Indian Journal of Science and Technology, Vol 8(27), DOI: 10.17485/ijst/2015/v8i27/83917, October 2015
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Design and Implementation of Shahmukhi Spell Checker

Kawarbir Singh Dhanju[1*], Gurpreet Singh Lehal[1], Tejinder Singh Saini[2] and Arshdeep Kaur[1]

[1]DCS, Punjabi University, Patiala - 147 002, Punjab, India; kbs.dhanju@gmail.com, gslehal@gmail.com, akaur448@yahoo.com

[2]ACTDPL, Punjabi University, Patiala - 147 002, Punjab, India tej@pbi.ac.in

Abstract

Spellchecker is a software tool that identifies and corrects any spelling mistakes in a text document. Designing a spell checker for Punjabi language is a challenging task. Punjabi language can be written in two scripts, Gurmukhi script (a Left to Right script based on Devanagari) and Perso-Arabic Script (a Right to Left script) which is also referred as Shahmukhi. Gurmukhi script follow 'one sound - one symbol' principle where as Shahmukhi follows 'one sound - multiple symbol' principle. Thus making Shahmukhi text even more challenging which complicates the design of spell checker for Shahmukhi text. The text written in Shahmukhi normally does not have short vowels and diacritic marks. So missing some of diacritic marks should not be considered as a mistake. But for Holy books like Quran, missing diacritic marks are considered as a mistake. So spell checker is designed in such a way that it can spell check with and without diacritic marks compulsion, which depends on user's selection to spell check. In addition to this, Shahmukhi text has complex grammatical rules and phonetic properties. Thus it needs different algorithms and techniques for expected efficiency. This paper presents the complete design and implementation of a spell checker for Shahmukhi text.

Keywords: Edit Distance, Gurmukhi, Punjabi, Shahmukhi, Spellchecker, Typing Errors

1. Introduction

Spellchecker is a software tool that identifies and corrects any spelling mistakes in a text by checking the spellings of the words in a document, validate them i.e. checks whether they are right or wrongly spelled and in case the spell checker has doubts about the spelling of the word, it suggests possible alternatives.

The main steps performed by the spell checker are:

- Input Shahmukhi words from user document.
- Pre-process the words.
- Detect the erroneous word by searching it from the dictionary.
- In case, the word is erroneous, suggest possible alternatives.

Even though it looks simple but to write Punjabi in Shahmukhi script is complex than other languages such as English, Hindi. Thus existing algorithms and techniques are not suitable for the design of spell checker for Shahmukhi script.

2. Brief Description of Punjabi Language

Punjabi language is 10th most widely spoken language in the world. It is spoken by 102 million speakers worldwide[12]. It is the native language of the Punjabi people who inhabit the historical Punjab region of Pakistan and India. Punjabi can be written in two Scripts Gurmukhi and Shahmukhi. Gurmukhi is used to write Punjabi in India and Shahmukhi is used to write Punjabi in Pakistan. Shahmukhi is basically Punjabi text written is Perso-Arabic Script (a Right to Left script). Shahmukhi text has complex grammatical rules and phonetic properties. In Pakistan Punjabi written in Shahmukhi script is not

*Author for correspondence

Table 1. Common aspirated consonants

Sr.	Shahmukhi	Unicode	Sr.	Shahmukhi	Unicode
1	ھب [bco	06BE + 0628	6	ھچ [ʧʰ]	0686 + 06BE
2	ھپ [p6	067E + 06BE	7	ھد [d̪ʰ]	062F + 06BE
3	ھت [t̪ t̪]	062A + 06BE	8	ھڈ [dˌʰ]	0688 + 06BE
4	ھٹ [ʈ ʈ]	0679 + 06BE	9	ھک [k9	06A9 + 06BE
5	ھج [ʤʰ]	062C + 06BE	10	ھگ [gC	06AF + 06BE

Table 2. Non-Aspirated consonants

Sr.	Shahmukhi	Unicode	Sr.	Shahmukhi	Unicode
1	ب [b]	0628	21	ط[t]	0637
2	پ [p]	067E	22	ظ[z]	0638
3	[ت [t]	062A	23	ع[?]	0639
4	ٹ [t]	0679	24	غ[r]	063A
5	ث[s]	062B	25	ف[f]	0641
6	ج [d3]	062C	26	ق[q]	0642
7	چ[tf]	0686	27	ک[k]	06A9
8	ح [h]	062D	28	گ[g]	06AF
9	خ[x]	062E	29	ل[l]	0644
10	د [d]	062F	30	م[m]	0645
11	ڈ[d]	0688	31	ن[n]	0646
12	ز [z]	0632	32	ڻ[n]	06BB
13	ر[r]	0631	33	ں[33	06BA
14	ڑ [t]	0691	34	و[v]	0648
15	ذ[z]	0630	35	ہ[h]	06C1
16	ژ [3]	0698	36	ھ[h]	06BE
17	س[s]	0633	37	ی[j]	06CC
18	[*f*] ش	0634	38	ے[j]	06D2
19	ص[s]	0635	39	۽[l]	0644
20	ض[z]	0636			

an official language so very little support and resources are available for Shahmukhi script. In fact this is the first time a spell checker support for Shahmukhi text has been designed and implemented.

