Design of Odia Spell Checker with word Prediction

Amrutanshu Pradhan
Dept.of IT
College of Engineering & Technology,
Bhubaneswar, India

Sasanka Sekhar Dalai
Dept.of CSE
Einstein Academy of Technology & Management,
Bhubaneswar, India

***Abstract*—Spell checking means to detect the error and correct the error. Spell checking is a well known task in natural language processing. Spelling error detection and correction is the process that will check the spelling of words in a document and in occurrence of any error, list out the correct spelling in the form of suggestions. Spelling checker tools use a dictionary as a database. Every word from the text is looked up in the dictionary. When a word is not present in the dictionary it will be treated as error. To overcome the error spell checker searches the dictionary for words that resemble the erroneous word most. These words are listed as suggestions for that error word and the use has to choose the best word from the list of suggestions. In this work a dictionary is created which contains odia words. Edit distance algorithm is implemented for the design of the Odia spell checker with prediction.**

Keywords—Spell checking,word prediction,edit distance

I. INTRODUCTION

Spell checking is a well-known task in Natural Language Processing which deals with error detection and correction of spelling errors. Spell checker will check the spelling of words in a document, and in occurrence of any error, a list of words will be shown as suggestion. This is a NLP problem which deals with spelling error correction. An error detector detects misspelled words, a candidate spelling generator that provides spelling suggestions for the detected misspelled word. All spelling checker tools use a dictionary as a database. The written text is looked up in the dictionary. When a word is could not find in the dictionary, it is detected as an error. To correct that error, a spell checker searches the dictionary for words that resemble the erroneous word most. These words are then listed as suggestions to the user who chooses the best word that was expected. There are two main steps in a spell checker, are Error detection and Error correction.

All the misspelled words are classified into two levels such as word level and sentence level error. At the character level the error patterns are formed. Detection of real word error need to compared with the detection of non word error. Several approaches based on Minimum edit distance, Similarity key, rules and N-grams are proposed to accomplish the task. Non-word errors can be classified into four major types such as substitution error, deletion error, insertion error, and transposition error. Most of the misspellings take place by slips or omissions, like slips of matras (matra or phala). Complete omission of vowel diacritical markers or some part of the diacritical markers. Wrong use of vowel diacritical markers is also noticed. Wrong uses of characters, which are phonetically similar to the correct ones, have also been observed. In the case of compound consonants (called yuktaksar in Odia), mistakes takes place are due to ignorance .A great deal of confusion has also been seen to occur in the use of dental and cerebral nasal consonant. As a result there is a chance of committing mistakes if proper spelling rules are not remembered. Similar confusion is seen in the case of three sibilant sounds in Odia (palatal, dental and cerebral).From our observation it is clear that most of the misspellings are due to

(i) phonetic similarity of Odia characters,
(ii) the difference between the graphemic representation and phonetic utterances, and
(iii) lack of proper knowledge of spelling rules.

II. SPELLING CORRECTION

It is a process of detecting and sometimes providing suggestions for incorrectly spelled words in a text. In computing, Spell Checker is an application program that flags words in a document that may not be spelled correctly. Spell Checker may be stand alone capable of operating on a block a text such as word processor, electronic dictionary. Spelling errors can be divided into two categories: Real word errors and Non word errors. Real word errors are those error words that are acceptable words in the dictionary. Non word errors are those error words that cannot be found in the dictionary. Basically spell checking consist of two functions i.e

A. Error Detection.
B. Error Correction.

A. Error Detection

Error Detection means to detect the error. A string of character is separated by space bar or punctuation marks may be called a candidate words. A candidate word is a valid if it has a meaning else it is a non word. The error detection process usually consists of checking to see if an input string is a valid index or dictionary word. Efficient techniques have been devised for detecting such types of errors. Spellcheckers rely mostly on dictionary lookup.

B. Error Correction

Error Correction means to correct the error. Correction of spelling errors is an old problem. Much research has been done in this area over the years. Most existing spelling correction techniques focus on isolated words, without taking into account the textual context in which the string appears. Error correction consists of two steps: the generation of candidate corrections and the ranking of candidate corrections. The ranking process usually invokes some lexical similarity measure between the misspelled string and the

candidates or a probabilistic estimate of the likelihood of the correction to rank order the candidates. These two steps are most of the time treated as a separate process and executed in sequence. The isolated-word methods that will be described here are the most studied spelling correction algorithms, they are: edit distance, similarity keys, rule-based techniques, n-gram-based techniques, probabilistic techniques and neural networks. All of these methods can be thought of as calculating a distance between the misspelled word and each word in the dictionary or index. The shorter the distance the higher the dictionary word is ranked.

