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\(^1\). 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
Design and Implementation of Shahmukhi Spell Checker

Indian Journal of Science and Technology

Vol 8 (27) | October 2015 | www.indjst.org

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>
<thead>
<tr>
<th>Sr.</th>
<th>Shahmukhi</th>
<th>Unicode</th>
<th>Sr.</th>
<th>Shahmukhi</th>
<th>Unicode</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[bco]</td>
<td>06BE + 0628</td>
<td>5</td>
<td>[g(^3)]</td>
<td>0686 + 06BE</td>
</tr>
<tr>
<td>2</td>
<td>[p6]</td>
<td>067E + 06BE</td>
<td>7</td>
<td>[d(^f)]</td>
<td>062F + 06BE</td>
</tr>
<tr>
<td>3</td>
<td>[d(^g)]</td>
<td>062A + 06BE</td>
<td>8</td>
<td>[d(^f)]</td>
<td>0688 + 06BE</td>
</tr>
<tr>
<td>4</td>
<td>[d(^g)]</td>
<td>0679 + 06BE</td>
<td>9</td>
<td>[k(^9)]</td>
<td>06A9 + 06BE</td>
</tr>
<tr>
<td>5</td>
<td>[d(^g)]</td>
<td>062C + 06BE</td>
<td>10</td>
<td>[g(^C)]</td>
<td>06AF + 06BE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sr.</th>
<th>Shahmukhi</th>
<th>Unicode</th>
<th>Sr.</th>
<th>Shahmukhi</th>
<th>Unicode</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[b]</td>
<td>0628</td>
<td>21</td>
<td>[t]</td>
<td>0637</td>
</tr>
<tr>
<td>2</td>
<td>[p]</td>
<td>067E</td>
<td>22</td>
<td>[z]</td>
<td>0638</td>
</tr>
<tr>
<td>3</td>
<td>[t]</td>
<td>062A</td>
<td>23</td>
<td>[?]</td>
<td>0639</td>
</tr>
<tr>
<td>4</td>
<td>[t]</td>
<td>0679</td>
<td>24</td>
<td>[r]</td>
<td>063A</td>
</tr>
<tr>
<td>5</td>
<td>[s]</td>
<td>062B</td>
<td>25</td>
<td>[f]</td>
<td>0641</td>
</tr>
<tr>
<td>6</td>
<td>[d(^3)]</td>
<td>062C</td>
<td>26</td>
<td>[q]</td>
<td>0642</td>
</tr>
<tr>
<td>7</td>
<td>[t(^f)]</td>
<td>0686</td>
<td>27</td>
<td>[k]</td>
<td>06A9</td>
</tr>
<tr>
<td>8</td>
<td>[h]</td>
<td>062D</td>
<td>28</td>
<td>[g]</td>
<td>06AF</td>
</tr>
<tr>
<td>9</td>
<td>[x]</td>
<td>062E</td>
<td>29</td>
<td>[l]</td>
<td>0644</td>
</tr>
<tr>
<td>10</td>
<td>[d]</td>
<td>062F</td>
<td>30</td>
<td>[m]</td>
<td>0645</td>
</tr>
<tr>
<td>11</td>
<td>[d]</td>
<td>0688</td>
<td>31</td>
<td>[n]</td>
<td>0646</td>
</tr>
<tr>
<td>12</td>
<td>[z]</td>
<td>0632</td>
<td>32</td>
<td>[n]</td>
<td>06BB</td>
</tr>
<tr>
<td>13</td>
<td>[r]</td>
<td>0631</td>
<td>33</td>
<td>[l(^33)]</td>
<td>06BA</td>
</tr>
<tr>
<td>14</td>
<td>[t]</td>
<td>0691</td>
<td>34</td>
<td>[v]</td>
<td>0648</td>
</tr>
<tr>
<td>15</td>
<td>[z]</td>
<td>0630</td>
<td>35</td>
<td>[h]</td>
<td>06C1</td>
</tr>
<tr>
<td>16</td>
<td>[3]</td>
<td>0698</td>
<td>36</td>
<td>[h]</td>
<td>06BE</td>
</tr>
<tr>
<td>17</td>
<td>[s]</td>
<td>0633</td>
<td>37</td>
<td>[j]</td>
<td>06CC</td>
</tr>
<tr>
<td>18</td>
<td>[f]</td>
<td>0634</td>
<td>38</td>
<td>[j]</td>
<td>06D2</td>
</tr>
<tr>
<td>19</td>
<td>[s]</td>
<td>0635</td>
<td>39</td>
<td>[l]</td>
<td>0644</td>
</tr>
</tbody>
</table>

