SPELL CHECKER FOR GURMUKHI SCRIPT

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering

in

Software Engineering

By:

Rupinderdeep Kaur

(800831023)

Under the supervision of:

Mr. Parteek Bhatia

Assistant Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

PATIALA – 147004

June 2010

Certificate

I hereby certify that the work which is being presented in the thesis entitled, **"Spell Checker for Gurmukhi Script"**, in partial fulfilment of the requirements for the award of degree of Master of Engineering in Software Engineering submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Mr. Parteek Bhatia* and refers other researcher's works which are duly listed in the reference section.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

Rupinderdeep K.
(Rupinderdeep Kaur)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Mr. Parteek Bhatia)
Assistant Professor
Computer Science and Engineering Department,
Thapar University, Patiala

Countersigned by

(RAJESH BHATIA) 17/06/10
Head
Computer Science and Engineering Department,
Thapar University,

(R.K.SHARMA) 1/7/10
Dean (Academic Affairs)
Thapar University,
Patiala.

Acknowledgement

I wish to express my sincere gratitude to Mr. Parteek Bhatia, Assistant Professor, Computer Science and Engineering Department, Thapar University, Patiala, for providing invaluable guidance, suggestions and sympathetic attitude, which inspired me to submit this thesis report on time.

I am also thankful to Dr. Rajesh Bhatia, Head, Computer Science and Engineering Department, for his kind help and cooperation.

I would also like to thank all the staff members of Computer Science and Engineering Department for providing me all the facilities required for the completion of this work. I also want to thank my classmates and friends for helping me in the hour of need and providing me all the help and support for completion of my thesis.

I am deeply indebted to my parents, sisters and brother for the inspiration and ever encouraging moral support, which enabled me to pursue my studies.

Rupinderdeep Kaur

Abstract

A Spell Checker is a basic necessity for composing text in any language. Though considerable work has been done in the area for English and related languages, the Indian language scenario present a relatively more complex and uphill task. Punjabi is the world's 12th most widely spoken language. There is a very little amount of work done in this field. The only available Spell Checker for Punjabi language is 'AKHAR'.

First and the far most requirement for any Spell Checker is to have dictionary of different possible words of that language which will act as a corpus. A Spell Checker can be Stand-alone Application capable of operating on a block of text, or as part of a larger application, such as a word processor, email client, electronic dictionary, or search engine. When some Punjabi text is given as an input to Spell Checker, it list outs the incorrect words separately by checking their availability in the dictionary. Finally it provides the suggestions for the incorrect words from the dictionary.

In this thesis, I have discussed various types of errors, techniques for detecting and correcting errors and the various Spell Checkers available in different Indian languages. A brief overview of origin and symbols of Gurumukhi Script are explained. In this thesis, I have developed a Spell Checker for Gurmukhi Script. The design, techniques and implementation of the newly developed Spell Checker for Gurmukhi Script are explained. Creation of dictionary, error detection and error correction and replacement are the main features of this system. The system detects approximately 87% of the errors and provides 90% of the correct suggestions.

Contents

List of Figures

List of Tables

Chapter 1: Introduction

1.1 What is Spell Checker?

In computing, a Spell Checker (or Spell Check) is an application program that flags words in a document that may not be spelled correctly. Spell Checker may be stand-alone capable of operating on a block of text, or as part of a larger application, such as a word processor, email client, electronic dictionary, or search engine [1].

1.2 Types of Errors

Spelling and typing errors are abundant in human generated electronic text. The problem of detecting error in words and automatically correcting them is a great research challenge. The word error can belong to one of the two distinct categories, namely, non-word error and real-word error. A word which is not a valid word is a nonword error. A valid but not the intended word in the sentence is a real word error. The knowledge about error pattern is necessary to model an efficient Spell Checker. The newly developed Spell Checker only deals with nonword errors.

1.3 Issues related with Spell Checker

There are several issues to be addressed in the error correction problem. The first issue concerns the error patterns generated by different text generating media such as typewriter and computer keyboard, typesetting and machine printing, OCR system, speech recognizer output, and of course, handwriting. Usually, the error pattern of one media does not match with that of the other. The error pattern issue of each media concerns the relative abundance of insertion, deletion, substitution and transposition error, run-on and split word error, single versus multiple character error, word length effect, positional bias, character shape effect, phonetic similarity effect, heuristic tendencies *etc*. The knowledge about error pattern is necessary to model an efficient Spell Checker.

Another important issue is the computerized dictionary which concerns the size of the dictionary, the problem of inflection and creative morphology, the dictionary file structure, dictionary partitioning, word access techniques and so on.

1.4 Origin and Symbols of Gurmukhi Script

The word 'Gurmukhi' literally means from the mouth of the Guru. Gurmukhi Script is used primarily for the Punjabi language, which is world's 12th most widely spoken language. Punjabi Language is used in both parts of Punjab in India and Pakistan. In East Punjab (India) Punjabi is written in Gurmukhi Script. This Script was created by Guru Angad Dev Ji. This is written from left to right. Punjabi is the mother tongue of more than 110 million people of Pakistan (66 million), India (44 million) and many millions in America, Canada and Europe. It has been written in two mutually incomprehensible scripts Shahmukhi and Gurmukhi for centuries.

Gurmukhi script is syllabic in nature. Gurmukhi script-consists of 41 consonants called *vianjans*, 9 vowel symbols called *laga* or *matras*, 2 symbols for nasal sounds (. , °), one symbol for reduplication of sound of any consonant (`)and three half characters [2]. Table 1.1 and 1.2 shows the description of Punjabi consonants, vowels, semi vowels and half characters.

The consonants of first row are classified as open syllabics and called vowel consonants or semi consonants or "Matra Vahak" due to their inherent property that they are never used in work without any 'Laga' or 'Vowel'. The next two consonants are classified as root class consonants. The rest of the consonants except to the last two groups namely the - "Antim" and "Naveen" group, are categorized according to their phonetic structure. The last but one group consisting of 5 independent consonants is called the "Antim" group and the last group is the "Naveen" group which has been introduced to accommodate the words of Persian, Arabic and Sanskrit.

Table 1.1: Consonants in Punjabi language

Consonents	ੳ A	ਅ A	ੲ e				**Matra Vahak**
				ਸ S	ਹ h		**Mul Varag**
	ਕ k	ਖ K	ਗ g	ਘ G	ਙ |		**Kavarg Toli**
	ਚ c	ਛ C	ਜ j	ਝ J	ਞ \		**Chavarg Toli**
	ਟ t	ਠ T	ਡ f	ਢ F	ਣ x		**Tavarg Toli**
	ਤ q	ਥ Q	ਦ d	ਧ D	ਨ n		**Tavarg Toli**
	ਪ p	ਫ P	ਬ b	ਭ B	ਮ m		**Pavarg Toli**
	ਯ X	ਰ R	ਲ l	ਵ V	ੜ V		**Antim Toli**
	ਸ਼ S	ਖ਼ (	ਗ਼ z	ਜ਼ Z	ਫ਼)	ਲ਼ L	**Naveen Toli**

Table 1.2: Vowels, Semi Vowels and Half Characters in Punjabi language

Vowels	◌ਾ **w**	◌ਿ **i**	◌ੀ **I**	◌ੁ **U**	◌ੂ **U**	◌ੇ **y**	◌ੈ **Y**	◌ੋ **o**	◌ੌ **O**
Semi vowels	◌ਂ **N**	◌ੰ **M**	◌ਁ **’**	◌ੱ **ˋ**					
Half Characters	◌੍ਹ **H**	◌੍ਰ **R**	੬ **6**						

It can be noted that most of the characters have a horizontal line at the upper part. The characters of word are connected mostly by this line called head line. A word in Gurmukhi Script can be partitioned into three horizontal zones. The upper zone denotes the region above the head line, where the vowels reside, while the middle zone represents the area below the head line where the consonants and some sub-parts of vowels are present. The middle zone is the busiest zone. The lower zone represents the area below middle zone where some vowels and certain half characters lie in the foot of consonants. For example, the division of Punjabi word ਸਕੂਲਾਂ is shown in Table 1.3.

Table 1.3: Division of Punjabi Word

◌ਂ	Upper zone
ਸ ਕ ਲ◌ਾ	Middle zone
◌ੂ	Lower zone

Chapter 2: Literature Survey

There are two main issues related with the Spell Checker. These are Error Detection and Error Correction. Further depending upon the types of errors *i.e.* non-word errors and real word errors, there are many techniques available for detection and correction of errors. The survey of these techniques and their use in different Spell Checkers is described in this chapter.

2.1 Non-word Error Detection

A string of characters separated by spaces or punctuation marks may be called a candidate word. A candidate word is a valid word if it has a meaning; else it is a nonword. The two main techniques that have been explored for nonword errors detection are n-gram analysis and dictionary lookup. N-grams are n-letter sub sequences of words or strings where n usually is one, two or three .One letter n-grams are referred to as unigrams or monograms; two letter n-grams are referred to as bi-grams and three letter n-grams as trigrams.

In general n-gram detection technique work by examining each n-gram in an input string and looking it up in a precompiled table of n-gram statistics to ascertain either its existence or its frequency of words or strings that are found to contain nonexistent or highly infrequent n-grams are identified as either misspellings. N-gram techniques usually require either dictionary look up technique or a large corpus of text in order to pre-compile an n-gram table. Dictionary looks up techniques work by simply checking to see if an input string appears in a dictionary, *i.e.*, a list of acceptable words.

2.1.1 N-gram Analysis Technique [3]

N-gram tables can take on a variety of forms. The simplest is a binary bi-gram array which is a two dimensional array of size 41x41 whose elements represent all possible two letter combinations of the alphabet. The value of each element in the array is set to either 0 or 1 depending on whether that bi-gram occurs in at least the word in a predefined lexicon or dictionary. A binary tri-gram array would have three dimensions. Both of the above arrays

are referred to as non-positional binary n-gram arrays because they do not indicate the position of the n-gram within a word.

More of the structures of the lexicon can be captured by a set of positional binary n-gram array, for example, in a positional binary tri-gram array the *i, j,* k^{th} element would have the value 1 only if there exists at least one word in the lexicon with the letters *l, m* and *n* in positions *i, j,* and *k.* The trade-off for representing more of the structure of the lexicon is the increase in storage space required for the complete set of positional arrays. Any word can be checked for errors by simply looking up its corresponding entries in binary n-gram arrays to make sure they are all 1's.

2.1.2 Dictionary Lookup Technique

The most popular method of detecting errors in a text is simply to look up every word in a dictionary; any words that are not there are taken to be errors. Dictionary lookup is a straightforward task. However response time becomes a problem when dictionary size exceeds a few hundred words. In document processing and information retrieval, the number of dictionary entries can range from 25000 to more than 250,000 words.

The most common technique for gaining fast access to a dictionary is the use of a Hash Table. To look up an input string, one simply computes its hash address and retrieves the word stored at that address in the pre-constructed hash table. If the word stored at the hash address is different from the input string or is null, a misspelling is indicated. The main advantage is that the random access nature of a hash code eliminates the large number of comparisons needed for sequential or even tree based searches of the dictionary. The main disadvantage is the need to device a clever hash function that avoids collisions without requiring a huge hash table.

2.2 Techniques for Non-word Error Correction

Error pattern analysis of each language helps in making an efficient spellchecker. It includes analysis of various types of errors (insertion, deletion, transposition, substitution, run-on, split word error) positional analysis, word length effects, phonetic errors, first position error

analysis, keyboard effects *etc.* In Punjabi there are many ways of writing the same word and all the ways could be correct.

So it might be wrong to collect the raw typed text as the data for analysis. Because analysis of that raw text does not surely direct us to the typing mistake but can mislead us to the spelling mistake of that word. Our main interest is to analyse the typing mistakes instead of spelling mistakes since, the study will be used to design a suggestion list for a Punjabi Spellchecker.

2.2.1 Commonly Encountered Punjabi Errors [2]

There are large number of errors encountered in human generated Punjabi text and these errors mostly belong to one of the following error types.

- **Substitution Error (SE):** When at least one character is substituted by the other character. The maximum of misspellings in Punjabi contain substitution errors.
- **Deletion Error (DE):** When at least one character is deleted in the desired word.
- **Insertion Error (IE):** When at least one extra character is inserted in the desired word.
- **Transposition Error (TE):** When two adjacent characters are transposed.
- **Run-on Error (ROE):** When there is space missing between two or more valid words**.**
- **Split Word Error (SWE):** This is opposite of Run-on error when there is some extra space is inserted between parts of a word. The error can be removed by removing the extra space.

2.2.1.1 Substitution Errors

This error occurs when at least one character is substituted by the other character. For example, ਅਸਦਾ-> ਵੁਸਦਾ, ਸ਼ਾਲ ->ਸਾਲ, ਸੁਣਦਾ-> ਬੁਣਦਾ *etc.*

In the above three words, ਅ->ਵ, ਸ਼-> ਸ, ਸ ->ਬ are the various substitution character pairs respectively.

