DOI: http://dx.doi.org/10.26483/ijarcs.v8i7.4364

ISSN No. 0976-5697

Volume 8, No. 7, July – August 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

N-GRAMS SOLUTION FOR ERROR DETECTION AND CORRECTION IN HINDI LANGUAGE

Shailza Kanwar, Manoj Sachan and Gurpreet Singh
Dept. of CSE, SLIET Longowal
Sangrur, India

Abstract: Hindi is the National language of India, which is still in its early stage of research and development regarding natural language processing applications in comparison to other languages like English, Chinese. Natural language processing is a field of Artificial Intelligence, which includes major tasks such as information retrieval, word segmentation, speech recognition, parsing, part of speech tagging, text classification, automatic text summarization etc. Spelling detection and correction in Hindi language is an important task of NLP which has not gotten sufficient attention till date. Spelling detection and correction for Indian languages such as Hindi is considered as a difficult task. Hindi Language is very different from English language in its phonetic properties and grammatical rules. Thus the existing techniques and methods that are being used to check the errors in English language can't be used for Hindi Language. There are mainly two types of error: Non word error and real word error. Error detection for non-word error in Hindi language has been done but for real word error no work has been done till date. This paper focused on Real word spelling error detection and correction in Hindi text by using N Grams Model and Levensthein edit distance algorithm..

Keywords: Natural Language Processing, Real Word Error, Spell Checking, N Grams

I. INTRODUCTION

Language processing is a very complex area of research. Automatic processing on language contents by machines directly has been considered as one of the most difficult tasks [1,2]. Till the time a lots of research work has been done on English and other languages and still more research work is going on for these languages to make them more user friendly, however very less work has been done for Hindi language to making it simpler, easier, corrective, understandable and computerized language. So it becomes more and more important to make a Hindi language user friendly as the government is emphasizing to increase its official use in letters and notifications. So this is my first step to making it more user friendly. The Spelling detection and correction techniques have an important role in various applications such as search engines, text editors and NLP tools like OCR, machine translations, text authoring systems etc. Now a days, there are several word processing applications are in use, so the solutions for spelling error correction become more important to provide accurate and quality information through text [12]. There are lots of applications and research available for English spelling detection and correction, but for Hindi very less work has been done.

English language has 21 consonants and 5 vowels, but Hindi language has 21 consonants, 10 vowels, 10 vowel sign (modifier), half characters and halant etc. So the methods applied to English language can't be directly applied to Hindi Language. Hindi language has an ambiguity due to presence of matras and many other symbols which confuses the user in writing the correct word for example (□□, □□) (□□□□, □□□) [13-14]. Both the words in pair looks similar, but have different meaning. User sometimes mistype the character and the resultant word leads to an error.

Spelling detection and correction techniques are used in Spell checker which are the basic tools required for word processing and document preparation. A spell checker is a tool which check the spellings in the text file, validates 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, suggests possible alternatives.

The main steps performed by the spell checker are:

1. Take the word from the file as its input.
2. Pre-process the word
3. Look for the word in dictionary
4. In case, the word exists, pass onto the next one.
5. If the word is not found, then seek for the closest matching patterns and put them up in the form of suggestions [11]

Spelling errors are broadly classified in two main categories: Non word Error and Real word error. Non word errors are those errors which do not exist in the dictionary. These errors have no meaning and can be detected easily by the spell checker. Real word occurs when a user mistakenly type the correct spelt word when the other word intended. These errors are difficult to find out as they exist in the dictionary, but incorrect according to the context. In this paper, author focused on correction and detection of real word errors in Hindi text.

This paper is organized as follow: Section 2 covers a brief overview of the related work. We describe our proposed method in Section 4. Evaluation and experimental results are discussed in Section 5. We conclude in Section 6.

II. RELATED WORK

Several methods have been proposed to handle real word spelling error problem. They are mainly based on either semantic information or machine learning and statistical