Table 1. Common aspirated consonants

Table 2. Non-Aspirated consonants
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 CHASHMA.

The remaining six aspirated consonants are: [rh],[mh],[th],[vh],[h],[nh].

<table>
<thead>
<tr>
<th>Table 3. Long vowels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unicode</td>
</tr>
<tr>
<td>0627</td>
</tr>
<tr>
<td>0622</td>
</tr>
<tr>
<td>0648</td>
</tr>
<tr>
<td>06CC</td>
</tr>
<tr>
<td>06D2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 4. Short vowels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unicode</td>
</tr>
<tr>
<td>064E</td>
</tr>
<tr>
<td>064F</td>
</tr>
<tr>
<td>0650</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 5. Optional diacritics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unicode</td>
</tr>
<tr>
<td>0656</td>
</tr>
<tr>
<td>0652</td>
</tr>
<tr>
<td>064E</td>
</tr>
<tr>
<td>0670</td>
</tr>
<tr>
<td>0650</td>
</tr>
<tr>
<td>064C</td>
</tr>
<tr>
<td>064D</td>
</tr>
<tr>
<td>0651</td>
</tr>
<tr>
<td>064F</td>
</tr>
<tr>
<td>0657</td>
</tr>
<tr>
<td>0658</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 6. Shahmukhi punctuations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Punctuations</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>06D4</td>
</tr>
<tr>
<td>021</td>
</tr>
<tr>
<td>061F</td>
</tr>
<tr>
<td>05D</td>
</tr>
<tr>
<td>061B</td>
</tr>
<tr>
<td>060D</td>
</tr>
<tr>
<td>066D</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 7. Numerals in Shahmukhi text</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numerals</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 8. Joiners in Shahmukhi Script</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unicode</td>
</tr>
<tr>
<td>062E</td>
</tr>
<tr>
<td>062D</td>
</tr>
<tr>
<td>0686</td>
</tr>
<tr>
<td>062C</td>
</tr>
<tr>
<td>0645</td>
</tr>
<tr>
<td>06A9</td>
</tr>
<tr>
<td>062B</td>
</tr>
<tr>
<td>0679</td>
</tr>
<tr>
<td>062A</td>
</tr>
<tr>
<td>0638</td>
</tr>
<tr>
<td>0636</td>
</tr>
<tr>
<td>0649</td>
</tr>
<tr>
<td>0634</td>
</tr>
<tr>
<td>0647</td>
</tr>
</tbody>
</table>

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.
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**. 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.

#### 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,

<table>
<thead>
<tr>
<th>Code</th>
<th>Shahmukhi</th>
<th>Code</th>
<th>Shahmukhi</th>
</tr>
</thead>
<tbody>
<tr>
<td>0698</td>
<td>ئ [3]</td>
<td>0622</td>
<td>[a]</td>
</tr>
<tr>
<td>0691</td>
<td>ئ [r]</td>
<td>0627</td>
<td>ئ [z]</td>
</tr>
<tr>
<td>0631</td>
<td>ئ [f]</td>
<td>06D2</td>
<td>ئ [e]</td>
</tr>
<tr>
<td>0630</td>
<td>ئ [z]</td>
<td>0648</td>
<td>ئ [v]</td>
</tr>
<tr>
<td>0688</td>
<td>ئ [d]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Unicode</th>
<th>Shahmukhi</th>
<th>Unicode</th>
<th>Shahmukhi</th>
</tr>
</thead>
<tbody>
<tr>
<td>0698</td>
<td>ئ [3]</td>
<td>0622</td>
<td>[a]</td>
</tr>
<tr>
<td>0691</td>
<td>ئ [r]</td>
<td>0627</td>
<td>ئ [z]</td>
</tr>
<tr>
<td>0631</td>
<td>ئ [f]</td>
<td>06D2</td>
<td>ئ [e]</td>
</tr>
<tr>
<td>0630</td>
<td>ئ [z]</td>
<td>0648</td>
<td>ئ [v]</td>
</tr>
<tr>
<td>0688</td>
<td>ئ [d]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 9. Non-Joiners in Shahmukhi Script

