A SYSTEM FOR CHECKING SPELLING, SEARCHING NAME & PROVIDING SUGGESTIONS IN BANGLA WORD

MD HAIBUR RAHMAN
Student Id: 012102013

A Thesis

in

The Department

of

Computer Science and Engineering

Presented in Partial Fulfillment of the Requirements

For the Degree of Master of Science in Computer Science and Engineering

United International University

Dhaka, Bangladesh

February, 2018

Approval Certificate

This thesis titled " **Avro for Bangla and its Application to Spelling checker, Transliteration and Name Searching**" submitted by **Md Habibur Rahman**, Student ID: **012102013**, has been accepted as Satisfactory in fulfillment of the requirement for the degree of Master of Science in Computer Science and Engineering on 27.02.2018.

Board of Examiners

1.
_____ Supervisor
Prof. Dr. Mohammad Nurul Huda
Professor & Coordinator - MSCSE
United International University

2.
_____ Head Examiner
Novia Nurain
Assistant Professor, CSE
United International University

3.
_____ Examiner-I
Mohammad Moniruzzaman,
Assistant Professor, CSE
United International University

4.
_____ Examiner-II
Suman Ahmmed,
Assistant Professor, CSE
United International University

5.
_____ Ex-Officio
Swakkhar Shatabda,
Associate Professor
United International University

Declaration

This is to certify that the work entitled "**Avro for Bangla and its Application to Spelling checker, Transliteration and Name Searching** " is the outcome of the research carried out by me under the supervision of Prof. Dr. Mohammad Nurul Huda.

_____

Md Habibur Rahman
Studen ID: 012102013
Department: Computer Science and Engineering

In my capacity as supervisor of the candidate's project, I certify that the above statements are true to the best of my knowledge.

_____

Dr. Mohammad Nurul Huda
Professor & Coordinator - MSCSE

Abstract

This thesis presents an improved phonetic encoding for Bangla which can be used for spelling checking, transliteration, name searching application as well as cross-lingual information retrieval. To produce an appropriate phonetic code for Bangla is always a significant challenge because of the complex and often inconsistent rules of Bangla words. We propose a phonetic encoding technique for Bangla considering the various Context-sensitive rules which includes the large repertoire of conjuncts in Bangla. Here we used Edit Distance Algorithm, Soundex Algorithm and Metaphone Algorithm for our proposed system. After implementation all of the said algorithms, we will get our targeted word within shortest possible time.

ACKNOWLEDGMENTS

At the very outset, I would like to express my deep gratitude to Almighty Allah who gave me enough knowledge and patience to complete the thesis within the stipulated time.

I would like to give special thanks my supervisor Prof. Dr. Mohammad Nurul Huda for his precious as well as constructive suggestions during the planning and development of this research work.

I would also like to extend my thanks to the concerned faculty member of United International University for their kind support throughout the research work.

Finally, I wish to thank my parents for their encouragement through my study.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

Chapter I:
INTRODUCTION

Bangla is one of the widely spoken languages, especially in the Indian Subcontinent. The Bengali spelling rules are very complex in nature. One of the basic reasons for this is its consonant clusters or juktakkhors. Some other notable reasons for its complexity are phonetic similarity of the characters, the difference between the rapheme representation and the phonetic utterances etc.

Phonetic encoding for Bangla is always a great challenge for its complex nature of letters or words. The first encoding for Bangla was based on Soundex method which was not able to handle the complexity of Bangla spelling rules.

In this particular thesis paper, we will describe phonetic encoding elaborately in Chapter II. Then, we will discuss the scope and importance of our encoding as well as the limitations of other encoding in Chapter III. After that, we will propose our encoding with reasoning in Chapter IV. Next, we will explain the methodology of our new application for Bangla in details in Chapter V. Finally, we will summarize how our new system would perform better than the existing systems.

CHAPTER II:
PHONETIC ENCODING

2.1. Definition

Based on the pronunciation of string, code is done. The input of a phonetic encoding algorithm is a word and the result is an encoded key that should be same for all words which are pronounced similarly that allows for a reasonable amount of fuzziness.

As for instance, metaphone encoding gives the code RLS for the word analyze in English. It is known that analyze and analise have the same pronunciation. Hence, a good encoding in English should be able to give the same code RLS to analyze as well.

2.2 Phonetic Encoding for English

Various types of approximate string matching algorithms are Soundex, Metaphone, Double metaphone and PHONIX in English. These phonetic matching algorithms make partition the consonants by phonetic similarity then use a single key to encode each set. Only the first few consonant sounds are encoded unless the first letter is a vowel for the said algorithms.

2.3 Soundex

Soundex partitions the set of letters into 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 are considered to be silent and is not considered during encoding. The Soundex codes are shown in Table 1: Soundex encoding table. The Soundex algorithm transforms all but the first letter of each string into the code, then truncates the result to be at most four characters long. Zeros are added at the end if necessary to produce a four-character code. For example,Washington is coded W-252 (W, 2 for the S, 5 for the N, 2 for the G, remaining letters disregarded), and Lee is coded L-000 (L, 000 added).

Soundex deals with small table size and works based on letter-by-letter algorithm. As a result, it is more speedy than other phonetic methods.

Table 1: Soundex encoding table

Code	Letters
0(not coded)	**A, E, I, O, U, H, W, Y**
1	**B, F, P, V**
2	**C, G, J, K, Q, S, X, Z**
3	**D, T**
4	**L**
5	**M, N**
6	**R**

1. Capitalize all letters in the word and drop all punctuation marks. Pad the word with Right most blanks as needed during each procedure step.

2. Retain the first letter of the word.

3. Change all occurrence of the following letters to '0' (zero): 'A', E', 'I', 'O', 'U', 'H', 'W', 'Y'.

4. Change letters from the following sets into the digit given:

- 1 = 'B', 'F', 'P', 'V'
- 2 = 'C', 'G', 'J', 'K', 'Q', 'S', 'X', 'Z'
- 3 = 'D','T'
- 4 = 'L'
- 5 = 'M','N'
- 6 = 'R'

5. Remove all pairs of digits which occur beside each other from the string that resulted after step (4).
6. Remove all zeros from the string that results from step 5.0 (placed there in step 3)
7. Pad the string that resulted from step (6) with trailing zeros and return only the first Four positions, which will be of the form <uppercase letter> <digit> <digit> <digit>.

Figure 1: The Soundex algorithm

2.4 Metaphone

The Metaphone algorithm analyzes both single consonants and groups of letters called diphthongs based on a set of rules for grouping consonants, then mapping groups to Metaphone codes.

The Metaphone Rules

Metaphone reduces the alphabet to 16 consonant sounds:

B X S K J T F H L M N P R 0 W Y

That isn't an O but a zero - representing the 'th' sound.

Transformations

Metaphone uses the following transformation rules:

Doubled letters except "c" -> drop 2nd letter.

Vowels are only kept when they are the first letter.