The common reasons for substitution errors are:

- Naveen Group elements: It is seen that 9.91% of the substitution errors are due to the naveen group elements. For example, ਜ਼ੇਲ ->ਜੇਲ, ਸ਼ਿਹਦ ->ਸਿਹਦ

- Due to assignment to same keys (shifted and unshifted modes) on the keyboard, for example, ਨ-> ਲ, ਅ ->ਵ, ਣ-> ਦ
- Words that are usually used in various forms, for example, ਜੇਹਾ ->ਜਿਹਾ, ਕ੍ਰਮ ->ਕਰਮ
- Vowels having similar sounds, for example, ੇ ->ੈ, ੋ-> ੌ, ੁ-> ੂ, ਿ-> ੀ
- Due to substitution of half characters, for example, ਰ ->੍ਰ, ਹ ->੍ਹ

2.2.1.2 Deletion Error

When at least one character is deleted in the desired word. For example, ਗੱਲ ->ਗਲ, ਚੱਲ-> ਚਲ. These errors also give rise to real word errors, for example, ਪੁੱਲ ->ਪੱਲ, ਪਾਣੀ ->ਪਾਣ

In the above example ਪੱਲ, ਪਾਣ are two valid words but they are not the desired word. It is observed that deletion related errors contribute significantly after substitution errors. It is seen that the characters ੱ, ੰ are most commonly missing characters. The percentage of missing ੱ is 20.51% and the percentage of missing ੰ is 17.47% and these two characters alone contribute to 38% of deletion errors [2].

2.2.1.3 Insertion Error

When at least one extra character is inserted in the desired word. For example, ਗ਼ਹਰ ->ਗ਼ਿਹਰ here ਿ is the extra inserted character. These errors also give rise to real word errors, for example, ਯੋਗਤਾ ->ਯੋਗਿਤਾ, ਸਾਰਾ-> ਸਾਰ

In the above example ਯੋਗਿਤਾ, ਸਾਰ are two valid words but they are not the desired words. In the multiple form words confusion regarding insertion errors are due to:

- The use of ੱ, for example, ਸਿਖਿਆ-> ਸਿੱਖਿਆ words on both side are delivering the same meaning.

- The use of ◌ं, for example, ਸਾਹਿਤ ->ਸਾਹਿੰਤ words on both side are delivering the same meaning.

It is seen that the characters ◌ੱ, ◌ं are mostly extra inserted characters. The percentage of insertion ◌ं is 17.53 % and the percentage of ◌ੱ is 12.52% and these two characters contribute around 30% of Insertion errors [2].

2.2.1.4 Transposition Error

This type of error occurs when two adjacent characters of the word are typed in swapped manner. For example, ਸਵੇਰ ->ਸਵਰੇ, ਰਾਤ-> ਰਤਾ .In the above two words ◌ੇ-> ਰ, ◌ਾ-> ਤ are transposed character pairs.

It is found that these transpositions (like substitution) also give rise to real word errors. For example, ਕਰਮ-> ਕਮਰ, ਸੂਰਤ-> ਸੂਤਰ where ਕਮਰ, ਸੂਤਰ are two valid words. The percentage of transposition errors is 1.85% and 1.43% in single and multi-error misspellings respectively. No prominent transposition character pairs were found [2].

2.2.1.5 Run-on Errors [4]

This type of error occurs when two or more valid words are mistakenly written side by side without a space in between. For example, ਜਿਸ ਦਾ-> ਜਿਸਦਾ, ਦਾਦੀ ਮਾਂ-> ਦਾਦੀਮਾਂ

In the above two word substitutions ਜਿਸ, ਦਾ, ਦਾਦੀ, ਮਾਂ are four different words. Sometimes these errors give rise to real word errors. For example, ਉਸ ਦੇ ਉਸਦੇ, ਜਿਸ ਦੇ ਜਿਸਦੇ. Words ਉਸਦੇ, ਜਿਸਦੇ are two valid words. The percentage of run on error is found to be 5.20% in single and 0.61% in multi-error misspellings.

2.2.1.6 Split Word Errors [4]

This is opposite of Run-on error when there is some extra space is inserted between parts of a word. The error can be removed by removing the extra space. For example, ਸੁਖੀ ->ਸੁ ਖੀ, ਦੀਵਾਰ-> ਦੀ ਵਾਰ *etc.* Sometimes these errors give rise to more than one real word errors, for example, ਉਸਦੇ -> ਉਸ ਦੇ, ਜਿਸਦੇ ->ਜਿਸ ਦੇ Words ਉਸ, ਦੇ, ਜਿਸ, ਦੇ, are four valid words. The percentage of split word error is found to be 2.32% in single and 0.20% in multi-error misspellings.

2.2.2 Phonetically Similar Character Errors

Phonetic errors are a special class of cognitive errors in which the writer substitutes a phonetically correct but orthographically incorrect sequence of letters for the intended word. We have classified the phonetic errors into four categories:

1) Type 1 ਗ ->ਘ, ਜ ->ਝ, ਦ ->ਧ, ਡ-> ਢ, ਨ-> ਣ, ਬ-> ਭ

2) Type 2 ਸ਼ ->ਸ, ਖ਼ ->ਖ, ਜ਼ ->ਜ, ਫ਼ ->ਫ, ਲ਼-> ਲ, ਗ਼-> ਗ

3) Type 3 ੈ ->ੇ, ੁ ->ੂ, ੋ-> ੇ, ਿ ->ੀ, ਂ ->ਁ

4) Type 4 ਰ ->ਰ੍, ਹ ->ਹ੍,

It is analysed that 17% of the errors are due to phonetically similar substitution pairs in above four categories .Out of the total number of phonetic errors 59.28% are due to the Type 2 group elements. Percentage of phonetically similar sounding vowel pairs are also considerable. It is concluded that about 23.83% of the misspellings contains mistakes due to Type 3 vowel pairs and 8.09% of the misspellings are due to Type 4 phonetically similar pairs [2].

Distinctions are made between three different types of nonword misspelling:

Typographic Errors: In the case of typographic errors it is assumed that the writer or typist knows the correct spelling but simply makes a motor coordination slip. For example, the- teh, spell-speel.

Cognitive Errors: The source of cognitive errors is presumed to be a misconception or lack of knowledge on the part of the writer or typist. For example, receive-recieve, conspiracy-conspiricy.

Phonetic Errors: Phonetic errors are a special class of cognitive errors in which the writer substitutes a phonetically correct but orthographically incorrect sequence of letters for the intended word, for example, abyss-abiss, naturally-nacherly.

2.2.3 Minimum Edit Distance Technique

The term minimum edit distance was defined by Wagner [5] as the minimum number of editing operations (*i.e.* insertions, deletions and substitutions) required to transform one string into another. The first minimum edit distance spelling error correction algorithm was implemented by Damerau[6].

Although many minimum edit distance algorithms compute integer distance scores, some algorithms obtain finer-grained scores by assigning non integer values to specific transformations based on estimates of phonetic similarities or keyboard adjacencies.

2.2.4 The Soundex System

The notion behind similarity key techniques is to map every string into a key such that similarly spelled strings will have identical or similar keys. Thus, when key is computed for a misspelled string it will provide a pointer to all similarly spelled words in the lexicon. Similarity key techniques have speed advantage because it is not necessary to directly compare the misspelled string to every word in the dictionary. A very early, often cited similarity key technique, the SOUNDEX system, was patented by Odell and Russel for use in phonetic spelling correction applications.

This problem has been around for a long time in the context of retrieving names from a list of names. Suppose you are working at an enquiry desk of a large organization, with a terminal connecting your office to the central computer. A customer comes in with a query about her account. She says her name is *Zbygniewski.* You don't want to ask her to spell it - perhaps her English is poor and other customers are waiting. To make matters worse, the name may be

misspelt in the computer file. You want to be able to key in something that sounds like what she just said and have the system find a name that resembles it.

The Soundex system was deviced to help with this problem [7,8]. It dates, in fact, from the days of card indexes - the name stands for 'Indexing on sound' - but has been transferred to computer systems. A Soundex code is created for every name in the file. The idea of the code is to preserve, in a rough-and-ready way, the salient features of the pronunciation. It is fairly obvious how this system can be applied to spelling correction. Every word in the dictionary is given a Soundex code. A Soundex code is computed from the misspelling, and those words that have the same code are retrieved from the dictionary.
Take as an example the misspelling *disapont.* A corrector would compute the code D215 from *disapont* and then retrieve all the words with code D215: *disband, disbands, isbanded, disbanding, disbandment, disbandments, dispense, dispenses, dispensed dispensing, dispenser, dispensers, dispensary, dispensaries, dispensable, dispensation, dispensations, deceiving, deceivingly, despondent, despondency, despondently, disobeying, disappoint, disappoints, disappointed, disappointing, disappointedly, disappointingly, disappointment, disappointments, disavowing.*

2.2.5 Rule Based Techniques

Rule Based Techniques are algorithms or heuristic programs that attempt to represent knowledge of common spelling error patterns in the form of rules for transforming missellings into valid words. The candidate generation process consists of applying all applicable rules to a misspelled string and retaining every valid dictionary word those results. Ranking is frequently done by assigning a numerical score to each candidate based on a predefined estimate of the probability of having made the particular error that the invoked rule corrected.

2.2.6 N-gram Based Technique

Letter n-grams, including tri-grams, bi-grams and uni-grams have been used in a variety of ways in text recognition and spelling correction techniques. They have been used by OCR correctors to capture the lexical syntax of a dictionary and to suggest legal corrections.

Riseman and Hanson[3] provide a clear explanation of the traditional use of n-grams in OCR correction. After partitioning a dictionary into sub-dictionaries by word length, they construct positional binary n-gram matrices for each sub-dictionary. They note that because these matrices provide answers to the question "Is there some word in the dictionary that has letters a and b in positions i and j respectively?" the matrices capture the syntax of the dictionary. An OCR output string can be checked for errors by simply checking that all its n-grams have value 1. If a string has a single error, it will have a 0 value in at least one binary n-gram; if more than one 0 value is found, the position of the error is indicated by a matrix index that is common to the 0 value n-grams. If the intersection results in only one n-gram with value 1, the error can be corrected, if more than one potential n-gram correction is found the word is rejected as ambiguous.

2.2.7 Probabilistic Technique

N-gram based technique led naturally into the probabilistic technique in both the text recognition and spelling correction paradigms. Two types of probabilities have been exploited:

- **Transition Probabilities** represent probabilities that a given letter will be followed by another given letter. These are language dependent. They are sometimes referred to as *markov probabilities* based on the assumption that language is a Markov Source. They can be estimated by collecting n-gram frequency statistics on a large corpus of text from the domain or discourse.
- **Confusion Probabilities** are estimates of how often a given letter is mistaken or substituted for another given letter. Confusion probabilities are source dependent. Because different OCR devices use different techniques and features to recognize characters, each device will have a unique confusion probability distribution. For that reason confusion probabilities are sometimes referred to as *channel characteristics*.

2.2.8 Neural Net Techniques

Neural nets are likely candidates for spelling correctors because of their inherent ability to do associative recall based on incomplete or noisy input. Furthermore, because they can be trained on actual spelling errors, they have the potential to adapt to the specific error patterns

of their user community, thus maximizing their correction accuracy for that population. Back propagation algorithm is the most widely used algorithm for training a neural net.

A typical back propagation net consists of three layers of nodes: input layer, an intermediate layer, usually referred to as hidden layer and an output layer. Each node in the input layer is connected by a weighted link to every node in the hidden layer. Similarly each node in the hidden layer is denoted by a weighted link to every node in the output layer. Input and output information is represented by on-off patterns of activity on the input and output nodes of the net. A 1 indicates that a node is turned on and a 0 indicates that a node is turned off.

Processing in a back propagation net consists of placing a pattern of activity on the input nodes, sending the activity forward through the weighted links to the hidden nodes, where a hidden pattern is computed and then on to the output nodes, where an output pattern is computed. Weights represent connection strengths between the nodes. For example, in a spelling correction application, a misspelling represented as a binary n-gram vector might serve as an input pattern to the net. Its corresponding output pattern might be a vector of *m* elements, where *m* is the number of words in the lexicon and only the node corresponding to the correct word is turned on.

2.3 Available Spell Checkers

There are many spell checkers available for different Indian languages. Following section involves a brief discussion of some available Spell Checkers.

2.3.1 Bangla Spell-Checker [9]

Bangla Spell Checker can act in both offline and online modes. It has a Graphics User Interface via an editor. Whenever the user types Bangla text, it checks for wrong spelling and gives suitable suggestions. On the other hand, one can run this software on previously typed material such as an OCR document. All misspelled words are captured by the system. For a single-error word, intended word is found within top four words in the suggestion list. Words having more than one error are so captured and for most of them intended words are in the

upper half of the suggestion list. However, suggestions cannot be given on some inflected words. It has facility to add new words in the dictionary against which spellings are checked.

