BANTU SPELL CHECKER AND CORRECTOR USING MODIFIED EDIT DISTANCE ALGORITHM (MEDA)

Boago Okgetheng, Gabofetswe Malema,
Ariq Ahmer, Boemo Lenyibi and Ontiretse Ishmael

Department of Computer Science,
University of Botswana, Gaborone, Botswana

ABSTRACT

Automatic spelling correction for a language is critical since the current world is almost entirely dependent on digital devices that employ electronic keyboards. Correct spelling adds to textual document accessibility and readability. Many NLP applications, such as web search engines, text summarization, sentiment analysis, and so on, rely on automatic spelling correction. A few efforts on automatic spelling correction in Bantu languages have been completed; however, the numbers are insufficient. We proposed a spell checker for typed words based on the Modified minimum edit distance Algorithm (MEDA), and the Syllable Error Detection Algorithm (SEDA). In this study, we adjusted the minimal edit distance Algorithm by including a frequency score for letters and ordered operations. The SEDA identifies the component of the word and the position of the letter which has an error. For this research, the Setswana language was utilized for testing, and other languages related to Setswana will use this spell checker. Setswana is a Bantu language spoken mostly in Botswana, South Africa, and Namibia and its automatic spelling correction are still in its early stages. Setswana is Botswana's national language and is mostly utilized in schools and government offices. The accuracy was measured in 2500 Setswana words for assessment. The SEDA discovered incorrect Setswana words with 99% accuracy. When evaluating MEDA, the edit distance algorithm was utilized as the baseline, and it generated an accuracy of 52%. In comparison, the edit distance algorithm with ordered operations provided 64% accuracy, and MEDA produced 92% accuracy. The model failed in the closely related terms.

KEYWORDS

Bantu Spell Checker, Edit Distance algorithm, morphologically rich, Syllable Error Detection Algorithm.

1. INTRODUCTION

The written form of a language is a significant aspect in communication. We do not need to know right spelling of a word when speaking, and communications can be done even without proper grammatical attention, but perfect spelling is critical when writing. Spelling errors are prevalent, especially in a complex language especially in Bantu languages which are morphologically rich. Most of Bantu languages are agglutinative with heavy use of affixes, suffixes, and prefixes. The most notable aspects involving Bantu syllable structure, consonant/vowel inventories, and phonological processes are firmly established throughout the Bantu region [1], despite the enormous number of languages and vast geographic expanse that they span. The notion of what constitutes a word is therefore different from European languages like English.

David C. Wyld et al. (Eds): ARIA, SIPR, SOFEA, CSEN, DSML, NLP, EDTECH, NCWC - 2022
pp. 303-316, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.121524

Spelling mistakes are widespread in Bantu languages due to the difficult word formation. Spelling mistakes are not necessarily caused by their complexity; they can also arise because of rapid typing or a lack of expertise and some are generated from other close terms. Spelling errors could be divided into two types: typographical errors and cognitive errors [2]. Cognitive errors include phonetic errors as well as errors associated with homonyms, which can result in a valid word that is incorrect in context. When developing a spell-checking tool, these two types of errors must be considered.

Since spelling mistakes are so common, automatic spelling corrective systems are critical. The practice of detecting improperly written words in a particular language is known as spell checking [3]. The spell-checking process may also make suggestions for improperly misspelled terms in the language. Many potential NLP applications, like text summarization, sentiment analysis, and machine translation, require spelling correction. [4]. There has been a lot of research on this area, although there aren't many significant publications on Spell checkers in Bantu languages. Some studies employed the minimal edit distance approach in conjunction with a considerably bigger dictionary/database.

We present a multilingual Bantu spell checker and corrector that employ a modified Edit distance algorithm (MEDA) and Syllable error detection Algorithm (SEDA), with a focus on spell application usability. The method employed SEDA to detect the error in the term and MEDA to fix it. The MEDA scores the frequency of the letters in the word using a mathematical method. The frequency approach calculates the likelihood of letter *a* being followed by letter *b*. MEDA additionally incorporates the optimal operation ordering from the edit distance for efficient and effective outcomes. This proposed paradigm was tested using Setswana, a Bantu language. This study is divided into the following sections: literature review, methodology, implementation, results, results analysis, and conclusions.