method. Among them Fossati, D., & Di Eugenio, B. (2007)[5], address the problem of real-word spell checking. They proposed a methodology based on a mixed trigrams language model. Their experiments show promising results with respect to the hit rates of both detection and correction, even though the false positive rate is still high. Wilcox-O'Hearn, A., Hirst, G., & Budanitsky, A. (2008) [6] proposed a model which uses trigram probabilities to detect errors. Their method identifies words that are semantically related to their context by calculating the semantic distance between words. They achieved recall and precision of 23%-50%, 18%-25% respectively. Bassil, Y. (2012) [7] proposed a parallel shared-memory spell-checking algorithm that uses rich real-world word statistics from Yahoo! N-Grams Dataset to correct non-word and real-word errors in computer text. Jayalatharachchi, E. et. Al (2012, December) [8] use the combination of n-gram and minimum edit distance based data-driven spelling detection and correction algorithm to improve the spell checking coverage and accuracy. Samanta, P., & Chaudhuri, B. B. (2013) [9] presented a real-word error detection and correction using local word bigram and trigrams.Jain, A., & Jain, M. (2014, September) [10] proposed the techniques to detect and correct the non-word spelling error for Hindi.Baljeet Kaur and Harsharndeep Singh [2015] [11] presented "Design and Implementation of HINSPELL -Hindi Spell Checker using Hybrid approach" implementing Dictionary lookup technique for detecting errors and Minimum Edit Distance for correction.

Based on spelling error detection and correction, a lot of work is available for English language, but for Hindi language not much work is available.

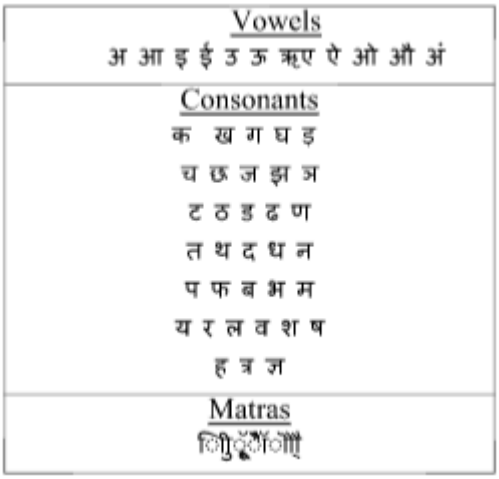
III. OVERVIEW OF HINDI LANGUAGE

Hindi is the National language of India which is spoken in different parts of India. Hindi is written in Devanagari script. Some of the major features of Devanagari script are:

- Type of writing system: alphasyllabary / abugida (**alphasyllabary/ abugida**, is a segmental writing system in which consonant–vowel sequences are written as a unit: each unit is based on a consonant letter, and vowel notation is secondary.)
- Direction of writing: left to right in horizontal lines.
- Consonant letters carry an inherent vowel which can be altered or muted by means of diacritics or *matra*.
- Vowels can be written as independent letters, or by using a variety of diacritical marks which are written above, below, before or after the consonant they belong to. This feature is common to most of the alphabets of South and South East Asia.
- When consonants occur together in clusters, special conjunct letters are used.
- The order of the letters is based on articulatory phonetics.
- There are total 47 primary characters Devanagari script, out of which 14 characters are vowels and 33 letters are consonants

IV. PROPOSED METHOD

Our proposed method initially tokenizes the user input sentence into words and then creates confusion set for each candidate word by using Levenshtein distance equals to two from the corpus. Then it calculates the probability of the elements of the confusion set by using bigrams and trigrams corpus. On basis of this error is detected and words are suggested. The detailed description of the proposed method is presented below.



Tokenization

Tokenization is the process to break the block of text into a list of words. The text is broken with help of some boundary delimiter and blank space. The boundary delimiters here are several punctuation marks. The Boundary delimiter could differ from font to font. In ASCII font several punctuation marks are part of the text but it is not case in UNICODE encoding system. But there are certain punctuation delimiters which are to be ignored. One of them and the most important one is hyphen (-). Some words in the dictionary contains hyphen. Thus, hyphen can't be used as a boundary delimiter

Creation of Confusion Set

A Confusion set for the main word E^i is a set of words (E_1^i, E_2^i ..., $E_{j\ i}$,...,E_{ki}^i) which contains words that are likely to be confused with the main word [15]. These words are convertible to the main word with single edit operation. We use Levenshtein Distance, which is the minimum number of edit operations required to convert one word into another. An edit operation is either asubstitution, deletion or aninsertionof a character in the word.The confusion set is represented as

$S(E^i)=\{E_1^i, E_2^i ...,E_{j\ i\ ,}...,E_{ni}^i$

- where $E^i$is the i-th word in the test sentence and n_i is number of elements in the set*S(Ei)*..

Estimation ofN-Gram Probabilities

The Probability of the sentence in bigram model is calculated by using Hidden Markov Chain rule. According to Markov rule, probability of an upcoming event (next word) depends upon prior events (previous words).[7] For example in a bigram language model for a sentence of m words $E_1,E_2,...,E_m$ it can be calculated as

$P(E^1,E^2,...,E^m)= P(E^1|Ь)P(E^2|E^1)P(E^3|E^2)...P(E^m|E^{m-1})P(Ь|E^m)$
where Ь denotes blank.