2.1 Shahmukhi Script

The meaning of "Shahmukhi" is "from the King's mouth"[1–3,10]. The Shahmukhi text was first used by the Sufi poets of the Punjab, and then Muslim populace in

Table 3. Long vowels

Unicode	Vowel	Name
0627	ا [ə]	Alif
0622	آ [a]	Alif Madda
0648	و [o]	Vav
06CC	ی[i]	Ye Chhote
06D2	ے [e]	Ye Bari

Table 4. Short vowels

Unicode	Vowel	Name
064E	[ə]	Zabar
064F	ُ[u]	Pesh
0650	[i]	Zer

Table 5. Optional diacritics

Unicode	Optional Diacritics	Name
0656	ٖ	(khari zabar)
0652	ْ	(Sukun)
064E	ً	(zabar)
0670	ٰ	(Superscript Alef)
0650	ِ	(zer)
064C	ٌ	(dammatan)
064D	ٍ	(do zer)
0651	ّ	(tasdid)
064F	ُ	(pesh)
0657	ٗ	(ulta pesh)
0658	٘	(Superscript Noon Ghunna)

Pakistan uses Shahmukhi text to write Punjabi. Some of the Major properties of Shahmukhi text:

- Shahmukhi text is written in Nastaleeq style and from right to left, a highly complex writing system that is cursive and context-sensitive. It has 49 common and 6 rare consonants, 16 diacritical marks or vowels, etc.
- Consonants can be further subdivided into two groups: aspirated and non-aspirated consonants.

In Shahmukhi, aspirated consonants are represented by the combination of a consonant (to be aspirated) and HEH-DO CHASHMEE.
The remaining six aspirated consonants are:

[rh]ھر,[mh] ھم,[lh]ھل,[th]ھڑ,[vh]ھو,[nh]ھن.

Table 6. Shahmukhi punctuations

Punctuations	Unicode	Punctuations	Unicode
۔	06D4	،	060C
!	0021	:	003A
؟	061F	٪	066A
]	005D	[	005B
؛	061B	٬	066C
٫	060D	ـ	0640
٭	066D		

Table 7. Numerals in Shahmukhi text

Numerals	Urdu Numerals	Unicode
0	۰	06F0
1	۱	06F1
2	۲	06F2
3	۳	06F3
4	۴	06F4
5	۵	06F5
6	۶	06F6
7	۷	06F7
8	۸	06F8
9	۹	06F9

Table 8. Joiners in Shahmukhi Script

Unicode	Shahmukhi	Unicode	Shahmukhi
062E	خ [x]	067E	پ [p]
062D	ح [h]	0646	ن [n]
0686	چ [tf]	0641	ف [f]
062C	ج [d3]	0628	ب [b]
0645	م [m]	06BB	ڻ [n]
06A9	ک [k]	063A	غ [r]
062B	ث [s]	0642	ق [q]
0679	ٹ [t]	0644	ل [l]
062A	ت [t]	0639	ع[?]
0638	ظ [z]	0637	ط [t]
0636	ض [z]	06AF	گ [g]
0649	ی [i]	0635	ص [s]
0634	ش [s]	0633	س [s]
0647	ہ [h]		

In case of **non-aspirated consonants,** Shahmukhi has more consonants than Gurmukhi, which follows the one symbol for one sound principle. On the other hand there are more than one characters for a single sound in Shahmukhi.

Table 9. Non-Joiners in Shahmukhi Script

Unicode	Shahmukhi	Unicode	Shahmukhi
0698	ژ [3]	062F	د [d]
0632	ز [z]	0622	آ [a]
0691	ڑ [r]	0627	ا [ə]
0631	ر [r]	06D2	ے [e]
0630	ذ [z]	0648	و [v]
0688	ڈ [d]		

Diacritics are used to specify the vowels. In Shahmukhi, there are five **long vowels**.

