# MYANMAR AUTOMATIC SPELLING CORRECTION BASED ON N-GRAM MODEL

**THAZIN WIN**

**M.C.Sc.** **SEPTEMBER 2022**

# MYANMAR AUTOMATIC SPELLING CORRECTION BASED ON N-GRAM MODEL

**By**

**Thazin Win**

**B.C.Sc.**

**A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of**

**Master of Computer Science**

**(M.C.Sc.)**

**University of Computer Studies, Yangon**

**September 2022**

# ACKNOWLEDGEMENTS

I would like to express my immense gratitude to those who assisted me with different parts of directing examination and composing this thesis. To finish this thesis, numerous things are required like my diligent effort as well as the supporting of many individuals.

I, first and foremost, would like to offer my most profound thanks to **Dr. Mie Mie Khin**, Rector, the University of Computer Studies, Yangon, for her caring consent to present this thesis.

My thanks and respects go to my supervisor, **Dr. Win Pa Pa**, Professor, Natural Language Processing Lab,Faculty of Computer Science of the University of Computer Studies, Yangon, for her help, direction, management, tolerance and support during the time of study towards fulfillment of this thesis.

I would like to communicate my appreciation to **Dr. Si Si Mar Win**, and **Dr. Tin Zar Thaw**, the course coordinators, Software Department of the University of Computer Studies, Yangon, for their unrivaled idea, managerial backings and support during my academic study.

I also want to offer my most profound thanks to **U Aung Myint Than**, Assistant Lecturer, English Department, the University of Computer Studies, Yangon, for altering this thesis according to the language perspective.

Besides, I would like to stretch out my gratitude to all my respectful teachers who taught and guided me throughout the master's degree course and my friends for their participation.

I particularly thank my parents, each of my classmates, and their support and help during my thesis.

# STATEMENT OF ORIGINALITY

I hereby certify that the work embodied in this thesis is the result of original research and has been submitted for a higher degree to any other University or Institution.

……... …..……..……

Date Thazin Win

# ABSTRACT

Myanmar spelling correction intended for real-word errors and non-word errors. There are three main modules in this thesis. They are error detection,candidates generation, error correction. Dictionary look up method is used for detecting errors, Levenshtein Distance Algorithm is used for generating candidates and N-gram model is used for correcting errors. There can be human-generated misspellings which can be distinguished into three groups (i) Typographic Errors (Non-word error) (ii)Phonetic Errors (Cognitive error) (iii) Context Errors (Real word errors). This spelling correction can solve all of these three misspellings problem and the main contribution of this system is to solve the context errors using n-gram model in sentence level. Moreover, this spelling correction can solve the pali and patsint misspelling errors. Experimental results show that each of error types can be solved by this spelling correction. The general accuracy of all error types is greater than 85%. This system is implemented by using python programming language with Linux system.

# CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF EQUATIONS

# LIST OF ABBRIVIATIONS

| | |
|---|---|
| IBM | International Business Machines |
| TB | Tibeto-Burman |
| ST | Sino-Tibetan |
| OCR | Optical Character Recognition |
| MT | Machine Translation |
| NLP | Natural Language Processing |

# CHAPTER 1
# INTRODUCTION

Spelling correction for Myanmar Language has more difficulties than that for English. Spell checking detects misspelled words in text. Spell correction gives advice after errors are detected. When only the former works, it is spell checking and when the latter is also involved, it is spell correction. The work in this system will focus on both tasks. For English spell checking, many studies have been made and good results have been obtained. For Myanmar spell checking,it is still challenging work due to Myanmar writing system. Unlike English,word boundaries are not marked with spaces in Myanmar sentences. So, it is difficult to tokenize. Spelling errors can be classified into two main categories (i)non-word error and (ii)real-word error. The former is one in which the input word is definitely incorrect and cannot be found in dictionary. For example, using “fcrm” rather than “farm”. The latter is one in which the input word is found in the dictionary but is incorrectly used. For example, using “come form” than “come from”. The most common reasons for misspelled and misused errors are caused by phonetic similarity and typing error of Myanmar characters. In this work, first we study the details on Myanmar Language to identify the problem area of Myanmar spell errors and then we develop Myanmar Spell Correction. It consists of three phases: error detection,candidates generation and error correction. Myanmar Language Commission (MLC) dictionary is used to detect errors. Levenshtein Distance Algorithm is used for generating candidates and N-gram Model is applied for error correction.

## 1.1 Objectives of the System

The main objectives of this system are:

- To detect and correct typographic errors (non-word errors)
- To detect and correct phonetic errors (cognitive errors)
- To detect and correct context errors (real-word errors)

## 1.2 Related Work

Like many other natural language processing tasks, spell checking is one the most important tasks. Many researches for English spell checking have been done for four decades. Even though other Asian spell checker researches have been done for two decades, Myanmar spell checker research is still challenging work. In this section, we discuss briefly some of related work.

In July 2015,Chinese Spelling Checker System Based on N-gram Model is presented by Bingzhou Chen and Lei Huang. In July 2009,Developing A spell Checker for Myanmar Unicode System is presented by Soe Moe Aye, Master Thesis of dissertation at University of Computer Studies,Yangon.

In December 2010,Myanmar Words Spelling Checking Using Levenshtein Distance Algorithm is presented by Nwe Zin Oo,Master Thesis of dissertation at University of Computer Studies,Yangon. The above systems only generate candidates for misspelled errors and cannot correct misspelled words automatically. They only check the misspelled words and cannot check as the sentence. That system can correct the sentence which includes misspelled words automatically.

## 1.3 Organization of the Thesis

This thesis is organized into five chapters. They are as follows:

**In Chapter 1,** introduction of the system, objectives of the system, related work and thesis organization are described. **Chapter 2** presents the background theory of Myanmar spelling correction. **Chapter 3** discusses the architecture of the proposed system. **Chapter 4** expresses the implementation and experimental results of the proposed system. Finally, **Chapter 5** presents the conclusions, advantages of the system, limitations and further extensions of the system.

# CHAPTER 2
# BACKGROUND THEORY

Spelling Checking is a software program or program feature designed to locate misspelled words and notify the user of the misspellings. Depending on the spell checker, the feature may either autocorrect the word or allow the user to select from potential corrections on the misspelled word.

## 2.1 Background History of Spelling Checking

This Spelling checking is an essential part for almost all application software. In software, a spell checker is a software feature that checks for misspellings in a text. Spell-checking features are often embedded in software or services, such as a word processor, email client, electronic dictionary, or search engine.

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). This might contain just a list of words, or it might also contain additional information, such as hyphenation points or lexical and grammatical attributes. 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.

The first spell checkers were widely available on mainframe computers in the late 1970s. A group of six linguists from Georgetown University developed the first spell-check system for the IBM corporation. The first spell checkers for personal computers appeared in 1980, such as "WordCheck" for Commodore systems which was released in late 1980 in time for advertisements to go to print in January 1981. by the mid-1980s developers of popular word-processing packages like WordStar and WordPerfect had incorporated spell checkers in their packages, mostly licensed from the above companies, who quickly expanded support from just English to many European and eventually even Asian_language. However, this required increasing

sophistication in the morphology routines of the software, particularly with regard to heavily-agglutinative languages like Hungarian and Finnish.

Firefox 2.0, a web browser, has spell check support for user-written content, such as when editing Wikitext, writing on many webmail sites, blogs, and social networking websites. The web browsers Google Chrome, Konqueror, and OPera, the email client Kmail and the instant messaging client Pidgin also offer spell checking support, transparently using previously GNU ASpell and currently Hunspell as their engine.

English is unusual in that most words used in formal writing have a single spelling that can be found in a typical dictionary, with the exception of some jargon and modified words. In many languages, words are often concatenated into new combinations of words. In German, compound nouns are frequently coined from other existing nouns. Some scripts do not clearly separate one word from another, requiring word-splitting algorithms. Each of these presents unique challenges to non-English language spell checkers.

There has been research on developing algorithms that are capable of recognizing a misspelled word, even if the word itself is in the vocabulary, based on the context of the surrounding words. For example, baht in the same paragraph as Thai or Thailand would not be recognized as a misspelling of bath. The most common example of errors caught by such a system are homophone errors.

The problem of detecting misspelled words in sentences and automatic correcting them is a great challenge in the world. Its solution has enormous application potentials in texts and code editing, computer aided authoring, optical Character Recognition (OCR), Machine Translation (MT), Natural Language Processing (NLP), database retrieval and information retrieval interface, Speech recognition, text to speech and speech to text conversion.

In spelling checking, there are two types of errors namely, non-word error and real-word error. Non-word error is the result of a spelling error where the word itself is not in the dictionary and is not a known word. For example, mistakenly spelling “apple” into “appll” is a non-word error because “appll” is not in the dictionary. Real-word error is due to misspelling a word to make another word that is in the dictionary. For example, mistakenly spelling “apple” in “I have an apple” as "apply", makes the

sentence “I have an apply”. This is a real-word error because “apply” is in the dictionary, but is not the right word. So ,it is a great challenge that a spell checker can check and correct both non-words and real-words.

## 2.2 Background History of Myanmar Language

The Burmese language belongs to the Tibeto-Burman group of the Tibeto-Chinese family of languages, but, unlike Chinese, it is not ideographic. That is, it does not have characters which originated as pictures, but an alphabet, of eleven vowels and thirty-three consonants, derived from the Pahlavi script of South India. Sino-Tibetan(ST)’s language family tree is shown in figure 2.1.