In our model, probability of sentence is not calculated. We are taking the hidden Markov model assumptionthat the appearance of any event (word) depends upon it next and previous words only and independent of any other event. [7]. By using Maximum Likelihood Estimation, we get the left, right bigrams and trigrams probabilities as follows

$$P_1\left(E_j^i \mid E^{i-1}\right) = \frac{count\left(E^{i-1}E_j^i\right)}{\sum_{r=0}^{ki} count\left(E^{i-1}E_r^i\right)} \quad (1)$$

$$P_2\left(E_j^i \mid E^{i+1}\right) = \frac{count\left(E^{i+1}E_j^i\right)}{\sum_{r=0}^{ki} count\left(E_r^i E^{i+1}\right)} \quad (2)$$

$$P_3\left(E_j^i \mid E^{i-1}, E^{i+1}\right) = \frac{count\left(E^{i-1}E_j^i E^{i+1}\right)}{\sum_{r=0}^{ki} count\left(E^{i-1}E_r^i\right)} \quad (3)$$

By equation (1), we calculate P_1 of each element of confusion set for each word using left bigram count. The denominator here represents the summation of all bigrams consisting of the previous word and one word from the confusion set. The frequency of words is given in Bigrams corpus. Similarly, we compute P_2 of each element using right bigram count in the corpus by equation (2). We compute it for every element in respective confusion set so that the following condition is satisfied:

$$\sum_{r=0}^{ki} P_1\left(E_r^i | E^{i-1}\right) = 1$$

$$\sum_{r=0}^{ki} P_2\left(E_r^i | E^{i+1}\right) = 1$$

$$\sum_{r=0}^{ki} P_3\left(E_r^i | E^{i-1}, E^{i+1}\right) = 1$$

We combine the probability estimates of equations (1), (2) and (3) into a score of evidence that a may be correct alternative to . The score can be obtained by simple addition as follows. The values obtained from equation (1) and (2) can be combined to get the final score by adding up.

$$Score\left(E_j^i\right) = P_1\left(E_j^i \mid E^{i-1}\right) + P_2\left(E_j^i \mid E^{i+1}\right) + P_3\left(E_j^i \mid E^{i-1}, E^{i+1}\right) \quad (3)$$

Error Detection

Consider a sentence as an example "□□□□□□ □□□□□ □□□□□ □□□□ □□□" The word '□□□□□' is a real word error in this sentence. In the proposed method, the system initially tokenizes the sentence into words and then starts processing each word one by one. While processing the word '□□□□□', we have the confusion set {□□□□□□, □□□□, □□□□} For each of these words we calculate the probability score and arrange it in decreasing order of score, as shown in Table1.

Considering one more example: "□□□□□□□ □□ □□□□□□ आ □□□ □□□"Here word '□□□□□□'is a real word error. The correct word

against '□□□□□□' is '□□□□□□'.The word '□□□□□□' means extent and the word '□□□□□□' means result. While processing the word '□□□□□□', we have the confusion set {□□□□□□,□□□□□□, □□□□□□}. The probability score of each word with rank and frequency is shown in Table 2

Table 1: Confusion Set for word □□□□□ with rank, frequency and score

Rank	Confusion Set Word	Frequency	Score
1	□□□□	9	0.52
2	□□□□□□	6	0.35
3	माता	2	0.11

Table 2: Confusion Set for word '□□□□□□' with rank, frequency and score

Rank	Confusion Word, Next Word	Frequency	Score
1	परिणाम	9	0.81
2	परिमाण	1	0.09
3	प्रणाम	1	0.09
4	परिनाम	0	0

V. RESULTS AND DISCUSSIONS

In order to assess our method, we collected test data which containapproximately 1000 words. There are total 150 real word errorspresent in the test data. The real-word error in our test data are generated by single operation like substitution, deletion or insertion. People make single position character mistake during typing which becomes real-word error. The performance of our approach can be examined by two metrics: PrecisionandRecall. These measures are defined as

$$Precision = \frac{Number\ of\ errors\ correctly\ detected}{Total\ Number\ of\ errors}$$

$$Recall = \frac{Number\ of\ errors\ correctly\ detected}{Number\ of\ errors\ detected}$$

Precision is a measure of the exactness of the spellchecker's responses and recall is a measure of the completeness of the spellchecker. The number errors not detected by the system can be calculated as

Non Detected errors = Total Number of errors- Number of errors correctly detected