<table>
<thead>
<tr>
<th>Unicode</th>
<th>Shahmukhi</th>
<th>Unicode</th>
<th>Shahmukhi</th>
</tr>
</thead>
<tbody>
<tr>
<td>0698</td>
<td>ئ [3]</td>
<td>0622</td>
<td>[a]</td>
</tr>
<tr>
<td>0691</td>
<td>ئ [r]</td>
<td>0627</td>
<td>ئ [z]</td>
</tr>
<tr>
<td>0631</td>
<td>ئ [f]</td>
<td>06D2</td>
<td>ئ [e]</td>
</tr>
<tr>
<td>0630</td>
<td>ئ [z]</td>
<td>0648</td>
<td>ئ [v]</td>
</tr>
<tr>
<td>0688</td>
<td>ئ [d]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 10. Characters having similar phonetic Code

<table>
<thead>
<tr>
<th>S</th>
<th>س [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>ح [h]</td>
</tr>
<tr>
<td>K</td>
<td>ق [k]</td>
</tr>
<tr>
<td>J</td>
<td>ز [z]</td>
</tr>
<tr>
<td>T</td>
<td>ئ [t]</td>
</tr>
</tbody>
</table>

### Table 11. Characters having similar Shape

<table>
<thead>
<tr>
<th>ب</th>
<th>ب [b]</th>
</tr>
</thead>
<tbody>
<tr>
<td>ج</td>
<td>ج [g]</td>
</tr>
<tr>
<td>د</td>
<td>د [d]</td>
</tr>
<tr>
<td>ر</td>
<td>ر [r]</td>
</tr>
<tr>
<td>ف</td>
<td>ف [f]</td>
</tr>
<tr>
<td>ض</td>
<td>ض [z]</td>
</tr>
<tr>
<td>ط</td>
<td>ط [t]</td>
</tr>
<tr>
<td>خ</td>
<td>خ [x]</td>
</tr>
<tr>
<td>ض</td>
<td>ض [z]</td>
</tr>
<tr>
<td>ر</td>
<td>ر [r]</td>
</tr>
<tr>
<td>ت</td>
<td>ت [t]</td>
</tr>
<tr>
<td>ن</td>
<td>ن [n]</td>
</tr>
</tbody>
</table>

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

For Example,

اُمَرِكَہ /omrikh/ =/اُمَرِکَہ/omriko/

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

#### Table 12. Characters having similar Shape

<table>
<thead>
<tr>
<th>ب</th>
<th>ب [b]</th>
</tr>
</thead>
<tbody>
<tr>
<td>ج</td>
<td>ج [g]</td>
</tr>
<tr>
<td>د</td>
<td>د [d]</td>
</tr>
<tr>
<td>ر</td>
<td>ر [r]</td>
</tr>
<tr>
<td>ف</td>
<td>ف [f]</td>
</tr>
<tr>
<td>ض</td>
<td>ض [z]</td>
</tr>
<tr>
<td>ط</td>
<td>ط [t]</td>
</tr>
<tr>
<td>خ</td>
<td>خ [x]</td>
</tr>
<tr>
<td>ض</td>
<td>ض [z]</td>
</tr>
<tr>
<td>ر</td>
<td>ر [r]</td>
</tr>
<tr>
<td>ت</td>
<td>ت [t]</td>
</tr>
<tr>
<td>ن</td>
<td>ن [n]</td>
</tr>
</tbody>
</table>

