RULE-BASED PHONETIC MATCHING APPROACH FOR HINDI AND MARATHI

Sandeep Chaware[1], Srikantha Rao[2]

[1]Research Scholar, MPSTME, NMIMS University, Mumbai, INDIA
{smchaware@gmail.com}
[2]PhD Guide, MPSTME, NMIMS University, Mumbai, INDIA
{dr_s_rao@hotmail.com}

ABSTRACT

Phonetic matching plays an important role in multilingual information retrieval, where data is manipulated in multiple languages and user need information in their native language which may be different from the language where data has been maintained. In such an environment, we need a system which matches the strings phonetically in any case. Once strings match, we can retrieve the information irrespective of languages. In this paper, we proposed an approach which matches the strings either in Hindi or Marathi or in cross-language in order to retrieve information. We compared our proposed method with soundex and Q-gram methods. We found better results as compared to these approaches.

KEYWORDS

Soundex, Q-gram, Edit distance, Indic-phonetic, phonetic matching.

1.INTRODUCTION

Phonetic matching is needed when many diversified people come together. They either speak with different pronunciation styles or write many languages in various writing styles, but their meaning is same. It deals with identified the similarity of two or more strings by pronunciation, regardless of their actual spelling. A typical example is Just-Dial service in Mumbai, where a telephone operator is given a name for finding out the information for the same. The operator guesses for the possible spelling of it, and then searches from the database (or spelling may be provided which may be incorrect).

Phonetic matching plays an important role in information retrieval in multilingual environment. Information retrieval means searching the data in a database for retrieval. For this indexing and ranking will be used. Information retrieval needs an exact match for a given string. Phonetic matching can be defined as a process of identifying a set of strings those is most likely to be similar in sound to a given keyword [1]. The strings can be spelled using different writing styles, but they can be matched phonetically. All the strings represent the same keyword, only way of writing is different. Since in rural areas, the word may be spelled or pronounce either wrongly or differently. We can retrieve the data using phonetic matching. There is no need of exact string matches.

DOI : 10.5121/cseij.2011.1302

There are many techniques had been proposed in order to find the phonetic matching of strings like Soundex, Q-gram, Caverphone, Phonix etc. Each technique generates a code for the strings and matches them through edit distances. If the edit distance is minimum, those strings are more close to each other [2]. But, these methods have been not found suitable for all the strings. There are some limitations like first, generation of code follows long procedure, and second, they are generating the same code for mismatch strings or generate different codes for match strings. Other technique uses text-to-phonetic (TTP) approach, which may not work for Indian languages, as those codes are not available or TTP conversion is complex. [2].

In this paper we proposed an approach, which provides a simple and efficient way of matching the strings. Our scheme will work on text-conversion technique for any Indian languages, especially Hindi, Marathi. It has been found that most of the rural people don't either spell or make correct pronunciation the keywords properly.

The main objectives of the proposed system are, first, to convert the entered strings into its equivalent phonetic forms by applying phonetic rules for each language instead of using TTP and to compare our methodology with other methods. Second, is to retrieve the information needed for the entered string from the database if it is matching phonetically.

2.PHONETIC MATCHING APPROACHES: A SURVEY

In this section, we describe the various techniques for phonetic matching. Existing techniques were developed to match the string approximately, where as our approach matches the strings exactly. These approaches used for phonetic matching.

A. Soundex for English

In this, each string is converted into a code, which consists of a first letter of a string and three numbers. The numbers are assigned to each letter as per guidelines described by the algorithm [2][3]. Zeros are added at the end if necessary to produce four-character code. Additional letters are discarded. The soundex codes and algorithm are given in table1 and algorihm1 respectively. The pitfall of this algorithm is producing the same code for phonetically two different strings or producing different codes, even they are same.