The spell-checker is a tool to detect errors in Bangla words and correcting them by providing a set of correct alternatives which includes the intended word. An erroneous word can belong to one of two distinct categories, namely, non-word error and real-word error. Let a string of characters separated by spaces or punctuation marks be called a candidate string. A candidate string is a valid word if it carries a meaning. A meaningless string is a non-word error. By real word error we mean a valid but not the intended word in the sentence, thus making the sentence syntactically or semantically ill-formed or incorrect. In both cases, the problem is to detect the erroneous word and either suggest correct alternatives or automatically replace it by the appropriate word. In this spellchecker, only the non-word errors are considered.

2.3.2 Oriya Spell Checker [10]

Error detection and automatic or manual correction for misspelled words is taken care of successfully by Oriya Spell Checker (OSC). There has been developed some algorithms to perform OSC in order to find out more accurate suggestions for a misspelled word. Then searching algorithm of the OSC is so fast that it processes nearly 170000 Oriya words for each misspelled word. The words are indexed according to their word length in word database in order for effective searching. On the basis of the misspelled word, it (OSC) takes into account the number of (i) matching characters, (ii) corresponding matching characters in forward direction and (iii) corresponding matching characters in backward direction to give more accurate suggestive words for the misspelled word. Moreover, it also takes help of the Oriya Morphological Analyser (OMA) for ascertaining the mistakes of inflection (VIBHAKTI), derived and compounding words. The Correction engine corrects automatically the error like phonetically similar characters; OCR's miss recognised characters with the help of confusion matrix and misspelled word correction by using most phonetically confused word database through n-gram model. The OSC is running successfully in Word Processor. Using this technique, **Hindi Spellchecker** (270000 words) and **English Spellchecker** (20000 words) have also been developed for Word Processor (WP). The word files are stored in ASCII format and supports ORTTSarala (Oriya), DV-

TTYogesh (Hindi) and any font in English. This software is designed using Java and Java Swing (Front-end) for both the Windows-98/2000/NT and the Linux O/S.

2.3.3 Marathi Spell Checker [11]

Under this ongoing project, a standalone spellchecker is being built for Marathi. The spellchecker will be available to spell check documents in a given Encoding. From the CIIL (Central Institute of Indian Language) corpus, 12,886 distinct words have been listed. Similarly other Marathi texts at the centre are being used to build a basic dictionary. A morphological analysis is being carried out on the collection of words. For example, an automatic grouping algorithm identified 3,975 groups out of 12,886 distinct words. First word is usually the root word. Thus, there are approximately 4000 root words from Marathi corpus. A manual proof reading will be done on these results. The morphology will be enriched.

A motivation behind the stand-alone spellchecker is that it can be used without an editor through a packaged interface, or it can be integrated with other compatible applications such as OCR.

2.3.4 Assamese Spell Checker [12]

The development of a Spell Checker for Assamese has been undertaken at the Resource Centre, and code has been developed in Perl. The Spell Checker exists in the form of separate modules for error detection and correction, as well as a stand-alone system in which all the spell checking routines have been integrated. The strategy used is described below. Figure 2.1, shows the flowchart of Assamese Spell checker.

The *non-words* are detected by looking up document words in a dictionary of valid words. The dictionary used is actually a word list of around 72000 Assamese words. A hash table has been used as a lexical lookup data structure. The performance of a dictionary using a hash table is quite adequate, even for complex access patterns. Perl has an efficient implementation of a hash table data structure, which has been used in the non-word detection module. A three-pronged strategy has been used for generating suggestions comprising of the Soundex, Edit-distance and Morphological processing methods. This method maps every

word into a key (code) so that similarly spelled words have similar keys. A Soundex encoding scheme for Assamese has been designed based on the encoding scheme for English which comprises of a set of rules for encoding words and 14 numerical codes. The Soundex code of the misspelt word is computed, and the dictionary is searched for words, which have similar codes. Candidate suggestions are obtained by 'Damerau's error reversal' method. The four well known error-inducing edit actions are the insertion of a superfluous letter, deletion of a letter, transposition of two adjacent letters and substitution of one letter for another. In 'Damerau's error reversal' each edit action is applied to the misspelt string and a set of strings is first generated. These are checked in the dictionary to see which of them are valid words to finally produce the suggestions.

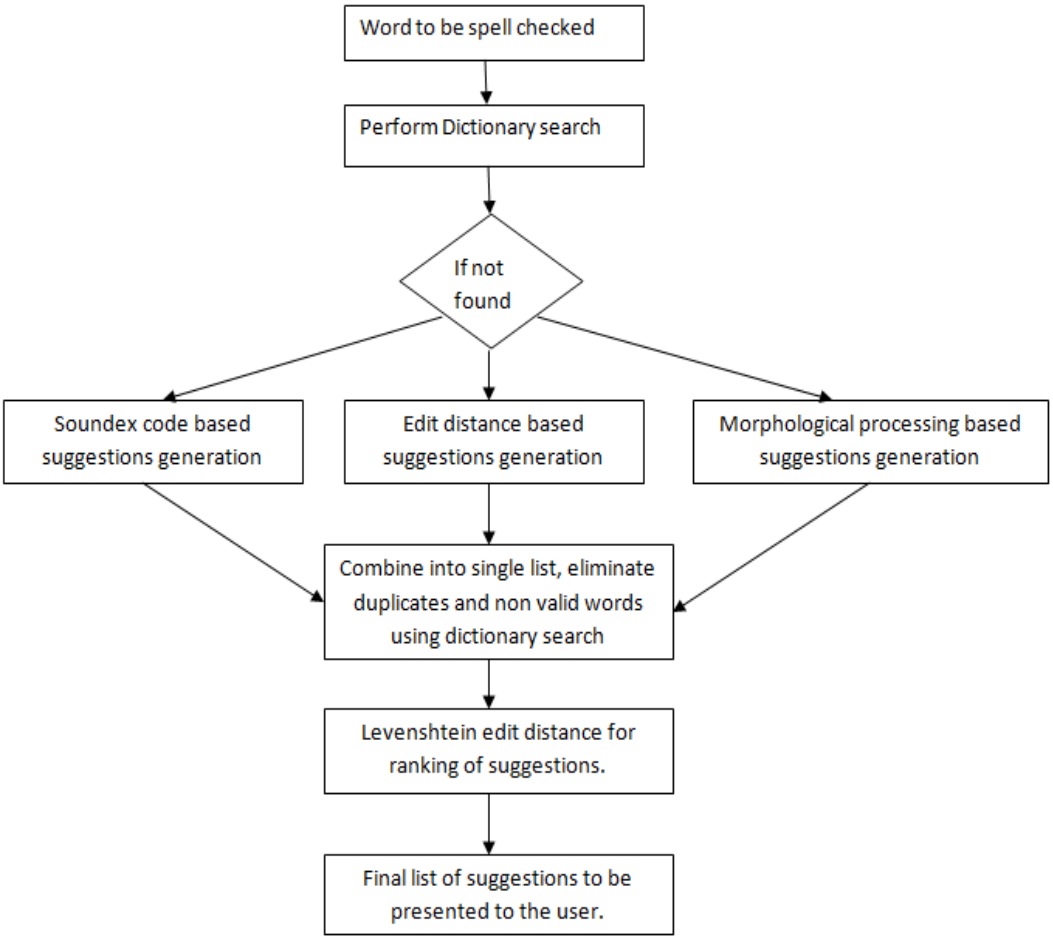

Figure 2.1: Strategy followed by the Assamese Spell Checker [12].

2.3.5 Annam (Tamil Spell Checker) [13]

Tamil Spell Checker is a tool used to check the spelling of Tamil words. It provides possible suggestions for erroneous words. User has the provision to select the suggestion among the list, ignore the suggestion or add the particular word to the dictionary. These modules extract the root word from the given word (noun/verb) with the help of Morphological Analyser and the root word is checked in the dictionary and if found, the word is termed as correct word. Otherwise, the correction process is activated. The correction process includes error handling and suggestion generation modules. After finding the type of error, the right form of suffixes; nouns or verbs are given as input to suggestion generation module. With the help of Morphological Generator, the correct word is generated. This module also handles the operations like select, change or ignore the suggested word and adding the word to the dictionary.

2.3.6 Malayalam Spell Checker [14]

It is a software subsystem that can be integrated with Microsoft word as a macro or the Malayalam editor, stylepad developed by CDAC, Thiruvananthapuram, to check the spelling of words in a Malayalam text file. While running as a macro in word, it functions as an offline spell checker in the sense that one can use this software with a previously typed text file only. Both off line and online checking are possible when it is integrated with the text editor. It generates suggestions for wrongly spelt words.

The system adapts a rule cum dictionary-based approach for spell checking. It incorporates a fully developed Morphological Analyser for Malayalam. This module splits the input word into root word, suffixes, post positions *etc.* and checks the validity of each using the rule database. Finally it will check the dictionary to find whether the root word is present in the dictionary. If anything goes wrong in this checking it is detected as an error and the error word is reprocessed to get 3 to 4 valid words, which are displayed as suggestion. The user can add new words into a personalised data base file, which can be added to the dictionary if required.

2.3.7 AKSHARA: Telugu Spell-Checker [15]

Perhaps the best and most thoroughly worked out morphological analyser for Telugu is the one which has been developed at University of Hyderabad over the past 10 years or so. The system has been tested on large scale corpora and, enhancements and refinements have been going on for years. During this project, a thorough re-engineering work was taken up and a new version was developed. The new version is far simpler, more transparent, portable, well documented, and conforms to standards. Research work has also been taken up on developing stemming algorithms for Telugu. A pure corpus based statistical stemming algorithm has been developed. The performance of this stemmer for the spell checking application has been studied in various combinations with dictionary and morphology based approaches.

Large scale spelling error data has been obtained from our 10 Million word Telugu corpus. The raw corpus as it was typed has been compared with the final version after three levels of proof reading and certification by qualified and experienced proof readers. A number of tools have been developed to prepare such a data. Quantitative study of spelling error patterns in Telugu is being conducted. This will help to build better spell checkers in future.

Since words are large and complex and hence too numerous in Telugu and proper morphological analysis is difficult, it would be useful to perform studies at lower levels of linguistic units. The syllable level is a natural choice since writing in Indian languages is primarily syllabic in nature. N-gram models have been built at syllable level.

2.3.8 Akhar [16]

Bilingual Spell Checker for Punjabi and English with support for additional dictionaries. A language sensitive Punjabi/English spell checker has been provided in Akhar. Akhar can automatically detect the language and invokes the respective spell checker. The Unicode compliant Punjabi spellchecker is font independent and can work on any of the popular Punjabi fonts such as AnanpurSahib, AmritLipi, Jasmine, Punjabi, Satluj *etc*. This removes the constraint on the user to type the text in predefined fonts only. Figure 2.2, shows the snapshot of AKHAR.

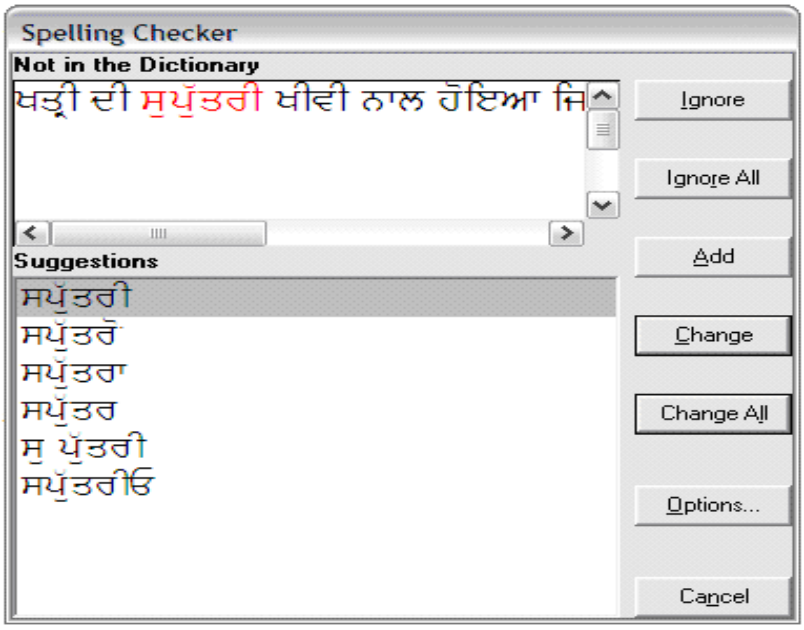

Figure 2.2: Snapshot of AKHAR [16]

It provides the facility to select the words from alternatives list by using Alt keys only. No need to shift to mouse input while using the keyboard, resulting in saving of time. User can add words to the Roman-Punjabi database easily, with no previous knowledge of Punjabi typing required. The on screen keyboard can be used to click the Gurmukhi keys. Typing text from Gurbani made simpler. Separate database of Gurbani words provided to suggest the most appropriate word for Gurbani text and pop up the other similar sounding alternatives. More words added to Roman-Punjabi database.