For Example,

اُمَرِکَہ /omrikh/ =/اُمَرِکَہ/omriko/
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

\[
\text{r} \text{ک} \text{ھ} \text{ے} \text{ک} \text{ّ} \text{ر} = \text{ر} \text{ُ} \text{ک} \text{ّ} \text{ے} \text{ھ}
\]

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

\[
\text{r} \text{ک} \text{ھ} \text{ے} \text{ک} \text{ّ} \text{ر} = \text{ر} \text{ُ} \text{ک} \text{ّ} \text{ے} \text{ھ}
\]

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 (ڻ), 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:

\[
	ext{س} \text{n} \text{ب} \text{ا} \text{ن} \text{د} \text{ھ} \text{ی} / \text{sanbhandhi}/
\]

\[
	ext{س} \text{n} \text{ب} \text{ا} \text{ن} \text{د} \text{ھ} \text{ی} / \text{sanbhandhi}/
\]

### 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, \text{/ulfat/} word can be written as \text{شا} or \text{ش} 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: رابتعا لیباق (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:

\[
	ext{رابتعا لیباق/kabil-i-aitbar/}
\]

\[
	ext{راجب و گاب/baag-o-bahaar/}
\]

\[
	ext{مظعا مغول/ Mughal-i-azam/}
\]

\[
	ext{مرگ اگرم/ah-i-garm/}
\]

### 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.

<table>
<thead>
<tr>
<th>Unicode</th>
<th>Optional Diacritics</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0656</td>
<td></td>
<td>(khari zabar)</td>
</tr>
<tr>
<td>064E</td>
<td></td>
<td>(zabar)</td>
</tr>
<tr>
<td>0670</td>
<td></td>
<td>(Superscript Alef)</td>
</tr>
<tr>
<td>0650</td>
<td></td>
<td>(zer)</td>
</tr>
<tr>
<td>0651</td>
<td></td>
<td>(tasdid)</td>
</tr>
<tr>
<td>064F</td>
<td></td>
<td>(pesh)</td>
</tr>
</tbody>
</table>

Table 12. Characters having zero width
In Assamese spell checker, 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. 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 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, ना वाला/ having two optional diacritic have four variations ना वाला/ ना वाला/ ना वाला/ ना वाला/ but in lexicon 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 "With diacritic" spell checking. For the consideration of all the cases, words having all the diacritic marks is stored in the lexicon.

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.

### 4.2 Format of the Words

In Unicode, there are more than one code points for a single letter. For example, अ (alef-madda) letter can be written by a single key (that means single Unicode 0622) and अ + ી, 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 मार + व (Aaram)

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

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. 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:
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:

\[ \text{ہا} / hua/ \]

This word can be written in multiple forms such as:

\[ ا \, ج \, ہ \, ا \, ج \, ہ \]

but in Lexicon, it is stored as: 
\[ ا \, ج \, ہ \, ا \, ج \, ہ \]

Thus, the word \( \text{ہا} \) typed in any format will be normalized as \{ ا, ج, ہ, ا \}.

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, 

\[ \text{ہا} \rightarrow \text{ہا} \] (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, 
\[ 
\text{سغلا} \rightarrow \text{سغلا} 
\] (after removing diacritics)

Here \( \text{سغلا} \) has ‘four’ characters so it will be compared with the dictionary of length ‘four’.

5.2 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 ؤہُ آَ, here “ُ” (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 diacritic 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 ؤُآَ, 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\(^4\). 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, 
  Ṣṣṭa → Ṣṣṭa (ustat t least)
- Deletion Error: When at least one character is deleted in the desired word. For example, 
  Ṣṣṭa → Ṣṣṭa (ustat least)
- Substitution Error: This error occurs when at least one character is substituted by the other character. For example, 
  Ṣṣṭa → Ṣṣṭa (ustat rorr oc)
- Phonetic Error: Shahmukhi has certain characters which phonetically sounds similar and thus are reason for the misspelled words. For example, 
  Ṣṣṭa → Ṣṣṭa (ustat khi has)
- Transposition Error: When two adjacent characters are transposed. For example, 
  Ṣṣṭa → Ṣṣṭa (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, 
  Ṣṣṭa → Ṣṣṭa (uṣṭa → uṣṭ) where 'f' is the missing diacritic. These errors also give rise to real word errors. For example, 
  Ṣṣṭa → Ṣṣṭa (as the misṣṣod as the m)
- Run-On Error: When there is space missing between two or more valid words. For example, 
  Ṣṣṭa Ṣṣṭa → Ṣṣṭa Ṣṣṭa (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, Ṣṣṭa Ṣṣṭa → Ṣṣṭa Ṣṣṭa (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 "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 "رزگا" 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
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, لگن