Table 1: Soundex phonetic codes

Code:	0	1	2	3	4	5	6
Letters:	a,e,i,o,u, y,h,w	b,p	c,g,j,k q,s,x,z	d,t	l	m, n	r

Algorithm 1: Soundex for Phonetic Matching
Input: Two string to match
Output: Soundex code

1. Retain the first letter of the string.
2. Remove all occurrences of the following letters, unless it is the first letter: a, e, i, o, u, y, h, w.
3. Assign numbers as given in code table.
4. If two or more letters with same number were adjacent, then omit all except first.
5. Return the first four bytes padded with 0 if necessary.

EXAMPLE

Consider the two strings in English as, '**sandy'** and '**sandhya'** for phonetic matching. After applying the rules from table 1 and steps from algorithm 1, we are getting the same code as '**s530'.** So, by looking at the code we have to say that both strings are phonetically matching, but both the strings are phonetically different.

B. Soundex for Hindi

If we consider the soundex algorithm for Hindi and assign the code as per the table 2. If we match the strings with this algorithm, we will get different codes for the same string or we can not generate the code. From this methodology, the code can be generated by taking halant for each consonants concatenated together with vowel. This will change the code and it takes long time to parse the string and assign the code to it. We have found out that, there is no code for many alphabets for Hindi language, such as ण, झ, छ, घ etc.

Table 2: Soundex code for Hindi as per algorithm

Code:	0	1	2	3	4	5	6
Letters:	अ, इ, ओ, उ, व, ह	ब, प	च, ग, ज, क, स	ड, ट, त	ल, ळ	म, न	र

EXAMPLE

If we consider the same rules and algorithm for Hindi as shown in table 2, the codes for 'सँडी' and 'संध्या' strings are same as 'स530'. By using soundex method, we are getting the same code for both the strings entered in Hindi. As codes are same, we have to say that both strings match phonetically, but they are not matching. There is an ambiguity to match.

C. Q-Gram for English

This method measures the string distances based on q-gram counts, where q-gram of a string s is any substring of s of some fixed length q. A simple such measure is to count number of q-grams for the two strings, with a higher count yielding a stronger match [2]. But, this algorithm is not exactly phonetic, since they do not operate based on comparison of the phonetic characteristics of words. The steps for the algorithm are as given in algorithm 2. Since phonetically similar words often have similar spellings, this technique can provide favorable results, yet it also successfully matches misspelled or otherwise muted words, even if they are rendered phonetically disparate. For example, for the strings, '**sandy'** and '**sandhya'**, with **q = 2**, the substrings are as shown in table 3. With q-gram algorithm we are getting 3-grams matches out of 4-grams and for 2 grams there are no gram to match from other string. Depending on number of q-grams, we have to find out similarity of strings as phonetically.

Table 3: Q-gram Example

String 1 : 'Sandy'	Sa	an	nd	dy	----	----
String 2: 'Sandhya'	Sa	an	nd	dh	hy	ya

Algorithm 2: Q-Gram
Input: Two strings to match
Output: Number of n-grams

1. Select the value for q.
2. Make substrings of the strings with number of letters equal to q.
3. for q value
   i. First string = first q letters.
   ii. Next substring = second letter of previous string + next letter.
   iii. Repeat.
4. Count the number of substrings in common.
5. Maximum count results in most matches.

D. Q-gram for Hindi

If we apply the same rules as in algorithm 2 in order to form q-grams for Hindi strings then the entire string will change or we will not get any q-gram that matches. One problem with this method is that if parsing of a string goes wrong then it affects the generated code.

EXAMPLE

Consider the same strings as, 'सँडी' and 'संध्या', after applying Q-gram method for Hindi, we will get only two 2-grams that match and two does not matches. There is also an ambiguity for phonetically match.

E. Edit Distance for English

A simple edit distance, which counts the minimal number of single-character insertions, deletions and replacements needed to transform one string into another, could be used for phonetic matching since similar sounding words are often spelled similarly. Each string is converted into phonetic string, and then the edit distance is calculated between them as shown in algorithm 3. For conversion, text-to-phonetic algorithm can be used. In this method, we need TTP algorithm for each language. TTP may not result appropriate for Indian languages.

