the initial position followed by a 'n' is silent and 'gh' together is silent if it is not at the end or before a vowel, considering these cases before giving a phonetic code can generate same code for *knight*, *night* and *nite*. Lawrence Philips in 1990 invented a phonetic encoding called Metaphone encoding [15,16] that handles these context issues before giving a phonetic code. This gives accurate phonetic encoding for English in most of the cases but there was another problem that Philips followed. There are some words, which have multiple established pronunciations. For example, Basinger is pronounced in both ways as "Basin-gger" or "Basin-jer". But in Metaphone encoding we only get one code, which cannot represent multiple codes (which eventually is multiple pronunciation) at the same time. If we can give multiple codes to words with multiple pronunciations based on their pronunciations then this problem can also be solved.

Philips, in 2000, came with a better phonetic encoding, which is an extension of Metaphone encoding with some modifications and also gives multiple codes to words with multiple pronunciations. He named his new phonetic encoding Double Metaphone encoding [16].

3.2.2.2. Phonetic encoding in Bangla. Phonetic encoding has been tried before in Bangla as a solution of spelling checker. Haque and Kaykobad [6] and UzZaman and Khan [7] tried the Soundex approach of disjointing letters of similar sound in Bangla and give them same code. As mentioned earlier this solution solved the problems of phonetically similar characters in Bangla.

Reviewing the challenges of phonetic errors in Bangla and phonetic encoding of English we can come to the conclusion that following the approach of English encoding we can solve our problems. Metaphone encoding considers the context of letter in a word before giving it a phonetic code. Using this method we can give phonetic code to the word based on their pronunciation.

Challenge: Consonant clusters or conjuncts with unusual pronunciation. ক্ষ = ক+্+ষ; ক্ষত [KA HASANT SSA TA] /k^{h}ɔ̪to/ is pronounced as খত [KHA TA] /k^{h}ɔ̪to/, where ষ does not have any sound.

Solution: We found that here ক্ষ is sounded as খ. If we can give ক্ষ the code of খ then we solve this problem.

Challenge: Different uses of Phalaa's. For example, BA phalaa after a consonant of initial position does not have any sound. ব in the word স্বামী does not have any sound.

Solution: ব in the context of phalaa is coded differently than in the usual context. We are just considering the context of ব phalla before giving the code.

Challenge: Different pronunciation of letters or conjuncts in different contexts. At the beginning of word, ক্ষ is pronounced as খ /k^{h}/. (ক্ষত → খত /k^{h}ɔ̪to/); in the middle or at the end of a word, it is pronounced as কখ /kk^{h}/, (দক্ষ → দকখ /d̪okk^{h}o/)

Solution: If we consider the context of ক্ষ before encoding then this problem is solved too.

From the cases above we understood that we could easily solve these problems using Metaphone encoding approach of giving phonetic code considering the context of letters.

There is still one challenge left, which is multiple pronunciations of same letters in same context. For example, হ with ব: According to Bangla phonological rules, হ should be pronounced as ও or উ and ব should be pronounced as **ভ:** আহ্বান → আওভান /aovan/. However, most native speakers pronounce these words the same way as it is written. For example, আহ্বান is usually pronounced as আহভান /ahob^{h}an/. Both pronunciations are considered correct. We can solve this problem too but we have to use the double metaphone encoding approach of giving multiple codes to words with multiple pronunciation.

Using the approaches of English encoding we can generate a phonetic code for Bangla which represents the pronunciation of a word. The best part is, even though Bangla has so many rules, in most cases these

grammatical rules are consistent which leads to a very successful phonetic encoding for Bangla. So we used the phonetic encoding for Bangla, Double Metaphone for Bangla proposed by UzZaman and Khan [2] and described in detail in [18]. This phonetic encoding handle all the cases described above.

3.2.2.3. Method of generating suggestion for phonetic error. Phonetic encoding is a method to increase the performance of spelling checker but it alone cannot generate suggestions. We need to use the approximate string-matching algorithm to generate the suggestion from this phonetic encoded list. When we have the phonetic encoding then the method of generating suggestion for phonetic error is simpler than it seems. At first we will generate the phonetic codes using [2, 18] of all the words in the word list. Then, instead of looking up the words in the word list, we will use this phonetically encoded word list instead. This way, all the phonetic variations are handled inside the phonetic encoding.

4. Ranking suggestions

Sorting the suggestions according to the relevance of the misspelled word is the most important part of a spelling checker.

4.1. Previous work on ranking suggestions

Levenshtein edit distance algorithm [12] is very efficient for any language to rank the suggestions and even in Bangla so far most of the spelling checkers recommended this method to rank the suggestions.