Chapter 3: Problem Statement

A Spell Checker is a basic necessity for composing text in any language. Though considerable work has been done in the area for English and related languages, the Indian Language scenario present a relatively more complex and uphill task. Punjabi is the world's 12th most widely spoken language. There is a very little amount of work done in this field. The only available Spell Checker for Punjabi language is 'AKHAR'.

Designing a Spell Checker for Indian languages such as Punjabi poses many new challenges not found in English, which complicates the design of the Spell Checker. Punjabi language is far different from Western languages in phonetic properties and grammatical rules. Thus, the existing algorithms and techniques that are being used to check the spelling and to generate efficient suggestions for mis-spelt words of English and other Western languages are not actually suitable for Punjabi; rather it needs different algorithms and techniques for expected efficiency. There are many spell checkers available in different Indian languages which uses different techniques were discussed in chapter 2. The Spell Checker for Gurmukhi Script can use any of these techniques because the syntax and semantics of Gurmukhi Script are similar to other Indian languages as compared to English.

First and the far most requirement for any spell checker is to have dictionary of different possible words of that language which will act as a corpus. A Spell Checker can be stand-alone, capable of operating on a block of text, or as part of a larger application, such as a word processor, email client, electronic dictionary, or search engine. When some Punjabi text will be given as an input to Spell Checker, it should list out the incorrect words separately by checking their availability in the dictionary. Finally it should provide the suggestions for the incorrect words from the dictionary and change the word in the input text with the selected suggestion word. Thus, the main issues related with any Spell Checker are Dictionary Creation, Error Detection, Suggestion Creation and finally Word Replacement.

Based on the above problem statement we have set the following Objectives:-

i) To analyse different types of errors in Punjabi text.

ii) To create a corpus of different Punjabi words which will act as dictionary.

iii) To detect the errors in the input Punjabi text.

iv) To design algorithm for suggestions of nearest matching words for detected errors.

v) To change the word in the input text with the selected suggestion word.

vi) Development and Testing of System.

Chapter 4: Design and Implementation of SUDHAAR-Punjabi Spell Checker

The newly developed system is named as SUDHAAR. It is a Punjabi word which means correction and improvement. It is a Stand-alone application developed in java with Net beans 6.8 platform. This chapter covers the features, architecture, design and implementation of SUDHAAR. Creation of dictionary, error detection and error correction & replacement are the main features of this system. The techniques used for implementation of these modules are Dictionary Look up and Minimum Edit Distance. The working of the system with all the snapshot are explained in last section of this chapter.

4.1 Features of SUDHAAR

The main features of designed Spell Checkers are:-

1) It contains a dictionary of around one million Punjabi words.

2) It checks each word of the input text for its presence in dictionary for error detection

3) It gives the suggestions for detected errors and replaces the selected suggestion in input text.

4) The user can append word in the dictionary.

5) It is a Standalone application capable of taking input from any source.

6) It is open source software freely available to everyone.

4.2 Architecture of SUDHAAR

There are two different applications in the system. One creates the dictionary which is already been developed and executed by us and thus, the dictionary of about one million words has been created.

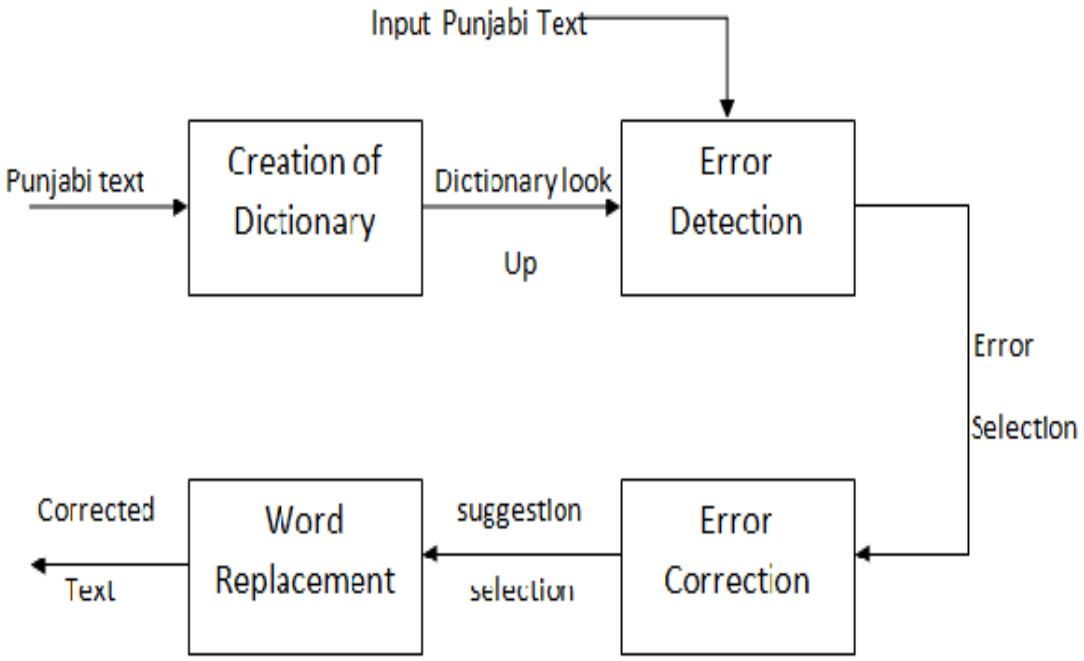

Figure 4.1: Architecture of SUDHAAR

Second is designed for Spell Checking of Punjabi Text where user gives the input Punjabi text and the system detects the errors by looking up for that word in dictionary and provide the suggestions for those errors. Then user can select the suggestion from the list and make changes accordingly and the final output is the corrected text without errors. Figure 4.1, shows the architecture of the system. The system is divided into three modules. Creation of Punjabi Dictionary, Error Detection and Error correction and Replacement. Various techniques and methods are used to develop these modules.

4.3 Dictionary Creation Tool

This application is a part of spell checker but is not needed to be executed every time because a dictionary that acts as database for Spell Checker is created using this tool. A corpus of around one million words which contains unique Punjabi correct words has been created using this application which will act as Dictionary for the Spell Checker.

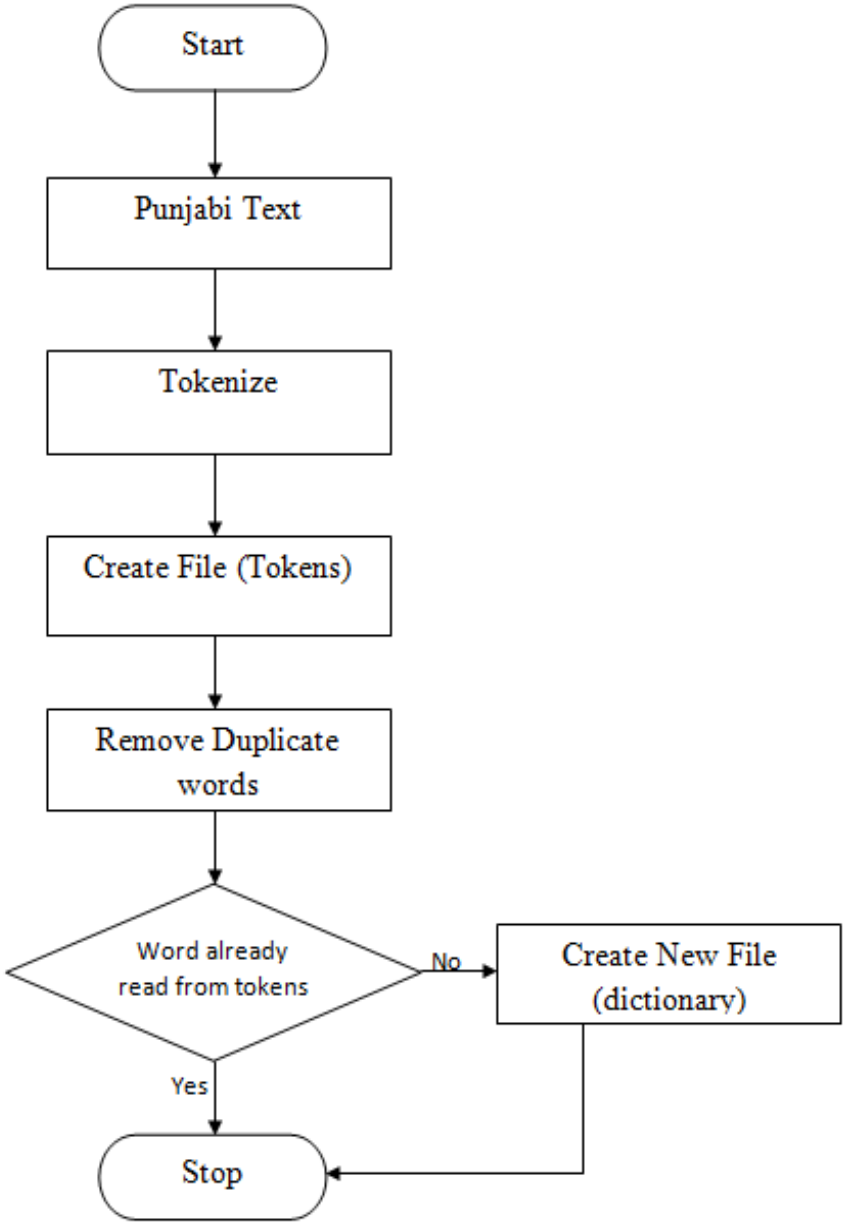

Figure 4.2: Flowchart of Dictionary Creation Tool using Hash Function

An application is developed in java to create that dictionary where a large Punjabi text is given as an input.

Figure 4.2, shows the flowchart of Dictionary Creation Tool. It tokenizes each word and calculates unique hash code of each word. A Hash Set holds a set of words, but in a way that it allows you to easily and quickly determine whether an object is already in the set or not. It does so by internally managing an array and storing the object using an index which is calculated from the hash code of the word, thus, resulting in a set of unique words.

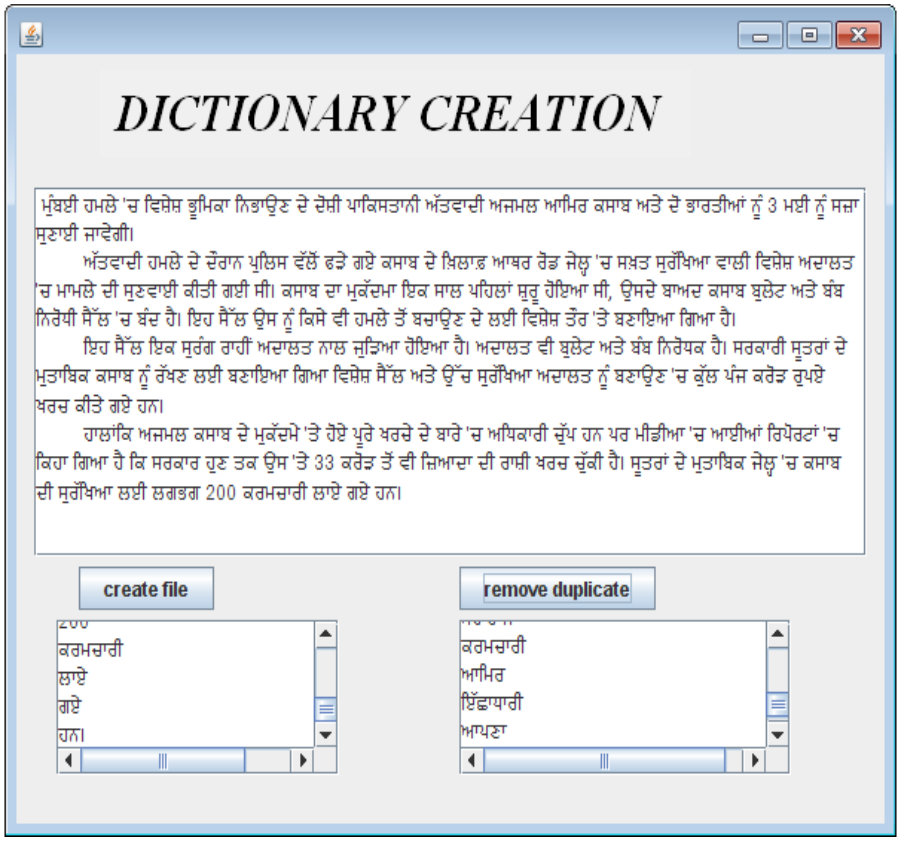

Figure 4.3: Snapshot of Dictionary Creation Tool

Figure 4.3, shows the snapshot of the dictionary creation tool where it takes a large Punjabi text as an input. When the “create file” button is clicked, it creates a text file which contains all the tokens of the input text. When the “remove duplicate” button is clicked, it creates another text file which saves only unique tokens of previously created file, which then acts as dictionary.