2. LITERATURE REVIEW

With the digitization of almost everything becoming the norm, in this case, text data, it is important to have spell checker tools in word processors or software programs put in place to validate and verify textual information. Although error detection and correction are the basis of a professional spell checker, thorough investigations indicate that for the Bantu languages this has not been conducted and is still in its primitive stages due to the languages' rich morphology and high agglutination. Spellchecking entails both error correction and detection. Text error correction has mostly concentrated on three areas: non-word error detection, isolated-word error detection, and context-dependent word correction. Error detection has been successfully implemented; however, error repair is currently being worked on incrementally.

2.1. Existing Systems

There are some existing techniques that are used for spell checking. These include R-rule based technique system – where a number of associated rules captures frequent spelling errors and typos and uses them to on the misspelled words, Dictionary lookup – where the word is checked against a dictionary and N-gram technique [5]. Other techniques include using Clustering Algorithm (PAM) to identify errors in the spelling [6] and using statistical model to check for spelling errors [7].

2.2. Related Work

A study in the development of a Turkish spell-checking and error-correcting application by Ince [8] also suggests that the development of a spell checker and error corrector for an agglutinative language proves to be challenging and different. The author emphasizes that for this to take place; mathematical preliminaries and morphological analysis are required for such languages. Ince developed a spell checker and error corrector for the Turkish language using the n-gram and edit distance algorithms using the C#.Net and Microsoft Visual Studio platform. An nZembrek file was then used as the Turkish dictionary's sustainability of the word roots and compliance with suffixes. The file contained a text file, a suffix file, a letter file, and an optional syllable finder. Ince says the n-gram was used to learn the rules of the letters and the word sequences and was dependent on the word length the user input. The n-gram was used to determine the distance of a misspelled word as n, where (n-1) and (n-2) are the misspelled word's properties. The edit distance algorithm was used to add the nearest 2 distance words to the suggestion list. The results obtained showed an accuracy of 95% and a success rate of 95% for suggestions for spelling errors.

Mjaria [9] conducted a study similar and came to the same conclusion that stated although it has become a common norm for the use of spellcheckers due to the increasing usage of text-based communication, be it at the workplace or on social media, it has been very gruesome when it comes to support for spellchecking of Bantu languages. Spellcheckers are designed such that the body consists of text representing the language, basically the "corpus", an error model, and a language model. The language model is used for determining how frequently a word occurs in a dictionary. The error model is just used for the modelling of spelling errors, whose accuracy is liable to the corpus.

Bantu languages are mostly known to be ambiguous, thus making it difficult for error correction when it comes to spell checking. Mashiane [10] pointed that there are yet efficient algorithms and tools that are yet to be found when it comes to error correction of Bantu languages.in this work a minimum edit distance algorithm was used for error detection, where the algorithm searches for the smallest number of insertions or deletions of a word. The structure of African languages differs significantly from that of the languages supported by conventional spell checker software. It has been easier for non-word error detection compared to error correction. A binary spell checker has been used to achieve this. Spelling errors in languages with complex morphology and structure are common and could either be cognitive or typographical in nature, so error corrections are required.

A discovery was made by Islam [11], who conducted research on the Bangla language, which was prone to errors due to its complex structure. Islam created a model for correct word prediction in Bangla using the fractional accuracy function for non-error words and the direct dictionary method and edit distance algorithm to check correctness and word suggestions and compute the accuracy. Islam further states that in handling the typographical non-word errors, direct dictionary lookup was used to detect erroneous words, and if the typed word was not available, it was an error. The edit distance algorithm was then used to correct the word with an estimate of the similarity of its structure to the misspelled word and possible correct words. The model produced an accuracy of 98.83% from a suggestion list of length 9 and an accuracy of 65.17% from a suggestion list of length 1. Contrary to the use of the edit distance algorithm, the reverse edit distance algorithm may be used in spell checkers as it uses fewer word comparisons and has proved to be efficient.

Iqbal [12], in his study of creating a spell checker for the Urdu language, used the reverse edit distance algorithm. Iqbal indicated that the reverse edit distance algorithm minimized the

comparisons between misspelled words and words in the dictionary for error correction. The reverse edit distance proposed for the Urdu language uses the edit distance permutation, which is generated and compared to the lexicon word in alphabetical order. With the Urdu language comprised of 42 alphabets, a word length of n resulted in a total of 86n+41 comparisons. The edit distance algorithm was then used on the generated permutation to avoid the occurrence of non-word errors that resulted in a zero match in the lexicon Some of the difficulties faced in building the Urdu spell checking model, according to Iqbal, were its complicated morphological structure, diction problems, word separators, and loan words.