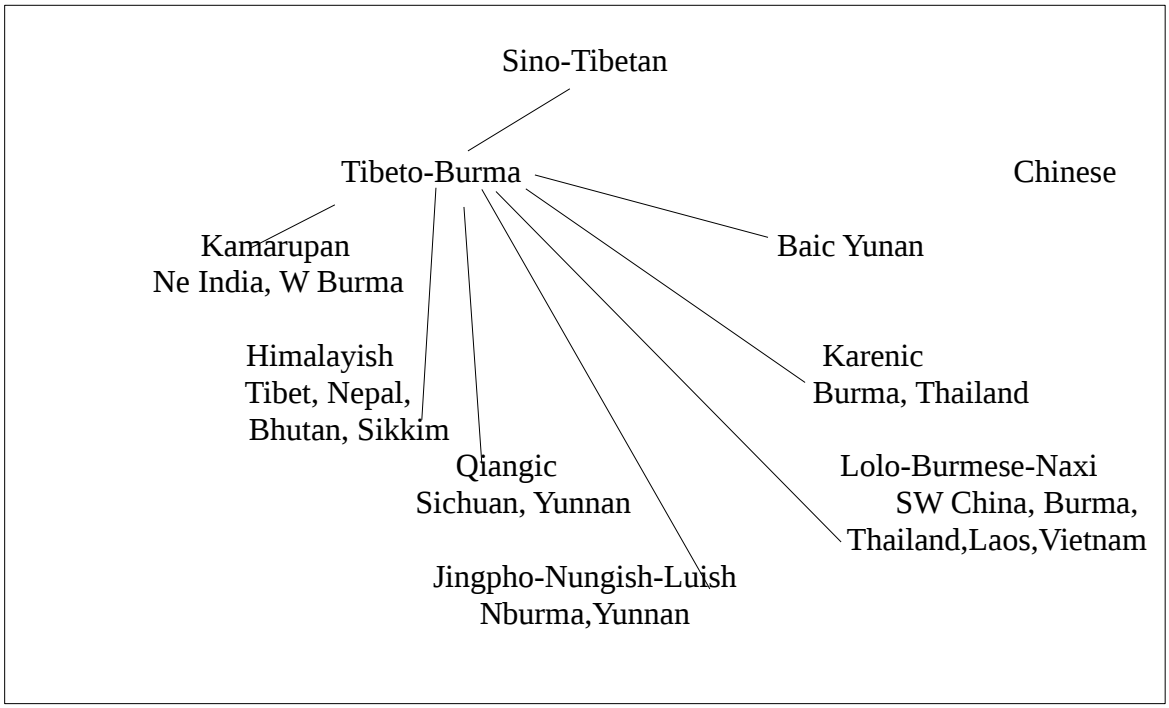


**Figure 2.1 Language Family Tree of Burmese**

Sino-Tibetan is one of the great language families of the world, containing hundreds of languages spoken by over 1 billion people, from Northeast India to the Southeast Asian peninsula. The best-known languages in the family are Chinese, Tibetan, and Burmese. Tibeto-Burman (TB) comprises hundreds of languages besides Tibetan and Burmese, spread over a vast geographical area (China, India, the Himalayan region, peninsular SE Asia).

## 2.3 History of Computerization in Myanmar

The history of computerization in Myanmar started in early 1970s. In 1988, mainframes, minicomputers and microcomputers were used by the governmental organizations, and were not available to the public sector due to the economic policies of the government. With the shift of Government's policy to the market economy in 1988, private sector was able to embrace computerization. The introduction of limited email connection in 1995 allowed computerization to be extended to cover Information and Communication Technology (ICT) development.

The history of computerization in Myanmar started in early 1970s with the installation of the first computer at the Universities' Computer Center (UCC) under a United Nations funded project. The UCC project also introduced minicomputer and microcomputers into the country in the early 1980s. Training courses were conducted by UCC and post graduate programs in computing were offered in conjunction with the Mathematics Department, Rangoon Arts and Science University (RASU). In the mid 1980s, the government implemented a UN funded computer development project at the Central Statistical Organization (CSO). The PC revolution and a shift to market economy in 1988 resulted in the second phase of computerization. Many private computer companies were established mainly in training and hardware. Software applications and development followed with the opening of foreign and local private enterprises in the country. In 1997, Myanmar Post and Telecommunications (MPT), under the Ministry of Communication, Post and Telegraph provided Internet and email connections. The third phase of computerization came with the ICT development. At the urging of local computer companies and computer users, the Myanmar ICT Park was established in Yangon in 2002. A second Internet Service Provider (ISP), BaganNet was also established. Recent developments included the establishment of Yadanarbon Cyber City in upper Myanmar in December 2007.The history of computerization in Myanmar will be presented in accordance with the three development periods: the first from early 1970s to 1987, the second from 1988 to 1994, and the third from 1995 to present. For each period, the infrastructure will be covered, hardware, software, training, and applications and

also present how the widespread use of computer and information technology in daily life in Myanmar lead to the localization efforts.

## 2.4 Myanmar Language Features

In Myanmar language, there are 33 consonants (byi), 4 basics medials (byi twe) and 11 vowels (thara) , special characters, punctuation mark, digits and stacked consonants to represent vowels sounds,tones marks, specified symbols and punctuation marks.

### 2.4.1 Myanmar consonants

Myanmar consonant is a transformation of the Pyu content, or Old Mon content and it is at last of South Indian beginning. The Burmese letters in order is organized into gatherings of five letters for stop consonants called sway (ဝဂ်) in light of enunciation. Inside each gathering, the primary letter is tenuis ("plain"), the second is the suctioned homologue, the third and fourth are the voiced homologues and the fifth is the nasal homologue. This is valid for the initial 25 letters in the Burmese letter set, which are assembled gathered as classified consonants or "wag character" (ဝဂ်ဗျည်း). The excess eight letters (ယ, ရ, လ, ဝ, သ, ဟ, ဠ, အ) are gathered as different consonants or "awag character" (အဝဂ်ဗျည်း), as they are not organized in a specific example.

A letter is a consonant or consonant cluster that occurs before the vowels of a syllable. The Burmese script has 33 letters to indicate the initial consonant of a syllable and four diacritics to indicate additional consonants in the onset. Burmese diacritics are placed above, below, before or after the consonant character. The categorized characters and miscellaneous characters are shown in Table 2.1.

**Table 2.1: Categorized Characters and Miscellaneous Characters**

| Group name | Unaspirated | Aspirated | Voiced | Voiced | Nasal |
|---|---|---|---|---|---|
| Velars | က | ခ | ဂ | ဃ | င |
| Palatals | စ | ဆ | ဇ | ဈ | ဉ/ည |
| Alveolars | ဋ | ဌ | ဍ | ဎ | ဏ |
| Dentals | တ | ထ | ဒ | ဓ | န |
| Labials | ပ | ဖ | ဗ | ဘ | မ |
| Without group | ယ | ရ | လ | ဝ | သ |
| Without group | | ဟ | ဠ | အ | |

### 2.4.2 Myanmar Medials

Consonant letters may be modified by one or more medial diacritics (three at most), indicating an additional consonant before the vowel. These diacritics are:

- *Ya pin* (ယပင်) - Written - ◌ျ ( palatalization of a velar consonant)
- *Ya yit* (ရရစ်) - Written ◌ြ – (palatalization of a velar consonant)
- *Wa hswe* (ဝဆွဲ) - Written ◌ွ (medial)
- *Ha hto* (ဟထိုး) -Written ◌ှ (a sonorant consonant is voiceless)

The 4 basic medials (byi twe) are shown in Table 2.2 and all the possible diacritic combination with consonant " မ " are shown in figure 2.2

**Table 2.2: Basic Medials (Byi Twe)**

| Ya pin | Ya yit | Wa hswe | Ha hto |
|---|---|---|---|
| ◌ျ | ◌ြ | ◌ွ | ◌ှ |

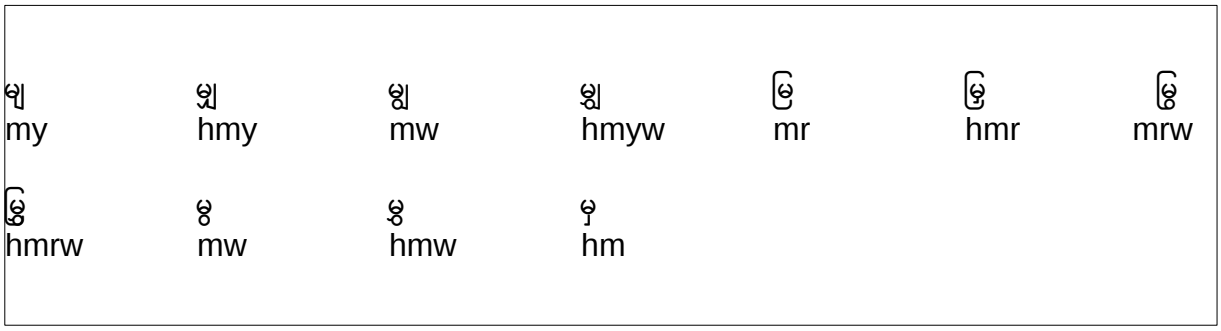


**Figure2.2 Possible diacritic combination with consonant " မ "**

### 2.4.3 2.4.3 Myanmar Vowels

Vowels are the basic building blocks of formation in Myanmar Language, although a syllable or a word can be formed from just consonants, without a vowel. Vowels are known as “ Thara ” . The 11 vowels symbols are shown in Table 2.3 and the 22 Myanmar vowels are shown in table 2.4.

