**Spelling Error Detection and Correction Methods for Indian Languages - A Study**

\*Padmadhar Mishra1 Jyothi Pillai2 Ani Thomas3

 Research Scholar (CSVTU, Bhilai)  Professor (BIT, Durg) Professor (BIT, Durg)

 Durg, India Durg, India Durg, India

 padmadhar.mishra@bitdurg.ac.in jyothi.pillai@bitdurg.ac.in ani.thomas@bitdurg.ac.in

**ABSTRACT**

Spelling error detection techniques play an important role for any language. Spelling errors occur while typing the text document. The applications like search engines, information retrieval, newspapers, etc., require user typing. Languages like English are very powerful and can handle any type of spelling errors but efficient spelling checkers are not available for Indian language. The authors have made a study on the developmental approaches as well as roles of spell checkers with respect to various applications based on Indian languages.

**Keywords-**  Text Error Detection, Error Correction, Spell Checker, Machine translation, Information retrieval, N-gram.

1. **INTRODUCTION**

Spelling correction is a process of replacing an incorrectly spelled word with the most likely intended one in a document written in any Natural Language. It has been an active research area in the field of Natural Language Processing (NLP). The goal of the NLP group is to design and build software that will analyze, understand and generate languages that humans use naturally. Various applications are used in NLP like Text Summarization, Question Answering, Machine Translation, Parsing, Information Retrieval and Optical Recognition.

The basic aim of spell checking is to discover problems in written text. N-gram analysis and dictionary lookup are two ways for error detection. The error detection procedure often entails determining whether or not an input string is a legitimate dictionary word. For detecting such errors, efficient approaches have been developed. Dictionary lookup and n-gram approaches are used by most spellcheckers. When a word in a written document is discovered as a mistake, spelling correction procedures are used to correct the word or provide right options. For text error correction, many techniques such as rule-based techniques, edit distance techniques, N-gram techniques, and deep learning techniques are available.

Therefore, automatic word correction research may be viewed as focusing on three increasingly broader issues for descriptive purposes: 1) Detection of non-word errors 2) Correction of isolated word errors and 3) Word correction based on the context. From the beginning of the 1970s to 1980s, work on the first issue listed. During that time, effort was directed mainly toward exploring efficient pattern-matching and string comparison techniques for deciding whether an input string appears in a predefine word list or dictionary. From the 1970s to the present, more time was spent working on the second problem. At that time, a variety of general and specialized methods for correcting spelling errors were developed, some of which were used in conjunction with examples of spelling error patterns. The development of automatic natural language-processing models sparked interest in the third problem in the early 1980s, and the development of statistical language models rekindled it.

1. **Types of Spelling Errors**

In this section, various techniques that were designed on the basis of spelling errors and trends also called error patterns have been summarised. The most notable among these are the studies performed by various researchers. According to these studies, spelling errors are general divided into two types: Typographic errors and Syntactic errors.

Phonological Error

Transmutation Error

Deletion Error

Insertion Error

Substitution Error

Spelling Errors

Typographic errors

Syntactic errors

Non-Word Errors

Real Word Errors

* **Typographic errors**

Typographic errors are made by humans while writing text. This kind of error can be caused by typing carelessly, not knowing how to spell, pressing the wrong key accidentally or pressing the keys in the wrong order. The typographic errors can further divide into two subcategories. The first type of error is known as a “non-word error” and it occurs when a string of letters lacks any meaning. For example, there may be 24 possible combinations for the letters L, I, O, and N, but the correct word is only “LION”. Apart from all 23 permutations will result in non-word error. Real-word error, on the other hand, result in a string of letters that are in the word dictionary but do not belong in the phrase. The terms "शंकर" and "संकर" have the same pronunciation. Both of these are appropriate terms, but if they are not used in the right context, they might lead to mistakes. Five categories of real world are further subdivided. A) Phonological Error: when two words sound identical but have very distinct meanings.eg. “शंकर” and “संकर”. B) Transmutation Error: When two consecutive letter positions are changed, a new word is created that is likewise grammatically correct. eg. “कलम” and “कमल”. C) Deletion Error: These are the errors made when a letter is unintentionally erased.eg. “शैलजा” and “शैल”. D) Insertion Error: This error was caused by accidentally adding one or more letters.eg. “पुत्र”and “कुपुत्र”. E) Substitution Error: These kinds of mistakes are caused by replacing one or more letters with a different set of letters.eg. “बाग” and “राग”.