3. METHODOLOGY

Our method for spell Checker is to extend the existing edit distance algorithm to make it work for Bantu languages, based on how words are formed or written. The proposed has been described in this section.

3.1. Syllable Error Detection Algorithm (SEDA)

In our methodology, we first detect the dissimilarities between the words using our error detection algorithm. We firstly input documents containing the words of that language in the model. Then the model learns the different types of syllables present in the language and stores them in a file. Then we take our input words and pass them through our SEDA. This model then extracts the syllables and looks them up the file of syllables to look for any inconsistencies. An inconsistency represents that the model detected an error in the word and then outputs that error to the next phase.

3.2. Edit Distance Algorithm (EDA)

We now consider the Edit Distance Algorithm (EDA). This algorithm attempts to determine the difference between two words and the fewest number of operations required to change one word to another. [13]. The error that is identified from the SEDA is then passed to this phase of the Edit Distance Algorithm. The EDA takes this error into account and performs its operations on the word to be transformed. We propose a system of ordering of the fundamental operations of EDA: insert, delete, substitute and transpose. The ordering of these operations is done to optimize the discovery of the smallest number of operations needed to change the word. Section 3.3 shows the orderings of the operations that are proposed for the transformation.

3.3. EDA with Ordering

In this step, take into account the order of operations that are available for the general EDA. We run the proposed ordering of operations (as shown in below) and run each order against the EDA to get a general performance of which ordering to take. An average score for each of the ordering is calculated which gives the performance of the ordering. This score determines which operations ordering will be used as we assume that this performs the best on average for every word. We determine the average performance for each operation and then arrange the results based on the scores. This is done to investigate how the ordering of operations affects the spell checker in Bantu languages. The best operation ordering will then be used in our MEDA.

Insert Ordering

1. insert , replace , delete, transpose
2. insert, transpose, replace, delete

3. insert, delete, transpose , replace
4. insert, replace, transpose, delete
5. insert, transpose , delete , replace
6. insert, delete, replace, transpose

Transpose Ordering

1. transpose, insert , delete, replace
2. transpose, insert, replace, delete
3. transpose, replace, delete, insert
4. transpose, replace, insert, delete
5. transpose, delete ,insert, replace
6. transpose, delete, replace, insert

Delete Ordering

1. delete, insert , transpose , replace
2. delete, insert, replace, transpose
3. delete, replace, transpose, insert
4. delete, replace, insert, transpose
5. delete, transpose, insert, replace
6. delete, transpose, replace, insert

Replace Ordering

1. replace , delete, transpose, insert
2. replace, delete, insert, transpose
3. replace, insert, delete, transpose
4. replace, insert, transpose, delete
5. replace, transpose, delete, insert
6. replace, transpose, insert, delete

3.4. Modified Edit Distance Algorithm (MEDA)

MEDA is a component of our suggested solution since it employs algorithms based on the structure of Bantu languages. This algorithm incorporates operations ordering and a mathematical model together with the general EDA. The mathematical model is used to find the score of each alphabet in the given word relative to its position in that word. This allows us to find the probable letter that may be the error in the word. Once this score is found, we employ the ordered operations of the general EDA to the "incorrect" word to find the minimum number of operations to be taken to transform the word. The operations of the EDA are ordered to explore which set of operations are optimized for the error correction of any given word in time and to see whether ordering said operations actually influences the final yield.

3.4.1. Mathematical Model

Let $W = \{c_0, c_1, \dots, c_n\}$ be the set of characters in a given word W.
Let $D_{js} = \{\{a_{00}, a_{10}, \dots, a_{j0}\}, \dots, \{a_{0s}, a_{1s}, \dots, a_{js}\}\}$ be the dictionary that consists of words whose characters are denoted by a_{js} of a word D_s.
Let $p = |W|, q = |D_s|, r = |D|$ where D_sa word in the dictionary is at any given time.

Let $O = \{O_0, O_1, \dots, O_n\}, 0 \leq n \leq p$ be the set consisting of objects that hold the frequency of each character c_n of the word W. Then the frequency of the characters relative to their position is calculated as follows.

$$O_i = \frac{1}{r}\sum_{k=0}^{r} \Phi_i$$

Where,

$$\Phi_i = \begin{cases} 1, & if\ c_i = D_{js}, i = j, 0 \leq j \leq q, 0 \leq s \leq r \\ 0, & Otherwise \end{cases}$$

The function Φ basically keeps track of how many times the letter c_i appears in the position i in the set of words in D. If there is a character that is common in that position, then a 1 is added to Φ and 0 if the characters don't match. The figure is then normalized by taking the average of the set Φ over the total number of words in the dictionary and place it in the i-th position of set O. Note that the position of the elements in the set W match with that of set O and $|W| = |O|$.