**Table 2.3: The 11 vowels symbols**

| -ာ (ရေးချ) | –ိ (လုံးကြီးတင်) | –ီ (လုံးကြီးတင်ဆန်ခတ် | –ု (တစ်ချောင်းငင်) |
|---|---|---|---|
| –ူ (နှစ်ချောင်းငင်) | ေ– (သဝေထိုး) | –ဲ (နောက်ပစ်) | ေ–ာ (သဝေထိုးရေးချ) |
| ေ–ာ် (သဝေထိုးရေးချ ရှေ့ထိုး) | –ံ (သေးသေးတင်) | -း (ဝစ္စပေါက်) | |

**Table 2.4: The 22 Myanmar vowels**

| အ | အာ | အား |
|---|---|---|
| အိ | အီ | အီး |
| အု | အူ | အူး |
| အေ | အေ့ | အေး |
| အဲ | အဲ့ | |
| အော | အော့ | အော် |
| အံ | အံ့ | |
| အို | အို့ | အိုး |

### 2.4.4 Special Characters(Independent Vowels)

The special characters, represented by separated codes, are mostly complete words or syllables by themselves. They formed independent words with complete meaning and do not require any extra consonants, medials, vowels to become a word. So, these characters are also known as independent vowels. They are shown in Table 2.5.

**Table 2.5: The Independent Vowels**

| | | | |
|---|---|---|---|
| ဣ | ဤ | ဥ | ဦ |
| ဧ | ဩ | ဪ | |

### 2.4.5 Punctuation

There are two primary break characters in Burmese, drawn as one or two downward strokes: ။ (called ပုဒ်ဖြတ်, ပုဒ်ကလေး, ပုဒ်ထီး, or တစ်ချောင်းပုဒ်) and ။ (called ပုဒ်ကြီး, ပုဒ်မ, or နှစ်ချောင်းပုဒ်), which respectively act as a comma and a full stop. Other abbreviations used in literary Burmese are: " ၏ " is used as a full stop if the sentence immediately ends with a verb and also used as possessive particle( 's, of), " ၍ " is used as a conjunction. " ၌ " is used as locative ('at'), " ၎င်း " is used in columns and lists.

### 2.4.6 Myanmar Digits

A decimal numbering system is used, and numbers are written in the same order as Hindu–Arabic numerals. The digits from zero to nine are: ၀၁၂၃၄၅၆၇၈၉. Separators, such as commas, are not used to group numbers. The comparison English digits and Myanmar digits is shown in Table 2.6.

**Table 2.6: The Comparison of English Digits and Myanmar Digits**

| English Digits | Myanmar Digits |
|---|---|
| 0 | ၀ |
| 1 | ၁ |
| 2 | ၂ |
| 3 | ၃ |
| 4 | ၄ |
| 5 | ၅ |
| 6 | ၆ |
| 7 | ၇ |
| 8 | ၈ |
| 9 | ၉ |

### 2.4.7 Stacked Consonants

Certain arrangements of consonants are kept in touch with one on the other, or stacked. A couple of stacked consonants shows that no vowel is articulated between them, with respect to model ကမ္ဘာ is comparable to ကမ်ဘာ. As Stacked consonants are consistently homorganic (articulated in a similar spot in the mouth), which is demonstrated by the conventional game plan of the Burmese letters in order into five-letter columns of letters called ဝဂ်. Ordered consonants must be multiplied - that is, stacked with themselves. At the point when stacked, the primary consonant is composed not surprisingly, while the subsequent consonant is subscripted underneath it. The stacked consonants and models are displayed in Table 2.7.

**Table 2.7: The Possible Combinations and Examples of Stacked Consonants**

| Possible Combinations | Examples |
|---|---|
| က္က, ဂ္ဂ, ဂ္ဂ, ဣ္ဣ | ဒုက္ခ |
| စ္စ, ဆ္ဆ, ဇ္ဇ, ဈ္ဈ, ဈ္ဈ, ဈ္ဈ, ဈ္ဈ | ဝိဇ္ဇာ |
| ဉ္ဉ, ဋ္ဋ, ဍ္ဍ, ဿ, ဏ္ဏ, ဏ္ဏ | ကဏ္ဏ |
| တ္တ, ထ္ထ, ဒ္ဒ, ဓ္ဓ, ဏ္ဏ, ဏ္ဏ, ဒ္ဒ, ဓ္ဓ, ဓ္ဓ | မန္တလေး |
| ပ္ပ, ဖ္ဖ, ဗ္ဗ, ဘ္ဘ, မ္မ, မ္မ, မ္မ, မ္မ, | ကမ္ဘာ |
| ဿ, လ္လ, ဠ္ဠ | ဝိဿာ |

## 2.5 Edit Distance Algorithms

In computational linguistics and computer science, edit distance is a string metric, i.e. a way of quantifying how dissimilar two strings (e.g., words) are to one another, that is measured 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. In bioinformatics, it can be used to quantify the similarity of DNA sequences, which can be viewed as strings of the letters A, C, G and T.

There are many edit distance algorithms for spelling checking such as, Levenshtein Distance, Longest Common Subsequence, Hamming Distance, Damerau-Levenshtein distance and Jaro distance. The Levenshtein distance allows deletion, insertion and substitution. The longest common subsequence (LCS) distance allows only insertion and deletion, not substitution. The Hamming distance allows only substitution, hence, it only applies to strings of the same length. The Damerau–Levenshtein distance allows insertion, deletion, substitution, and the transposition of two adjacent characters. The Jaro distance allows only transposition. Among them, Levenshtein Distance is suitable for measuring the amount of difference between two sequences.

### 2.5.1 Levenshtein Distance

In data hypothesis, phonetics, and software engineering, the Levenshtein distance is a string metric for estimating the contrast between two successions. Casually, the Levenshtein distance between two words is the base number of single-character alters (inclusions, cancellations or replacements) expected to transform single word into the other. It is named after the Soviet mathematician Vladimir Levenshtein, who thought about this distance in 1965.

For instance, the Levenshtein distance among "kitten" and "sitting" is 3, since the accompanying 3 alters change one into the other, and it is absolutely impossible to do it with less than 3 alters:

kitten → sitten (substitution of "s" for "k"),

sitten → sittin (substitution of "i" for "e"),

sittin → sitting (insertion of "g" at the end).

The Levenshtein algorithm that takes two strings, *s* of length *m*, and *t* of length *n*, and returns the Levenshtein distance between them is shown in figure 2.3

```
function LevenshteinDistance(char s[1..m], char t[1..n]):
 set each element in d to zero
 for i from 1 to m:
   d[i, 0] := i
 for j from 1 to n:
   d[0, j] := j

 for j from 1 to n:
   for i from 1 to m:
     if s[i] = t[j]:
       substitutionCost := 0
     else:
       substitutionCost := 1

     d[i, j] := minimum(d[i-1, j] + 1,
                        d[i, j-1] + 1,
                        d[i-1, j-1] + substitutionCost)

 return d[m, n]
```

**Figure 2.3: Levenshtein Distance Algorithm**

### 2.5.2 Longest Common Subsequence

The longest common subsequence (LCS) problem is the problem of finding the longest subsequence common to all sequences in a set of sequences (often just two sequences). It differs from the longest common substring problem: unlike substrings, subsequences are not required to occupy consecutive positions within the original sequences. The longest common subsequence problem is a classic computer science problem, the basis of data comparison programs such as the diff utility, and has applications in computational linguistics and bioinformatics. It is also widely used by

revision control systems such as Git for reconciling multiple changes made to a revision-controlled collection of files.

For example, consider the sequences (ABCD) and (ACBAD). They have 5 length-2 common subsequences: (AB), (AC), (AD), (BD), and (CD); 2 length-3 common subsequences: (ABD) and (ACD); and no longer common subsequences. So (ABD) and (ACD) are their longest common subsequences. Longest common subsequence algorithm is shown in figure 2.4.

```
function LCSLength(X[1..m], Y[1..n])
    C = array(0..m, 0..n)
    for i := 0..m
        C[i,0] = 0
    for j := 0..n
        C[0,j] = 0
    for i := 1..m
        for j := 1..n
            if X[i] = Y[j]
                C[i,j] := C[i-1,j-1] + 1
            else
                C[i,j] := max(C[i,j-1], C[i-1,j])
    return C[m,n]
```

**Figure 2.4: Longest Common Subsequence Algorithm**

### 2.5.3 Hamming Distance

The number of positions at which the corresponding symbols differ between two strings of equal length is referred to as the Hamming distance in information theory.To put it another way, it measures the smallest number of mistakes that could have transformed one string into the other or the smallest number of substitutions required to change one string into the other. The Hamming distance is one of several string metrics that can be used to measure the edit distance between two sequences in a broader sense. The American mathematician Richard Hamming is the inspiration for its name.

The symbols may be letters, bits, or decimal digits, among other possibilities. For example, the Hamming distance between: "karolin" and "kathrin" is 3, "karolin" and "kerstin" is 3, "kathrin" and "kerstin" is 4, 0000 and 1111 is 4, 2173896 and 2233796 is 3.

```
def hamming_distance(string1, string2):
    dist_counter = 0
    for n in range(len(string1)):
        if string1[n] != string2[n]:
            dist_counter += 1
    return dist_counter
```

**Figure 2.5: Hamming Distance Algorithm**

### 2.5.4 Damerau -Levenshtein Distance

A string metric for determining the edit distance between two sequences is the Damerau–Levenshtein distance.Informally, the minimum number of insertions, deletions, or substitutions of a single character, or transpositions of two adjacent characters, required to change one word into the other is referred to as the Damerau–Levenshtein distance between two words.