B -> B unless at the end of a word after "m" as in “dumb"

C -> X (sh) if -cia- or -ch-

S if -ci-, -ce- or -cy-

K otherwise, including -sch-

D -> J if in -dge-, -dgy- or -dgi-

T otherwise

F -> F

G -> silent if in -gh- and not at end or before a vowel
in -gn- or -gned- (also see dge etc. above)
J if before i or e or y if not double gg
K otherwise
H -> silent if after vowel and no vowel follows
H otherwise

J -> J
K -> silent if after "c"
K otherwise
L -> L

M -> M

N -> N
P -> F if before "h"
P otherwise
Q -> K

R -> R

S -> X (sh) if before "h" or in -sio- or -sia-
S otherwise

T -> X (sh) if -tia- or -tio-
0 (th) if before "h"
silent if in -tch-
T otherwise
V -> F

W -> silent if not followed by a vowel
W if followed by a vowel
X -> KS

Y -> silent if not followed by a vowel

Y if followed by a vowel

Z -> S

Initial Letter Exceptions

Initial kn-, gn- pn, ac- or wr- -> drop first letter

Initial x- -> change to "s"

Initial wh- -> change to "w"

CHAPTER III:
PROPOSED ENCODING

We needed to keep few things in our mind while proposing this encoding. We particularly considered the phonetic similarity of letters to give them the same code and also to keep in mind the orthographic or spelling rules as well as to know how letters spell in different context so that we can encode the letters with similar sounding letters considering the context.Using this encoding, anyone would be able to work as an intermediate code in multi-lingual applications. We will be encoding our Bangla letters to a set of Latin alphabets so that it can easily work as an intermediate language to work with English.

We assume that the Bangla text is encoded using Unicode Normalization Form C (NFC).

3.1 Proposed phonetic encoding for words

We will have two encoding- mainly one for words and a few variations from it for names as well. This section describes about the words encoding. Throughout the thesis paper, we termed our proposed phonetic encoding by Avro phonetic encoding or proposed phonetic encoding. In order to encode Bangla words, we need to consider context and also need to generate multiple codes for the same string. These constraints can be handled in Edit Distance, Soundex and metaphone algorithm, which we did for Bangla here. That's why, we termed it as metaphone phonetic encoding.

3.2 Phonetic Encoding

Following Table 2: Phonetic Encoding table for words is the table of proposed Avro phonetic encoding for words. Followed by the table, there will be reasoning of each of the encoding.

Table 2: Phonetic Encoding Table

Letter	Name	ASCII Code
O	অ	2437
A	আ	2438

I	ই	2439
I	ঈ	2440
U	উ	2441
U	ঊ	2442
rri	ঋ	2443
E	এ	2447
OI	ঐ	2448
O	ও	2451
OU	ঔ	2452
K	ক	2453
kh	খ	2454
G	গ	2455
ghgh	ঘ	2456
ng	ঙ	2457
C	চ	2458
ch	ছ	2459
J	জ	2460
jh	ঝ	2461
NG	ঞ	2462
T	ট	2463
Th	ঠ	2464
D	ড	2465
Dh	ঢ	2466
N	ণ	2467
T	ত	2468
th	থ	2469
D	দ	2470
dh	ধ	2471
N	ন	2472
P	প	2474
ph	ফ	2475
B	ব	2476
bh	ভ	2477
m	ম	2478
Z	য	2479
R	র	2480
L	ল	2482
sh	শ	2486
S	ষ	2487

s	স	2489
a	া	2494
i	ি	2495
I	ী	2496
u	ু	2497
U	ূ	2498
e	ে	2503
OI	ৈ	2504
O	ো	2507
OU	ৌ	2508
hs	্	2509
TH	ৎ	2510
R	ড়	2524
Rh	ঢ়	2525
Y	য়	2527
o	ঁ	2433
ng	ং	2434

3.3 Existing Phonetic Encoding for Bangla

Eighty years old technique of phonetic encoding is new in Bangla which was first proposed by Hoque and Kaykobad in 2002. Then Zaman and Khan, 2004, proposed their version of soundex type Bangla phonetic encoding. Both of the encoding use “soundex” in their encoding name. The cause behind it is they follow the general principal of soundex encoding, to partition the letters in to disjoint sets.

CHAPTER IV:
APPLICATIONS OF PHONETIC ENCODING

Without being properly used in applications, phonetic encoding would not be able to play significant role for a language. Name searching was first such application in which phonetic encoding was used after that spelling checker adopts this phonetic encoding technique.

We have used our phonetic encoding in many applications like spelling checker, transliteration, cross-lingual information retrieval and name searching for Bangla. In every case, we will first show how that application were developed earlier, how they perform and then how phonetic encoding improves its performance.

4.1 Translation using Direct Mapping

Some software exactly uses this mapping. We are giving a mapping, which we used for our direct mapping transliteration. Since this direct mapping is still a phonetic mapping, the difference is, it will not look up in the dictionary if it has any word with same pronunciation. We have introduced an intermediate encoding which will be used to encode before converting. We need it because in some cases it should not be converted directly, like bool pronounce as bul, hence before mapping we convert “oo” to “u”. One more thing is we will not only consider one letter for one to one mapping, we may sometime consider bigrams for mapping. Because, to represent some Bangla letters phonetically in English we use those bigrams. Like for Bangla letter খ/kh/ we use kh.

Table 3: Table for direct mapping

case (char)2433: engText.Append("o"); break; //chandra-bindu

(char)2434: engText.Append("ng"); break; //onesh-kar

case (char)2435: /*engText.Append(":");*/ break; //khandata

case (char)2437: engText.Append("o"); break; //'অ'