Algorithm 3: Edit Distance
Input: Two strings to match
Output: Edit Distance between them

1. edit (0 , 0) = 0
2. edit (i , 0) = i
3. edit (0 , j) = j

4. edit (i , j) = min [edit (i-1),j + 1,
   edit (i,j-1) + 1,
   edit (i-1, j-1) + r(s_i + t_j)]
5. if r(a, b) = 0, strings are similar
   else strings are non-similar.

Where s and t are strings of length m and n respectively.

3.PROPOSED INDIC-PHONETIC APPROACH

In this approach, we are providing the user interface to provide the keywords or strings in local languages like Hindi or Marathi for phonetically matching. Each entered string will be parsed to get exact combination of vowels, consonants and/or modifiers as phonetic string according to phonetic rules for each language mentioned below. We are assigning the codes for each phonetic string, in order to match. The difference between those codes are compared with the threshold value, if it is less or equal to threshold value, then these two strings are phonetically matched else not.

A. DESIGN

Overall system architecture for proposed Hindi-Marathi phonetic approach is as shown in figure 1. The system includes modules as user interface module, parsing module, phonetic equivalent string module, comparison module and display or result module. Each module is described below in brief.

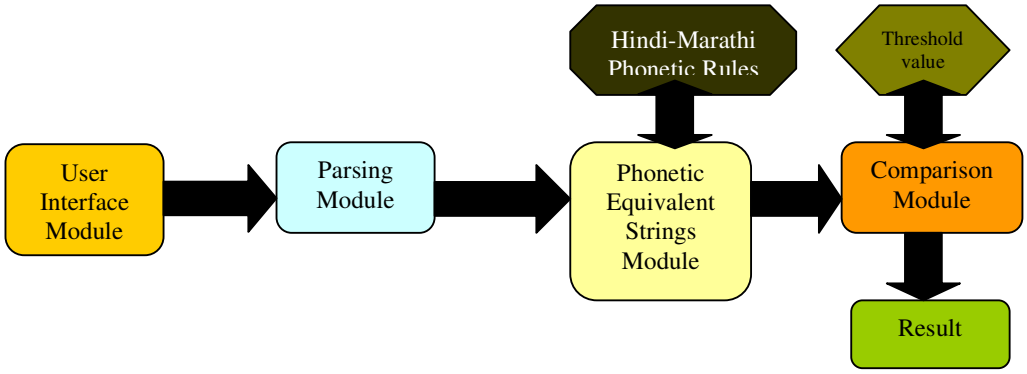

Figure 1: Hindi-Marathi Phonetic Approach System Architecture

1. **User Interface Module:** This module will give user an interface to interact with the system in order to find phonetic equivalency between two strings. The user can enter either two Hindi/Marathi strings or two strings in two different languages. Assuming that the two entered strings are exactly as they should be written in those languages.
2. **Parsing Module:** Entered strings are being parsed by this module in order to extract each alphabet as vowel, consonant or modifier. This will help us to get the equivalent phonetic strings.
3. **Phonetic Equivalent String Module:** This module will translate the entered string into its equivalent phonetic string by applying phonetic rules for each language. We had

applied some rule directly from the grammar and designed some rules for Hindi and Marathi languages.

4. **Comparison Module:** In this module, we are assigning phonetic code as Unicode as per the phonetic group for each language and finally got the Unicode for the entire string by summing up all the Unicode in a string. With some threshold value, we can compare those two strings for phonetic matching. If found matching, then both the strings are very closely phonetic matches.
5. **Display/result Module:** This module will display whether two strings are phonetically equivalent or not as a message to the user.

B. IMPLEMENTATION STRATEGY

In this approach, we are forming the rules as per pronunciation of alphabets in Hindi and Marathi languages. Each string will be interpreted and converted to its phonetic form using these rules. For phonetic matching, we are forming the codes for each string as per Unicode table [6], which will be common for both the languages. Then these codes are compared with 5% threshold fixed value. If they are within a range, then we can say that they are phonetically matched.