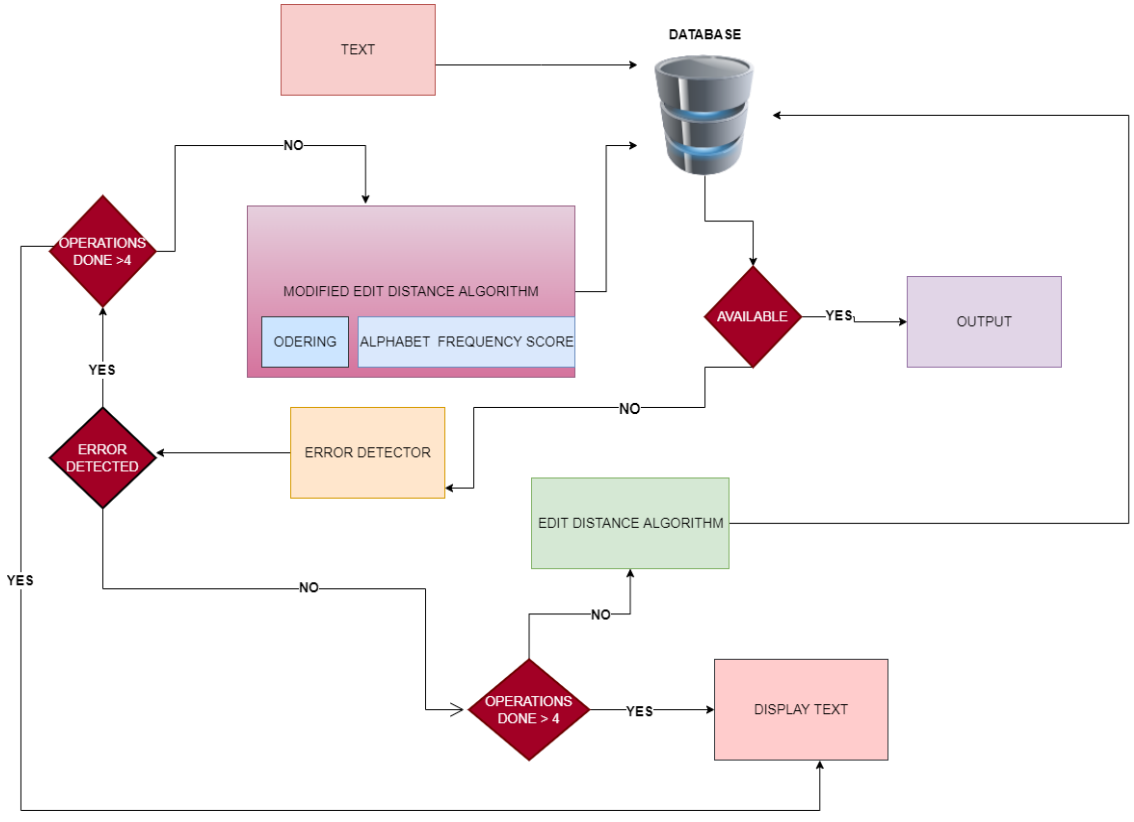

Figure 1. Proposed Solution

We start off by getting a word input. The word is then checked against a dictionary of known words. If the word appears in the dictionary, we assume that it is accurate, therefore we display it right away. In the case that it is not found, the word is passed through an error detector. If there are no errors detected and we also know that the word is non-existent in the dictionary, then the word is passed through the Edit Distance algorithm (EDA) in four passes. The operations in each

pass are accounted for before passing to EDA. Each pass consists of an operation of the following: Insert, Delete, Replace and Transpose. The operations are ordered such that the optimized operation is performed on average. If, after executing the aforementioned operations, the word is still not located in the dictionary, we conclude that it does not exist and display it directly. However, if there is (are) error(s) detected after passing through the error detector, then the word is passed to check if the said number of operations (as mentioned above) are done before passing it to "Modified Edit Distant" algorithm (MEDA). In this method, the algorithm detects where the error is and then splits the syllables and characters. Each character is then evaluated with a frequency score (see section 3.4.1) relative to its position in the word. The character with the least score is taken to be the anomaly. Then, the four standard operations are applied to this character in an attempt to correct the word. After the operations are done, the word is checked against the dictionary to see if it exists. If the word does not exist and we exceed the number of operations, then we conclude that the word in non-existent and display it accordingly.