case (char)2438: engText.Append("a"); break; //'আ'

```
case (char)2439: engText.Append("e"); break;      //'ই'
case (char)2440: engText.Append("E"); break;      //'ঈ'
case (char)2441: engText.Append("u"); break;      //'উ'
case (char)2442: engText.Append("U"); break;      //'ঊ'
case (char)2443: engText.Append("rri"); break;    //'ঋ'
case (char)2447: engText.Append("e"); break;      //'এ'
case (char)2448: engText.Append("OI"); break;     //'ঐ'
case (char)2451: engText.Append("O"); break;      //'ও'
case (char)2452: engText.Append("OU"); break;     //'ঔ'
case (char)2453: engText.Append("k");/*engText.Append("ko");*/break;              //'ক'
case (char)2454: engText.Append("kh");/*engText.Append("kha");*/break;            //'খ'
case char)2455: engText.Append("g");/*engText.Append("go");engText.Append("G");*/break;  //'গ'
case (char)2456: engText.Append("GH"); break;                                    //'ঘ'
case (char)2457: engText.Append("N");/*engText.Append("g");*/break;     //'ঙ'

case (char)2458: engText.Append("c");/*engText.Append("co");*/break;    //'চ'
case (char)2459: engText.Append("ch");/*engText.Append("CH");*/break;   //'ছ'
case (char)2460: engText.Append("j");/*engText.Append("jo");*/break;    //'জ'
case (char)2461: engText.Append("jh"); break;                           //'ঝ'
case (char)2462: /*engText.Append("n");*/ break;                        //nio

case (char)2463: engText.Append("T");/*engText.Append("To");*/break;    //'ট'
case (char)2464: engText.Append("Th");/*engText.Append("TH");*/break;   //'ঠ'
case (char)2465: engText.Append("D");/*engText.Append("Do");*/break;    //'ড'
case (char)2466: engText.Append("Dh");/*engText.Append("DH");*/break;   //'ঢ'
case (char)2467: engText.Append("N");/*engText.Append("No");*/break;    //'ণ'

case (char)2468: engText.Append("t");/*engText.Append("to");*/break;    //'ত'
case (char)2469: engText.Append("th");/*engText.Append("tho");*/break;  //'থ'
case (char)2470: engText.Append("d");/*engText.Append("do");*/break;    //'দ'
case (char)2471: engText.Append("dh"); break;                           //'ধ'
case (char)2472: engText.Append("n");/*engText.Append("no");*/break;    //'ন'
```

```
case (char)2474: engText.Append("p");/*engText.Append("po");*/break;    //'প'
case (char)2475: engText.Append("ph");/*engText.Append("f");*/break;    //'ফ'
case (char)2476: engText.Append("b");/*engText.Append("bo");*/break;    //'ব'
case (char)2477: engText.Append("bh");/*engText.Append("BH");engText.Append("v");*/break;   //'ভ'
case (char)2478: engText.Append("m");/*engText.Append("mo");*/break;    //'ম'

case (char)2479: engText.Append("z"); break;                            //'য'
case (char)2480: engText.Append("r");/*engText.Append("ro");*/break;    //'র'
case (char)2482: engText.Append("L");/*engText.Append("Lo");*/break;    //'ল'
case (char)2486: engText.Append("sh");/*engText.Append("S");*/break;    //'শ'
case (char)2487: engText.Append("S");/*engText.Append("h");*/break;     //'ষ'
case (char)2488: engText.Append("s");/*engText.Append("so");*/break;    //'স'
case (char)2489: engText.Append("h");/*engText.Append("ho");*/break;    //'হ'

case (char)2494: engText.Append("a"); break;    // a-kar
case (char)2495: engText.Append("i"); break;    // rossi-kar
case (char)2496: engText.Append("I"); break;    // dirghi-kar
case (char)2497: engText.Append("u"); break;    // rossu-kar
case (char)2498: engText.Append("U"); break;    // dighu-kar

case (char)2503: engText.Append("e"); break;    //a-kar
case (char)2504: engText.Append("OI"); break;    //oi-kar
case (char)2507: engText.Append("O"); break;    //o-kar
case (char)2508: engText.Append(","); break;    //oaau-kar
case (char)2509: /*engText.Append("OU");*/break;    //hosonta
case (char)2510: engText.Append("t"); break;    //khandata

case (char)2524: engText.Append("R");/*engText.Append("Ro");*/break;    //'ড়'
case (char)2525: engText.Append("Rh"); break;                           //'ঢ়'
case (char)2527: engText.Append("Y");/*engText.Append("Yo");*/break;    //'য়'
case 'ক্ষ': engText.Append("kkh");break;
```

4.2 Phonetic mapping

In phonetic mapping, the basic idea is to check in the dictionary if we have the word with same pronunciation. Following is the algorithm of phonetic mapping.

Algorithm of phonetic mapping

```
if there is a word with the same pronunciation
in the dictionary
    then convert it to that word
else if there are multiple words with the same
pronunciation in the dictionary
     then give suggestions for that word and
the user will select which one to use
else if there are not words with the same
pronunciation in the dictionary
  then convert it using direct mapping
```

Now our main challenge is how we can get the pronunciation of a Bangla word to check it with an English word and understand it has the same pronunciation. We have used the phonetic encoding for Bangla proposed in section 4.1. That encoding encodes Bangla word in to an English word that represents the pronunciation of a word. So, our only challenge is to convert the English words in the same manner so that both encoding are consistent. For example, is encoded in to klm.

4.3 Spelling Checker

Spelling Checker may be used for various applications like Optical Character Recognition (OCR), Machine Translation (MT), Natural Language Processing (NLP and so on.

4.4 Spelling error patterns

There are two types of word-error such as non-word error and real-word error. Again, there are two types of errors which are typographical error phonetic error. Typographical errors may occur because of typing mistakes, negligence, lack of concentrations and may

be for any other reasons. Phonetic errors may be happened because of not knowing the spelling of a desired word although the user knows the pronunciation of the word.
In non-word errors, there are mainly two types of errors. One is typographical error and another is phonetic error. Description of typographical error is as follows.
In an early study, found that 80% of all misspelled words (non-words errors) in a sample of human keypunched text were caused by single error misspellings: a single one of the following errors:

- Substitution error: mistyping the as ther
- Deletion error: mistyping the as th
- Insertion error: mistyping the as thw
- Transposition error: mistyping the as hte

These are the type of typographical errors, which occurred due to typing mistakes, negligence, and lack of concentrations or other reasons. But if computer gives a red underline into the word, then we can easily correct it without seeing the spelling suggestions.

But scenarios of phonetic errors are different. Phonetic errors occur when the user do not know the spelling of a desired word but knows the pronunciation of the word. So, using the pronunciation the user may write a word but in suggestion it is impossible to get the desired word in case of Bangla, because of complex Bangla rules.

4.5 Approximate string matching algorithm

In our Thesis we use Levenshtein Edit Distance method for approximate string-matching algorithm. The algorithm use to check the closeness of dictionary words with the misspelled word. It gives suggestion that is closed to misspelled word.

Levenshtein Edit Distance:

Definition:

The edit distance algorithm is similarity of two strings, s1 and s2, is defined as the minimum number of point mutations required to change s1 into s2, where a point mutation is one of

- Insert Letter
- Delete Letter
- Replace Letter
- Transpose Letter

Levenshtein Edit Distance algorithm used various reporting purpose like Spell checking, Speech recognition, DNA analysis and Plagiarism detection.

Example:

e (“kitten”, “sitting”) = 3
Kitten sitten (substitution of “k” with “s”),
Sitten sittin (substitution of “e” with “I”),
Sittin sitting (insert “g” at the end),

For example, we assume our lexicon consist of following words.

□□□, □□□, □□□, □□□□

Our misspelled word is কল. Now when we check the lexicon dictionary we find that there are no such word কল. So, it is a misspelled word according to this dictionary. Now to generate and rank the suggestion, we will generate the edit-distance with all the words of the dictionary.

4.6 How to Rank Performance of Encoding

To rank the suggestion, we used both phonetic edit distance, which is edit distance between phonetic codes, and normal edit distance. We did not use the average of both, but preferred for a weighted average. For example, our

score = a * phonetic_edit_distance + (1-a) * normal_edit_distance
where, a > (1-a).

We rank the suggestions according to the scored achieved for a word.

Table 4: Example of Edit distance

Dictionary Word	Edit Distance with Word
□□□	2
□□□	2
□□□	1
□□□□	3

Hence, our ranked suggestion for কল will be কলা, কাক, কথা, মালা

4.7 Performance of our proposed encoding

In our Lexicon Dictionary we have 110750 words for suggestions. Our proposed Encoding performance shows the performance when it used on 1607 commonly misspelled words. Firstly, we apply our encoding to both the correct and misspelled words, after complete the encoding of both word we use Edit Distance Algorithm for minimum distance measure. After implement the Edit Distance algorithm we have found few words which is lowest minimum distance (like=1). The words which is found from Edit Distance Algorithm at this stage will be implemented this words in Soundex Algorithm. The number of words will be reduced after the implementation of Soundex Algorithm. That means, the words which have less Edit Distance will be brought using Soundex Algorithm. Later on, the words found using Soundex Algorithm will be used in Metaphone Algorithm. After implementation of Metaphone Algorithm, we will get our targeted word it is considered correct if the edit distance is 0. In our case 130 out of 1607 words do not produce an edit distance of 0 with the correct word, which are termed as error, resulting in an accuracy of 91.91%.