* **Syntactic errors:**

A Syntactic error is an error in utilizing a language that includes coordinating words and expressions that don't seem OK. Essentially, syntax reveals the structure and wording of a sentence, which is easy to misunderstand. For example अध्यापक विद्यार्थी बुलाए but the correct is अध्यापक ने विद्यार्थी को बुलाए ।

1. **NON-WORD ERROR DETECTION**

 N-gram analysis and dictionary lookup are two main approaches investigated for the purpose of identifying non-word errors. n-grams are sub sequences of words or strings composed of n letters, where n is typically one, two, or three. One-letter n-grams are referred to as unigrams or monograms, bi-grams or di-grams are the names given to n-grams with two letters, furthermore three-letter n-grams are referred to as trigrams. In most cases, n-gram error detection methods work by looking up each n-gram in an input string and looking it up in a pre-compiled table of n-gram statistics to find out if it exists or how often it occurs. To preassemble an n-gram table using n-gram techniques, a dictionary or a large corpus of text is typically required. Dictionary lookup techniques work by simply checking to see, if an input string appears in a dictionary, i.e., a list of acceptable words. If not, the string is flagged as a misspelled word. There are subtle problems involved in the compilation of a useful dictionary for a spelling correction application.

Historically, text recognition systems have tended to rely on n-gram techniques for error detection while spelling checkers have tended to rely on dictionary lookup techniques. In both cases, problems arise when errors cross word boundaries resulting in run-on or split words.

1. **N-gram Analysis Techniques:**

 Typically, text recognition systems concentrate on one of two text modes: handwritten text (sometimes referred to as cursive script) and text printed by a machine. Devices for optical character recognition (OCR) can process two modes. OCR devices typically recognize individual characters within words using feature analysis. The count of a character's vertical, horizontal, curved, and crossing lines are examples of features. n-gram analysis has proven useful for detecting such errors because they typically result in improbable n-grams. Errors made by OCR devices typically involve characters with similar features being misinterpreted, such as O and 0, S and 5, or t and f.

N-gram tables can take on various structures. The least difficult is a twofold bigram exhibit, which is a two-layered cluster of size 26 X 26 whose components address all conceivable two-letter blends of the letters in order. Depending on whether that bigram appears in at least one word in a predefine lexicon or dictionary, the value of each element in the array is set to either 0 or 1. The dimensions of a binary trigram array are three. Both of the above exhibits are alluded to as non-positional two-fold n-gram clusters since they don't show the place of the n-gram inside a word. A greater amount of the construction of the dictionary can be caught by a bunch of positional paired n-gram clusters. The i, j, and kth element in a positional binary trigram array, for instance, would have the value 1 if and only if there is at least one word in the lexicon with the letters l, m, and n in positions z, j, and k. However, the increase in storage required to represent more of the lexicon's structure comes at the expense of this trade off. Space that is required for the entire set of positional arrays. Any word can be checked for errors by simply looking up its corresponding entries in binary n-gram arrays to make sure that they are all one-digit numbers. To check a document for spelling errors, Srinivasa and Shree Devi [2017] generate a trigram frequency table based on the document itself. Then, for each unique word in the document they compute an index of peculiarity as a function of the trigram frequencies of the word. Finally, they rank the words in decreasing order of peculiarity. They hypothesize that misspelled words will tend to appear near the top of the list. As an example of the technique’s success, they point out, it took only ten minutes for an author of a 108-page document to scan the output list and identify misspelled words that 23 of the 30 misspelled words in the document occurred in the top 100 words of the list.

1. **Dictionary Lookup Techniques:**

A dictionary lookup is a simple process. In any case, reaction time turns into an issue when word reference size surpasses two or three hundred words. In archive handling and data recovery, the quantity of word reference passages can go from 25,000 to in excess of 250,000 words. This issue has been handled in three ways, by means of productive word reference query and additionally design matching calculations, through word reference dividing plans, and by means of morphological-handling procedures. The most well-known method for acquiring quick admittance to a word reference is the utilization of a hash table [Knuth 1992]. To look into an information string, one just figures its hash address and recovers the word put away at that location in the pre-constructed hash table. In some cases, a chance of crash may happen during development of the hash table. If the word put away at the hash address is not the same as the info string or is invalid, an incorrect spelling is shown.