4. IMPLEMENTATION

4.1. Data Pre Processing

Test Data

We created a json file including Setswana words with right Setswana words and different erroneous words next to each, for example "bolela": "bolel blela bulela" the first is the proper word 'bolela' and the following are alternative mistake terms. The target word is then used to construct a list of spelling mutations within a distance of two, yielding the wrong Setswana terms. This file served as an input. We used 2500 correct terms and 7728 wrong words. These were nouns and verbs in Setswana.

setwanaTest1.json

```
"abang": "abing", "aba": "abn", "amogela":
"amogella amogel", "atla": "atha", "adima":
"adim adema", "adimela": "ademela", "apaya": "apaa apya",
"apayang": "apayng", "agisa": "agiisa", "arabisisa": "arbisis",
"amusa": "amisa amsa", "akanya": "aknya", "akela": "akle",
"atlana": "atlna atlsna", "bolotsa": "bulotsa", "bapatsa": "baphatsa",
"betla": "beta", "betsega": "btsega", "bidile": "bedele", "bintsha":
"bintsa bitsha", "bidikamisa":"bedikamisa bidekamesa bidekamisa bidikamsa bidikamasa", "botsosolosa": "botsoslosa  botsosolo
"bina": "bena", "baka": "bak",
"bidisa": "bidis bedesa", "bankanya": "bankana", "bakisa": "bkisa",
 "biletsa": "biltsa bilets billetsa",
"boifa": "biofa", "boifisisa": "boifsisa", "bokegelwa": "bokegelw",
"bolotsana": "bolotsna", "batola": "batlola", "budusa": "bodusa",
"belega": "bilega", "belegisa": "belegsa", "bitisiwa":
"btisiwa bitlisiwa botisiwa", "balela":"balila balla", "dikologa": "dkologa dekologa", "dirisiwa":
"deresewa deresiwa", "dipa": "depa", "dirolola": "dirolla","dia": "dhia", "emelela": "emella", "epa":"epha", "eta": "etla",
"etsa": "etsha", "ega": "eha", "ema": "emha",
"eletsa": "elletsa", "emisa": "emsa"
```

Figure 2. Test Data Sample

Dictionary/database

This is a list of Setswana terms. The dictionary is used as a lookup to see if the term is in the dictionary; if it is, we presume it is the proper targeted words. If the entered word is not accessible, it is considered an error.

4.2. Module Implementation

We used python as a programming language for implementation. Python comes with a support for advanced text manipulation that is easy to code which in our case meant that it helped in pre-processing our text input easily.

4.2.1. Frequency score Module

We require a corpus of words from a certain language that has most of the forms, or a method that understands how to generate all of the forms of a word. We next performed our frequency score method, which gave us the probability of specific alphabets following each other. To train the model, we utilized a big dataset containing Setswana phrases to score our alphabets in Setswana. Figure 3 illustrates the findings of the frequency score for the Setswana word "bontshantse."

Figure 3. Frequency score

4.2.2. EDA, EDA by ordering and MEDA Modules

We implemented the three approaches, and they were all tested with the same dataset of 2500 Setswana words. The results of each approach were recoded as indicated in section 5.

5. RESULTS

A set of words with varying morphologies from a text were used as input to test the performance of the three techniques. The accuracy was measured by comparing the correct number of words to the total number of words provided, as shown in equation 1 below.

Equation 1. Accuracy [14]

$$\text{Accuracy} = \frac{\text{No. of correct words}}{\text{No. of words}} \times 100$$

To evaluate the spelling behaviour of Setswana words, the three methodologies were explored. In this experiment, 2500 words were analysed, and the accuracy of each approach was measured. Except for overlapping terms, the SEDA detector 99 percent accurate (Words which are wrong but having correct syllables). The edit distance method by Levenstein has 52 percent accuracy in correcting misspelled words, the edit distance algorithm by ordering has 64 percent accuracy, and the modified edit distance algorithm has 92 percent accuracy. The failure causes are described in the section on outcomes.