In contrast to the traditional Levenshtein distance, the Damerau–Levenshtein distance allows for transpositions in addition to the three traditional single-character edit operations (insertion, deletion, and substitution).Figure 2.5 depicts the Damerau–Levenshtein distance algorithm.

```
algorithm DL-distance is
    input: strings a[1..length(a)], b[1..length(b)]
    output: distance, integer

    da := new array of |Σ| integers
    for i := 1 to |Σ| inclusive do
        da[i] := 0

    let d[−1..length(a), −1..length(b)]

    maxdist := length(a) + length(b)
    d[−1, −1] := maxdist
    for i := 0 to length(a) inclusive do
        d[i, −1] := maxdist
        d[i, 0] := i
    for j := 0 to length(b) inclusive do
        d[−1, j] := maxdist
        d[0, j] := j

    for i := 1 to length(a) inclusive do
        db := 0
        for j := 1 to length(b) inclusive do
            k := da[b[j]]
            ℓ := db
            if a[i] = b[j] then
                cost := 0
                db := j
            else
                cost := 1
            d[i, j] := minimum(d[i−1, j−1] + cost,
                               d[i,   j−1] + 1,
                               d[i−1, j  ] + 1,
                               d[k−1, ℓ−1] + (i−k−1) + 1 + (j-ℓ−1))
        da[a[i]] := i
    return d[length(a), length(b)]
```

**Figure 2.6: Damerau–Levenshtein Distance Algorithm**

### 2.5.5 Jaro–Winkler Distance

A string metric that measures the edit distance between two sequences is the Jaro–Winkler distance. A prefix scale called p is used in the Jaro–Winkler distance, and for a given prefix length, it gives higher ratings to strings that match from the beginning. The closer two strings are to one another, the closer their Jaro–Winkler

distance is. The score is normalized so that a value of 1 indicates an exact match and a value of 0 indicates no similarity. The metric was actually defined in terms of similarity in the initial system, so the distance is the inversion of that value.The Jaro–Winkler distance is not a metric in the mathematical sense of that term because it does not adhere to the triangle inequality, despite the fact that it is frequently referred to as such. In figure 2.6, the Jaro_Winkler algorithm is depicted.

```
double jaro_Winkler(string s1, string s2)
{    double jaro_dist = jaro_distance(s1, s2);
     if (jaro_dist > 0.7) {
          int prefix = 0;
          for (int i = 0;i < min(s1.length(), s2.length()); i++)
          {
                    if (s1[i] == s2[i])
                              prefix++;
                    else
                              break;
          }
          prefix = min(4, prefix);
          jaro_dist += 0.1 * prefix * (1 - jaro_dist);
     }
     return jaro_dist;
}
```

**Figure 2.7: Jaro–Winkler Distance Algorithm**

## 2.6 Language Model

A language model is a probability distribution over sequences of words. Given such a sequence of length m, a language model assigns a probability P ( w 1 , … , w m ) to the whole sequence. Language models generate probabilities by training on text corpora in one or many languages. Given that languages can be used to express an infinite variety of valid sentences (the property of digital infinity), language modelling faces the problem of assigning non-zero probabilities to linguistically valid sequences that may never be encountered in the training data. Several modelling approaches have been designed to surmount this problem, such as applying the Markov assumption or using neural architectures such as recurrent neural networks or transformers.

Language models are useful for a variety of problems in computational linguistics; from initial applications in speech recognition to ensure nonsensical (i.e. low-probability) word sequences are not predicted, to wider use in machine translation (e.g. scoring candidate translations), natural language generation (generating more human-like text), part-of-speech tagging, parsing, Optical Character Recognition, handwriting recognition, grammar induction, information retrieval,spell checking and other applications.

### 2.6.1 N-gram Model

An n-gram model is a type of probabilistic language model that uses a (n 1)–order Markov model to predict the next item in a sequence.Probability, communication theory, computational linguistics (such as statistical natural language processing), computational biology (such as biological sequence analysis), and data compression all make use of N-gram models.Simpleness and scalability are two advantages of n-gram models. With a larger n, a model can store more context with a well-understood space–time tradeoff, allowing small experiments to grow quickly.

An n-gram of size 1 is referred to as a "unigram"; size 2 is a "bigram"; size 3 is a "trigram"; size 4 is a "four-gram"; size 5 is a "five-gram", and so on.

### 2.6.2 Bi-gram Model

A sequence of two adjacent elements from a string of tokens—typically letters, syllables, or words—is referred to as a bigram or digram. An n-gram with size n=2 is a bigram. In numerous fields, such as computational linguistics, cryptography, speech recognition, and so on, the frequency distribution of each bigram in a string is frequently utilized for straightforward statistical text analysis.

When the relation of the conditional probability is applied, Bigrams assist in providing the conditional probability of a token in light of the token that came before it. The equation of bigram calculation is shown as follow.

$$P(W_{n|}|W_{n-1}) = \frac{Count\,(Wn-1, Wn)}{Count\,(Wn-1)} \qquad (2.1)$$

### 2.6.3 Bigram probability estimate of a word sequence

For Calculating bigram probability estimate of a word sequence, we need a corpus and the test data. Let us assume that the following is a small corpus;

Training corpus:

<s> I am from Vellore </s>

<s> I am a teacher </s>

<s> students are good and are from various cities</s>

<s> students from Vellore do engineering</s>

Test data:

<s> students are from Vellore </s>

P(<s> students are from Vellore </s>)

= P(students | <s>) * P(are | students) * P(from | are)

* P(Vellore | from) * P(</s> | Vellore)

= 1/4 * 1/2 * 1/2 * 2/3 * 1/2 = 0.0208

# CHAPTER 3
# MYANMAR SPELLING CORRECTION

A Myanmar Spelling Checker is an essential component of many of the applications such word processors as well as the more exotic applications. In this system, it proposes the process of checking the spelling of a Myanmar input sentence which includes misspelled words, produces suggestion list if it is misspelled Myanmar word and then correct automatically.

## 3.1 Myanmar Spelling Correction

For every language, spell checker is an essential component of many of the common Desktop applications, Machine Translation system and Office Automation system. In Myanmar, Myanmar Language is used as an official language. Myanmar Pronunciation and orthography has differences because spelling is often not an accurate reflection of pronunciation. In this system, we developed Myanmar Spell Checker which can handle Typographic Errors (Non-word Errors), Phonetic Errors(Cognitive error) and Context Errors(Real-word Errors) of Myanmar words. If misspelled word contains in the input sentence, this system can correct misspelled Myanmar words automatically. We apply Myanmar Dictionary to find misspelled words, Levenshtein Distance Algorithm to generate candidates, N-gram Model using Myanmar text Corpus to correct Myanmar words. The system can improve the quality of suggestion for misspelled Myanmar words and users' efficiency when the users cannot figure out the correct spelling by themselves.

Most spelling checkers can only generate the candidates suggestion for the misspelled words. Spelling suggestion features are commonly included in search engine, word processors, spell checker, medical transcription,automatic query reformation and frequency-log statics reporting.

A Myanmar spelling correction is an essential component of many of the common desktop applications such word processor,search engine and dictionary.

## 3.2 Types of misspelled words

This system can correct three types of misspelled words. Generally,misspelled words can be distinguished into three groups: (i) Typographic Errors (Non-word

errors) (ii) Phonetic Errors (Cognitive errors) and (iii) Context Errors (Real-word errors).

### 3.2.1 Typographic Errors(Non-word errors)

These mistakes have been made by the typist incidentally presses some unacceptable key. These mistakes are made accepting that the author or typist knows how to spell the word yet may have composed the word hurriedly bringing about a blunder. For instance, "ယခုအခါဗိုလ်ချုပပြတိုက်ကိုဖွင့်လစ်ထားပြီဖြစ်သည်" , the mistyped word is (ဗိုလ်ချုပ) .The typist need to embed ( ်, vowel). The word (ဗိုလ်ချုပ) has no significance and the right word is (ဗိုလ်ချုပ်). The model sentence of typographic errors is displayed in figure 3.1.

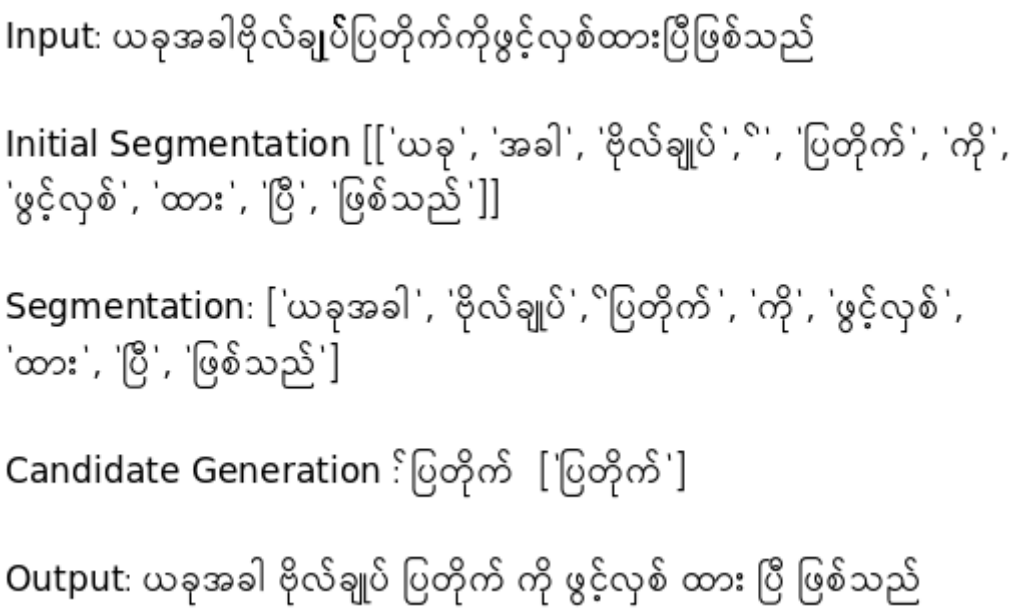


**Figure 3.1: Example Sentence of Typographic Error**

### 3.2.2 Phonetic Errors(Cognitive errors)

They have been made by an absence of information on the essayist. These blunders are made when the essayist subbed letters they accept sound right into a word. The incorrectly spelled word(နေလှမ်း) is articulated equivalent to the right word(နေလှန်း).