Table 5: Performance of Encoding

No of Word	**1607**
Correct (Edit Distance 0)	1477
Error	130
Rate of Accuracy	91.91%
Rate of Error	8.08%

The numbers of unmatched words fall to 107 and 23 if we consider edit distances of 1 and 2 respectively, as shown in Table 6.

Table 6: distribution of Error

Error	130
Edit Distance =1	107
Edit Distance =2	23

After complete of our proposed technique we have got some suggestion list of words. It show words suggestion which have Edit Distance >=2. So, we can always get our expected words in suggestion list and more than 91.91% time's word at the top of the suggestion list.

4.8 Example of transliteration

ami bhal achi. Tomar khbor ki? Ajke shndha bela tumi ki Kroch. obak bepar hl, ami ekhon bangla likhte pari English diye. ar mjar bepar hl ami dui vhabe likhte pari. ek`ta daireckT arekta phnetik. Tmar desh e koto taka te Dlar. Ami abar jukt brn likhte pari.

Output in direct mapping will be following.

আমি ভালো আছি। তোমার খবর কি। আজকে সন্ধ্যা বেলা তুমি কি করছ। অবাক ব্যপার হল , আমি এখন বাংলা লিখতে পারি। অভ্র দিয়া। আর মজার বেপার হল আমি দুই ভাবে লিখতে পারি। একটা ডাইরেক্ট আরেকটা ফনেটিক। তোমার দেশ এ কত টাকা তে ডলার। আমি এই ভাবে আবার জক্ত বর্ন লিখতে পারি।

Bangle Text:

আমি ভালো আছি। তোমার খবর কি। আজকে সন্ধ্যা বেলা তুমি কি করছ। অবাক ব্যপার হল , আমি এখন বাংলা লিখতে পারি। অভ্র দিয়া। আর মজার বেপার হল আমি দুই ভাবে লিখতে পারি। একটা ডাইরেক্ট আরেকটা ফনেটিক। তোমার দেশ এ কত টাকা তে ডলার। আমি এই ভাবে আবার জুক্ত বর্ন লিখতে পারি।

Table 7: Proposed Name searching for Bangla using direct mapping

```
private void ShowOutput(List<string> matches, string code, bool isShowMessageBox ,
System.Windows.Forms.RichTextBox rtb)
        {
            if(isShowMessageBox)
            {
                StringBuilder builder = new StringBuilder();
                builder.Append("Searching for:\r\n");
                builder.AppendFormat("{0} ({1})\r\n\r\n", txtFind.Text, code);
                if (matches.Count > 0)
                {
                    builder.AppendFormat("Matches found ({0}):\r\n",
matches.Count);
                    foreach (string match in matches)
                    {
                        builder.AppendFormat("{0}\r\n", match);
                    }
                }
                else
                    builder.Append("No matches found");

                MessageBox.Show(builder.ToString());
            }
            else
            {
                StringBuilder builder = new StringBuilder();
                builder.Append("Searching for:\r\n");
                builder.AppendFormat("{0} ({1})\r\n\r\n", txtFind.Text, code);
                if (matches.Count > 0)
                {
                    builder.AppendFormat("Matches found ({0}):\r\n",
                        matches.Count);
                    foreach (string match in matches)
                    {
                        builder.AppendFormat("{0}\r\n", match);
                    }
                }
                else
                {
                    builder.Append("No matches found");
                }
                rtb.Text = builder.ToString();
            }
        }
        #endregion
```

The transformation or rules described in Table 6: Proposed Name searching for Bangla that derived names from the Dictionary using direct mapping, if the inputted word is exists it show the word is match. If the inputted word not exist in Dictionary then it show the word not found in Dictionary.