4.4 Error Detection

It detects the errors present in the input text. There are many techniques available which can be used for error detection.

Dictionary look up is one of the two principal ways of spelling error detection. The other approach, popularly used in OCR problems, is the n-gram approach. Construction of appropriate n-gram from raw text data is an important issue in this approach [2].
In this system, dictionary look up technique is used which checks each word of input text for its presence in dictionary. If that word is there in dictionary, then it is a correct word,

otherwise it is put into the list of error words. Figure 4.4, shows the flowchart of the Error Detection module of the system. The user gives the input Punjabi text and the system then tokenizes the text. Till it has more tokens, the system checks whether the token is present in dictionary or not. If not, it is an error.

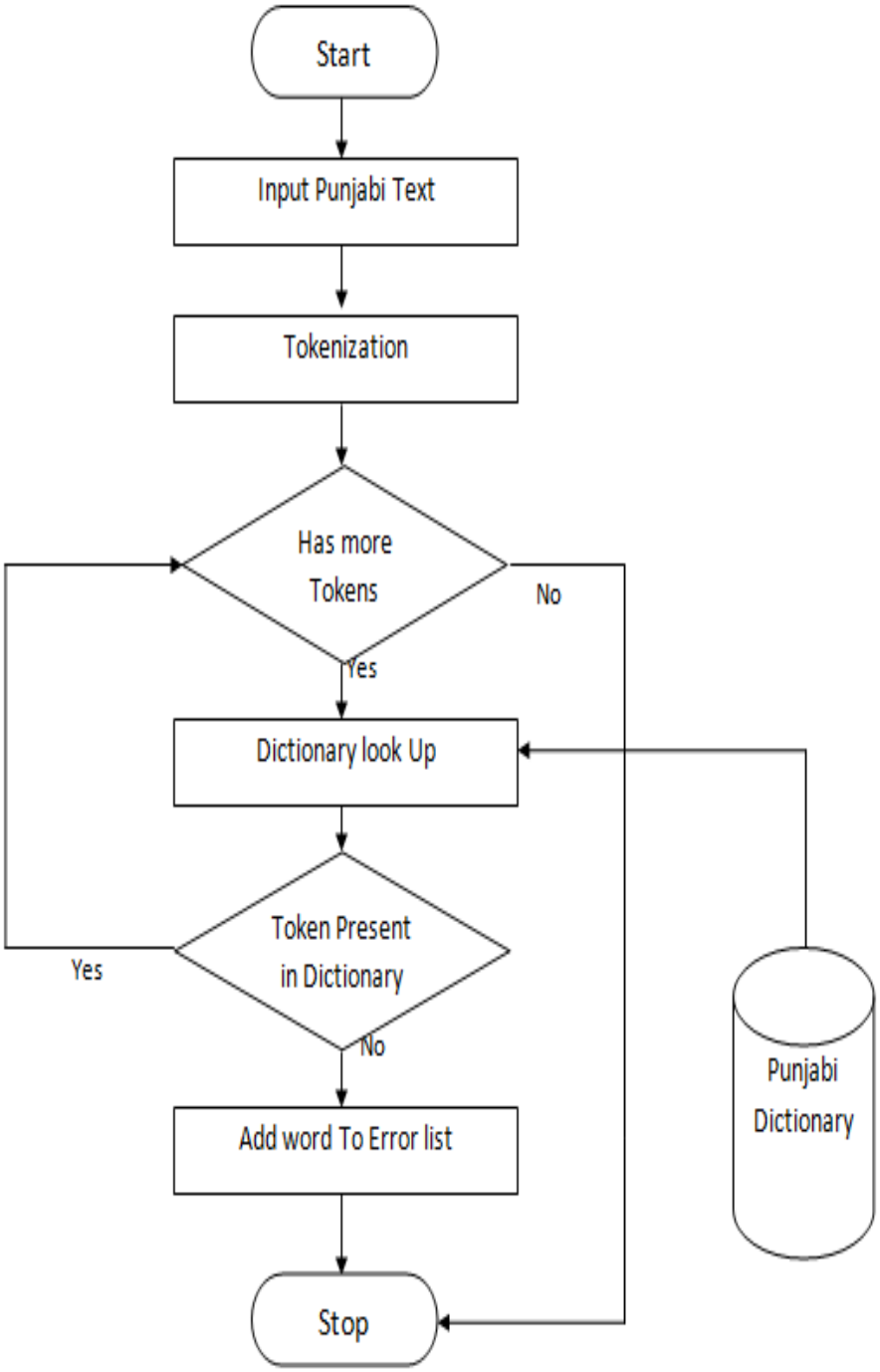

Figure 4.4: Flowchart of Error detection using Dictionary Look Up

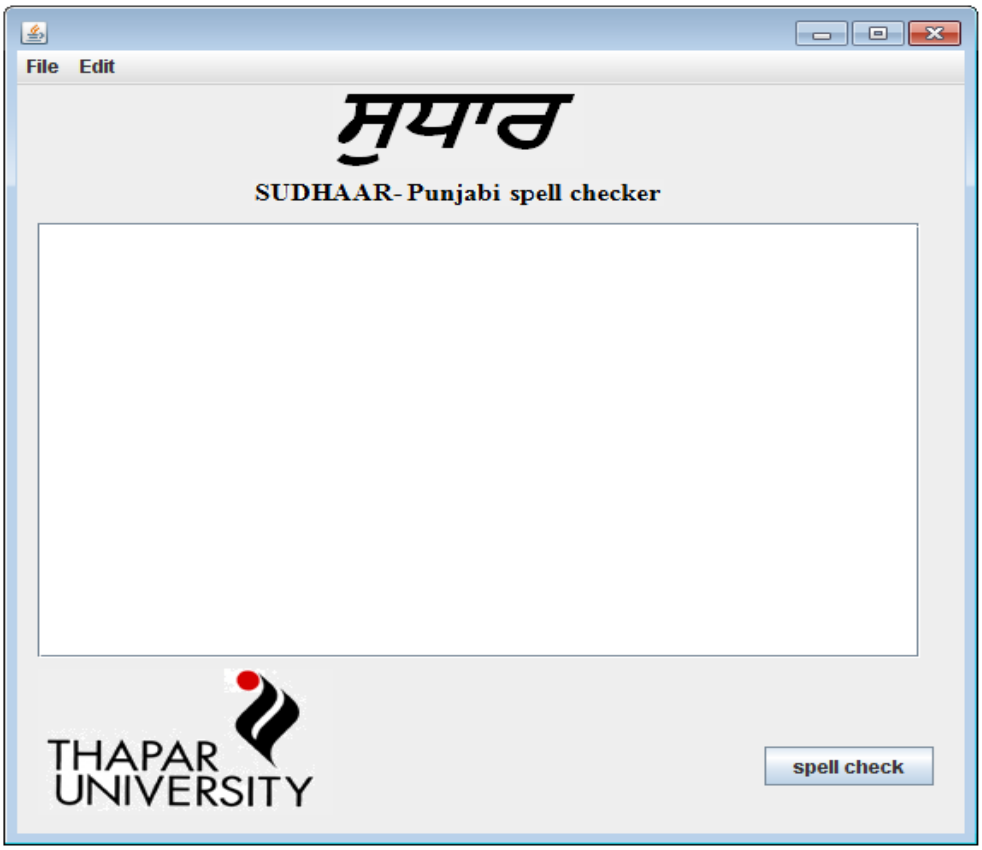

Figure 4.5: Snapshot of first screen of SUDHAAR

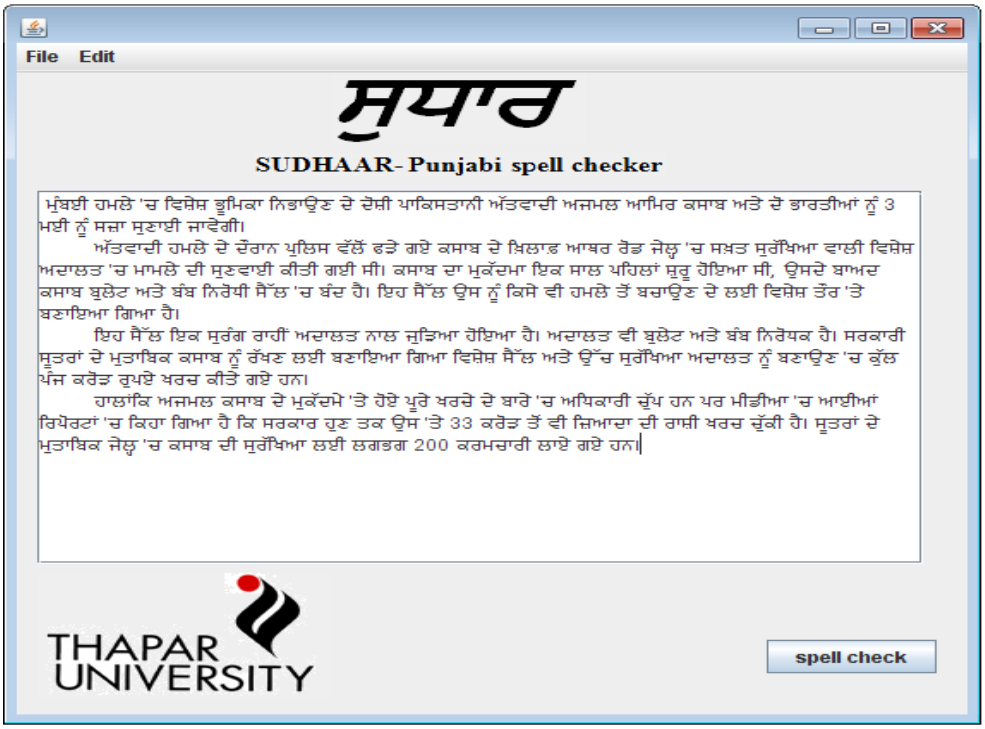

Figure 4.6: Snapshot of input text screen

The window where user enters the input Punjabi text is shown in Figure 4.5. When user provides the input as shown in Figure 4.6, and clicks on spell check button, a new window opens which shows all the detected errors as shown in Figure 4.7.

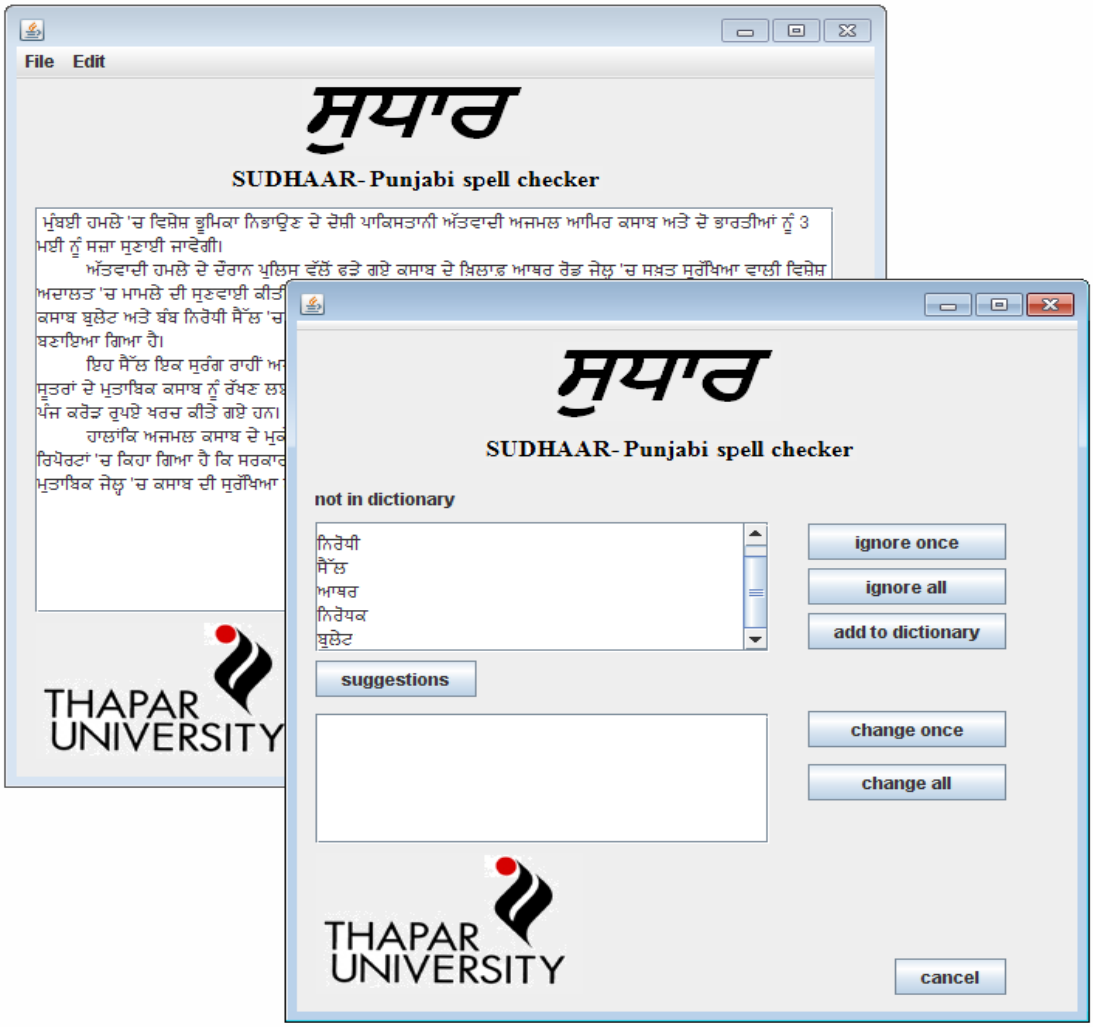

Figure 4.7: Snapshot of error detection screen

4.5 Error Correction and Replacement

Providing correct word as a suggestion for any error is the most important task of any Spell Checker and then replacing the selected suggestion in the input text. In SUDHAAR, we used Minimum Edit Distance technique to find out most suitable suggestion for the error words.