And three **short vowels**:

According to Analysis, below diacritics are considered to be optional:

2.2 Shahmukhi Punctuation

2.3 Shahmukhi Numerals

Shahmukhi characters can be divided into two groups, **non-joiners and joiners**[1]. The non-joiners can acquire only isolated and final shape and do not join with the next character On the contrary; joiners can acquire all the four shapes and get merged with the next following character. A group of joiners and/or non-joiner joined together form a ligature. A word in Urdu is a collection of one or more ligatures. The isolated form of joiners and non-joiners is shown in Tables 8 and 9.

3. Error Pattern in Shahmukhi Text

Shahmukhi text has complex grammatical rules and phonetic properties which makes Shahmukhi text open to different types of mistakes. The following error patterns were observed in Shahmukhi text:

3.1 Multiple Characters with Same Sound (Phonetic Nature)

In Shahmukhi script, there is more than one letter for single sound; some sounds have 5 to 6 letters, which is the major reason for spelling mistakes. Some of the examples are shown below.

Table 10. Characters having similar phonetic Code

S	س	ص	ث			
H	ہ	ح				
K	ک	ق				
J	ض	ز	ظ	ج	ژ	ذ
T	ط	ت	ت	ھٹ	ٹ	ة

Table 11. Characters having similar Shape

ث	ٹ	ت	پ	ب		
خ	ح	چ	ج			
ز	ڑ	ژ	ر	ذ	ڈ	د
س	ش					
ض	ص					
ظ	ط					
غ	ع					
ق	ف					
گ	ک					
ں	ن					

One of the most common type of error in Shahmukhi text is that "gol he(ہ)" is used at the end of word to produces sound of "a", but mostly the user misspelled it with ا (alef).

For Example,

امریکہ /əmrikh/=>امریکا/əmrikə/

Characters with Similar Shapes: - In Shahmukhi script, the characters such as given below, have same shapes and thus are reason for the misspelled words.

3.2 Characters with Zero Width

In Shahmukhi script, the characters such as given below in the Table 12, have zero width and so if by mistake a user makes multiple entries of such characters only a single entry is visible. If the spell checker flags such word as misspelled the user will not come to know where the error exist. This problem is also considered as **Visual Error** as well as **Dual Diacritic Error**. For example, consider the word,

اُ+ُ+ل+ف+ َ+ت=/ulfat/تفَلُاُ

Table 12. Characters having zero width

Unicode	Optional Diacritics	Name
0656	ٖ	(khari zabar)
064E	ً	(zabar)
0670	ٰ	(Superscript Alef)
0650	ِ	(zer)
0651	ّ	(tasdid)
064F	ُ	(pesh)

It has two "pesh" diacritic mark but visually the word looks correct but internally it has stored wrongly and the user will not be aware where the error lies.

3.3 Visual Errors due to Nastaleeq Style

As Shahmukhi is written in Nastaleeq Style, so sometime, when number of joiners (letter type- Joiner and Non-Joiner) gets combined to form the word, the diacritic marks is not visible to the users which might be having some mistake. Such problems are considered as Visual Error. As in the Example

ے + ھ + ّ + ک + ُ + ر = ےھکّرُ /rukkhe/

This word has 3 Joiners (ک,ھ,ے), so tasdid (ّ) is not visible when the letters are joined.

ے + ھ + ک + ُ + ر = ےھکـرُ/rukhe/

In this word tasdid (ّ) is missing which is considered as a mistake when diacritic marks are compulsory.

3.4 Presence of Nasal Sounds

There are five nasal sounds in Shahmukhi, Rnoon (ڻ), Meem (م), Noon Gunna (ں), Gunna (ن) and Do Zabar (ً). The user often gets confused about which character is to place among the above characters for the nasal sound. For Example:

سںبھدنھی/sanbhandhi/

سنبھدنھی/sanbhandhi/

3.5 Optional Diacritics

The problem which is typically related to Shahmukhi Script is that Short Vowels and diacritic marks are not compulsory to write. So if the word having two optional diacritics, the user may lose first diacritic, second diacritic or both or even both the diacritic marks may be considered. Thus, a single word with two diacritics have its four variations. All the cases have to be considered for spell checking.

For example, /ulfat/اُلفَتword can be written as اُلفَت or اُلفـت or الفَت or الفـت, all these variations are correct and has to be considered.