1. **Dictionary creation challenges:**

A lexicon must be carefully adjusted to the target area of conversation for a spelling checker or text recognition programme. Unacceptably high numbers of false acceptances, or actual errors that went unnoticed because they happened to form valid low-frequency or extra domain words (e.g., lave, fen, veery etc.), can result from lexicons that are too small or too large, respectively. However, there are certain complications in the link between word frequencies and misspellings.

However, Damerau and Mays [1989] disagree with this advice. They discovered that by expanding the size of their frequency rank-ordered word list from 50,000 to 60,000 words, they were able to reduce 1,348 erroneous rejections while incurring just 23 more false acceptances using a corpus of more than 22 million words of text from diverse genres. They advise the use of broader lexicons since this 50-to-1 differential mistake rate indicates a considerable improvement in corrective accuracy.

1. **The Issue of Word Boundaries:**

Word boundaries are determined by white space characters (such as blanks, tabs, carriage returns, etc.) for almost all spelling mistake detection and correction methods. This assumption seems to be incorrect since a large percentage of text mistakes include splitting a single word (e.g., sp ent, th ebook) or running together two or more words, sometimes with inherent faults (e.g., ofthe, understandhme). In a corpus of 40,000 words of typed textual conversations, Kukich [1992] discovered that 15% of all non-word spelling errors were of this type (i.e., 13% were run-on words, and 2% were split words), Mitton [1987] discovered that run-ons and splits frequently result in at least one valid word (e.g., forgot -> for got, in form -> inform), so one or both of these errors may occur.

Although certain spelling error correction programmes make an explicit effort to address some run-on and split-word problems, no spelling error detection programme treats word boundary violations any differently from other errors.

1. **Rule‑based Techniques:**

These methods use morphology-based heuristics, part of speech, proper, etc., which have other properties the word that does the spell check. Monisha Das et al. Assamese language designed speller based on morphology and dictionary search method to identify and error correction [24]. Dhanabalan et al. recommended Tamil spell check using morphological analysis for error detection and correction [25]. Many other changes to the spelling of languages ​​were later proposed using morphological analysis. Secondly rule-based spell checking was proposed by Fossati et al. [27] where they proposed the use of a part of speech (POS) tag for English spell check. Besides these few texts were developed using Hidden Markov model to further improve spell check performance [28]. The biggest disadvantage such methods require different heuristic rules and information related to the specific language it affects acting negatively.

1. **A Techniques based on statistical analysis:**

A specific language is not necessary to use statistical procedures. Spell-checking is done using tools like frequency-based, n-gram-based, and finite state automata-based spell-checkers, which base their work on word counts and word properties. For the purpose of identifying and fixing spelling mistakes, Abdullah et al. [28] created a spell-checker for Bengali based on the finite-state representation (FSR) and state table approach. Naseem et al. [29] presented an Urdu spell-checker that used the word-frequency and edit-distance methods. Another Urdu spell-checker that incorporates the reverse edit-distance method (REDM) and finite state automata (FSA) was also proposed by Iqbal et al. Finite-state automata were used by Manohar et al. [30] to spell-check Malayalam. Additionally, P. H. Hema et al. created a spell-checker using the N-gram and minimal edit distance methods for Malayalam language. The advantage of utilising the statistical technique is that it enhances performance greatly while not requiring knowledge of the specific language. These systems have a fault in that they spell-check using characteristics, frequency, word counts, etc., when certain spelling errors need knowledge of the target language. Many academics used rule-based and statistical approaches to solve this challenge. To get around the problems, a hybrid model combines statistical and rule-based approaches.