I) Phonetic Rules for Hindi

We can formulate some rules according to the occurrences and way of pronunciation of each character in Hindi[7]. Some rules are follows:

Rule 1: If 'स' or 'श' comes, then dot (.) will be pronounced as 'न'.

Rule 2: If 'ह' is preceded by anuswar (.) then it will be pronounced as 'घ'.

Rule 3: If 'ज्ञ' occurs in a string, then it will be pronounced as 'ग्य '.

Rule 4: If 'ड़' or 'ढ़' occur as last character from a string, then it is pronounced lightly and will have different code.

Rule 5: If 'ड' or 'ढ' occurs as a starting or after anuswar (.) in a string, then it is pronounced fully and will have different code.

Rule 6: For 'ऋ', the pronounced character is 'रि'.

Rule 7: If 'त्र' occurs, then it is pronounced as 'त् + र'.

We are applying these mentioned rules of pronunciation to the Hindi strings, in order to acquire the correct form of phonetic string. Now, for this phonetic form, we are assigning the codes from the mapping table in order to match phonetically. We had used the difference between the codes as a threshold value for matching. If the result crosses the threshold value, then the entered two strings are phonetically matched, else not.

II) Phonetic Rules for Marathi

We can formulate some rules according to the occurrences and way of pronunciation of each character in Marathi [8]. Some rules are follows:

Rule 1: Every consonant is a combination of consonant + halant + 'अ'.

Rule 2: If 'क्ष' occurs, then its pronounced character is 'क् ष'.

Rule 3: If 'ज्ञ' occurs, then its pronounced character is 'द् + न् + य'.

Rule 4: For last consonant of a string, it is pronounced with halant. For example, नवस = नवस्.

Rule 5: For last but-one consonant is pronounced with halant. For example, हसरा = हस्रा.

Rule 6: A string of 4 characters, the second consonant is pronounced with halant, for example, घरदार = घर्दार.

Rule 7: if 'य' after anuswar then it is pronounced as 'यँ', for example, संयुक्त = संय्युक्त.

Rule 8: if 'ल' after anuswar then it is pronounced as 'लँ', for example, संलाप = संल्लाप.

Rule 9: if 'र, व, श, ष, स, ह, ज्ञ' after 'सं' then it is pronounced as 'वँ', for example, संसद = संव्सद.

We are applying these mentioned rules of pronunciation to the Marathi strings, in order to acquire the correct form of phonetic string. Now, for this phonetic form, we are assigning the codes from the mapping table in order to match phonetically. We had used the difference between the codes as a threshold value for matching. If the result crosses the threshold value, then the entered two strings are phonetically matched, else not.

III) Proposed Rule-based Phonetic Matching for Hindi and Marathi Algorithm

Input: Two strings either in Hindi or Marathi to match
Output: Phonetic Matching Yes or No

1. Enter two Hindi or Marathi strings to match.
2. Each string is translated into its phonetic form by using phonetic rules for each language.
3. Obtain Unicode for each translated string, by summing the Unicode value of each characters of a string.
4. Compare the resultant Unicode by considering a threshold value of 5%.
5. If these values are within 5%, then we are saying that they are phonetically matched. Else they are not matching.

EXAMPLE

If we consider the strings as in either Hindi or Marathi, after applying our proposed Indic-phonetic approach, we can say that both the strings are NOT phonetically matching. As shown in table 4, we are not getting the same phonetic strings as well as same phonetic code.

Table 4: Indic-phonetic approach for Hindi and Marathi strings

String no.	Hindi strings	Corresponding phonetic code as per Indic-phonetic approach	Marathi strings	Corresponding phonetic code as per Indic-phonetic approach
1	‘सैंडी’	स्ऐंड्ई	‘सैंडी’	स्ऐंड्ई
2	‘संध्या’	स्अन्अध्य्आ	‘संध्या’	संध्य्आ

4. RESULTS

It has been found that our approach will give the exact match for any writing styles of a string, which will be equivalent to phonetically. There is no need to generate a code or forming the substrings, only conversion is needed. For this we had used mapping method, or Wordnet can be used for the same.