4.9 Code for Name searching using Dictionary

```
public partial class FormMeasurement : Form
   {
      private static SpellCheck _Dictionary;

      public FormMeasurement()
      {
         InitializeComponent();
      }

      #region Events
      private void buttonBrowse_Click(object sender, EventArgs e)
      {
         textBoxDictionaryPath.ReadOnly = false;
         openFileDialog.InitialDirectory =
System.IO.Path.GetFullPath(@"..\..\Dictionary");
         openFileDialog.Title = "Browse Text Files";
         openFileDialog.CheckFileExists = true;
         openFileDialog.CheckPathExists = true;
         openFileDialog.DefaultExt = "txt";
         openFileDialog.Filter = "Text files (*.txt)|*.txt|All files (*.*)|*.*";
         openFileDialog.FilterIndex = 2;
         openFileDialog.RestoreDirectory = true;
         openFileDialog.ReadOnlyChecked = true;
         openFileDialog.ShowReadOnly = true;
         if (openFileDialog.ShowDialog() == DialogResult.OK)
         {
            textBoxDictionaryPath.Text = openFileDialog.FileName;
            textBoxDictionaryPath.ReadOnly = true;
         }
      }
      private void btnSearch_Click(object sender, EventArgs e)
      {
         listViewSuggestionList.Items.Clear();
         _Dictionary = new SpellCheck(File.ReadAllText(textBoxDictionaryPath.Text),
textBoxDictionaryPath.Text.Contains("BD") ? true : false);
         string source = txtFind.Text;

         #region Edit Distance
         List<string> suggestions = _Dictionary.Correct(source);

         ListViewItem item;
```

```
            foreach (string targetString in suggestions)
            {
                int distance = EditDistance.Compare(source.ToLower(),
targetString.ToLower());
                item = new ListViewItem(targetString);
                item.SubItems.Add(distance.ToString());
                listViewSuggestionList.Items.Add(item);
            }
            #endregion

            #region Soundex
            string[] names = suggestions.ToArray();

            // List to hold matches
            List<string> matches = new List<string>();
            string code = SearchSoundex(txtFind.Text, names, matches);
            ShowOutput(matches, code, false, richTextBoxSoundex);
            #endregion

            #region Metaphone
            // List to hold matches
            matches = new List<string>();
            code = SearchMetaphone(txtFind.Text, names, matches);
            ShowOutput(matches, code, false, richTextBoxMetaphone);
            #endregion

            ShowOutput(matches, code, true, null);
        }
        #endregion

        #region Soundex
        private string SearchSoundex(string find, string[] names, List<string> matches)
        {
            find = ConvertSoundex(find);
            // Encode string we want to find
            string code = Soundex.Encode(find);

            // Search through the list of names
            foreach (string name in names)
            {
                string soundex_name = ConvertSoundex(name);
                // Compare against soundex-encoded version of name
                if (Soundex.Encode(soundex_name) == code)
```

```
            {
               // Found a match--add it to list
               //matches.Add(soundex_name);
               matches.Add(name);
            }
         }
         return code;
      }
      private string ConvertSoundex(string text)
      {
         StringBuilder engText = new StringBuilder();
         for (int index = 0; index < text.Length; index++)
         {
            switch (text[index])
            {
               case (char)2433: engText.Append("o"); break;    //chandra-bindu
               case (char)2434: engText.Append("ng"); break;    //onesh-kar
               case (char)2435: /*engText.Append(":");*/ break;    //khandata
               case (char)2437: engText.Append("o"); break;    //'অ'
               case (char)2438: engText.Append("a"); break;      //'আ'
               case (char)2439: engText.Append("e"); break;      //'ই'
               case (char)2440: engText.Append("E"); break;      //'ঈ'
               case (char)2441: engText.Append("u"); break;      //'উ'
               case (char)2442: engText.Append("U"); break;      //'ঊ'
               case (char)2443: engText.Append("rri"); break;    //'ঋ'
               case (char)2447: engText.Append("e"); break;      //'এ'
               case (char)2448: engText.Append("OI"); break;     //'ঐ'
               case (char)2451: engText.Append("O"); break;      //'ও'
               case (char)2452: engText.Append("OU"); break;     //'ঔ'
case (char)2453: engText.Append("k");/*engText.Append("ko");*/break;              //'ক'
case (char)2454: engText.Append("kh");/*engText.Append("kha");*/break;            //'খ'
case (char)2455: engText.Append("g");/*engText.Append("go");engText.Append("G");
         */break;    //'গ'
case (char)2456: engText.Append("GH"); break;                        //'ঘ'
case (char)2457: engText.Append("N");/*engText.Append("g");*/break;     //'ঙ'
case (char)2458: engText.Append("c");/*engText.Append("co");*/break;    //'চ'
case (char)2459: engText.Append("ch");/*engText.Append("CH");*/break;   //'ছ'
case (char)2460: engText.Append("j");/*engText.Append("jo");*/break;    //'জ'
case (char)2461: engText.Append("jh"); break;                        //'ঝ'
```

```
case (char)2462: /*engText.Append("n");*/ break;                    //nio
case (char)2463: engText.Append("T");/*engText.Append("To");*/break;   //'ট'
case (char)2464: engText.Append("Th");/*engText.Append("TH");*/break;  //'ঠ'
case (char)2465: engText.Append("D");/*engText.Append("Do");*/break;   //'ড'
case (char)2466: engText.Append("Dh");/*engText.Append("DH");*/break;  //'ঢ'
case (char)2467: engText.Append("N");/*engText.Append("No");*/break;   //'ণ'
case (char)2468: engText.Append("t");/*engText.Append("to");*/break;   //'ত'
case (char)2469: engText.Append("th");/*engText.Append("tho");*/break; //'থ'
case (char)2470: engText.Append("d");/*engText.Append("do");*/break;   //'দ'
case (char)2471: engText.Append("dh"); break;                        //'ধ'
case (char)2472: engText.Append("n");/*engText.Append("no");*/break;   //'ন'
case (char)2474: engText.Append("p");/*engText.Append("po");*/break;   //'প'
case (char)2475: engText.Append("ph");/*engText.Append("f");*/break;   //'ফ'
case (char)2476: engText.Append("b");/*engText.Append("bo");*/break;   //'ব'
case (char)2477:
engText.Append("bh");/*engText.Append("BH");engText.Append("v");*/break;  //'ভ'
case (char)2478: engText.Append("m");/*engText.Append("mo");*/break;   //'ম'
case (char)2479: engText.Append("z"); break;                         //'য'
case (char)2480: engText.Append("r");/*engText.Append("ro");*/break;   //'র'
case (char)2482: engText.Append("L");/*engText.Append("Lo");*/break;   //'ল'
case (char)2486: engText.Append("sh");/*engText.Append("S");*/break;   //'শ'
case (char)2487: engText.Append("S");/*engText.Append("h");*/break;    //'ষ'
case (char)2488: engText.Append("s");/*engText.Append("so");*/break;   //'স'
case (char)2489: engText.Append("h");/*engText.Append("ho");*/break;   //'হ'
case (char)2494: engText.Append("a"); break;    // a-kar
case (char)2495: engText.Append("i"); break;    // rossi-kar
case (char)2496: engText.Append("I"); break;    // dirghi-kar
case (char)2497: engText.Append("u"); break;    // rossu-kar
case (char)2498: engText.Append("U"); break;    // dighu-kar
case (char)2503: engText.Append("e"); break;    //a-kar
case (char)2504: engText.Append("OI"); break;    //oi-kar
case (char)2507: engText.Append("O"); break;    //o-kar
case (char)2508: engText.Append(","); break;    //oaau-kar
case (char)2509: /*engText.Append("OU");*/break;    //hosonta
case (char)2510: engText.Append("t"); break;    //khandata
case (char)2524: engText.Append("R");/*engText.Append("Ro");*/break;   //'ড়'
case (char)2525: engText.Append("Rh"); break;                        //'ঢ়'
case (char)2527: engText.Append("Y");/*engText.Append("Yo");*/break;   //'য়'
```

```
engText.Append("kkh");break;
            }
        }
        return engText.ToString();
    }
    #endregion

    #region Metaphone
    private string SearchMetaphone(string find, string[] names, List<string> matches)
    {
        find = ConvertMetaphone(find);
        // Encode string we want to find
        Metaphone metaphone = new Metaphone();
        string code = metaphone.Encode(find);

        // Search through the list of names
        foreach (string name in names)
        {
            string metaphone_name = ConvertMetaphone(name);
            // Compare against soundex-encoded version of name
            if (metaphone.Encode(metaphone_name) == code)
            {
                // Found a match--add it to list
                //matches.Add(metaphone_name);
                matches.Add(name);
            }
        }
        return code;
    }
    private string ConvertMetaphone(string text)
    {
        StringBuilder engText = new StringBuilder();
        for (int index = 0; index < text.Length; index++)
        {
            switch (text[index])
            {
                case (char)2437: engText.Append("o"); break;    //'অ'
                case (char)2451: engText.Append("o"); break;    //'ও'
                case (char)2438: engText.Append("a"); break;    //'আ'
                case (char)2494: engText.Append("a"); break;    // a-kar
                case (char)2439: engText.Append("i"); break;    //'ই'
                case (char)2440: engText.Append("i"); break;    //'ঈ'
```

```
case (char)2495: engText.Append("i"); break;    // rossi-kar
case (char)2496: engText.Append("i"); break;    // dirghi-kar
case (char)2441: engText.Append("u"); break;    //'উ'
case (char)2442: engText.Append("u"); break;    //'ঊ'
case (char)2497: engText.Append("u"); break;    // rossu-kar
case (char)2498: engText.Append("u"); break;    // dighu-kar
case (char)2447: engText.Append("e"); break;    //'এ'
case (char)2503: engText.Append("e"); break;    //a-kar
case (char)2448: engText.Append("oi"); break;   //'ঐ'
case (char)2504: engText.Append("oi"); break;   //oi-kar
case (char)2452: engText.Append("ou"); break;   //'ঔ'
case (char)2508: engText.Append("ou"); break;   //oaau-kar
case (char)2453: engText.Append("k"); break;    //'ক'
case (char)2454: engText.Append("k"); break;    //'খ'
//case 'ক্ষ': engText.Append("k"); break;
case (char)2455: engText.Append("g"); break;    //'গ'
case (char)2456: engText.Append("g"); break;    //'ঘ'
case (char)2457: engText.Append("ng"); break;    //'ঙ'
case (char)2434: engText.Append("ng"); break;   //onesh-kar
case (char)2458: engText.Append("c"); break;    //'চ'
case (char)2459: engText.Append("c"); break;    //'ছ'
case (char)2460: engText.Append("j"); break;    //'জ'
case (char)2461: engText.Append("j"); break;    //'ঝ'
case (char)2479: engText.Append("j"); break;    //'য'
case (char)4444: engText.Append("e"); break;    //'য'-phola
case (char)2462: engText.Append("n"); break;    //nio
case (char)2463: engText.Append("T"); break;    //'ট'
case (char)2464: engText.Append("T"); break;    //'ঠ'
case (char)2465: engText.Append("D"); break;    //'ড'
case (char)2466: engText.Append("D"); break;    //'ঢ'
case (char)2443: engText.Append("ri"); break;   //'ঋ'
case (char)2480: engText.Append("r"); break;    //'র'
case (char)2524: engText.Append("r"); break;    //'ড়'
case (char)2525: engText.Append("r"); break;    //'ঢ়'
case (char)2472: engText.Append("n"); break;    //'ন'
case (char)2467: engText.Append("n"); break;    //'ণ'
case (char)2468: engText.Append("t"); break;    //'ত'
```

```
                case (char)2469: engText.Append("t"); break;     //'থ'
                case (char)2470: engText.Append("d"); break;     //'দ'
                case (char)2471: engText.Append("d"); break;   //'ধ'
                case (char)2474: engText.Append("p"); break;    //'প'
                case (char)2475: engText.Append("p"); break;    //'ফ'
                case (char)2476: engText.Append("b"); break;    //'ব'
                case (char)2477: engText.Append("b"); break;   //'ভ'
                case (char)2478: engText.Append("m"); break;    //'ম'
                case (char)2527: engText.Append("y"); break;    //'য়'
                case (char)2482: engText.Append("l"); break;    //'ল'
                case (char)2486: engText.Append("s"); break;    //'শ'
                case (char)2487: engText.Append("s"); break;     //'ষ'
                case (char)2488: engText.Append("s"); break;    //'স'
                case (char)2489: engText.Append("h"); break;    //'হ'
                case (char)58: engText.Append("h"); break;    //bisarga
                case (char)2433: engText.Append("o"); break;    //chandra-bindu
               //case (char)2507: engText.Append("O"); break;    //o-kar
               //case (char)2509: /*engText.Append("OU");*/break;    //hosonta
            }
        }
        return engText.ToString();
    }
    #endregion

    #region Private Method
    private void ShowOutput(List<string> matches, string code, bool
isShowMessageBox , System.Windows.Forms.RichTextBox rtb)
    {
        if(isShowMessageBox)
        {
            StringBuilder builder = new StringBuilder();
            builder.Append("Searching for:\r\n");
            builder.AppendFormat("{0} ({1})\r\n\r\n", txtFind.Text, code);
            if (matches.Count > 0)
            {
                builder.AppendFormat("Matches found ({0}):\r\n", matches.Count);
                foreach (string match in matches)
                {
                    builder.AppendFormat("{0}\r\n", match);
                }
```

```
        }
        else
            builder.Append("No matches found");

        MessageBox.Show(builder.ToString());
    }
    else
    {
        StringBuilder builder = new StringBuilder();
        builder.Append("Searching for:\r\n");
        builder.AppendFormat("{0} ({1})\r\n\r\n", txtFind.Text, code);
        if (matches.Count > 0)
        {
            builder.AppendFormat("Matches found ({0}):\r\n",
                matches.Count);
            foreach (string match in matches)
            {
                builder.AppendFormat("{0}\r\n", match);
            }
        }
        else
        {
            builder.Append("No matches found");
        }
        rtb.Text = builder.ToString();
    }
}
#endregion
```