Edit Distance Algorithm by ordering

In our test input, we ordered the operations. In this case, we modified the Edit distance Algorithm to avoid using the shortest distance and instead complete the processes in chronological order. The key is **I-Insert, R-Replace, T-Transpose, and D-Delete.** We wanted to see the order in which Setswana words may be corrected so that we determine the most prevalent Setswana word error (it be swapping, omission or addition). The accuracy of all orders was calculated, as well as the average from each operation, as shown in tables 1, 2, 3, and 4. The average accuracy of the four operations is 64.3 percent for transpose, 59.9 percent for insert, and 56.9 percent for replace, with delete being the least accurate at 56.6 percent. The precision is depicted in figure 4 below.

Table 1. Accuracy of Transpose Ordering

Ordering	No. of words	No. correct words	No. incorrect words	Accuracy rate (%)
1. T,I,R,D	2500	1617	883	64.7
2. T,I,D,R	2500	1610	830	64.4
3. T,R,D,I	2500	1597	903	63.9
4. T,R,I,D	2500	1610	890	64.4
5. T,D,I,R	2500	1605	895	64.2
6. T,D,R,I	2500	1597	903	63.9
Average Accuracy				64.3

Table 2. Accuracy of Insertion ordering

Ordering	No. of words	No. correct words	No. incorrect words	Accuracy rate (%)
1. I,R,D,T	2500	1595	705	63.8
2. I,T,R,D	2500	1607	893	64.3
3. I,D,T,R	2500	1425	1075	57.0
4. I,R,T,D	2500	1460	1040	58.4
5. I,T,D,R	2500	1412	1088	56.5
6. I,D,R,T	2500	1485	1015	59.4
Average Accuracy				59.9

Table 3. Accuracy of Delete ordering

Ordering	No. of words	No. correct words	No. incorrect words	Accuracy rate (%)
1 D,I,T,R	2500	1460	1040	58.4
2 D,I,R,T	2500	1460	1040	58.4
3 D,R,T,I	2500	1365	1135	54.6
4 D,R,I,T	2500	1375	1125	55.0
5 D,T,I,R	2500	1450	1050	58.0
6 D,T,R,I	2500	1378	1122	55.1
Average Accuracy				56.6

Table 4. Accuracy of Replace ordering

Ordering	No. of words	No. correct words	No. incorrect words	Accuracy rate (%)
1 R,D,T,I	2500	1378	1122	55.1
2 R,D,I,T	2500	1390	1110	55.6
3 R,I,D,T	2500	1460	1040	58.4
4 R,I,T,D	2500	1463	1037	58.5
5 R,T,D,I	2500	1425	1075	57.0
6 R,T,I,D	2500	1413	1087	56.5
Average Accuracy				56.9

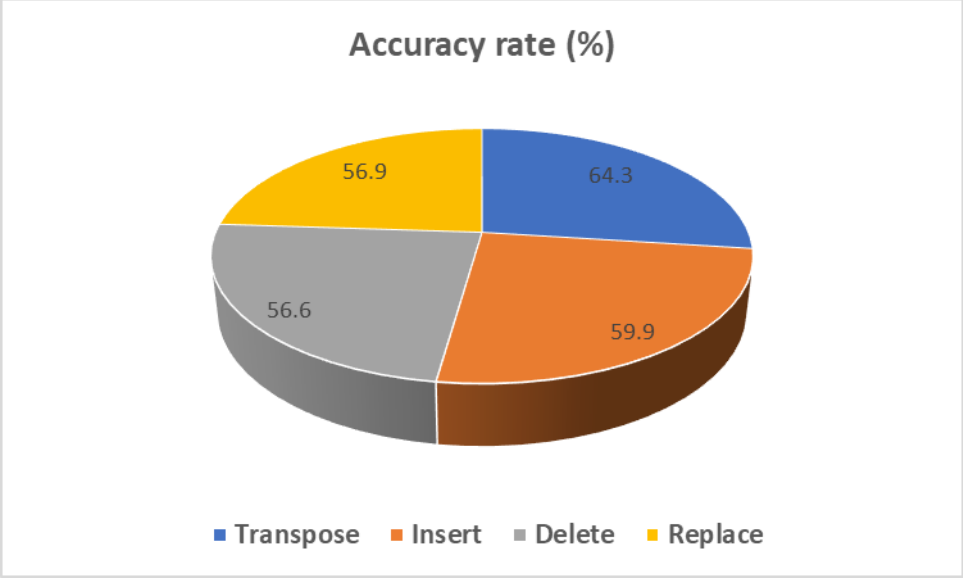

Figure 4. Accuracy per operation

From figure 4, it is clear that starting with the transpose operation yielded the best result. And in order of accuracy the order of operations {T, I, R, D} yielded best result which is in accordance with the average results obtained per operations order set. As this order was the best performing, we adopted this in our proposed solution (MEDA) which gave us optimal results

Edit Distance Algorithm

The **edit distance** is a measure of how similar two strings are. This checks the minimum number of editing operations. The accuracy of this algorithm given Setswana words gave 52% as indicated in table 5.