4.5.1 Minimum Edit Distance (M.E.D.)

Several approaches based on minimum edit distance, similarity key, rules, n-grams, probability and neural nets are proposed to accomplish the task of error correction. Of these, minimum edit distance based approaches are the most popular ones. The minimum edit distance is the minimum number of editing operations (insertions, deletions and substitutions) required to transform one text string into another. The distance is also referred to as Damerau-Levenshtein distance after the pioneers who proposed it for text error correction [6,17]. In its original form, minimum edit distance algorithms require *m* comparisons between misspelled string and the dictionary of *m* words. After comparison, the words with minimum edit distance are chosen as correct alternatives.

4.5.2 Implementation of Minimum Edit Distance

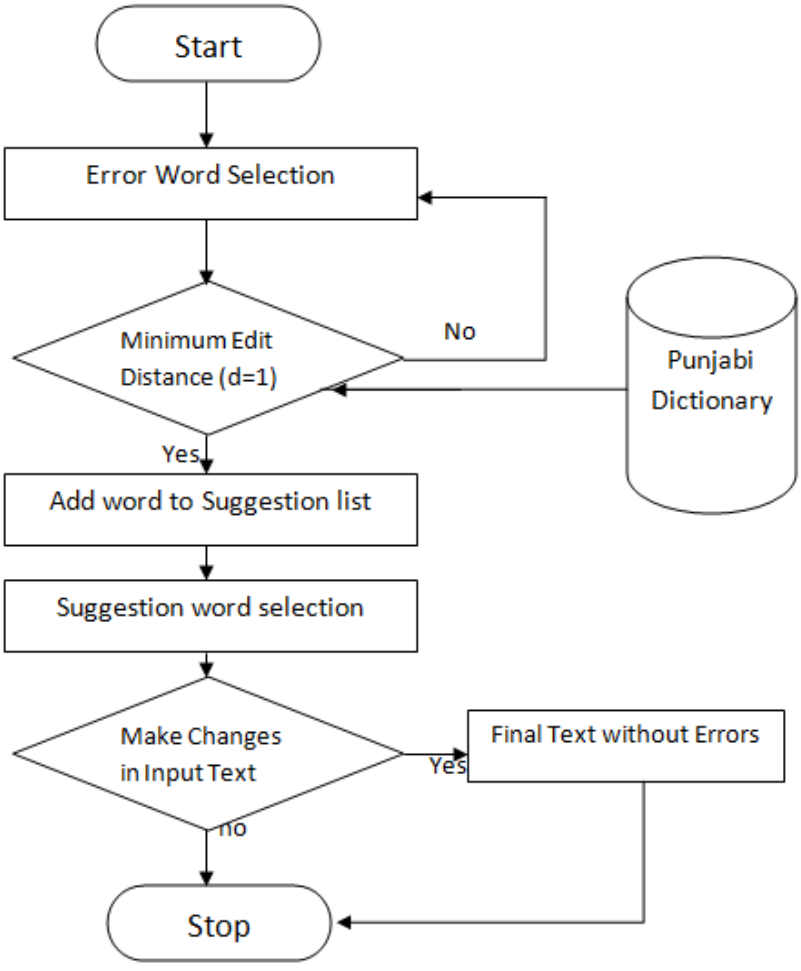

Figure 4.8: Flowchart of Minimum Edit Distance

Figure 4.8, shows the flowchart of error correction and replacement module using minimum edit distance algorithm. The user selects the error word from error list and system performs the minimum edit distance for that word, by comparing it with each word in the dictionary and gives the words in the suggestion list with distance(d) = 1. Then, user again selects the word from suggestion list and replaces it in input text.

As shown in figure 4.9, user selects the word from the generated error list, and clicks on the 'suggestions' button, a suggestion list is created in new text box. The user then selects the word from suggestion list and click on 'change all' button. The error word in the input text gets replaced by the selected suggestion, which is shown in figure 4.10. Similarly, user can perform tasks like ignore once, ignore all and add to dictionary, *etc*.

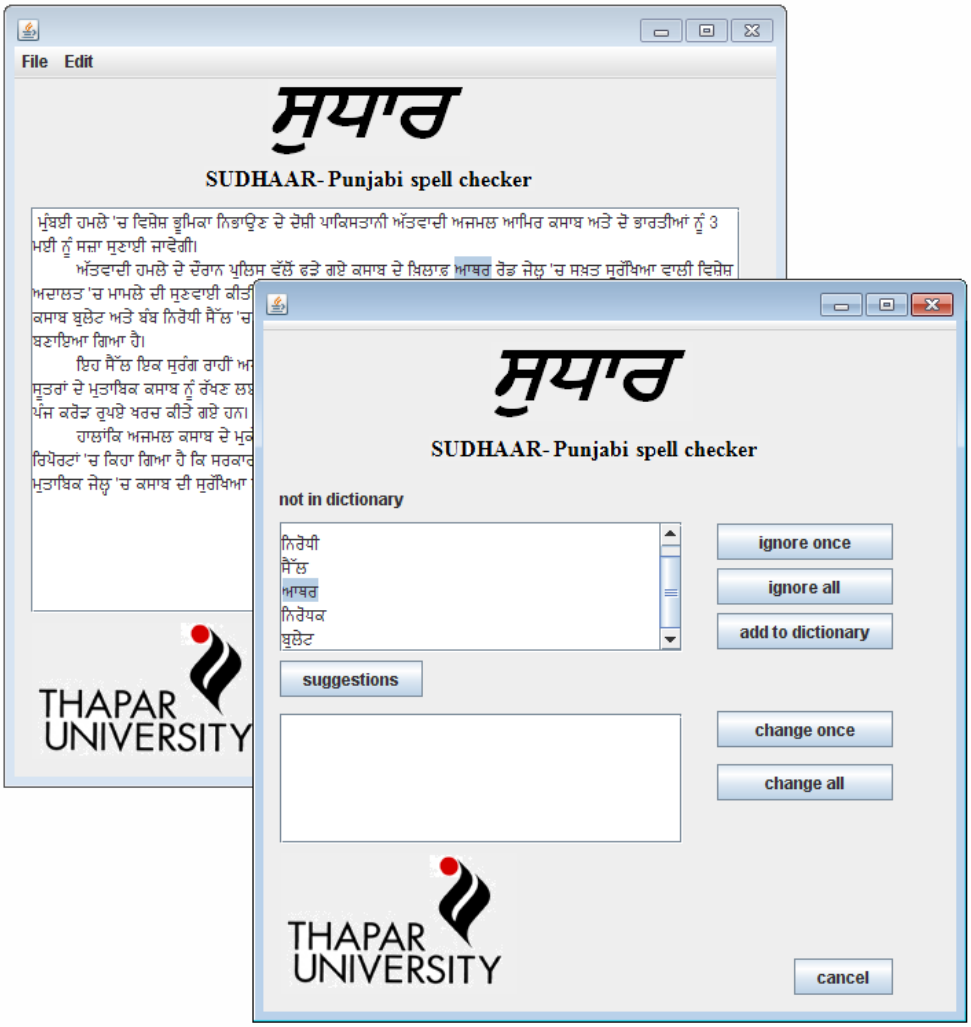

Figure 4.9: Snapshot of error selection screen

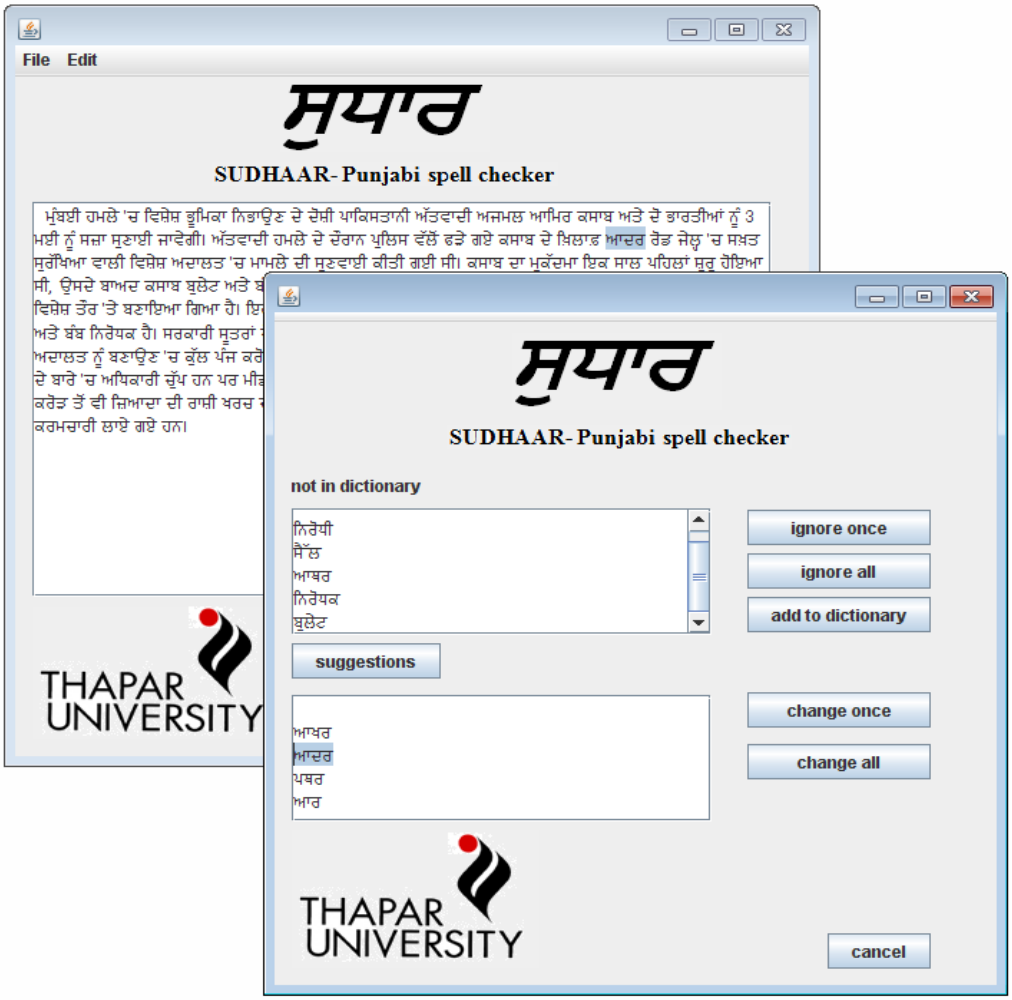

Figure 4.10: Snapshot of suggestion selection screen

4.6 Working of SUDHAAR

SUDHAAR, Punjabi Spell Checker, is a Standalone application. The working and the functionalities of various buttons of the system are described in this section.

4.6.1 Main Window

Whenever the application is started, the window shown in figure 4.11, appears which contains the text area, where user can enter the input text for spell checking.

Open File:- There are two options to enter text for spell checking:-

- The user can directly copy the content of text file from the destination and paste in the input area.
- The user can browse the file from the open menu item of file menu bar.

Figure 4.12, shows the snapshots of the open file process. Finally, the snapshot, where the text file is opened in the text area for spell check is shown in figure 4.13.