4.10 Proposed Technique:

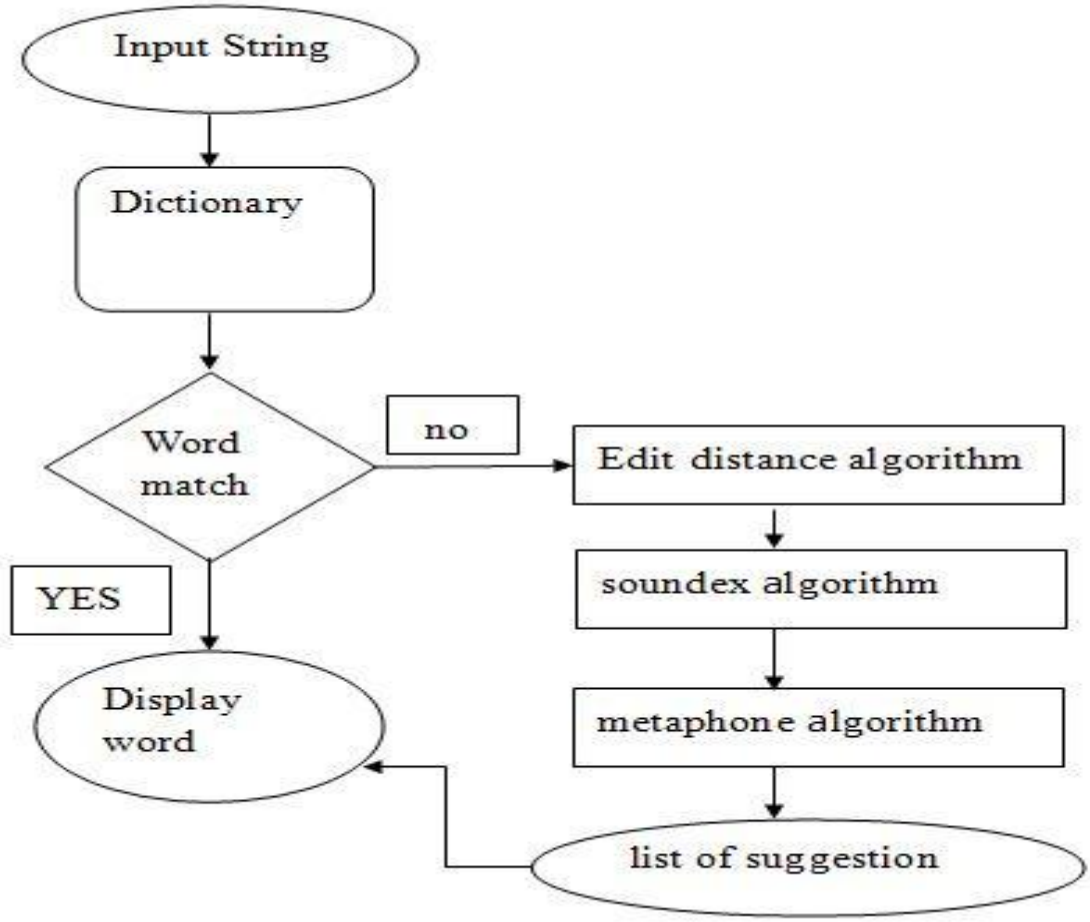

Fig 2: Proposed Techniques

In the figure 2: The sound that we have got after encoding the inputted sound is added in the dictionary for searching. If the given sound is found in the dictionary, then it will be displayed as correct word. Otherwise, at first using Edit Distance Algorithm of our proposed system, the least distance words will be brought from the dictionary. Here, only those words will be brought whose Distance Value = 1. The words found using Edit

Distance Algorithm at this stage will be implemented using Soundex Algorithm in future. The number of words will be reduced after the implementation of Soundex Algorithm. That means, the words which have less Edit Distance will be brought using Soundex Algorithm. Later on, the words found using Soundex Algorithm will be used in Metaphone Algorithm. After implementation of Metaphone Algorithm, we will get our targeted word. If the word will not be found, it will be shown as wrong word using error message. At last, the desired word will be found using our proposed technique. Otherwise, the words close to the desired word will be displayed / showed as suggestion. In this way, the user will get his desired word. If after using this system the desired word will not be found, it will be shown as wrong word to the user.