1. **Deep-learning-based techniques:**

Although statistical and rule-based approaches to spell-checking are effective, deep learning (DL) approaches have the potential to improve performance even further. These deep-learning techniques excel in detecting real-word mistakes, which depend on the word's context in relation to the sentence. The first researchers to employ deep learning techniques for mistake correction were Ghosh and Kristensson. They suggested a text correction model for English [31]. In the meanwhile, Keisuke Sakaguchi et al. investigated the word recognition and spelling correction capabilities of the semi character recurrent neural network (SCRNN) [32]. The trials showed that the SCRNN works better than several other current spelling checkers. Language processing using DL is still a relatively new field of study. The deep-learning-based spell-checker is only accessible for the Malayalam language when it comes to regional languages. Using an LSTM network, Sooraj et al. [33] created a spell-checker for the Malayalam language. The network used in this spell-checker has undergone training and testing to recognise spelling mistakes and pinpoint their locations.

1. **RESEARCH ON ISOLATED WORD ERROR CORRECTION**

It may be adequate for certain apps to only identify text problems, but this is not the case for the majority of them. For instance, output mistakes need to be found and fixed as text recognition software aims to faithfully copy input text. Similar to this, people now anticipate spelling checkers to offer fixes for any non-words they find. In fact, several applications for spelling correction, including text-to-speech synthesis, need that mistake be found and fixed automatically. Many isolated-word mistake correction algorithms have been developed to solve the issue of fixing words in text.

The design of isolated-word error correctors is subject to a variety of limitations depending on the features of the application in question, and several effective correction techniques have been developed by taking use of these constraints and qualities. Before getting into the specifics of each approach, it is worthwhile to analyse a few isolated-word mistake repair applications and their features.

The three main challenges that affect the majority of application-specific design considerations are (1) lexicon concerns, (2) computer-human interface issues, and (3) spelling mistake pattern issues. Issues with lexicons include topics like lexicon size and coverage, rates of new word entry, and if morphological processing, such affix handling, is necessary. These were covered in the section on dictionary construction challenges that came before it.

**Table1: Accuracy of Some Isolated-Word Spelling Correction Techniques**

|  |  |  |  |
| --- | --- | --- | --- |
| Technique | 521 -Word Lexicon | 1142-Word Lexicon | 1872-Word Lexicon  |
| Minimum Edit distance | **64%** | **62%** | **60%** |
| Similarity Key | **80%** | **78%** | **75%** |
| Simple N-gram Vector Distance - Dot Product - Hamming Distance - Cosine Distance | **58%** **69%** **76%** | **54%****68%****75%** | **52%****67%****74%** |
| SVD N-gram Vector Distance | **81%** | **76%** | **74%** |
| Probabilistic | **-** | **78%** | **-** |
| Neural Net | **75%** | **75%** | **-** |

**Figure 1: Accuracy of Some Isolated-Word Spelling Correction Techniques**

1. **RESEARCH ON CONTEXT-DEPENDENT WORD CORRECTIONS**

There will always be a subset of faults that isolated-word error correction methods cannot handle, despite the advancements achieved in the field. When one correctly spelt word is used in place of another, it falls under the category of real-word mistakes. Example of real-word error (कमल कीचड़ में खिलता है|) in this sentence कमल is a valid word but not intended. Correct word for this sentence is कमल, which is valid and intended.

All of these problems appear to require contextual information to be detected and corrected. Contextual information might be beneficial for enhancing the detection of non-word errors during repair. All of these problems appear to require contextual information to be detected and corrected. Additionally, contextual information would help to increase the accuracy of non-word error correction. All of these problems appear to require contextual information to be detected and corrected. Additionally, contextual information would help to increase the accuracy of non-word error correction. All of these problems appear to require contextual information to be detected and corrected. Additionally, contextual information would help to increase the accuracy of non-word error correction.

Constructing context-sensitive word correction tools has remained a difficult task. This is mostly because of the problem's seeming intractability, which seems to demand fully developed natural language processing (NLP) skills, such as robust natural language parsing, semantic comprehension, pragmatic modelling, and discourse structure modelling. Successful NLP systems up to this point have been limited to a small number of discourse domains, and while some of these systems have addressed the need to handle input that is not well-formed, none of them were intended for widespread application. However, recent developments in robust syntactic parsing have resulted in the creation of at least two broad writing assistance programs that can identify and fix mistakes brought on by a few syntactic-constraint violations. Additionally, improvements in statistical language modelling and a consistent rise in.