Figure 4.11: Snapshot of Main Screen

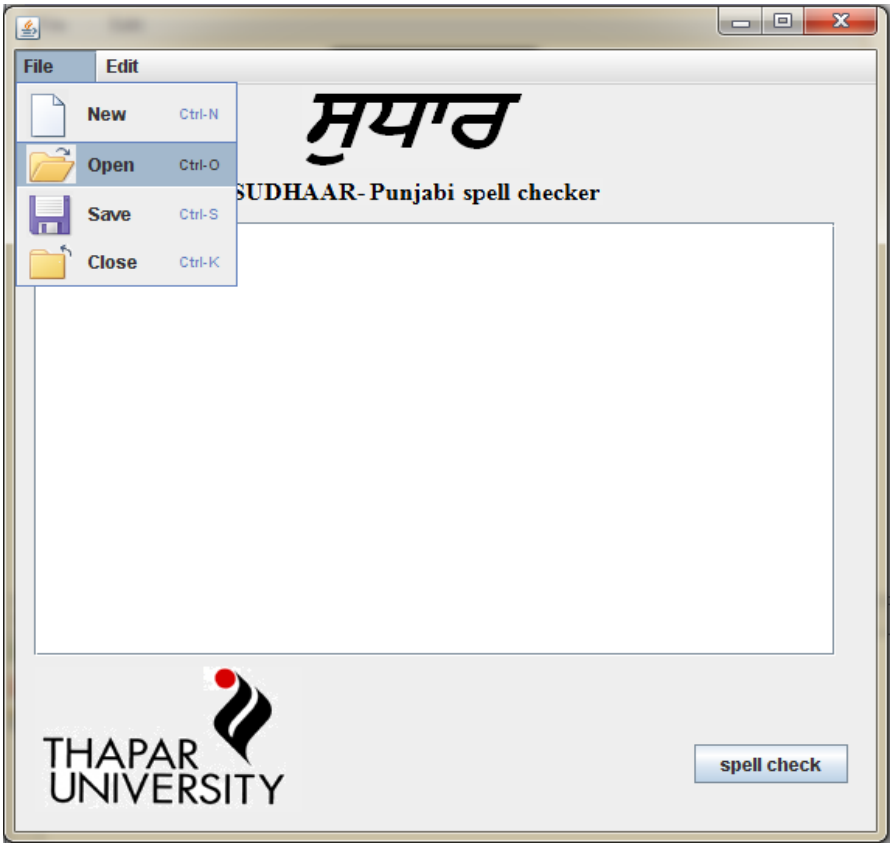

Figure 4.12: Snapshot of Open menu

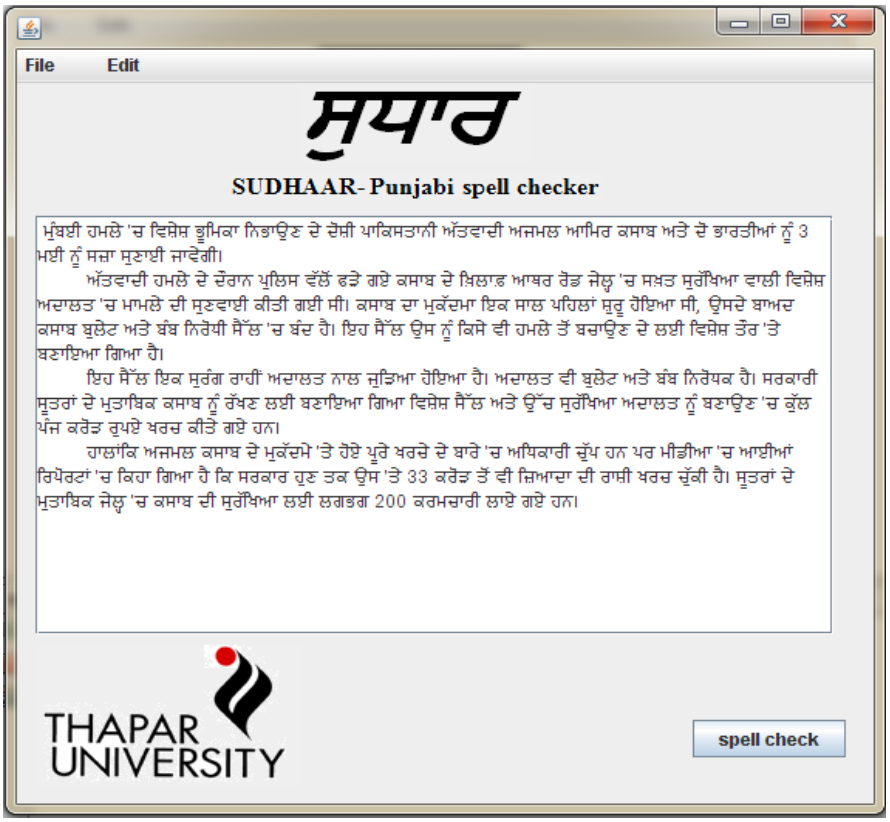

Figure 4.13: Snapshot of Input Text Screen

4.6.2 Spell Check

There is a button on main window *i.e.* Spell Check and is clicked when user wants to check the text file entered in the text area for spelling errors. When user clicks on this button, a new window gets opened which displays the list of word which are not present in the dictionary and are considered as errors. There are seven buttons on the second window namely ignore once, ignore all, add to dictionary, change once, change all, suggestions and exit. The functionalities all these buttons are described latter in this section.

Initially the change once and change all buttons are disabled. The user can select the word from error words list and can perform any of three tasks from ignore once, ignore all or add to dictionary. The snapshots of the new window, when user selects the word to take any action are shown in figure 4.14 and 4.15.

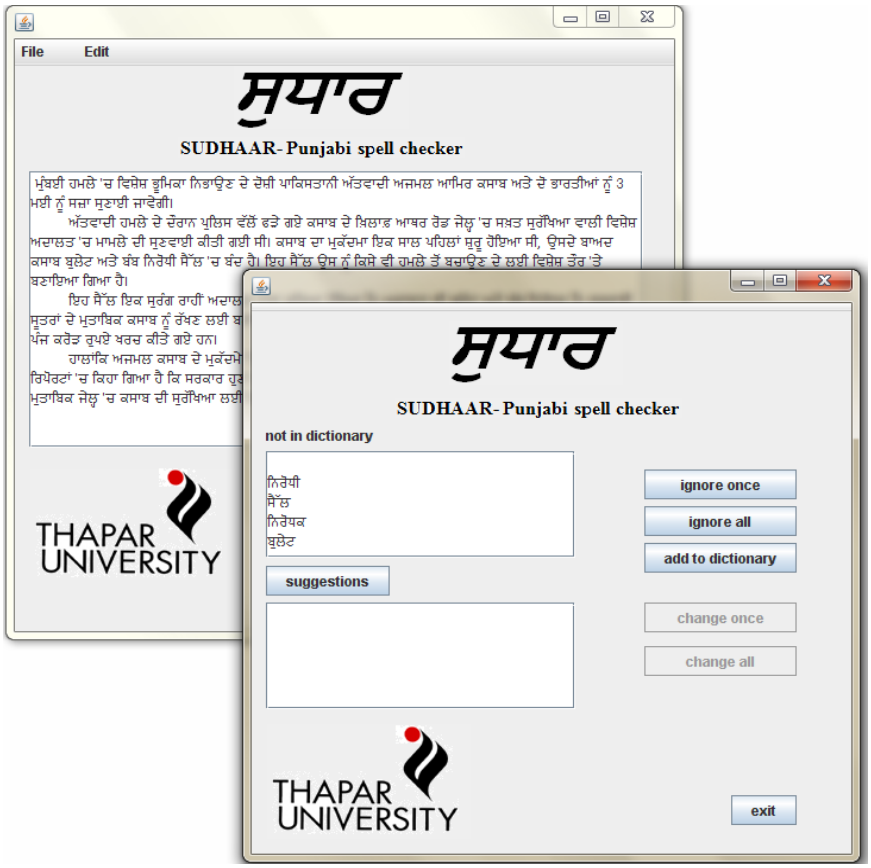

Figure 4.14: Snapshot of Error Detection Screen

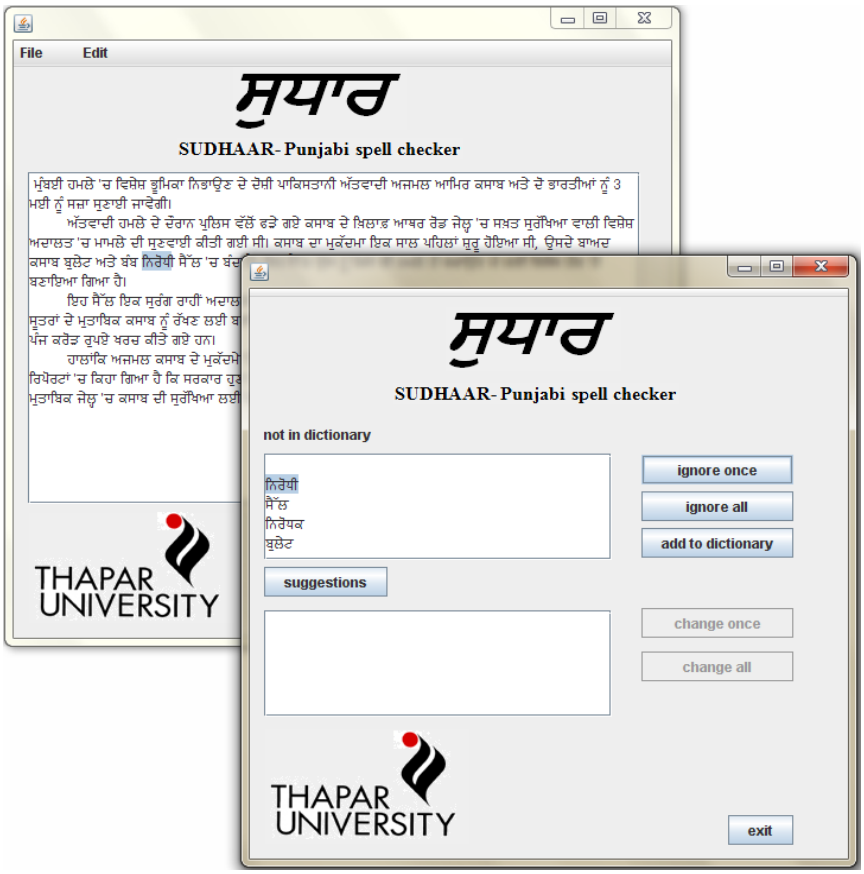

Figure 4.15: Snapshot of Word Selection

Ignore once: - After the selection of the word from the list, when user clicks the 'ignore once' button, the spell checker checks if the word comes more than once in the text or only once. If it comes only once, then the word gets deleted from the error words list, else it remains there in the error list.

Ignore all:- Once the ignore all button is clicked, no matter it comes how many times in the input text, it gets ignored *i.e.* it gets deleted from the error words list.

Add to dictionary: - When the 'add to dictionary' button is clicked, the selected word gets added to the dictionary and gets deleted from the error words lists automatically.

When no word is selected from the list and any of these three buttons are clicked, the Spell Checker shows the message that 'no word is selected' and performs no action. The snapshots of actions of these three buttons are shown in figures 4.16, 4.17.

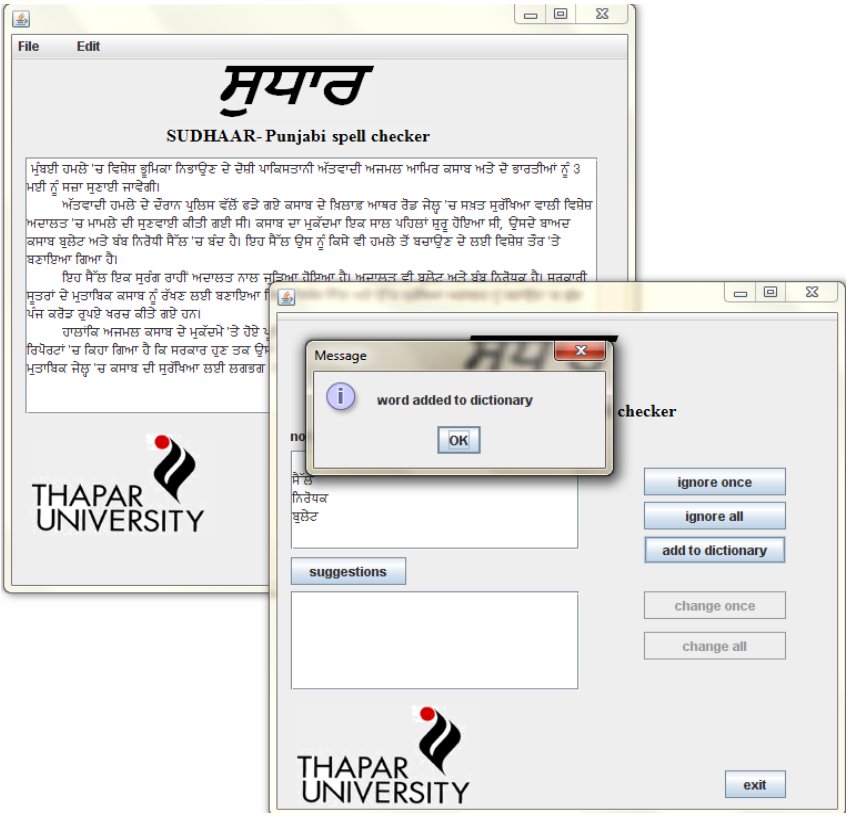

Figure 4.16: Snapshot of Actions performed by ‘add to dictionary’ button

Figure 4.17: Snapshot of Action performed by ‘ignore all’ button

4.6.3 Suggestions

When ‘suggestion’ button is clicked after selecting the word from error list, suggestion words for that selected word are displayed in the text area below the suggestion button and the ‘change once’ and ‘change all’ buttons gets activated.