The Suggestion list generated is:

\[\text{لگن, نگن, مگن, یگن, رگن}\]

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

<table>
<thead>
<tr>
<th>Code</th>
<th>Characters with similar sounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>ان آ اا ان ان</td>
</tr>
<tr>
<td>A</td>
<td>ع ع ع ع ع</td>
</tr>
<tr>
<td>K</td>
<td>خ خ خ خ خ</td>
</tr>
<tr>
<td>G</td>
<td>ن ن ن ن ن</td>
</tr>
<tr>
<td>C</td>
<td>ج ج ج ج ج</td>
</tr>
<tr>
<td>J</td>
<td>ز ز ز ز ض ض</td>
</tr>
<tr>
<td>T</td>
<td>ت ت ت ت ط ط</td>
</tr>
<tr>
<td>D</td>
<td>د د د د ه ه</td>
</tr>
<tr>
<td>P</td>
<td>هب هب هب ف ف</td>
</tr>
<tr>
<td>B</td>
<td>بب بب بب</td>
</tr>
<tr>
<td>M</td>
<td>م م م م</td>
</tr>
<tr>
<td>R</td>
<td>ر ر ر ر ز ز</td>
</tr>
<tr>
<td>L</td>
<td>ل ل ل ل</td>
</tr>
<tr>
<td>S</td>
<td>س ص ص ص</td>
</tr>
<tr>
<td>U</td>
<td>و و و و و</td>
</tr>
<tr>
<td>H</td>
<td>ح ح ح ح ح</td>
</tr>
<tr>
<td>E</td>
<td>ئ ئ ئ ئ ئ</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Phonetic</th>
<th>Shahmukhi</th>
</tr>
</thead>
<tbody>
<tr>
<td>HUA</td>
<td>اوُ/hua/</td>
</tr>
<tr>
<td></td>
<td>اوُ/hawa/</td>
</tr>
<tr>
<td></td>
<td>اوُ/haawa/</td>
</tr>
</tbody>
</table>
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: 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.

<table>
<thead>
<tr>
<th>Table 16. Percentage of occurrence of Error in Optional Diacritic case</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type of Error</strong></td>
</tr>
<tr>
<td>Single Error</td>
</tr>
<tr>
<td>Double Error</td>
</tr>
<tr>
<td>Multiple Error</td>
</tr>
<tr>
<td>Name Entities</td>
</tr>
<tr>
<td>Foreign words (like English or Hindi words spelled in Punjabi)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 17. Percentage of occurrence of Error in Compulsory Diacritic case</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type of Error</strong></td>
</tr>
<tr>
<td>Single Error</td>
</tr>
<tr>
<td>Double Error</td>
</tr>
<tr>
<td>Multiple Error</td>
</tr>
<tr>
<td>Name Entities</td>
</tr>
<tr>
<td>Foreign words (like English or Hindi words spelled in Punjabi)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 18. Percentage of occurrence of Error in Optional as well as Compulsory Diacritic case</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type of Errors</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Insertion Error</td>
</tr>
<tr>
<td>Substitution Error</td>
</tr>
<tr>
<td>Deletion Error</td>
</tr>
<tr>
<td>Transposition Error</td>
</tr>
<tr>
<td>Diacritic</td>
</tr>
<tr>
<td>Run-On Error</td>
</tr>
<tr>
<td>Split Word Error</td>
</tr>
<tr>
<td>Phonetic Nature of Character e.g. (ṱ, ´ṱ)</td>
</tr>
</tbody>
</table>
• 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:


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