The incorrect spelling is articulated equivalent to the right word yet the spelling is off-base which coincidentally produce a genuine word(e.g., incorrect spelling 'စိမ်းလန်း' as 'စိမ်းလမ်း'). The model sentence of phonetic mistake is displayed in figure 3.2.

Input: သမင်သည်ကြော့ကွင်းတွင်မိနေသည်

Initial Segmentation [['သမင်', 'သည်', 'ကြော့', 'ကွင်း', 'တွင်', 'မိ', 'နေသည်']]

Segmentation: ['သမင်', 'သည်', 'ကြော့ကွင်း', 'တွင်', 'မိနေသည်', '']

Candidates Generation : ကြော့ကွင်း ['ကျော့ကွင်း']

Output: သမင် သည် ကျော့ကွင်း တွင် မိနေသည်

**Figure 3.2: Example Sentence of Phonetic Error**

### 3.2.3 Context Errors(Real-word errors)

They should be visible to be a subset of phonetic mistakes which produce a phonetic word blunder. In the sentence, 'အခန်းအနား စတော့မည်,the word(ခန်း) is articulated equivalent to the right word (ခမ်း).In the sentence, 'သူ အခန်း ထဲသို့ ဝင်လာသည်',the word(ခန်း) is right word however incorrectly spelled word for the above sentence. Myanmar words have same elocution however various implications and various utilizations. The model sentence of context blunder is displayed in figure 3.3.

Input: သူအခမ်းထဲသို့ဝင်လာသည်

Initial Segmentation [['သူ', 'အ', 'ခမ်း','ထဲသို့', 'ဝင်လာသည်']]

Segmentation: ['သူ', 'အခမ်း','ထဲသို့', 'ဝင်လာသည်']

Candidate Generation : အခမ်း ['အခန်း']

Output: သူ အခန်း ထဲသို့ ဝင်လာသည်

**Figure 3.3: Example Sentence of Context Error**

# CHAPTER 4

# IMPLEMENTATION AND EXPERIMENTAL RESULTS

In this chapter, the implementation of the system and experimental results of the system are described. The experimental results show the comparison of typographic errors, cognitive errors and context errors.

## 4.1 Implementation of the System

The proposed system presents the correct words for misspelled words. If the user type the sentence including the misspelled words, the system will generate the candidates for misspelled words and then choose the best ones for misspelled words. Finally, the system presents the sentence including the correct words.

## 4.2 The Detail of the System's Design

The proposed system can check and correct the sentence with spaces or with not spaces which use Unicode System automatically. The sentence which includes any number of errors including Pali or Patsint word errors can also be corrected.

When the user types the sentence with spaces or with not spaces, the system can check and correct the sentence automatically. Firstly,the system segments the input sentence. Initial segmentation is the process of splitting a sentence into tokens by using myan-word-breaker: Myanmar Word segmentation Tool to segment initially. And then, the system segments tokens into possible words by using simple left to right maximum longest matching: making all possible combinations by using Levenshtein Distance Algorithm. It combines tokens if the minimum distance between tokens and dictionary word is less than 3. The system consider these two steps: initial segmentation and segmentation as pre-processing.

Error detection is the first step of spell correction. This step checks to see whether an input word is in the dictionary by using Myanmar Dictionary to detect misspelled words. Candidate generation is the second step of spell correction. It generates a list of possible candidates for every misspelled errors from the first step by using the Levenshtein distance algorithm to generate candidates. The final step of spell correction is error correction. It chooses the best candidate in candidates

generating from second step and correct by using N-gram model. The detailed design of Myanmar Spelling Correction system is shown in figure 4.1.

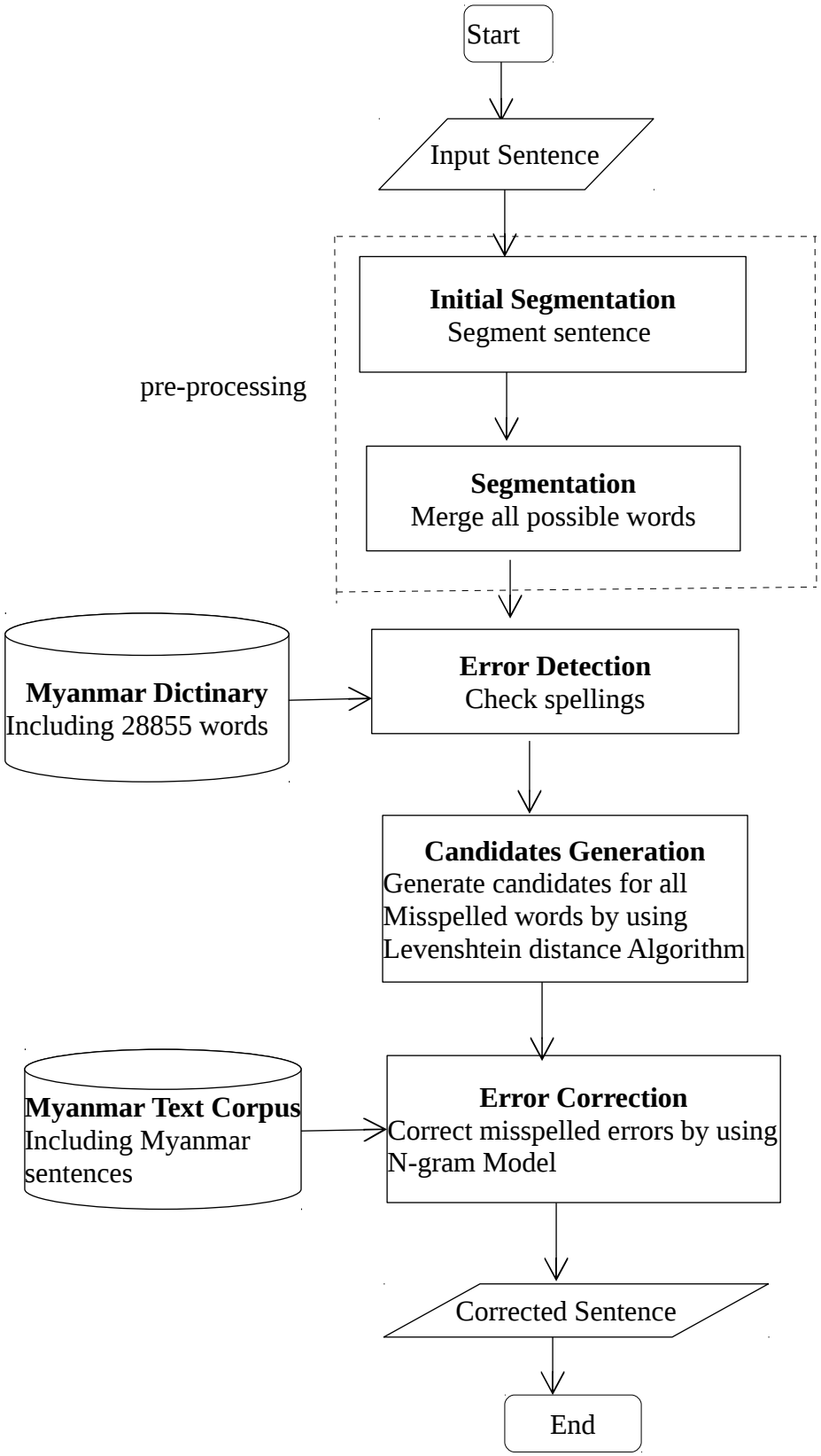


**Figure 4.1: Detail Design of Myanmar Automatic Spelling Correction System**

### 4.2.1 Error Detection

Error detection is the first step of spell correction. This step checks to see whether an input word is in the dictionary by using Myanmar Dictionary to detect misspelled words.

Example of Error Detection

The input : ကျွန်တော်သည်ကျောင်းသာတစ်ယောက်ဖြစ်သည်

Initial Segment : ကျွန်တော် သည် ကျောင်း သာ တစ်ယောက် ဖြစ်သည်

Segment : ကျွန်တော် သည် ကျောင်းသာ တစ်ယောက် ဖြစ်သည်

The item " ကျောင်းသာ " is a misspelled error because it is not in dictionary.

### 4.2.2 Myanmar Dictionary

Myanmar language commission (MLC) dictionary includes approximately 28,857 words. It Includes stem words e.g. သွား, လာ, စား, compound words e.g. မှန်တင်ခုံ,ရေအိုး,မိုးကာအင်္ကျီ, derivative words e.g. ဒါရိုက်တာ,ဗက်တီးရီးယား.

### 4.2.3 Candidates Generation

Candidates generation is the second step of spell correction. It generates a list of possible candidates for every misspelled errors from the error detection step by using the Levenshtein distance algorithm to generate candidates in this system. The Levenshtein distance between two words is the minimum number of single-character edits (insertions, deletions or substitutions) required to change one word into the other. It generates all words that are at a Levenshtein distance of 2 or less than 2 from misspelled words.