Types of error correction

- Automatic Error Correction

Error is automatically replaced with correction without user intervention and spell checker should be able to provide one best correction suggestion.

- Interactive Error Correction

User can interactively select one of the suggested corrections for replacement spellchecker can suggest multiple corrections.

III. DESIGN OF SPELL CHECKER

A basic spell checker carries out the following processes:

- It scans the text and extracts the words contained in it.
- It then compares each word with a known list of correctly spelled words (i.e. a dictionary).

An additional step is a language-dependent algorithm for handling morphology. Even for a lightly inflected language like English, the spell-checker will need to consider different forms of the same word, such as plurals, verbal forms, contractions, and possessives. For many other languages, such as those featuring agglutination and more complex declension and conjugation, this part of the process is more complicated.

It is unclear whether morphological analysis—allowing for many different forms of a word depending on its grammatical role—provides a significant benefit for English, though its benefits for highly synthetic languages such as German, Hungarian or Turkish are clear.

As an adjunct to these components, the program's user interface will allow users to approve or reject replacements and modify the program's operation.

An alternative type of spell checker uses solely statistical information, such as n-grams, to recognize errors instead of correctly-spelled words. This approach usually requires a lot of effort to obtain sufficient statistical information. Key advantages include needing less runtime storage and the ability to correct errors in words that are not included in a dictionary.

The flow chart of the process of Spell Checker is as follows:-

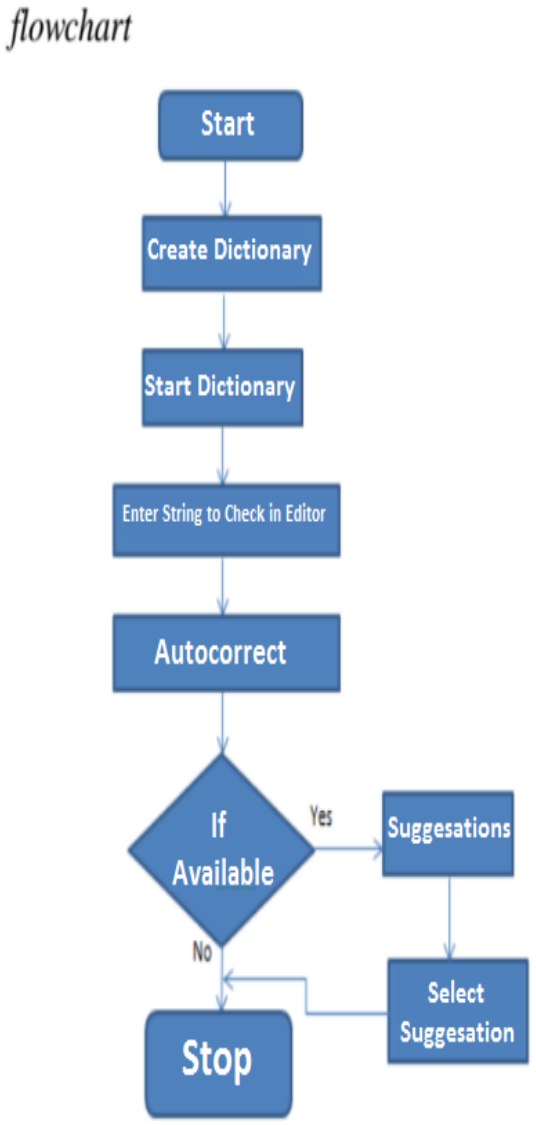

Figure.1 The Process of Spell Checker

IV. PROPOSED WORK

Edit Distance Algorithm is implemented using Java Programming Language.

- First we have to take odia words as corpus.
- The corpus will act as a dictionary, for that we have to take large number of odia articles from the newspaper.
- By removing the multiple words from that we can get odia words which will be a good dictionary.
- While typing an odia word that will go through dictionary look up method.
- If that word is present in the dictionary then no error, but if that word is not present in the dictionary then that word is a misspelled word.
- Now edit distance will be calculated with the input word with the words present in the dictionary.