Table 5. Accuracy of the three Methods

	Accuracy (%)		
No. of Words	**Edit Distance Algorithm**	**Edit Distance Algorithm by Ordering**	**Modified Edit Distance Algorithm (MEDA)**
2500	52%	64.3%	92%

Modified Edit Distance Algorithm

This recommended option outperformed the other two described approaches above. Table 5 shows that the Modified algorithm performed at 92 percent. As the approach employed the score of the alphabets in terms of following each other, this is a clear depiction that the Setswana words are constructed in a consistent pattern. The MEDA also employed the syllables *i*, *i*+1 where *i* represents the position of the word's incorrect syllable and the syllable at. Because Setswana has few syllables, this made it easy for the algorithm to correct the words.

The algorithm corrects the syllable and checks to see if the word is valid; if not, it employs a mathematical method that scores each letter in a word. In this example, it begins with the order of operations from the ordering Edit distance algorithm. In this instance, for example, we have "thlola" as the input word (incorrect spelling), and the correct word is "tlhola." The term has three syllables:"th", "lo", and "la". And the error detector found a mistake in the initial syllable "th," which is not a proper syllable in Setswana. The MEDA then attempted to remedy this by using the nearest syllable, "tho," and it produces a correct Setswana word, "Tholola," which is in the dictionary but not the intended term. The frequency score will then be used by the MEDA to perform the operations in the ordering. In this situation, we have "th" and "lo," which are syllables at i and i+1, respectively. Because the scoring of l following h is low, we perform operations on l. We begin with transpose, which means we swap h and l to get "Tlhola," which is the proper and predicted word. If the term is incorrect, we proceed with the other operations in the order listed above.

6. RESULTS DISCUSSIONS

Different factors influenced the MEDA performance. Setswana is morphologically rich, which means that most of the words are derived from other words, therefore there is a good probability that if someone misses the intended word they wanted to write, they will receive a legitimate Setswana term. Also, because most words are derived from other words, utilizing the edit distance algorithm to do different operations may result in the word being rectified by any operation to yield an acceptable Setswana word when it was not the desired word. The 8% was from the words which were added as correct while were not the desired one. The following are examples of Setswana word errors which affected the performance of MEDA.

6.1. Omission and Addition of a Character

Botswana has various ethnic groupings in different demographics northern and southern regions. This influences how they speak Setswana. People in the north commonly omit the 'l' letter while pronouncing some Setswana words, which influences how they write it. This affected MEDA performance, because some words, when 'l' is omitted, they fall into a different term which is legitimate and most of the time having a different meaning. E.g.

+beta 'beta' betla
+thapa 'thapa' tlhapa
+fapa 'fapa' fapha
+ikgantse 'ikgantse' ikgantshe

According to the examples above, 'beta' is an acceptable Setswana term, however some people pronounce 'betla' as beta, and beta was found in the dictionary whereas we wanted it to give us betla. The edit distance is the shortest distance and represents the number of operations required in a word to produce a valid word. Then there could be 0 operations, indicating that the word was already in the dictionary. As though we expected 'tlhapa' and the user typed 'thapa' because most individuals who have a problem with this 'l' character sometimes add it when it is not supposed to be included. And certain Setswana words have the letter 'l,' which will confirm the term as correct. This can also happen with the character 'h,' as shown in the list above.

6.2. Swapping of characters

This is mainly a keyboard mistake; there are characters near to one other in the "QWERTY" keyboard, and when one of them is used, it can result in a legitimate word. For example, u and I are on the same row of the keyboard. We anticipated the term amusa in the example below, but because the user made an error by inputting u, it becomes legal because amisa is a word in Setswana.e.g
+amisa amisa amusa

6.3. Morphology

Morphology is the study of words, how they are formed, and their relationship to other words in the same language. It analyzes the structure of words and parts of words, such as stems, root words, prefixes, and suffixes. So, there are words in Setswana which has suffixes/ prefixes, which when removed the remaining sub-string becomes valid.