B.B. Choudhury [4] suggests the edit distance algorithm for ranking the suggestions. Abdullah and Rahman in [5] states the necessity of "highly efficient algorithm" for sorting the suggestions considering the phonetic similarity but they did not discuss their method of solving this problem. In another paper by the same authors Abdullah and Rahman [8] states that they used edit distance in their case to rank the suggestions but they reports the necessity of a "highly efficient algorithm" to consider the phonetic similarity in this paper too.

4.2. Our proposal for ranking suggestions

In this section we propose for a solution that can consider the phonetic similarity to rank the suggestions. At this point we have generated suggestions for our misspelled words, which includes words having edit distance maximum 2 between the misspelling word and words of word list for typographical error and we term this distance as "Typo edit distance". We also have words having edit distance 2 between the phonetic code of misspelling word and the phonetic code of words of word list for phonetic error, we term this distance as "Phonetic edit distance". Now we need to rank the suggestions.

In our case we always prioritize phonetic error than typographical error. To rank we need to consider both the scores but we give a higher weight to the phonetic edit distance so that words with lower phonetic edit distance appear in the higher position in the suggestion list. In our case we give weight of 60 to phonetic edit distance and a weight of 40 to typographical edit distance. Using these we will generate a score, which is our determinant to rank the suggestions.

```
Score = Typo edit distance * Typo weight + Phonetic edit distance * Phonetic weight
```

(3)

Below is the table (Table 1) with all possible values of Score, considering up to the edit distance of 2 for both typographical and phonetic error. It is clearly shown that because of higher weight phonetic edit distance with lower value will always be in the top of the list.

Table 1: Possible scores of suggestion ranking

Typo edit dis	Typo weight	Phonetic edit dis	Phonetic weight	score
0	40	0	60	0
1	40	0	60	40
2	40	0	60	80
0	40	1	60	60
1	40	1	60	100
2	40	1	60	140
0	40	2	60	120
1	40	2	60	160
2	40	2	60	200

5. Performance

In our spelling checker to handle phonetic error we used the phonetic encoding proposed in [2]. This phonetic encoding [2] was used in 1607 commonly misspelled words found in [19] and showed the encoding performance. It generated the encoding [2] of both the correct and misspelled words, and then

compute the edit distance between two phonetic codes. It showed Error if the edit distance between their phonetic codes is not zero. Edit distance 0 means encoding of the two words were same.

Table 2: Encoding performance of [2]

No of words	1607
Edit Distance 0	1473
Error	134
Rate of accuracy	91.67%
Rate of error	8.33%

From the table above (Table 2) we can see that we do not need to consider the typographical errors in 91.67% to get the suggestion. Phonetic encoding is giving the right suggestion for us. And to handle rest of the cases we included up to the edit distance of 2. Now we have to check if these errors fall in this region or not. We have another table in [2] that describes the error distribution of these 8.33% words, which is shown in Table 3.

Table 3: Error distribution

Error	134
Edit Distance 1	107
Edit Distance 2	27

It shows that that words that does not have the same phonetic code with the misspelling word has an edit distance of either 1 or 2 between their phonetic codes. So, after handling the edit distance of 2 we are now including all the possible words in our suggestion list and we are not missing any word. And our ranking scheme ensures to rank according to phonetic relevance because of giving a higher weight to the phonetic edit distance. So we are able to generate the right suggestion and also able to rank them according to phonetic relevance.

6. Evaluation

Kukich [10] lists certain parameters that should be considered during the evaluation of spelling checkers for isolated-word error correction. These are:

- lexicon size,
- test set size,
- correction accuracy for single error misspellings,
- correction accuracy for multi-error misspellings, and
- type of errors handled (phonetic, typographical, OCR generated etc.);

Another paper on Bangla spelling checker [9] also considers these parameters for evaluation of Bangla spelling checkers. We are also considering these parameters to evaluate our spelling checker.

Lexicon size: We need to have an extensive lexicon. Using the morphological parser can reduce this lexicon size, which should be considered in future spelling checker for Bangla.
Test set size: We tested our spelling checker on 1607 words that list the most common misspelling words of Bangla [19].
Correction accuracy for single error misspellings: Phonetic encoding is our part of spelling checker. So the combination of phonetic encoding and single error misspelling can correct 98% of errors for this sample.
Correction accuracy for multi-error misspellings: We used edit-distance for typographical error. We can handle multi-error misspelling if we want to but it become expensive in terms of time. So, we handled up to 2-error misspellings, which lead us to 100% accuracy for this sample. B.B. Chaudhuri [4] notices that more than 70% errors of 15,162,317 words are single and 2-error misspelling. Hence we can be assured that in very large corpus 2-error misspelling will work well.
Type of errors handled (phonetic, typographical, OCR generated etc.): We consider only phonetic and typographical error. In case of OCR generated error, substitution error between similar looking characters (e.g. ‘e’ and ‘c’ or ‘m’ and ‘nn’) will be more common than those between similar sounding characters (e.g. ‘c’ and ‘k’ or ‘f’ and ‘ph’). We have not considered OCR generated errors in our paper.