Example of Candidates Generation

The input : ကျွန်တော်သည်ကျောင်းသာတစ်ယောက်ဖြစ်သည်

Initial Segment : ကျွန်တော် | သည် | ကျောင်း | သာ | တစ် | ယောက် | ဖြစ်သည် |

Segment : ကျွန်တော် | သည် | ကျောင်းသာ | တစ်ယောက် | ဖြစ်သည်

Candidates Generation : ကျောင်းသာ [‘ကျောင်းသား’,’ကျောင်းစာ’,’ကျောင်းသူ’]

The Levenshtein distances between misspelled word “ကျောင်းသာ” and the candidates “ ကျောင်းသား ”, “ ကျောင်းစာ ”, “ ကျောင်းသူ ”are calculated by using the Levenshtein distance algorithm.

The distance calculations between misspelled word and candidates are shown in Table 4.1, 4.2, 4.3.

**Table 4.1: Distance Calculation between “ကျောင်းသာ” and “ကျောင်းသား”**

| | Candidate | ေ | က | ျ | ာ | င | ်ဒ် | း | သ | ာ | း |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Misspell | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| ေ | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| က | 2 | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| ျ | 3 | 2 | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| ာ | 4 | 3 | 2 | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| င | 5 | 4 | 3 | 2 | 1 | 0 | 1 | 2 | 3 | 4 | 5 |
| ်ဒ် | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 1 | 2 | 3 | 4 |
| း | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 1 | 2 | 3 |
| သ | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 1 | 2 |
| ာ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 2 | 0 | 1(min-d) |

If s is ကျောင်းသာ and t is ကျောင်းသား, then min-d(s,t) = 1, one insertion (insert “ း” after “ာ”) is needed to transform s into t.

s[1] =t[1] // "ေ" and "ေ" → substitution_cost = 0

Above =d[0,1]+1 = 2

Left =d[1,0]+1 = 2

Dia =d[0,0]+substitution_cost =0

d[1,1] =min(Above,Left,Dia) =0

s[2] ≠ t[2] // "ေ" and "က" → substitution_cost = 1

Above =d[0,2]+1 = 3

Left =d[1,1]+1 = 1

Dia =d[0,1]+substitution_cost =2

d[1,1] =min(Above,Left,Dia) =1

s[9] ≠ t[10] // "ာ" and "း" → substitution_cost = 1

Above =d[8,10]+1 = 3

Left =d[9,9]+1 = 1

Dia =d[8,9]+substitution_cost =2

d[1,1] =min(Above,Left,Dia) =1

So, the minimum distance between ကျောင်းသာ and ကျောင်းသား is 1.

**Table 4.2: Distance Calculation between "ကျောင်းသာ" and "ကျောင်းစာ"**

| | Candidate | ေ | က | ျ | ာ | င | ်ဒ် | း | စ | ာ |
|---|---|---|---|---|---|---|---|---|---|---|
| misspell | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| ေ | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | ၈ |
| က | 2 | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| ျ | 3 | 2 | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| ာ | 4 | 3 | 2 | 1 | 0 | 1 | 2 | 3 | 4 | 5 |
| င | 5 | 4 | 3 | 2 | 1 | 0 | 1 | 2 | 3 | 4 |
| ်ဒ် | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 1 | 2 | 3 |
| း | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 1 | 2 |
| သ | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 1 | 2 |
| ာ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 2 | 1(min-d) |

**Table 4.3: Distance Calculation between "ကျောင်းသာ" and "ကျောင်းသူ"**

| | Candidate | ေ | က | ျ | ာ | င | ်ိ | း | သ | ူ |
|---|---|---|---|---|---|---|---|---|---|---|
| Misspell | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| ေ | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| က | 2 | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| ျ | 3 | 2 | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| ာ | 4 | 3 | 2 | 1 | 0 | 1 | 2 | 3 | 4 | 5 |
| င | 5 | 4 | 3 | 2 | 1 | 0 | 1 | 2 | 3 | 4 |
| ်ိ | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 1 | 2 | 3 |
| း | 7 | 6 | 5 | 4 | 3 | 2 | 2 | 0 | 1 | 2 |
| သ | 8 | 7 | 6 | 5 | 4 | 3 | 3 | 1 | 0 | 1 |
| ာ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 1(min-d) |

### 4.2.4 Error Correction

Error correction is the final step of spell correction. It chooses the best candidate in candidates generating from candidates generation step and correct automatically by using N-gram model. N-gram model is now widely used in natural language processing such as speech recognition,machine translation and spell checking.

The input : ကျွန်တော်သည်ကျောင်းသာတစ်ယောက်ဖြစ်သည်

Initial Segment : ကျွန်တော် | သည် | ကျောင်း | သာ | တစ် | ယောက် | ဖြစ်သည် |

Segment : ကျွန်တော် | သည် | ကျောင်းသာ | တစ်ယောက် | ဖြစ်သည်

Candidates Generation : ကျောင်းသာ ['ကျောင်းသား','ကျောင်းစာ','ကျောင်းသူ']

Output : ကျွန်တော် သည် ကျောင်းသား တစ်ယောက် ဖြစ်သည်

### 4.2.5 Language Model for N-gram Model

Language models generate probabilities by using training data in one or many languages. Myanmar Text corpus is used as the training data. This corpus includes approximately 50,000 segmented sentences with space. These sentences are collected from the news pages and are written by Unicode System. The sentences are segmented into word level. The corpus which includes these sentences is used as training corpus.

<s>ကင်မရာ ပြောင်း အရှည် ကြီး တွေ နဲ့ ရိုက် ကြည့် တော့ တော်တော် ရှင်းရှင်း ကို
မြင် ရတယ် ။ </s>
<s>အဓိက ကတော့ သောက်ရေ လို တာပါ ။ </s>
<s>ပြီးခဲ့တဲ့ မတ်လ ၂၇ ရက်နေ့ မှာ ကျင်းပ တဲ့ လူ့ အခွင့်အရေးကောင်စီ အစည်းအဝေး မှာ ထောက်ခံမဲ ၂၀ မဲ ၊
ကန့်ကွက် မဲ ၁၁ မဲ နဲ့ အတည်ပြုခြင်း ဖြစ်ကြောင်း ကုလသမဂ္ဂ ရဲ့ ထုတ်ပြန်ချက် မှာ ဖော်ပြ ထား ပါတယ် ။ </s>

### 4.2.6 N-gram Model for Error Correction

N-gram model is a type of probabilistic language model for predicting the next item in a sequence. The item can be phonemes,syllables,letters or words according to the system. In this system, a word can be considered as an item. An n-gram of size 1 is referred to as a "unigram"; size 2 is a "bigram" ; size 3 is a "trigram". Bigram is used for this system.

Bigram probabilities are calculated by Laplace Smoothing method because it can avoid zero probability and increase a small positive number. Bigram probability calculation is shown in equation 4.1.

$$P(W_n|W_{n-1}) = \frac{Count(Wn-1, Wn)+1}{Count(Wn-1)+V} \quad (4.1)$$

### 4.2.7 Bigram Probability Calculation

Example corpus

<s> သူ သည် ကျောင်းသား တစ် ယောက် ဖြစ် သည် </s>

<s> မောင်မောင် သည် ကျောင်းသား တစ် ယောက် ဖြစ် သည်</s>

<s> မြမြ သည် ကျောင်းသူ တစ် ယောက် ဖြစ် သည် </s>

<s> ကျွန်တော် သည် ပန်းသီး တစ် လုံး ကုန် အောင် မ စား နိုင် ပါ </s>

P(ကျွန်တော်|<s>)=Count(<s> ကျွန်တော်)+1/Count(ကျွန်တော်)+V=0.061

P(သည်|ကျွန်တော်)=Count(ကျွန်တော်သည်)+1/Count(သည်)+V=0.051

P(ကျောင်းသား|သည်)=Count(သည်ကျွန်တော်)+1/Count(ကျောင်းသား)+V=0.061

P(တစ်|ကျောင်းသား)=Count(ကျောင်းသား)+1/Count(တစ်)+V=0.083

P(ယောက်|တစ်)=Count(တစ်ယောက်)+1/Count(ယောက်)+V=0.114

P(ဖြစ်|ယောက်)=Count(ယောက်ဖြစ်)+1/Count(ဖြစ်)+V=0.114

P(သည်|ဖြစ်)=Count(ဖြစ်သည်)+1/Count(သည်)+V=0.103

P(</s>|သည်)=Count(သည်</s>)+1/Count(</s>)+V=0.111

### 4.2.8 Sentence Probability Calculation

We can calculate the probability of a sentence by the probabilities of each component part. In the equation 4.2, the probability of the sentence can be calculated by using the probabilities of the sentence's individual bigrams.

$$P(W_1 \cdots W_n)=P(W_1)P(W_2|W_1)P(W_3|W_2)\cdots P(W_n|W_{n-1}) \qquad (4.2)$$

P('<s>ကျွန်တော်သည်ကျောင်းသားတစ်ယောက်ဖြစ်သည်</s>')=P('<s>')*P('ကျွန်တော်'|'<s>')*
P('သည်'|'ကျောင်းသား')*P(|'ကျောင်းသား'|'သည်')*P('တစ်'|' ကျောင်းသား')*('ယောက်'|'တစ်')
* P('ဖြစ်'|'ယောက်') * P('သည်'|'ဖြစ်') *P('</s>'|'သည်)'
=0.0833 * 0.061 * 0.051 * 0.088 * 0.083 * 0.114 * 0.114 * 0.103 * 0.111
$=3.376*10^{-9}$

P('<s>ကျွန်တော်သည်ကျောင်းစာတစ်ယောက်ဖြစ်သည် </s>') = P('<s>')*P('ကျွန်တော်'|'<s>')
*P('သည်'|'ကျောင်းသား')*P(|'ကျောင်းစာ'|'သည်')*P('တစ်'|'ကျောင်းစာ')*('ယောက်'|'တစ်') *
P('ဖြစ်'|'ယောက်') * P('သည်'|'ဖြစ်') *P('</s>'|'သည်)'

=0.0833 * 0.061 * 0.051 * 0.031 * 0.028 * 0.114 * 0.114 * 0.103 * 0.111

$=4.012*10^{-10}$

P('<s>ကျွန်တော်သည်ကျောင်းသူတစ်ယောက်ဖြစ်သည်</s>'= P('<s>') * P(‘ ကျွန်တော်'|’<s>’) *P('သည်'|'ကျောင်းသား')*P(|'ကျောင်းသူ'|'သည်')*P('တစ်'|'ကျောင်းသူ’)*('ယောက်'|'တစ်') * P('ဖြစ်'|'ယောက်') * P('သည်'|'ဖြစ်') * P('</s>'|'သည်)'

=0.0833 * 0.061 * 0.051 * 0.061* 0.056 * 0.114 * 0.114 * 0.103 * 0.111

$=1.579*10^{-9}$

For the sentence probabilities,the first sentence gets the highest probability and so the system chooses ကျောင်းသား as the best candidates and correct the misspelled word “ကျောင်းသာ" into the correct word “ကျောင်းသား" automatically.