3.6 Presence of Izafat

Izafat are words in which two valid words are connected like: قابلِ اعتبار (kabil-i-aitbar) and its meaning is words are connected like: diacritics, the user may lose first diacritic, second diacritic or both or even both the diacritic marks may be considered any two words can be connected to form Izafat. Some of the examples of Izafat are:

قابلِ اعتبار/kabil-i-aitbar/,

باغ و بہار/baag-o-bahaar/,

مغلِ اعظم/Mughal-i-azam/,

آہِ گرم/ah-i-garm/

4. Lexicon Creation

The first step in development of the spell checker is the creation of a lexicon of correctly spelled words, which will be used by the spell checker to check the spellings as well as generate the suggestions. Various Techniques has been used to create the Lexicon for different Spell Checker such as some of them are given below:

- In Bangla Spell Checker[4], phonetically similar characters are mapped into the single unit of character code. So the user input is checked using that character code.
- In Malayalam Spell Checker[5], "Rule cum Dictionary" based Approach is used, where it stores the root word in dictionary and user input is checked by deriving the root word using the Morphological Analyzer and Morphological Generator.
- In Oriya Spell Checker[6], the words in the dictionary are stored according to the length of the word for effective search. Only root words are stored in the dictionary, so root word is obtained from the user input by using Morphological Analyzer and this root word is then checked from the dictionary.

- In Assamese spell checker[7], Hash Table has been used as lexicon look-up data structure. The correct Assamese words are stored into the hash table. The user input is directly checked by dictionary search technique.

From the above observation, there are two issues involved in lexicon development:

- Size of the lexicon.
- Format of the words in lexicon.

4.1 Size of the Lexicon

It is observed that there are two approaches can be followed for storing the lexicon[8]. The first approach stores the root words of a language and the rest of the words are derived from these root words like Oriya spell checker. The other approach is to store all the possible words of the language in the lexicon. We have followed the second approach and stored all the possible forms of words of Shahmukhi words in the lexicon.

In Shahmukhi Script, as Optional Diacritics discussed in Error Pattern, a single word with two diacritics have its four variations and all the cases have to be considered for spell checking. For the consideration of all the cases, words having all the diacritic marks is stored in the lexicon.

For example, تفَلْاُ/ulfat/ having two optional diacritic have four variations (تفلْاُ تفَلْاُ تفَلْاُ تفلْاُ), but in lexicon تفَلْاُ word (word having both diacritic marks) is stored so that spell check and suggestion generation of wrong word is possible while Spell Checker is executing in e considered for as well as having both diacritic marks) is stored so that spell check and suggestion generation of wrong word is possible while Spell Checker is executing in "With diacritic" spell checking. For the conside identified and stored in the database.

In Lexicon, each word is given a Phonetic code according to Soundex Approach. Phonetic code itself acts as index key to all the words having same phonetic code.

Furthermore, the words are arranged according to the size of phonetic index. During program execution the words are loaded into Hash table and nineteen different Hash tables are used for different Phonetic code length. The key of the Hash Table is the Phonetic code and value of the Hash Table is the list of all possible words correspond to that Phonetic code.

Table 13. Some entries in Hash table of length 3

Phonetic	Shahmukhi words
HUA	ہوَاُ/hua/
	ہوا/hawa/
	ہواّ/haawa/

The advantage of this approach is Phonetic code, Word length base dictionary division and Hash Table of above discussed Spell Checkers are included to make it more powerful. The number of keys in each of these hash tables is shown in Figure 1.

4.2 Format of the Words

In Unicode, there are more than one code points for a single letter. For example, آ (alef-madda) letter can be written by a single key (that means single Unicode 0622) and ا + ◌ٓ two keys (that means two Unicode combined 0627 + 0653, correspond to a single letter). It was necessary to normalize the text in lexicon for storing so that the order of letters in the word stored in lexicon and that of user entered for spell checking is always same. Therefore the Normalization Form C (NFC) is used for storing the lexicon.

For an example, as discussed above is a word مارآ (Aaram)

مارآ=آ + ر + ا + م

Now if the user write that word by using different keys, like,

مارآ = ا + ◌ٓ + ر + ا + م

Then it is normalized to the above form so that it can be compared with the Lexicon.

5. Spell Checker Architecture

As proposed by many other researchers[4–8]. The major components of the spell checker architecture are shown in Figure 2. The basic modules are: Pre-processing Module (which consists of Tokenization, Normalization, Remove Optional Diacritics and Code Generation), Lexicon Look-Up/Error Detection Module and Error Correction/ Suggestion Generation Module.

5.1 Pre-Processing Module

This module pre-process the user text, so that it can be formalized into the predefined format of the lexicon. This module performs the following steps:

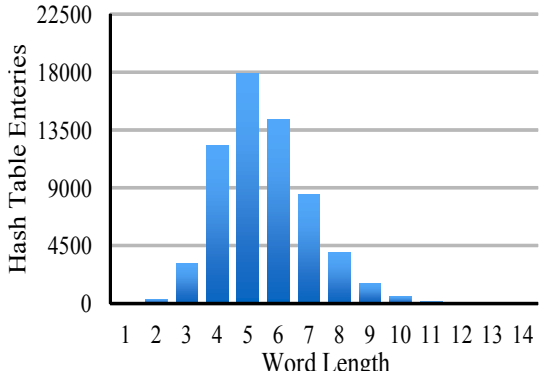


Figure 1. Dhanju: Keys in Hash Table of Different Word Length.