- After calculating the edit distance, those who have edit distance 1(which is very similar to the input word) will be shown as suggestions
- User has to take the suitable words from the list of suggestions.

Edit distance Algorithm

In computational linguistics and computer science, edit distance is a way of quantifying how dissimilar two strings (e.g., words) are to one another by counting the minimum number of operations required to transform one string into the other. Edit distances find applications in natural language processing, where automatic spelling correction can determine candidate corrections for a misspelled word by selecting words from a dictionary that have a low distance to the word in question.

```
EDITDISTANCE(s1, s2)
 1  int m[i, j] = 0
 2  for i ← 1 to |s1|
 3  do m[i, 0] = i
 4  for j ← 1 to |s2|
 5  do m[0, j] = j
 6  for i ← 1 to |s1|
 7  do for j ← 1 to |s2|
 8     do m[i, j] = min{m[i − 1, j − 1] + if (s1[i] = s2[j]) then 0 else 1fi,
 9                      m[i − 1, j] + 1,
10                      m[i, j − 1] + 1}
11  return m[|s1|, |s2|]
```

Edit distance algorithm for calculating edit distance between S1,S2

Given two character strings s1and s2 the edit distance between them is the minimum number of *edit operations* required to transform s1 into s2.ost commonly, the edit operations allowed for this purpose are: (i) insert a character into a string;(ii) delete a character from a string and (iii) replace a character of a string by another character; for these operations, edit distance is sometimes known as *distance* . For example, the edit distance between cat and dog is 3. In fact, the notion of edit distance can be generalized to allowing different weights for different kinds of edit operations, for instance a higher weight may be placed on replacing the character s by the character p, than on replacing it by the character a (the latter being closer to s on the keyboard). Setting weights in this way depending on the likelihood of letters substituting for each other is very effective in practice . However, the remainder of our treatment here will focus on the case in which all edit operations have the same weight.

The idea is to use the dynamic programming algorithm, where the characters in s1 and s2 are given in array form. The algorithm fills the (integer) entries in a matrix m whose two dimensions equal the lengths of the two strings whose edit distances is being computed; the (i,j) entry of the matrix will hold (after the algorithm is executed) the edit distance between the strings consisting of the first i characters of s1 and the first j characters of s2,where the three quantities whose minimum is taken correspond to substituting a character in s1, inserting a character in s1 and inserting a character in s2.

The spelling correction problem however demands more than computing edit distance: given a set S of strings (corresponding to terms in the vocabulary) and a query string q, we seek the string(s) in v of least edit distance from q. We may view this as a decoding problem, in which the codeword (the strings in V) are prescribed in advance. The obvious way of doing this is to compute the edit distance from q to each string in V, before selecting the string(s) of minimum edit distance. This exhaustive search is inordinately expensive. Accordingly, a number of heuristics are used in practice to efficiently retrieve vocabulary terms likely to have low edit distance to the query term(s).

The simplest such heuristic is to restrict the search to dictionary terms beginning with the same letter as the query string; the hope would be that spelling errors do not occur in the first character of the query. A more sophisticated variant of this heuristic is to use a version of the permuterm index, in which we omit the end-of-word symbol $. Consider the set of all rotations of the query string q. For each rotation r from this set, we traverse the B-tree into the permuterm index, thereby retrieving all dictionary terms that have a rotation beginning with . ‘r’ For instance, if q is mase and we consider the rotation r=sema, we would retrieve dictionary terms such as semantic and semaphore that do not have a small edit distance to q. Unfortunately, we would miss more pertinent dictionary terms such as mare and mane. To address this, we refine this rotation scheme: for each rotation, we omit a suffix of l characters before performing the B-tree traversal. This ensures that each term in the set R of terms retrieved from the dictionary includes a ``long" substring in common with q. The value of l could depend on the length of q. Alternatively, we may set it to a fixed constant such as 2.

The Edit distance algorithm is programmed using Java Programming Language. In this work a dictionary is created that contains Odia words. Some odia words are taken from a odia newspaper, extract words from it, removed multiple words and written back to a file named “odiaWord.txt”. The dictionary contains 36000 odia words.