## 4.3 System Architecture

This system implements Myanmar Spelling Correction based on N-gram Model. When the user uses this system, the user can see the home page as shown in figure 4.3.

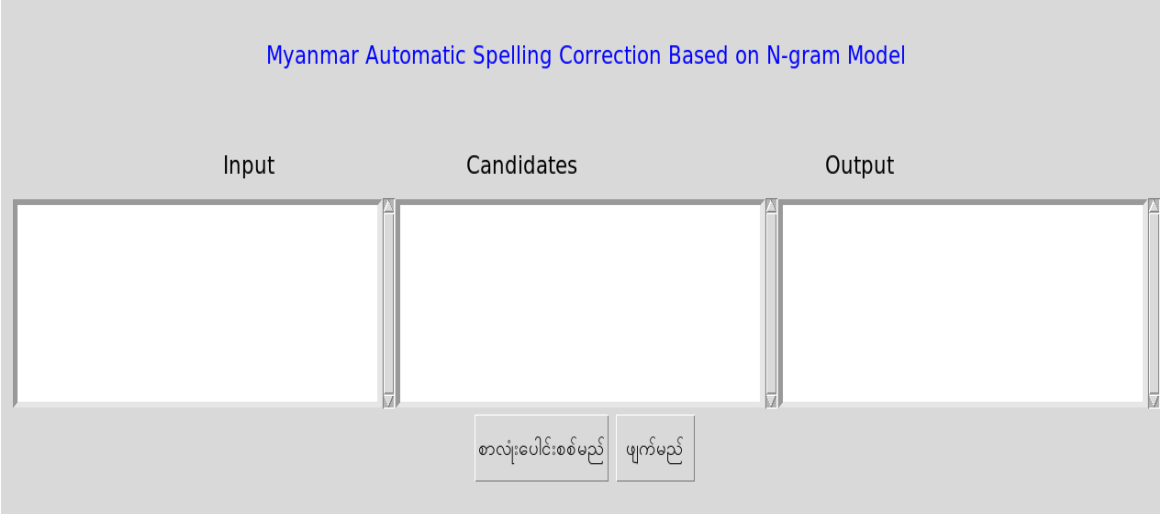


**Figure 4.3: The Main Page of Myanmar Spelling Correction System**

Figure 4.3 shows the main page of the system, which consists of two buttons. From the first button “စာလုံးပေါင်းစစ်မည်” , the user can check the Myanmar Spellings by typing Myanmar sentences or Myanmar words. When the user types the Myanmar sentence or word in the text box “Input” and click the first button “စာလုံးပေါင်းစစ်မည်”, the system will show the correct sentence or word in the text box “Output”. Moreover,

the system shows the words suggestion for misspelled words in the text box “Candidates”. Then, when the user want to check the another sentence, he or she can click the second button “ဖျက်မည်” to clear the text in all the text boxes.

### 4.3.1 Correction of Misspelled Myanmar Word

When the user wants to check the Myanmar word , he or she must type the Myanmar word in the text box “Input” and click the first button “စာလုံးပေါင်းစစ်မည်”. And then, the system will show the correct word in the text box “Output” and generated candidates in the text box “Candidates” as shown in figure 4.4. When the user type the correct word, the system will also show the correct word in the text box “Output” and show nothing in the text box “Candidates” as shown in figure 4.5.

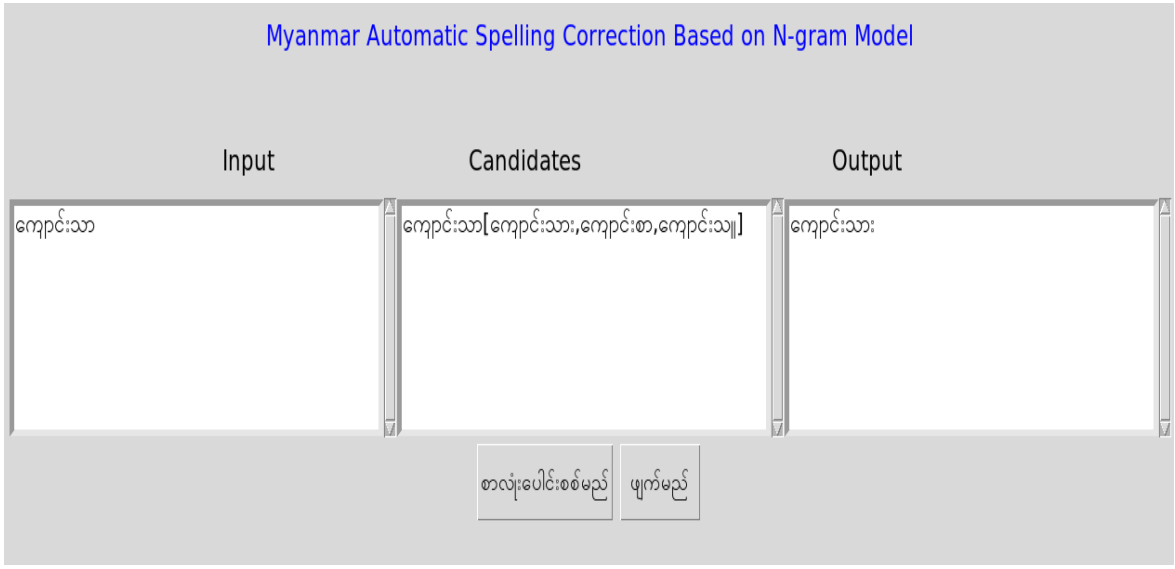


**Figure 4.4: Correction of Misspelled Myanmar Word**

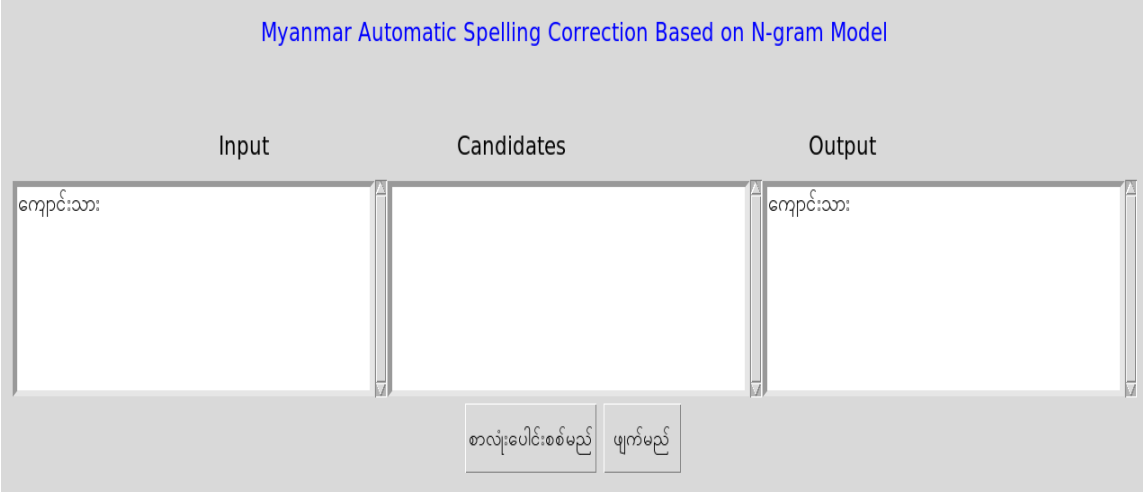


**Figure 4.5: Correction of Correct Myanmar Word**

### 4.3.2 Correction of Misspelled Myanmar Sentence

When the user want to check the Myanmar sentence and type the sentence in the text box "Input", the system will check the sentence and correct automatically. Firstly, the system segments the sentence initially and then merges all the possible words as word level. Then, the segmented words is detected to find the misspelled errors by using Myanmar Dictionary. After finding the misspelled words, the system will generate the candidates for all misspelled words by using Levenshtein Distance Algorithm. Finally, the system calculates the sentence probabilities for each generated candidates by using N-gram Model. Then, the system chooses the best candidate that gets the highest probability and correct the sentence automatically.

The sentence with one misspelled word is corrected in figure 4.6 and the sentence with more than one misspelled words is corrected in figure 4.7 respectively.