5.1.1 Tokenization

Tokenization is the process to break the block of text into a list of words. The text is broken with the help of some boundary delimiter and blank spaces. The boundary delimiters here are several punctuation marks. As Unicode is standardized for Shahmukhi script, so all boundary delimiter like punctuation marks, blank space etc. is considered for tokenization.

5.1.2 Word Normalization

The tokens are then made to pass through a normalization process to convert them to the format in which the lexicon has been stored. We have used CNF for this purpose. The purpose of normalization can be shown from the following example:

آؤُہَ/hua/,

This word can be written in multiple forms such as:

ہ ہَ ؤ ؤُ ا اُ orہ ہَ و وُ ا اُ

, but in Lexicon, it is stored as ہ ہَ ؤ ؤُ آ, Thus, the word آؤُہَ typed in any format will be normalized as { آ, ؤُ, ؤ, ہَ, ہ }.

5.1.3 Remove Diacritics

The normalized tokens are then passed through Remove Diacritics phase, in which the optional diacritics (as shown in Table 5) are removed from the normalized token. The purpose of this phase is to achieve a word of constant length which can be searched in a single dictionary of constant length. For an example,

آؤہ →آؤُہَ (after removing diacritics)

As this word is of constant length 'three', so it needs comparison with a dictionary of words having length 'three' only.Similarly,

تفلاتفَلاُ (after removing diacritics)

Here تفلا has 'four' characters so it will be compared with the dictionary of length 'four'.

diacritics. If match occurs then control passes to next token.

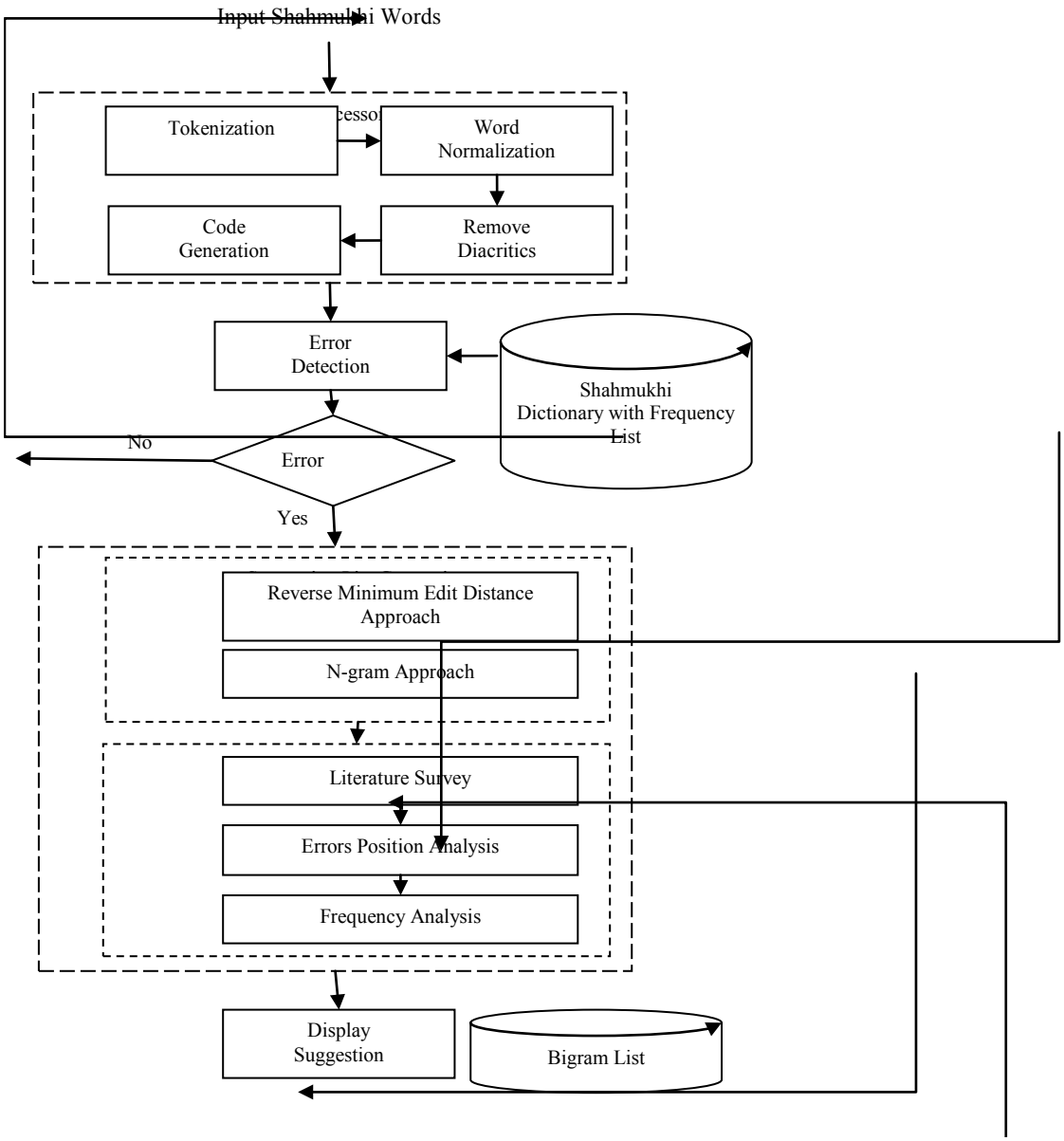
But if mismatch occurs due to other reasons like use of wrong diacritic or diacritic is placed at the wrong position, then it will considered as a miss and Error correction Module will start.