Review of context-dependent word correction research's historical development. There are four subsections in it. The first examines research on real-word mistakes' frequency and classification. The second examines prototype NLP systems that deal with the issue of handling improperly formatted input, including two tools for writing assistance that are based on syntactic rules and contain grammar and spelling checks. The third discusses current research on the use of statistical language models for text-dependent spelling error detection and repair. The final gives a summary of the spelling correction job that is depending on context.

 **Table 2. Distribution of Spelling Errors (from Atwell & Elliott)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Text source | **Total no of errors** | **% Non-Word Errors** | **% Local Syntactic Errors** | **% Global Syntactic Errors** | **% Semantic Errors** |
| Published Texts  | 50 | 52% | 28% | 8% | 12% |
| Student Essay | **50** | **36%** | **38%** | **16%** | **10%** |
| Non-Native Text | **50** | **4%** | **48%** | **12%** | **36%** |

**Figure 2 : Distribution of Spelling Errors**

Although the distributions of errors across text genres are very diverse (Table 2), they do show that a sizeable number of errors may be identifiable as local syntactic violations. the prototype error-detection system Atwell and Elliott developed.

1. **CONCLUSION**

In the review paper, the author suggested methods for identifying and fixing Indian language non-word spelling mistakes. Here, it is examined why approaches for English error detection and repair couldn't be used directly for Indian language. In addition to Hindi vowels and consonants, it also includes several types of symbols like the "vowel sign" "matrass," half letters, and halant, among others. Real-word mistake detection and correction is a crucial field for future research.

1. **REFERENCES**

[1]. Singh, S., & Singh, S. (2021). HINDIA: a deep-learning-based model for spell-checking of Hindi language. *Neural Computing and Applications*, *33*(8), 3825-3840.

[2]. Caryappa, B. C., Hulipalled, V. R., & Simha, J. B. (2020, October). Kannada Grammar Checker Using LSTM Neural Network. In *2020 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE)* (pp. 332-337). IEEE.

[3]. Etoori, P., Chinnakotla, M., & Mamidi, R. (2018, July). Automatic spelling correction for resource-scarce languages using deep learning. In *Proceedings of ACL 2018, Student Research Workshop* (pp. 146-152).

 [4]. Bopche, L., Dhopavkar, G., & Kshirsagar, M. (2011, December). Grammar checking system using rule based morphological process for an indian language. In *International Conference on Computing and Communication Systems* (pp. 524-531). Springer, Berlin, Heidelberg.

[5]. Islam, S., Sarkar, M. F., Hussain, T., Hasan, M. M., Farid, D. M., & Shatabda, S. (2018, December). Bangla sentence correction using deep neural network based sequence to sequence learning. In *2018 21st International Conference of Computer and Information Technology (ICCIT)* (pp. 1-6). IEEE.

[6]. Islam, S., Sarkar, M. F., Hussain, T., Hasan, M. M., Farid, D. M., & Shatabda, S. (2018, December). Bangla sentence correction using deep neural network based sequence to sequence learning. In *2018 21st International Conference of Computer and Information Technology (ICCIT)* (pp. 1-6). IEEE.

[7]. Dereza, O. (2018, October). Lemmatization for Ancient Languages: Rules or Neural Networks?. In *Conference on Artificial Intelligence and Natural Language* (pp. 35-47). Springer, Cham.

[8]. Li, H., Wang, Y., Liu, X., Sheng, Z., & Wei, S. (2018). Spelling error correction using a nested rnn model and pseudo training data.

[9]. Hu, Y., Jing, X., Ko, Y., & Rayz, J. T. (2020, September). Misspelling Correction with Pre-trained Contextual Language Model. In *2020 IEEE 19th International Conference on Cognitive Informatics & Cognitive Computing (ICCI\* CC)* (pp. 144-149). IEEE.

[10]. EL Atawy, S. M., & Ahmed, H. M. (2021). Spelling Checker for Dyslexic Second Language Arab Learners. *Journal of Theoretical and Applied Information Technology*, *99*(2).

[11]. Singh, S., & Singh, S. (2018, March). Review of real-word error detection and correction methods in text documents. In *2018 second international conference on electronics, communication and aerospace technology (ICECA)* (pp. 1076-1081). IEEE.