Figures from 2 to 8 shows an interface to enter two strings to compare phonetically either in Hindi or Marathi, its phonetic equivalent according to rules, and matching results for soundex, Q-gram and proposed Indic-phonetic methods. The results also show the information retrieval as phonetic matching from database. These phonetic representations are assigned codes as per rules mentioned. These codes are compared with threshold given compared for phonetically matching.

Figure 2: Phonetic Matching Approach for Hindi and Marathi

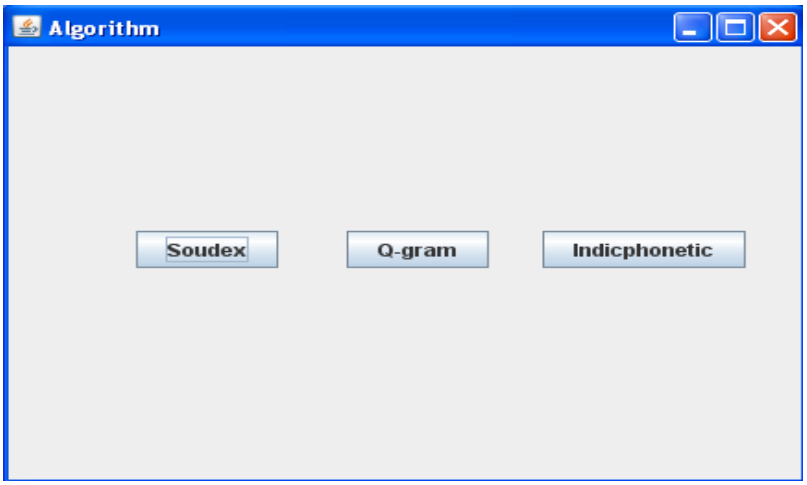

Figure 3: Three Phonetic matching algorithm to compare

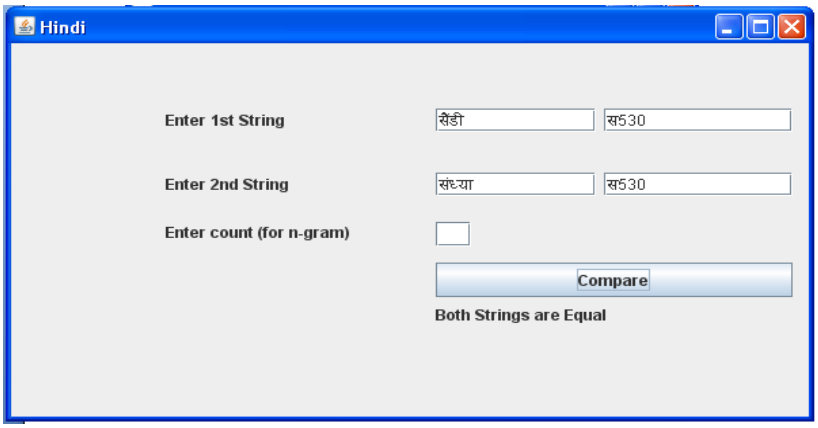

Figure 4: Soundex Approach for Hindi

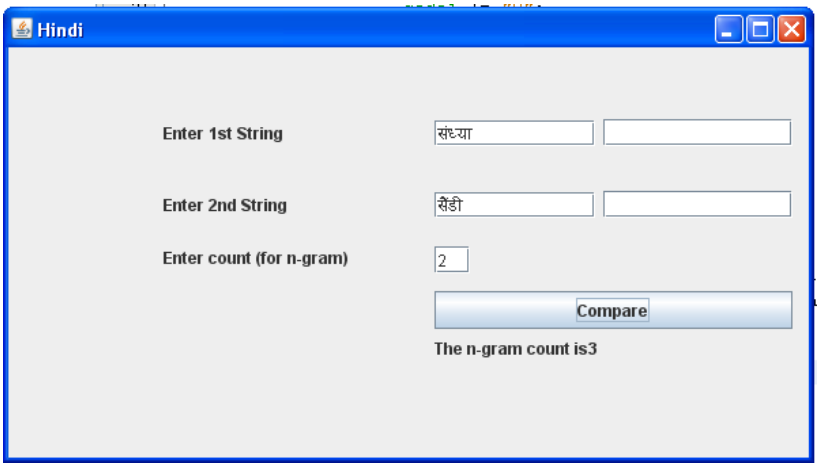

Figure 5: Q-gram approach for Hindi

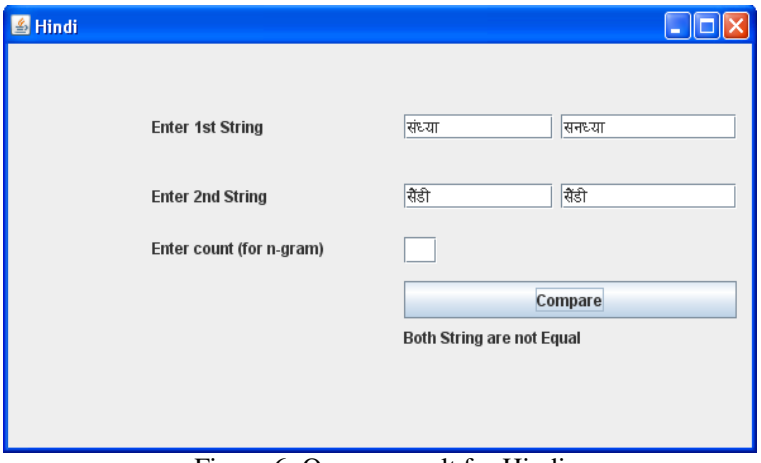

Figure 6: Q-gram result for Hindi

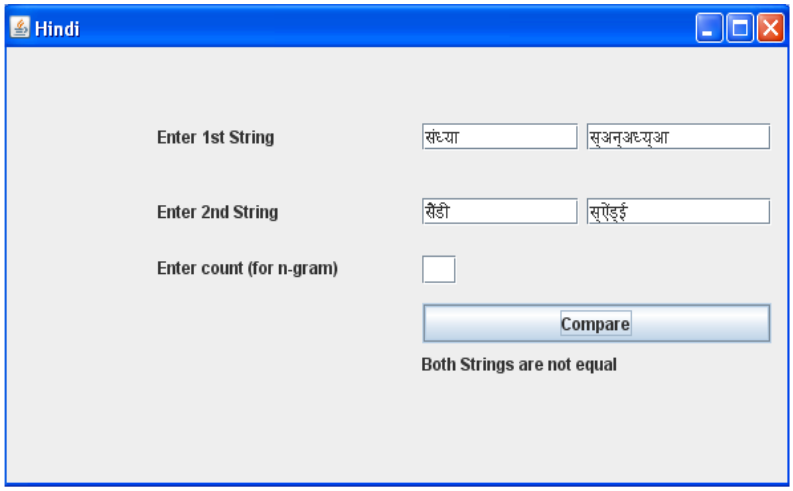

Figure 7: Indic-phonetic for Hindi

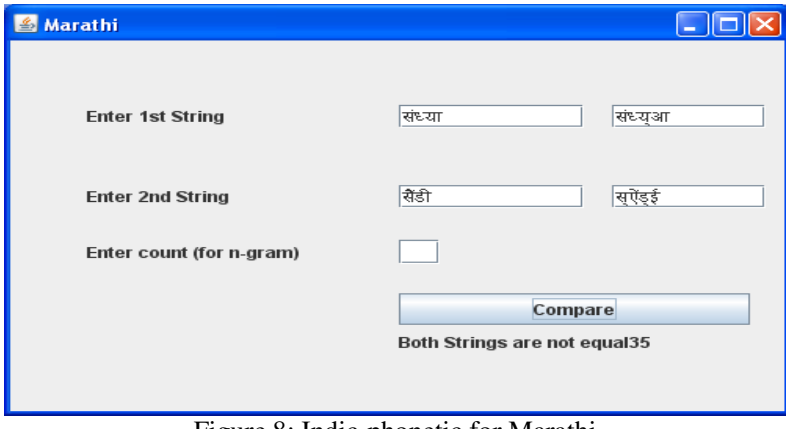