As an example, considering a wrong word آؤُہ from previous example, here ""(zabar) is placed at wrong position (after "pesh" instead of after "gol he"). So, when we compare wrong word أؤہ with right word آؤُہَ (from Table 15), here mismatch occurs and then this word is passed to the Error Correction Module. Using this approach we can easily handle Visual Error, Dual diacritic error and diacritic displacement Error.

Similarly, consider a word آؤُہ, here "" (zabar) is missing from the word in lexicon, so when we compare both words, it will not be considered as an error as the difference lies in optional diacritics and control passes to the next token.

5.2.2 Search with Diacritics

In case of Search with Diacritics, if the token does not matches with any word in the list correspond to the phonetic code, then Error correction phase starts.



As an example, consider a word آؤُہ, here "ؤ" (zabar) is not optional from the word in lexicon, so when we compare both words, it will be considered as an error as the difference lies in diacritics and in this phase, all the diacritics are compulsory. So wrong word passes to the Error Correction Module.

5.2.3 *Advantages of using Phonetic Dictionary in Error Detection Module*

Once the system has detected an erroneous word, all the words in the list we found earlier using the phonetic code, will be considered as a suggestion for the current user token as this list contain all the words that have either diacritics or phonetic differences. So at this stage while detecting error we are provided with the suggestions of most commonly occurring errors (i.e. diacritic errors and phonetic errors), which, are later feed to Ranking phase of Error Correction Module.

\For an example, In case of wrong word آؤُہ , the Table 15 is directly passed to the Ranking Phase of Error Correction Module as all the words in Table 15 have either diacritic or phonetic differences.

5.3 Error Correction Module

Once the Error Detection Module has detected an erroneous word, the erroneous word along with the previous and next word are passed to the Error correction module. Error correction Module performs the following steps:

- Suggestion Generation.
- Ranking of Suggestions.
- Suggestion Generation Phase.

We have used following approach for Suggestion Generation.
Advanced Reverse Minimum Edit Distance Approach using bi-gram to find suggestions for the wrong word.

5.3.1.1 Reverse Minimum Edit Distance Approach

We have used the reverse minimum edit distance approach to generate the primary suggestion list[8]. We generate suggestions from the wrong word by supposing Errors like

- Insertion Error: When at least one extra character is inserted in the desired word. For example,
  اُستتتسٹ اُستتت (ustat t least
- Deletion Error: When at least one character is deleted in the desired word. For example,
  اُستتتاُت (ustat least
- Substitution Error: This error occurs when at least one character is substituted by the other character. For example,
  اُستتتاُرتس (ustat rror oc
- Phonetic Error: Shahmukhi has certain characters which phonetically sounds similar and thus are reason for the misspelled words. For example,
  اُستتتاُتتتث (ustat khi has
- Transposition Error: When two adjacent characters are transposed. For example,
  اُستتتاُتسٹ (ustat o adjac
- Diacritic Error: In case of Search with Diacritics, at least one extra Diacritic is inserted or deleted in the desired word, which is considered as diacritic Error. For example,
  اُفا← اُف(uf → af as a+u=u) here ُ is the missing diacritic. These errors also give rise to real word errors. For example,اس←اِس (as the misاُس←اِس (as the m
- Run-On Error: When there is space missing between two or more valid words. For example,
  اُستتت رک نی←اُستتتکرنی (ustat karni ords).
- Split Word Error: This is opposite of Run-on error when there is some extra space is inserted between parts of a word. The error can be removed by removing the extra space. For example, اُستتت←اُستاُس(ustat -> us tat)

5.3.1.2 Advantages of using Phonetic Dictionary in Suggestion Generation Phase

We generate suggestions by inserting combination of errors ourselves in the wrong Token's Phonetic Code. Here every combination is tried but Our Combination span reduces to its 1/3rd almost because of:

- Phonetic Code: As each Symbol generally stands for three characters or more (Explained in Table 11). So by substituting single phonetic symbol we are checking for all phonetic characters corresponding to that symbol in a single match.