7. Conclusion

In this paper, we have proposed a comprehensive spelling checker application for Bangla. We discussed the steps of checking the spelling of a word, namely detecting misspelled words, generating suggestions for misspelled words, and ranking the suggestions so that the most likely candidate is placed first. We then discussed the existing solutions and explored their limitations, and proposed a complete spell checking methodology for Bangla. Finally we presented the performance and evaluation of our proposed solution.

8. Acknowledgement

This work has been supported in part by the PAN Localization Project (www.panl10n.net), grant from the International Development Research Center, Ottawa, Canada, administrated through Center for Research in Urdu Language Processing, National University of Computer and Emerging Sciences, Pakistan. We would also like to thank Arnab Zaheen, Naira Khan and other members of our research group.

9. References

[1] Wikipedia page of Bengali language http://en.wikipedia.org/wiki/Bengali.

[2] N. UzZaman and M. Khan, "A Double Metaphone Encoding for Bangla and its Application in Spelling Checker", *Proc. 2005 IEEE Natural Language Processing and Knowledge Engineering, Wuhan, China*, October, 2005.

[3] P. Kundu and B.B. Chaudhuri, "Error Pattern in Bangla Text", *International Journal of Dravidian Linguistics, 28(2)*, 1999.

[4] B.B. Chaudhuri, "Reversed word dictionary and phonetically similar word grouping based spell-checker to Bangla text", *Proc. LESAL Workshop, Mumbai*, 2001.

[5] A.B.A. Abdullah and A. Rahman, "A Different Approach in Spell Checking for South Asian Languages", *Proc. 2nd International Conference on Information Technology for Applications (ICITA), China*, 2004.

[6] M.T. Haque and M. Kaykobad, "Use of Phonetic Similarity for Bangla Spell Checker", *Proc. 5th International, Conference on Computer and Information Technology*, Dhaka, December, 2002, pp. 182-185.

[7] N. UzZaman and M. Khan, "A Bangla Phonetic Encoding for Better Spelling Suggestion", *Proc. 7th International Conference on Computer and Information Technology*, Dhaka, Bangladesh, December, 2004.

[8] A.B.A. Abdullah and A. Rahman, "Spell Checker for Bangla Language: An Implementation Perspective", *Proc. 6th International Conference on Computer and Information Technology*, Dhaka, Bangladesh, 2003.

[9] A. Bhatt, M. Choudhury, S. Sarkar and A. Basu, "Exploring the Limits of Spellcheckers: A comparative Study in Bengali and English", *Proc. The Second Symposium on Indian Morphology, Phonology and Language Engineering (SIMPLE'05)*. Published by CIIL Mysore, Kharagpur, INDIA, February 2005, pp. 60-65.

[10] K. Kukich, "Techniques for automatically correcting words in text", *ACM Computing Surveys, 24 (4)*, pp. 377-439.

[12] F.J. Damerau, "A technique for computer detection and correction of spelling errors", *Communication of ACM, 7(3)*, 1964, pp. 171-176.

[13] Levenshtein edit distance algorithm, available online at http://www.nist.gov/dads/HTML/Levenshtein.html.

[14] Definition of phonetic encoding available online at http://www.nist.gov/dads/HTML/phoneticEncoding.html.

[15] The Soundex Algorithm, available online at http://www.archives.gov/research_room/genealogy/census/soundex.html.

[16] L. Phillips, "Hanging on the Metaphone", *Computer Language, 7(12),* 1990.

[17] Lawrence Philip's Metaphone Algorithm, available online at http://aspell.sourceforge.net/metaphone/index.html.

[18] L. Phillips, "The Double Metaphone Search Algorithm", *C/C++ Users Journal, 18(6)*, June 2000, available online at http://www.cuj.com/documents/s=8038/cuj0006philips/.

[19] N. UzZaman, "Phonetic Encoding for Bangla and its Application to Spelling checker, Name searching, Transliteration and Cross language information retrieval", Undergraduate thesis (Computer Science), BRAC University, May 2005.

[20] K. Alam, *Bangla Banan Obhidhan*, Mirnava, Dhaka, Bangladesh.