Figure 8: Indic-phonetic for Marathi

Table 5: Strings with codes and matching status for Hindi and Marathi

Strings	HINDI			MARATHI		
	SOUNDEX	Q-GRAM	INDIC-PHONETIC	SOUNDEX	Q-GRAM	INDIC-PHONETIC
संतोष & संथोष	YES	YES	YES	YES	YES	YES
मुंबई & मुंबाई	YES	YES	NO	YES	YES	NO
रघुलिला & राघुलिला	YES	YES	YES	YES	YES	YES
संध्या & सैंडी	YES	YES	NO	YES	YES	NO

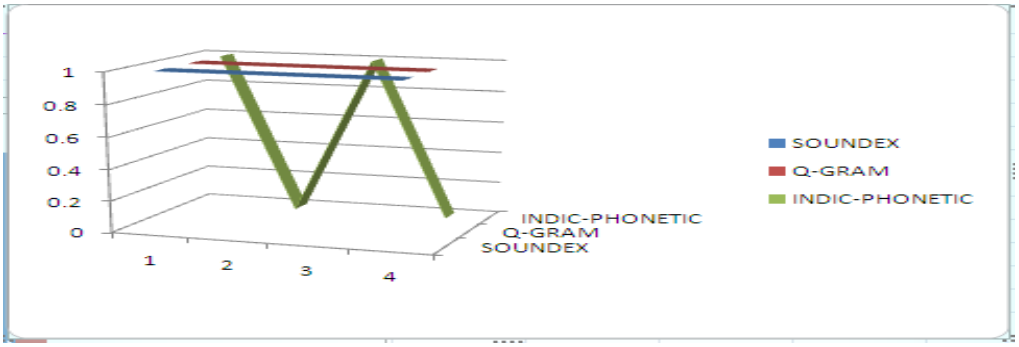

Figure 9: Phonetic Matching for soundex, q-gram and Indic-phonetic for Hindi and Marathi

5. CONCLUSIONS

There are many features of phonetic for Indian languages. In this chapter, we had tried to include most of them by forming the rules in order to match the strings for Hindi and Marathi languages. In this approach, we explored and compared some of the phonetic matching approaches for Indian languages like Hindi and Marathi. We have found out that these approached are lagging in accommodating all the characters form an alphabet for both Hindi and Marathi languages. We proposed a rule-based approach for phonetic matching. We had also proposed this approach for cross-language string matching. This method is without using IPA code, thus reducing the ambiguity for matching. Advantages of our approach are that it gives the user and developer a simple, easy and efficient way for phonetic matching.

REFERENCES

[1] Justin Zobel and Philip Dart, "Phonetic String Matching: Lessons from Information Retrieval".
[2] A. Kumaran, "Multilingual Information Processing on Relational Database Architectures", PhD Thesis, IISc Bangolre, 2005.
[3] Alexander Beider and Stephen P. Morse, "Phonetic Matching: A Better Soundex".
[4] http://en.wikipedia.org/wiki/Soundex
[5] www.bhashaindia.com
[6] http://tdil.mit.gov.in/unicodeapril03.pdf
[7] Hindi Grammar Book.
[8] Marathi Grammar Book.

Author [1]

Mr. Sandeep Chaware, Research Scholar at MPSTME, NMIMS University, Mumbai, INDIA.

Author [2]

Dr. Srikantha Rao, PhD Guide at MPSTME, NMIMS University, Mumbai, INDIA.