Hence Percent of non-detected errors =

$$\frac{Non-detected\ errors}{Total\ Number\ of\ words} * 100\ \%$$

Table 3 shows Precision, Recall and percentage of erroneous detection

Detected Real Word Error		**Non Detected Error**
Precision	**Recall**	
88 %	90%	1%- 3%

VI. CONCLUSION

Since no work has been done for detecting and correcting real word errors in hindi language so far. the results found shows that this method of detecting and correcting real word errors proved to be very useful for hindi language. this method will make the hindi language as the world class language, which helps to use the hindi language in computer world like presently used english language. we have used bigrams and trigrams data set to detect and correct the errors which makes our proposed method to achieve a precision of 88% and recall of 90%. the bigrams and trigrams data set used in this method are not vast, hence many valid bigrams and trigrams are not present are in it.

VII. REFERENCES

[1]. Sachan, M.K., Lehal, G.S., Jain, V.K. (2011) 'A Novel Method to Segment Online Gurmukhi Script', Proceedings of International Conference on Information Systems for Indian Languages, ICISIL 2011, Patiala, Communications in Computer and Information Science, Springer-Verlag Berlin Heidelberg, Germany,Vol. 139, pp. 1-8.

[2]. Sachan, M.K., Lehal, G.S., Jain, V.K. (2011), 'A System for Online Gurmukhi Script Recognition', Proceedings of International Conference on Information Systems for Indian Languages, ICISIL 2011, Patiala, Communications in Computer and Information Science, Springer-Verlag Berlin Heidelberg, Germany, Vol. 139, pp. 294-295.

[3]. F. J. Damerau. A technique for computer detection and correction of spelling errors, communication of ACM, 7(3), pages 171-176, 1964.

[4]. Jain, A., & Jain, M. (2014, September). Detection and correction of non word spelling errors in Hindi language. In Data Mining and Intelligent Computing (ICDMIC), 2014 International Conference on (pp. 1-5). IEEE.

[5]. Fossati, D., & Di Eugenio, B. (2007). A mixed trigrams approach for context sensitive spell checking. In Computational Linguistics and Intelligent Text Processing (pp. 623-633). Springer Berlin Heidelberg

[6]. L. A. Wilcox-O"Hearn, G. Hirst, and A. Budanitsky. R eal-word spelling correction with trigrams: A reconsideration of the mays, damerau, and mercer model. In Proceedings of CICLing-2008 (LNCS 4919, Springer-Verlag), pages 605–616, Haifa, February 2008.

[7]. Youssef Bassil, Parallel Spell-Checking Algorithm Based on Yahoo! N-Grams Dataset, International Journal of Research and Reviews in Computer Science (IJRRCS), ISSN: 2079-2557, Vol. 3, No. 1, February 2012.

[8]. Eranga Jayalatharachchl, Asanaka Wasala, Ruvan Weersinghe, Data-Driven Spell Checking: The Synergy of Two Algorithms for Spelling Error Detection and Correction, The International Conference on Advances in ICT for Emerging Regions - iCTer 2012.

[9]. Samanta, P., & Chaudhuri, B. B. (2013). A simple real-word error detection and correction using local word bigram and trigram. In ROCLING.

[10]. Jain, A., & Jain, M. (2014, September). Detection and correction of non word spelling errors in Hindi language. In Data Mining and Intelligent Computing (ICDMIC), 2014 International Conference on (pp. 1-5). IEEE.

[11]. Lehal, G. S. (2007). design and implementation of Punjabi spell checker. International Journal of Systemics, Cybernetics and Informatics, 70-75

[12]. Singh, G., Sachan, M. (2014) 'Multi-layer perceptorn (MLP) neural network technique for offline handwritten gurmukhi character recognition', IEEE International conference on computational intelligence and computing research. 221-225.

[13]. Singh, S.K., Sachan, M.K., "Opportunities and Challenges of Handwritten Sanskrit Character Recognition System", July 17 Volume 3 Issue 7 , International Journal on Future Revolution in Computer Science & Communication Engineering (IJFRSCE), PP: 18 - 22

[14]. Sharma, P., Sachan M.K., "A Technique for Character Segmentation in Middle zone of Handwritten Hindi words using Hybrid Approach", July 17 Volume 3 Issue 7 , International Journal on Future Revolution in Computer Science & Communication Engineering (IJFRSCE), PP: 01 - 10

[15]. Singh, R.K., Rani, A., Sachan M.K., "Fuzzy Automata: A Quantitative Review", July 17 Volume 3 Issue 7 , International Journal on Future Revolution in Computer Science & Communication Engineering (IJFRSCE), PP: 11 - 17