[12]. Jain, A., Jain, M., Jain, G., & Tayal, D. K. (2018). “UTTAM” An Efficient Spelling Correction System for Hindi Language Based on Supervised Learning. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, *18*(1), 1-26.

[13]. Kumar, R., Bala, M., & Sourabh, K. (2018). A study of spell checking techniques for indian languages. *JK Research Journal in Mathematics and Computer Sciences*, *1*(1).

[14]. Etoori, P., Chinnakotla, M., & Mamidi, R. (2018, July). Automatic spelling correction for resource-scarce languages using deep learning. In *Proceedings of ACL 2018, Student Research Workshop* (pp. 146-152).

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

[16]. Jain, N., Singla, K., Tammewar, A., & Jain, S. (2012, December). Two-stage approach for hindi dependency parsing using maltparser. In *Proceedings of the Workshop on Machine Translation and Parsing in Indian Languages* (pp. 163-170).

[17]. Shalini, M., & Indira, B. (2017). Implementation of Hindi Word Recognitionand Classification System Using Artificial Neural Network. International Journal of Pure and Applied Mathematics, 117(15), 557-565.

[18]. Faili, H., Ehsan, N., Montazery, M., & Pilehvar, M. T. (2016). Vafa spell-checker for detecting spelling, grammatical, and real-word errors of Persian language. *Digital Scholarship in the Humanities*, *31*(1), 95-117.

[19]. Gupta, P. (2020, February). A context-sensitive real-time Spell Checker with language adaptability. In *2020 IEEE 14th International Conference on Semantic Computing (ICSC)* (pp. 116-122). IEEE.

[20]. Kukich, K. (1992). Techniques for automatically correcting words in text. *ACM computing surveys (CSUR)*, *24*(4), 377-439.

[21]. Damerau, F. J., & Mays, E. (1989). An examination of undetected typing errors. *Information Processing & Management*, *25*(6), 659-664.

[22]. Mitton, R. (1987). Spelling checkers, spelling correctors and the misspellings of poor spellers. *Information processing & management*, *23*(5), 495-505.

[23]. Atwell, E., & Elliott, S. (1987). Dealing with ill-formed English text, The Computational Analysis of English. *A Corpus-Based Approach*, *12*, 120-138.

[24]. Das M, Borgohain S, Gogoi J, Nair SB (2002) Design and implementation of a spell checker for assamese. In: Language engineering conference, proceedings IEEE, pp 156–162.

[25].. Dhanabalan T, Parthasarathi R, Geetha TV (2003) Tamil spell checker. In: Sixth tamil internet conference, Chennai, Tamilnadu, India, pp 18–27.

[26]. Fossati D, Di Eugenio B (2007) I Saw TREE trees in the park : how to correct real-word spelling mistakes. In: LREC, pp 896–901
[27]. Jain U, Kaur J (2015) Text chunker for Punjabi. Int J Curr Eng Technol 5(5):3349–3353.

[28]. Abdullah M, Islam Z, Khan M (2007) Error-tolerant finite-state recognizer and string pattern similarity based spelling-checker for Bangla. In: Proceeding of 5th international conference on natural language processing (ICON).

[29]. **Naseem T, Hussain S (2007) A Novel approach for ranking spelling error corrections for Urdu. Lang Resour Eval 41(2):117–128.**

**[30]. Manohar N, Lekshmipriya PT, Jayan V, Bhadran VK (2015) Spellchecker for Malayalam using finite state transition models. In: IEEE recent advances in intelligent computational systems, RAICS 2015, pp 157–161.**

**[31**]. **Sakaguchi K, Duh K, Post M, Van Durme B (2017) Robsut wrod reocginiton via semi-character recurrent neural network. In: Thirty-first AAAI conference on artificial intelligence, pp 3281–3287.**

**[32]. Sooraj S, Manjusha K, Anand Kumar M, Soman KP (2018) Deep learning based spell checker for malayalam language. J Intell Fuzzy Syst 34(3):1427–1434**

**[33]. Gumaei A, Hassan MM, Alelaiwi A, Alsalman H (2019) A hybrid deep learning model for human activity recognition using multimodal body sensing data. IEEE Access 7:99152–99160**