For e.g. consider a word"ربکا" رب e.g. conside code AKBR. If we insert phonetic symbol S in AKBR i.e. ASKBR then searching for ASKBR in hash table will be equal to searching of all form of (A= 5 characters, S=5 characters, K=4 characters, B=2 characters, R=3 characters) in a single comparison.

5.3.1.3 Advanced Reverse Minimum Edit Distance Approach using Bi-gram

We have reduced the comparisons of Reverse Minimum Edit Distance Approach by using Bi-gram. We had already found the possibility of occurrence of character after another character in each dictionary (dictionary is divided on length bases). By this we can find out the possible positions where error could have existed. Therefore it minimizes the positions where the symbols can be inserted, substituted, deleted or Trans-positioned.

If there is no possibility of occurrence of one character after the other character in user Token, then that combination is tried for reverse minimum Edit Distance Approach and the loop for all other combinations will not be tried whereas if all the bi-grams of user token exist, then every combination of Reverse Edit Distance Approach is tried.

For e.g. consider a word "رکا" having Phonetic code AKRR. The bi-gram for AKRR are: AK, KR, and RR. If we know that bi-gram 'RR' does not exist in dictionary of length 4. So either of the R can be substituted. No other symbol (i.e. A, K) can be substituted, as substituting either A or K will lead to existence of 'RR' in Phonetic Code and

Table 15. List of words correspond to a single Phonetic code

Code	Characters with similar sounds							
N	اً	ن	ں					
A	ءا	أ	ع	آ	ا			
K	ک	ھک	خ	ق				
G	غ	ھگ	گ					
C	چ	ھچ						
J	ض	ھج	ظ	ژ	ذ	ز	ج	
T	ط	ھت	ت	ھٹ	ٹ	ۃ		
D	ھڈ	ڈ	ھد	د				
P	ف	ھپ	پ					
B	ب	ھب						
M	م							
R	ڑ	ھڑ	ر					
L	ل							
S	ش	ص	ث	س				
U	ۇ	و	ؤ	ُو	ٗو			
H	ہ	ح	ۂ	ھ				
E	ے	ءے	ےَ	ئے				

Table 15. List of words correspond to a single Phonetic code

Phonetic	Shahmukhi
HUA	آؤُہَ/hua/
	اوہ/hawa/
	اوّحَ/haawa/

hence there will be no match of Phonetic Code in the dictionary. Therefore it leads to only 2 possible positions for substitution error (i.e. either of 'R').

Similarly if 'RR' does not exist in dictionary of length 4+1, then a symbol can only be inserted between 'RR'. So we do not need to try any other combination for deletion error.

We have observed that bi-gram is most successful with words of length 7 to 19.

First Character changes should be excluded as we have observed that the occurrence of error in first character is very rare. So it eliminates the change of first character in application of Reverse Minimum Edit Distance Approach.

Example of Suggestion Generation

For wrongly spelt word,

لگـن

The Suggestion list generated is:

رگـن , یگـن, مگـن,نگـن , لگـنِ

Table16. Percentage of occurrence of Error in Optional Diacritic case

Type of Error	Percentage of Occurrence
Single Error	83.23%
Double Error	6.45%
Multiple Error	2.40%
Name Entities	5.06%
Foreign words (like English or Hindi words spelled in Punjabi)	2.86%

Table 17. Percentage of occurrence of Error in Compulsory Diacritic case

Type of Error	Percentage of Occurrence
Single Error	71.75%
Double Error	11.34%
Multiple Error	13.03%
Name Entities	2.48%
Foreign words (like English or Hindi words spelled in Punjabi)	1.40%

Table 18. Percentage of occurrence of Error in Optional as well as Compulsory Diacritic case

Type of Errors	Percentage of Occurrence (With Diacritics)	Percentage of Occurrence (Without Diacritics)
Insertion Error	13.83%	28.96%
Substitution Error	19.76%	41.38%
Deletion Error	3.06%	6.40%
Transposition Error	0.19%	0.39%
'Diacritic'	54.39%	4.46%
Run-On Error	1.38%	2.91%
Split Word Error	0.32%	0.68%
Phonetic Nature of Character e.g.(ਤ ,ਟ)	7.07%	14.82%

ਗਨ , ਨਗਨ , ਹਗਨ, ਰ੍ਗਨ,ਗਨੇ,ਗਨੀ,

5.3.2 Ranking of Suggestions

Once the suggestion list has been generated, each suggestion is given weight according to the results of error analysis for Shahmukhi script carried out in detection of error patterns. According to the analysis following type of weights are assigned to the suggestions.