Sample Result#1

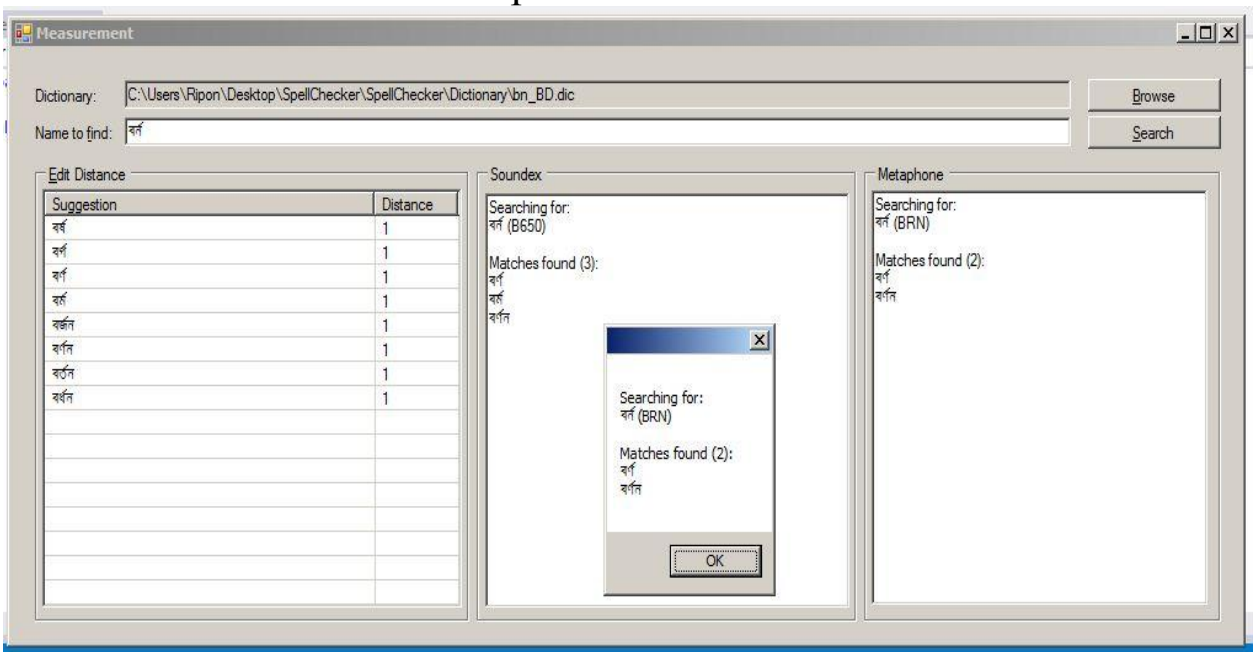

Fig: 3 sample output for বর্ণ

Figure #3 here user given word is **বর্ণ** . In our system firstly it goes to dictionary with the **বর্ণ.** In dictionary it **বর্ণ** and another word is বর্ণনা. And it show suggestion two word which your desired word for correction.

Sample Result#2

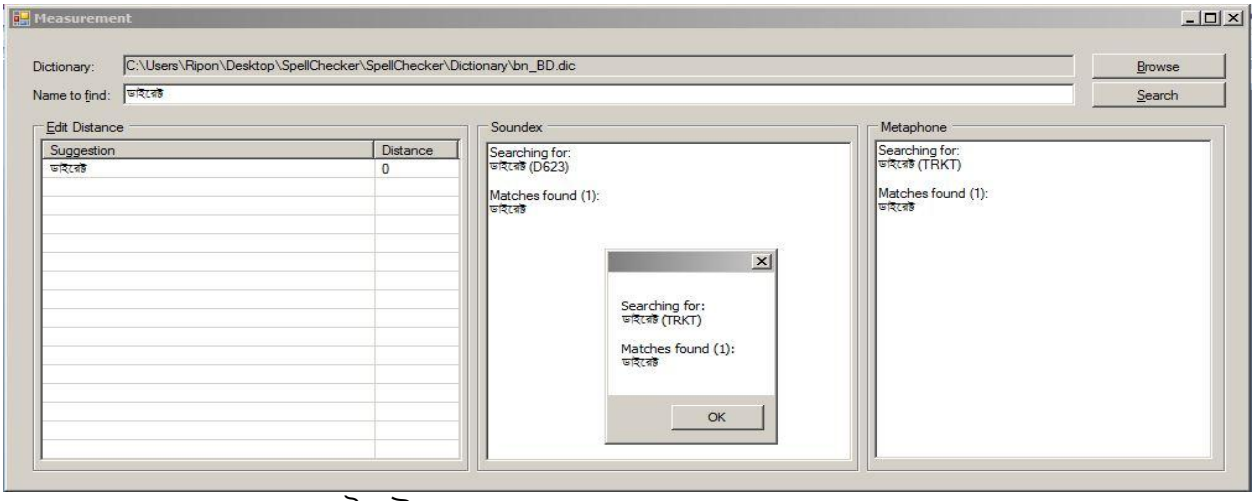

Fig #4: Sample output ডাইরেক্ট

In figure #4 user give word is ডাইরেক্ট which is direct found in dictionary and show your given word is correct.

Sample Result#3

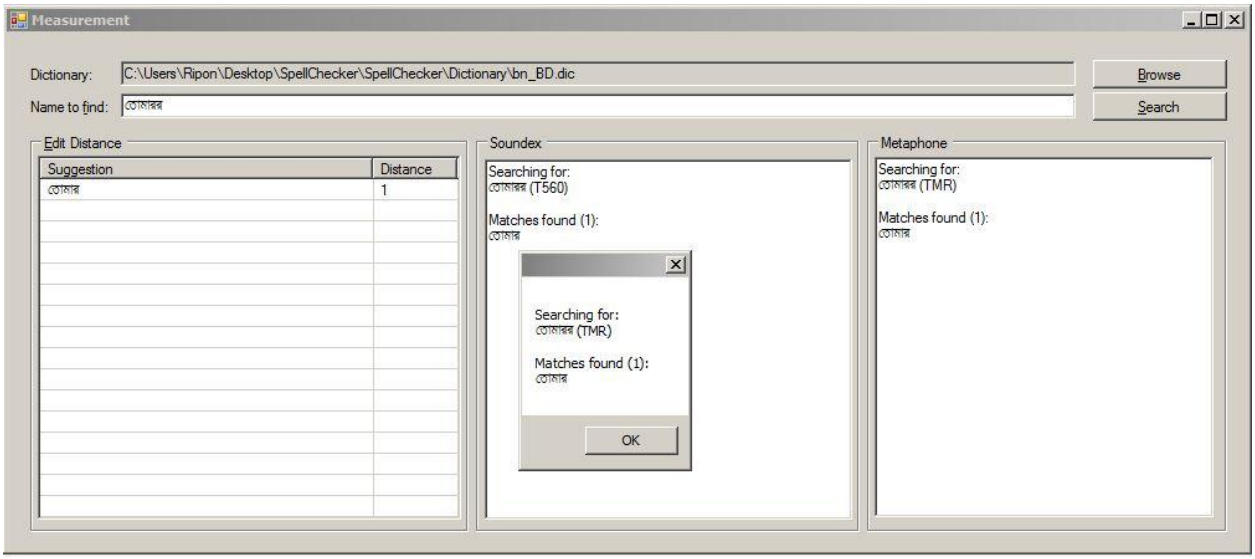

Fig 5: Sample output for তোমারর

In figure #5 user write তোমারর which misspelled add extra character. In our proposed system the input word firstly go to dictionary for matching. When it not found in dictionary it goes to Edit distance, soundex and Metaphone. It gives you the proper desired word like তোমার.