First the program will check, the given word with the dictionary words, If the word is present in the dictionary, then the word is correct. If not, Edit Distance is calculated. In the Edit Distance calculation, each word is being checked with previously defined words in odiaWords.txt & It will give suggestions for the wrong words. Those Edit Distances are either One or Two. Then I have implemented this concept in paragraphs. If a wrong word is detected, then suggestions will be given, by the program.

The Edit distance algorithm is to find the distance between two words. If the two words are same then the edit distance between two words is zero ,If both the words are different then edit distance is calculated .This concept is used to built a spell checker .First process is string matching, User will give a word and the program will check that word with the dictionary words. Edit Distance algorithm is also helps to predict the word for a wrong word. In the program the word given by the user is compared with all the words present in the dictionary and the edit distance is calculated .For an example suppose edit distance of a word is 1 means if is very similar to the word given by the user, That concept is used for the word prediction.

V. SIMULATION RESULT & FUTURE WORK

We have developed a model program for Odia Spell Checker, which acts as a intermediate between user and system to rectifier the odia script error. It also predicates the similar word or synonyms for inputted odia character. We have evaluated the edit distance method between the odia character in order to detect the wrong words as well as suggest the correct word. We have implemented all the necessary steps and a satisfactory result is achieved. In future work a lot of algorithm have to be implemented to built odia spell checker and also a GUI application based model is to be carried out.

REFERENCES

[1] Vibhakti V. Bhaire, Ashiki A. Jadhav, Pradnya A. Pashte and Mr. Magdum P.G "SPELL CHECKER" International Journal of Scientific and Research Publications, Volume 5, Issue 4, April 2015

[2] Amanjot Kaur, Dr. Paramjeet Singh and Dr. Shaveta Rani "Spell Checking and Error Correcting System for text paragraphs written in Punjabi Language using Hybrid approach" International Journal Of Engineering And Computer Science ISSN:2319-7242 Volume 3 Issue 9 September 2014 Page No. 8030-8032.

[3] Ambili T, Panchami K S and Neethu Subash "Automatic Error Detection and Correction in Malayalam" IJSTE - International Journal of Science Technology & Engineering | Volume 3 | Issue 02 | August 2016

[4] Neha Gupta and Pratistha Mathur "Spell Checking Techniques in NLP: A Survey" International Journal of Advanced Research in Computer Science and Software Engineering.

[5] Saadat Iqbal, Waqas Anwar, Usama Ijaz Bajwa and Zobia Rehman "Urdu Spell Checking: Reverse Edit Distance Approach" The 4th Workshop on South and Southeast Asian NLP (WSSANLP), International Joint Conference on Natural Language Processing, pages 58–65, Nagoya, Japan, 14-18 October 2013.

[6] Khyati Shah, Mayuri Patel, Jikitsha Sheth and Dr. kalpesh Lad " Comparative Study of Spell Checking Algorithms and Tools" Volume 3, No. 3, May-June 2012 International Journal of Advanced Research in Computer Science.

[7] Baljeet kaur and Harsharndeep Singh "Design and Implementation of HINSPELL -Hindi Spell Checker using Hybrid approach" International Journal of scientific research and management (IJSRM) ||Volume||3||Issue||2||Pages|| 2058-2061-2015

[8] Ritika Mishra and Navjot Kaur "A Survey of Spelling Error Detection and Correction Techniques" International Journal of Computer Trends and Technology- volume4Issue3- 2013

[9] Rajashekara Murthy S,Vadiraj Madi and Sachin D "A Non Word Kannad aSpell Checker Using Morphological AND Dictionary Lookup Method" International Journal of Engineering Sciences & Emerging Technologies, June 2012 Volume 2, Issue 2, pp: 43-52.

[10] Sumreet Kaur Randhawa1 ,Charanjiv Singh Saroa "STUDY OF SPELL CHECKING TECHNIQUES AND AVAILABLE SPELL CHECKERS IN REGIONAL LANGUAGES: A SURVEY" International Journal For Technological Research In Engineering Volume 2, Issue 3, November-2014

[11] https://en.wikipedia.org/wiki/Edit_distance

[12] https://en.wikipedia.org/wiki/Odia_language

[13] https://en.wikipedia.org/wiki/Unicode

[14] http://www.unicode.org/charts/