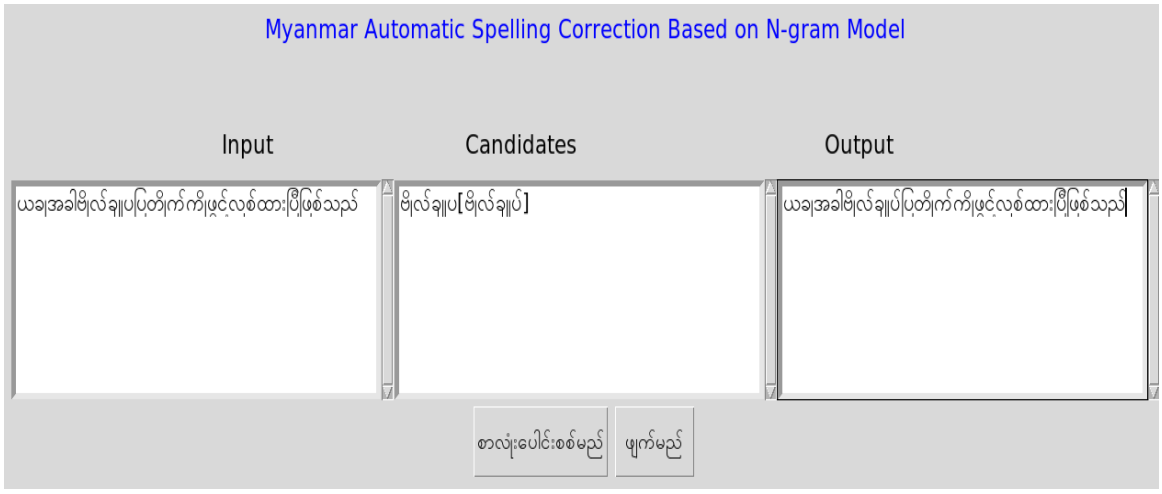


**Figure 4.6: Correction of Myanmar Sentence with One Misspelled Word**

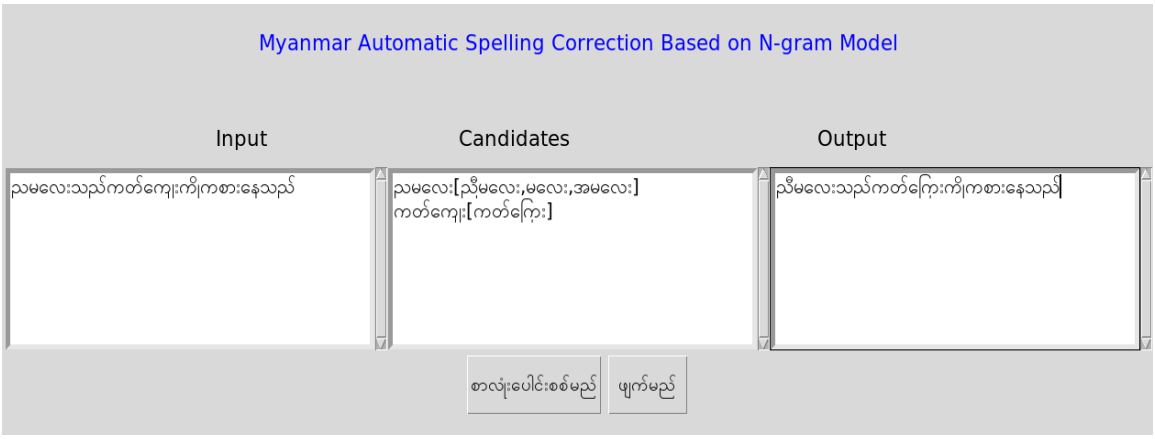


**Figure 4.7: Correction of Myanmar Sentence More than One Misspelled Word**

## 4.4 Experiments

The Proposed system checks and correct the sentence which include misspelled words automatically. The performance of the system is calculated by using accuracy, precision, recall, F1-score. The accuracy of typographic errors and phonetic errors is better than that of context errors in the proposed system.

### 4.4.1Experiment Data Set

In order to verify the validity of the system,some experiments are based on the monolingual Myanmar corpus. There are two types of sentences for experimental data. They are open sentences and closed sentences. The open sentences are the ones that do not include in the training corpus. The closed sentences are the ones that include in the training corpus.

1,000 sentences are tested to evaluate the experimental results of the system. The average number of words including in one sentence is 9 words. Experiment data set is shown in Table 4.4.

**Table 4.4: Experiment data set**

| Types of errors | Open sentences | Close sentences | Total |
|---|---|---|---|
| All | 350 | 650 | 1000 |
| Typographic | 100 | 150 | 250 |
| Phonetic | 200 | 400 | 600 |
| Context | 50 | 100 | 150 |

### 4.4.1Experimental Results

Spell correction performance is evaluated in terms of accuracy,precision, recall and F1-score which are the common way to measure spelling checking system's performance .Accuracy is simply the ratio of correctly predicted observation to the total observations .Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. Recall is the ratio of correctly predicted positive observations to the all observations in the system. F1-score is the mean of precision and recall. Experimental results are shown in Table 4.5, 4.6 and 4.7.

**Table 4.5: Spelling Error of Generation Results of Open Sentences**

| Types of errors | Accuracy | Precision | Recall | F1-score |
|---|---|---|---|---|
| All | 93.45% | 98.03% | 89.78% | 94.03% |
| Typographic | 91.38% | 97.39% | 84.89% | 90.48% |
| Phonetic | 94.26% | 97.82% | 91.71% | 94.61% |
| Context | 84.31% | 90.45% | 81.91% | 85.74% |

**Table 4.6: Spelling Error of Generation Results of Close Sentences**

| Types of errors | Accuracy | Precision | Recall | F1-score |
|---|---|---|---|---|
| All | 94.94% | 98.38% | 90.79% | 94.43% |
| Typographic | 92.07% | 98.57% | 85.19% | 91.39% |
| Phonetic | 95.73% | 98.51% | 92.74% | 95.54% |
| Context | 85.42% | 91.67% | 81.48% | 86.28% |

**Table 4.7: Spelling Error of Generation Results of all Types of Sentences**

| Types of errors | Accuracy | Precision | Recall | F1-score |
|---|---|---|---|---|
| All | 94.67% | 98.97% | 91.27% | 95.61% |
| Typographic | 92.93% | 98.91% | 86.31% | 92.96% |
| Phonetic | 96.23% | 98.93% | 93.38% | 96.02% |
| Context | 86.39% | 92.27% | 82.54% | 86.91% |

# CHAPTER 5
# CONCLUSION

This chapter presents the benefits of the system, the conclusion, limitation and further extension of the proposed system.

The spelling correction for Myanmar language was proposed. This spelling correction can check and correct Typographic errors, Phonetic errors, and Context errors and the performance of the system is measured in terms of accuracy, precision, recall, F1-score. A monolingual Myanmar Text Corpus is built for Myanmar Spelling Correction system. Dictionary Look Up Method is used for error detection, Levenshtein Distance Algorithm is utilized for generating candidates and N-gram Model is applied for error correction.

## 5.1 Benefits of the System

The system can check and correct every sentence that has any number of errors and any types of errors. Moreover, Pali and Patsint words can be handled by this spelling correction. One of the benefits of the system is that the system can check and correct the Myanmar spellings automatically. Myanmar spelling correction gave satisfiable results to apply in Myanmar NLP applications.

## 5.2 Limitations and Further Extensions

For the proposed system, the average accuracy of all errors is 92.07% typographic errors, 95.73% phonetic errors, 85.42% context errors. Therefore, the performance of context errors is not better than that of typographic errors and phonetic errors. Moreover, if the system cannot correct some misspelled words, the system will just provide these misspelled words. This is because there are only approximately 50,000 training sentences in the corpus. If there are more than 50,000 training sentences in the corpus, the spelling correction can provide a better performance.

# AUTHOR'S PUBLICATIONS

[1] Thazin Win, Win Pa Pa, "Myanmar Automatic Spelling Correction Based on N-gram Model", Nineteenth International Conference On Computer Applications (ICCA 2021), University of Computer Studies Yangon, Myanmar, 2021.

# REFERENCES


[1] AnshMehta, Vishal Salgond, Darshan Satra, Nikhil Sharma, Spell Correction and Suggestion Using Levenshtein Distance, Department of Information Technology, K.J.Somaiya College of Engineering, Mumbai, 2021

[2] Asif Mohaimen, Charkraborty, Bangla OCR Post Processing Word Based Longest Common Subsequence Techique, Department of Computer Science and Engineering, 2017

[3] Aye Myat Mon, International Journal of Science and Research (IJSR), 2013

[4] Hicham Gueddah, Abdellah Yousfi, Mostafa Belkasm, The filtered combination of the weighted edit distance and Jaro-Winkler distance to improve spell checking Arabic texts, University Mohammed V of Rabat, Morocco,2015

[5] Jui Feng Yeh, Chinese Spelling Check based on N-gram Model and String Matching Algorithm, Department of Computer Science and Information Engineering, National Chia-YI University, 2017

[6] Myanmar Words Commonly Misspelled and Misused Book, Department of Myanmar Language Commission, Ministry of education, Union of President Myanmar July, 2003

[7] Nur Hamidah, Novi Yusliani , Desty Rodiah, Spelling Checker using Algorithm Damerau Levenshtein Distance and Cosine Similarity, 2020

[8] Nwe Zin Oo, Myanmar Words Spelling Checking Using Levenshtein Distance Algorithm, University of Computer Studies, Yangon, 2010

[9] V.J. Hodge, J.Austin, A comparison of standard spell checking algorithms and a novel binary neural approach, Department of Computer Science, University of York , UK, 2003

[10] Weijian Xie,Peijie Huang, Chinese Spelling CheckSystem Based on N-gram Model,College of Mathematics and Informatics, South China Agricultural University,2015

[11] Win Pa Pa, Words Segmentation for Myanmar, University of Computer Studies, Yangon, 2008

[12] Youssef Bassil and Mohammad Alwani, Context -sensitive Spelling Correction Using Google Web 1T 5-Gram Information