4.11 Proposed technique code:

```
public partial class FormSoundexMetaphone : Form
   {
      public FormSoundexMetaphone()
      {
         InitializeComponent();
         textBoxDictionaryPath.ReadOnly = true;
         btnSearch.Enabled = false;
      }

      #region Form Events
      private void Form1_Load(object sender, EventArgs e)
      {
         cboAlgorithm.SelectedIndex = 0;
      }
      private void btnSearch_Click(object sender, EventArgs e)
      {
         // Get list of names to search
         string[] names = txtNames.Text.Split(new char[] { '\r', '\n' },
StringSplitOptions.RemoveEmptyEntries);

         // List to hold matches
         List<string> matches = new List<string>();

         // Call search method for the selected algorithm
         string code;
         if (cboAlgorithm.Text == "Soundex")
            code = SearchSoundex(txtFind.Text, names, matches);
         else  // Metaphone
            code = SearchMetaphone(txtFind.Text, names, matches);

         #region Show result
         StringBuilder builder = new StringBuilder();
         builder.Append("Searching for:\r\n");
         builder.AppendFormat("{0} ({1})\r\n\r\n", txtFind.Text, code);
         if (matches.Count > 0)
         {
            builder.AppendFormat("Matches found ({0}):\r\n",
               matches.Count);
            foreach (string match in matches)
            {
               builder.AppendFormat("{0}\r\n", match);
```

```
            }
        }
        else
        {
            builder.Append("No matches found");
        }
        MessageBox.Show(builder.ToString());
        #endregion
    }
    private void btnClose_Click(object sender, EventArgs e)
    {
        Close();
    }
    private void buttonBrowse_Click(object sender, EventArgs e)
    {
        textBoxDictionaryPath.ReadOnly = false;
        openFileDialog.InitialDirectory =
        System.IO.Path.GetFullPath(@"..\..\Dictionary");
        openFileDialog.Title = "Browse Text Files";
        openFileDialog.CheckFileExists = true;
        openFileDialog.CheckPathExists = true;
        openFileDialog.DefaultExt = "txt";
        openFileDialog.Filter = "Text files (*.txt)|*.txt|All files (*.*)|*.*";
        openFileDialog.FilterIndex = 2;
        openFileDialog.RestoreDirectory = true;
        openFileDialog.ReadOnlyChecked = true;
        openFileDialog.ShowReadOnly = true;
        If (openFileDialog.ShowDialog() == DialogResult.OK)
        {
            textBoxDictionaryPath.Text = openFileDialog.FileName;
            textBoxDictionaryPath.ReadOnly = true;

            String dictionary = File.ReadAllText(textBoxDictionaryPath.Text);
            List<string> wordList = dictionary.Split('\n', ' ').ToList();
            string[] s = wordList.Select(w=> w.Any(x => !char.IsLetter(x)) ?
w.Substring(0, w.IndexOf("/")==-1? w.Length: w.IndexOf("/")) : w).ToArray();
            txtNames.Lines = s;
            btnSearch.Enabled = true;
        }
    }
    #endregion

    #region Soundex
```

```
private string SearchSoundex(string find, string[] names, List<string> matches)
{
   find = ConvertSoundex(find);
   // Encode string we want to find
   string code = Soundex.Encode(find);

   // Search through the list of names
   foreach (string name in names)
   {
      string soundex_name = ConvertSoundex(name);
      // Compare against soundex-encoded version of name
      if (Soundex.Encode(soundex_name) == code)
      {
         // Found a match--add it to list
         matches.Add(soundex_name);
      }
   }
   return code;
}
private string ConvertSoundex(string text)
{
   StringBuilder engText = new StringBuilder();
   for(int index=0; index<text.Length; index++)
   {
      switch(text[index])
      {

      }
   }
   return engText.ToString();
}
#endregion

#region Metaphone
private string SearchMetaphone(string find, string[] names, List<string> matches)
{
   find = ConvertMetaphone(find);
   // Encode string we want to find
   Metaphone metaphone = new Metaphone();
   string code = metaphone.Encode(find);

   // Search through the list of names
   foreach (string name in names)
```

```
        {
            string metaphone_name = ConvertMetaphone(name);
            // Compare against soundex-encoded version of name
            if (metaphone.Encode(metaphone_name) == code)
            {
                // Found a match--add it to list
                matches.Add(metaphone_name);
            }
        }
        return code;
    }
    private string ConvertMetaphone(string text)
    {
        StringBuilder engText = new StringBuilder();
        for (int index = 0; index < text.Length; index++)
        {
            switch (text[index])
            {

            }
        }
        return engText.ToString();
    }
    #endregion
}
```

CHAPTER V:
CONCLUSION

We have improved Bangla spelling checking, transliteration and name searching application using Edit Distance, Sondex, Metaphone phonetic encoding. The summary regarding the improvements of our new system is as under:

- It can be used to develop a spelling checker that can provide the words of same pronunciation in suggestion.

- It can also be used to develop a transliteration that can be used not only a one to one direct mapping but also be able to give words with same pronunciation from dictionary.

It can be used to develop a name searching application as well in which similar sounding names can be easily found in dictionary and ranked in the suggestion.

Future research

We will try to upgrade the system which will be able to convert the inputted voice into text and as per that text, it will find the related words in the dictonary. If the word exactly

matched, it will be displayed directly. Otherwise, it will show suggestion for same type of words. This is how the system will be more effective in future.

REFERENCES

i. https://people.cs.pitt.edu/~kirk/cs1501/Pruhs/Spring2006/assignments/editdistance/Levenshtein%20Distance.htm

ii. http://creativyst.com/Doc/Articles/SoundEx1/SoundEx1.htm

iii. https://www.codeproject.com/Articles/162790/Fuzzy-String-Matching-with-Edit-Distance

iv. https://nlp.stanford.edu/IR-book/html/htmledition/edit-distance-1.html

v. https://nickgrattan.wordpress.com/2014/06/21/levenshtein-minimum-edit-distance-in-c/

vi. https://en.wikibooks.org/wiki/Algorithm_Implementation/Strings/Levenshtein_distance\

vii. https://en.wikipedia.org/wiki/Metaphone

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

[2] P Lekho, available online at http://lekho.sourceforge.net/.

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

[4] Lawrence Phillips, “Hanging on the Metaphone”, Computer Language, 7(12), 1990.

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