- Weightage to each Error according to Literature Survey.
- Weightage according to Frequency.
- Weightage according to the location of Errors.

Weightage to Each Error according to Literature Survey[11]: During Literature Survey, we have found the following hierarchy of errors which is from high occurring errors to low occurring errors.

- Diacritic Error is the most common error which occur due to non-consideration of diacritics in Shahmukhi.
- Phonetic Error mostly occurs after Diacritic Error because of the complexity of this language:
    - High and Low Tones of same sound.
    - Multiple Characters for same sound.
- Run-On Error and Split Error are the most common errors that occur due to presence of Non-joiners and typing Errors.
- Substitution Error is the next most common Error that occur due to substitution of wrong character.

- Insertion Error and Deletion Error has less probability of occurrence than above discussed errors.
- Transposition Error has lowest weightage according to survey but it is quite helpful in some special cases.

Weightage according to Frequency:

- The Results are also refined according to the frequency of their occurrence. It helps in rearranging the suggestions where a single type of error has occurred.

Weightage according to the location of Errors:

- Errors that occur at the end of the word (Token) has more weightage as compare to the error that are at beginning of the word (Token).

Example for Ranking of Suggestions:
After sorting, the list from suggestion Generation Phase, We obtain the reordered list:

نِ,لگـن ,نلگـن ,الگـنی, نگـن,لےگـن
نِ,ہگـن ,رّگـنِ ,رگـن ,یگـن,مگـن

6. Evaluation and Results

6.1 Test Words Preparation

We used most commonly mis-spelled words to analyze the performance of the spell checker. The words were drawn from several sources:

- Online Shahmukhi Newspapers.
- Online Shahmukhi stories.
- Shahmukhi Research Reports.

6.2 Test Results

- Error Analysis if Diacritics is not compulsory.
- Error Analysis if Diacritics is Compulsory
- General Error Analysis.

7. Conclusion

This is the first time that a spell checker for Shahmukhi Script has been designed and implemented. The spell checker is part of the Shahmukhi word processor. We have only taken care of non-real word errors. Detection and correction of real word errors and Izafat is a subject of further research.

8. References

1. Saini TS. Scripts of Punjabi Language: Comparative Study. Research in Shahmukhi to Gurmukhi Transliteration System: A Corpus based Approach; 2011. p. 65–93.
2. Malik MGA.Transliteration STransliteration. Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL. , Sydney; 2006 Jul. p. 1137–44.
3. A Study on Collation of Language from Developing Asia. 109–20. Available from: http://www.panl10n.net/english/outputs/Collation%20Book/Collation%20Book/Final%20Versions/pdfs/Urdu.pdf
4. Chaudhuri BB. Reversed word dictionary and phonetically similar word grouping based spell-checker to Bangla text. Proc LESAL Workshop: Mumbai; 2001.
5. Varghese ST, Kumar RR, Sulochana KG. Malayalam Spell Checker. Resource Centre forIndian Language Technology Solutions, TDIL Newsletter.
6. Mohanty S. Analysis and Design of Oriya Morphological Analyzer: Some Tests with OriNet. TDIL Newsletter.
7. Das D, Borgohain S, Gogoi J, Nair SB. Design and Implementation of a Spell Checker for Assamese. LEC, Language Engineering Conference (LEC'02); 2002 Dec 13-15. p. 156–62.
8. Singh S, Gohain L, Gogoi J, Nair SB. Design and Implementation of a journal of systemic cybernetics and informatics; 2007. p. 70–5.
9. M. G. Abbas Malik, ,gohain, JuliGogoi, S.B. Nair (2002), Design and Implementation of ajournal of systemic cybernetics and informatics, 2007, pp.70-75
10. Javaid S,Sattar H, Ali A, Malik MGA.Survey of Computational Support for Shahmukhi script of Punjabi language. Academic Research International. 2011 Jul; 1(1):292–300.
11. Iqbal S, Anwar W, Bajwa UI, Rehman Z. Survey of Compeverse Edit Distance Approach. The 4th Workshop on South and Southeast Asian NLP (WSSANLP). International Joint Conference on Natural Language Processing. Nagoya, Japan. 2011 Oct 14-18. p. 58–65.
12. Growth of Scheduled Languages-1971, 1981, 1991 and 2001. Census of India. Ministry of Home Affairs, Government of India. 2015 Feb 22.