For example if we perform delete operation, it affects most words which are coming from the other ones, usually words starting with ***n,i,m*** in Setswana, when we delete the first character, it gives us a valid Setswana word.

+ fitesa 'itesa' 'Fitisa'
+nkgakololo 'kgakololo' 'nkgakolole'

6.4. Closed Words

Since most words are closed to each other, and even though inserting ordering performing better, it has bad effects. E.g., Setswana words are almost the same but not meaning the same thing, we can say they are too close to each other. When we give insertion priority, we can end up in totally different words. E.g., the word 'lametsa' was misspelled and expecting the word ‘lamatsa’ and inserting with change the first letter with different letter and it will give us nametsa which is correct.

+ lametsa 'nametsa' 'lamatsa'
+ deresewa 'keresega' 'dirisiwa'

7. DISCUSSIONS

Due to the fact that the bulk of words in Setswana are loanwords, there is a high probability of accidentally writing a term that is both acceptable and not the one intended. This study revealed that all four edit distance operations are essential because they affect how prospective words are mutated, but the findings also indicated that the order in which the operations are carried out matters. MEDA is a morphologically rich language edit distance method. By utilizing the syllable algorithm and the frequency score of each letter in the word, this technique efficiently locates errors in words. The algorithm was tested in Setswana words and received a 92 percent, and it might be tested in additional languages which uses the same writing style.

REFERENCES

[1] Hyman, Larry M. , "Segmental Phonology" , in The Bantu Languages ed. Derek Nurse and Gerard Philippson (Abingdon: Routledge, 03 Jul 2003), accessed 07 Jul 2022 , Routledge Handbooks Online.

[2] F. J. Damerau, “A technique for computer detection and correction of spelling errors,” Common. ACM, vol. 7, pp.171-176, 1964

[3] Walfish, M., Hachamovitch, & Andrew, F. (2000). Patent No 6047300. United States of America.

[4] Yonatan , B., and Yonatan B. (2017). Synthetic and natural noise both break neural machine translation. arXiv preprint arXiv:1711.02173.

[5] Patil, C., Rodrigues, R., & Ron, R. (2020). Auto-Spelling Checker using Natural Language Processing.

[6] Cordeiro De Amorim, R. and Zampieri, M., 2022. Effective Spell Checking Methods Using Clustering Algorithms. [online] Uhra.herts.ac.uk. Available at: <https://uhra.herts.ac.uk/handle/2299/16916> [Accessed 7 July 2022].

[7] M. S. Rasooli, O. Kahefi and B. Minaei-Bidgoli, "Effect of adaptive spell checking in Persian," 2011 7th International Conference on Natural Language Processing and Knowledge Engineering, 2011, pp. 161-164, doi: 10.1109/NLPKE.2011.6138186.

[8] İnce, E. Y. (2017). Spell checking and error correcting application for Turkish. International Journal of Information and Electronics Engineering, 7(2), 68-71.

[9] Mjaria, F., & Keet, C. M. (2018, May). A statistical approach to error correction for isiZulu spellcheckers. In 2018 IST-Africa Week Conference (IST-Africa) (pp. 1-of). IEEE.

[10] Mashiane, N. An overview of digitization of African languages, spellchecking techniques & the progress of spellcheckers globally.

[11] Islam, M. I. K., Meem, R. I., Kasem, F. B. A., Rakshit, A., & Habib, M. T. (2019, May). Bangla spell checking and correction using edit distance. In 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT) (pp. 1-4). IEEE

[12] Iqbal, S., Anwar, W., Bajwa, U. I., & Rehman, Z. (2013, October). Urdu spell checking: Reverse edit distance approach. In Proceedings of the 4th workshop on south and southeast asian natural language processing (pp. 58-65).

[13] Haldar, Rishin & Mukhopadhyay, Debajyoti. (2011). Levenshtein Distance Technique in Dictionary Lookup Methods: An Improved Approach. Computing Research Repository - CORR

AUTHORS

B. Okgetheng: MSc, BSc Computer Science.

G. Malema: PhD Computer Engineering, MS Electrical and Computer Science, BS Computer Engineering.

Ariq Ahmer: Bsc Computer Science Student.

Boemo Lenyibi: Bsc Computing with Finance Student.

O. Ishmael: MSc Computer Science, BSc Computer Systems Engineering.