Change once: - User first selects the word from the suggestion list and when ‘change once’ button is clicked; the first occurrence of the error word in the input text is replaced by the selected suggestion word.

Change all: - When user clicks on the ‘change all’ button after selecting the word from the suggestion list, rest of the occurrences of the word in the input text are replaced by that word.

When no word is selected from the suggestion list and any of these buttons is clicked, the message box appears ‘no word is selected from suggestions’. When there is no word left in the error word list, the message box appears ‘spell check complete’. The snapshots of processing of these buttons are shown in figures 4.18 – 4.25.

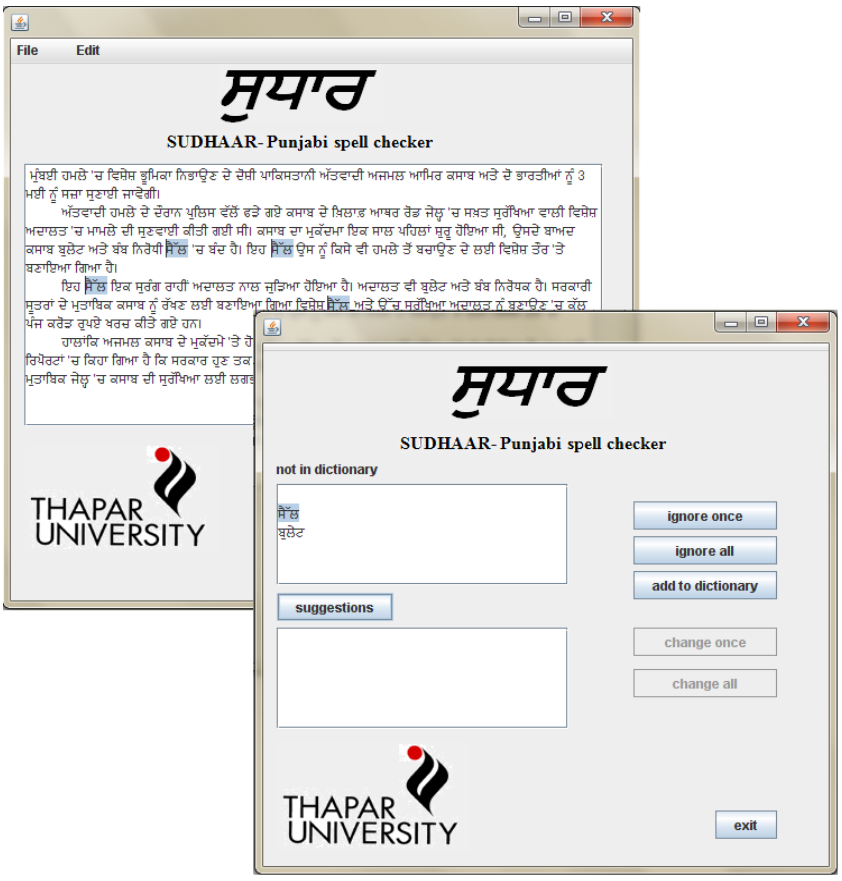

Figure 4.18: Snapshot of word selection from detected errors

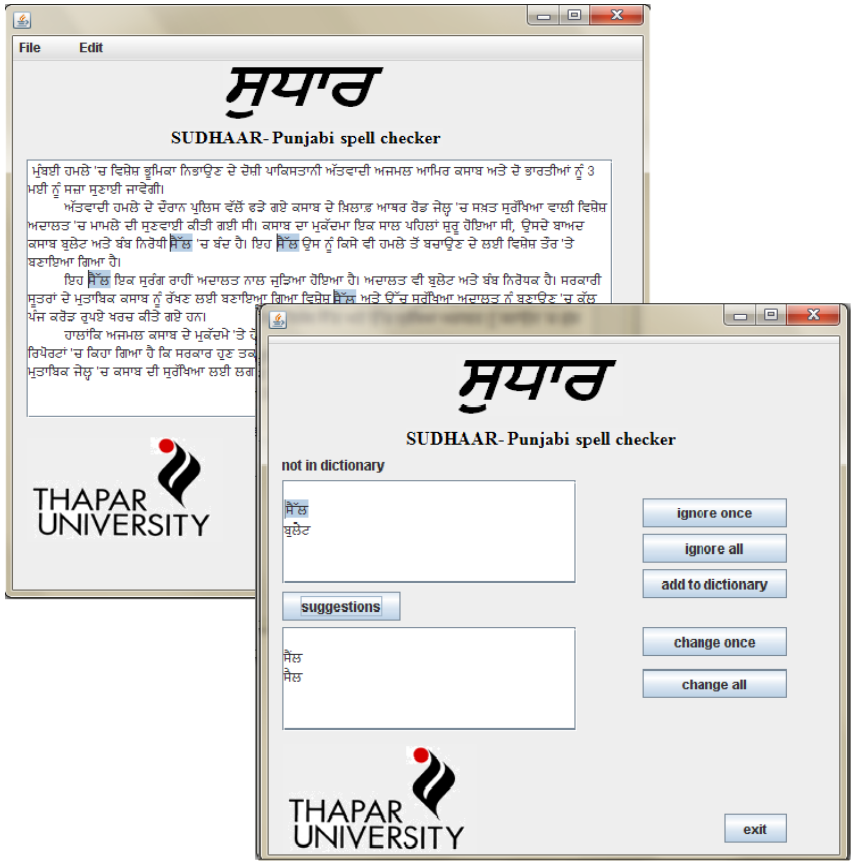

Figure 4.19: Snapshot of ‘suggestions’ button

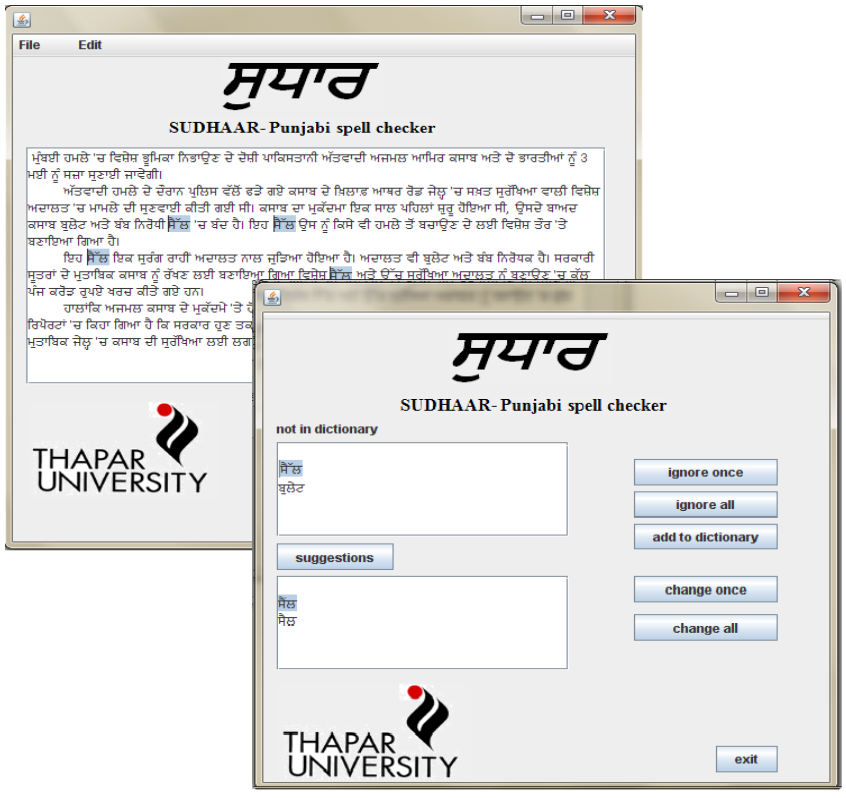

Figure 4.20: Snapshot of selection of word from suggestion list

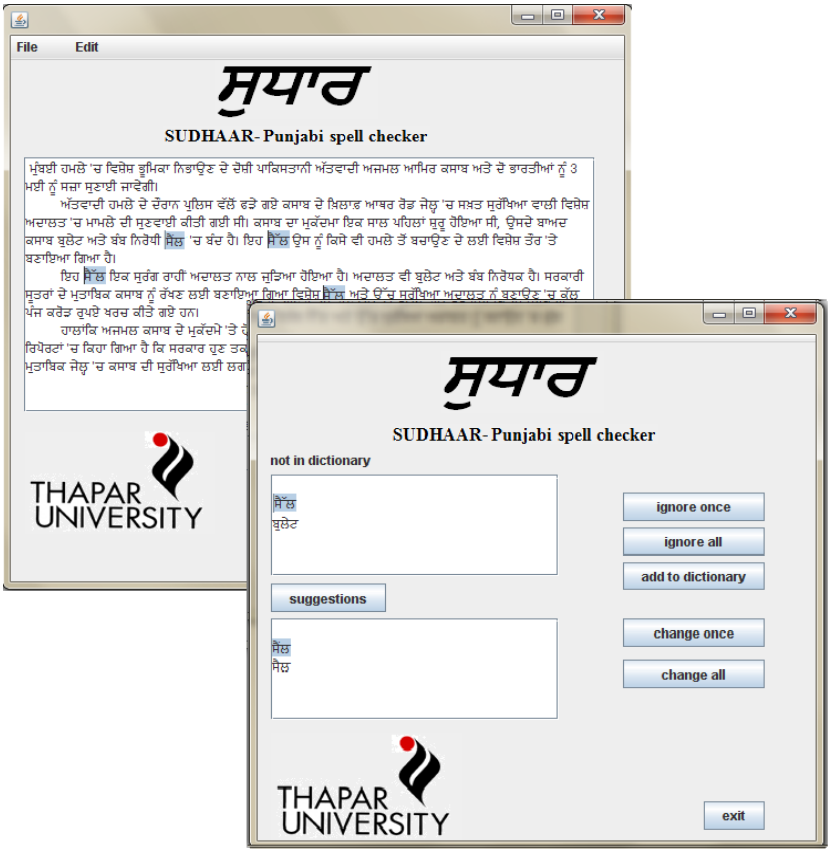

Figure 4.21: Snapshot of ‘change once’ button

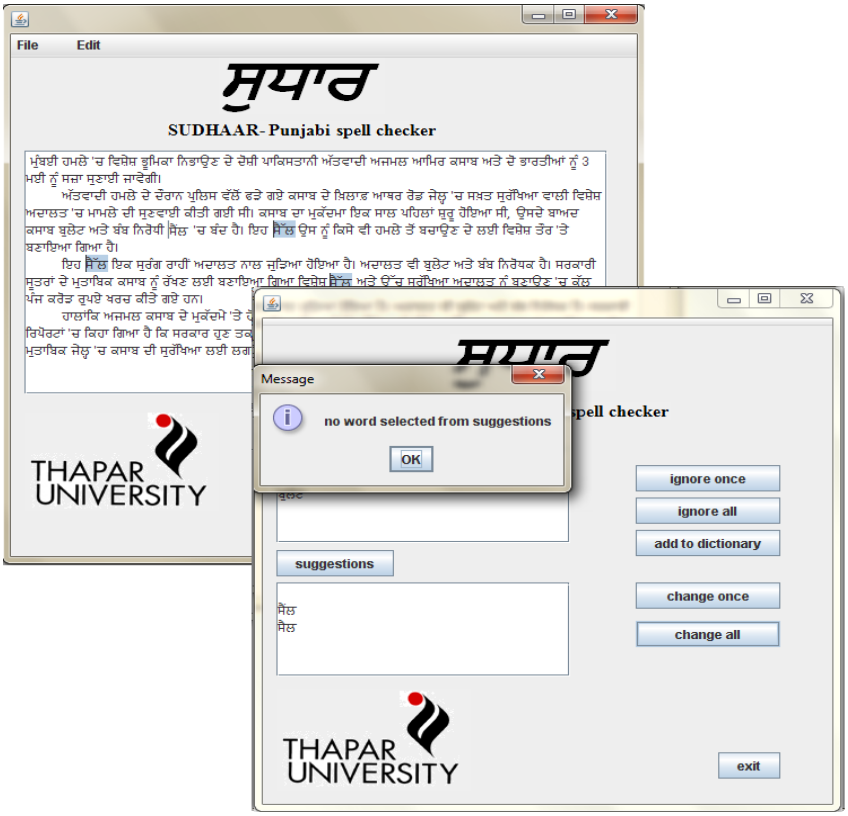

Figure 4.22: Snapshot of “no word selected”

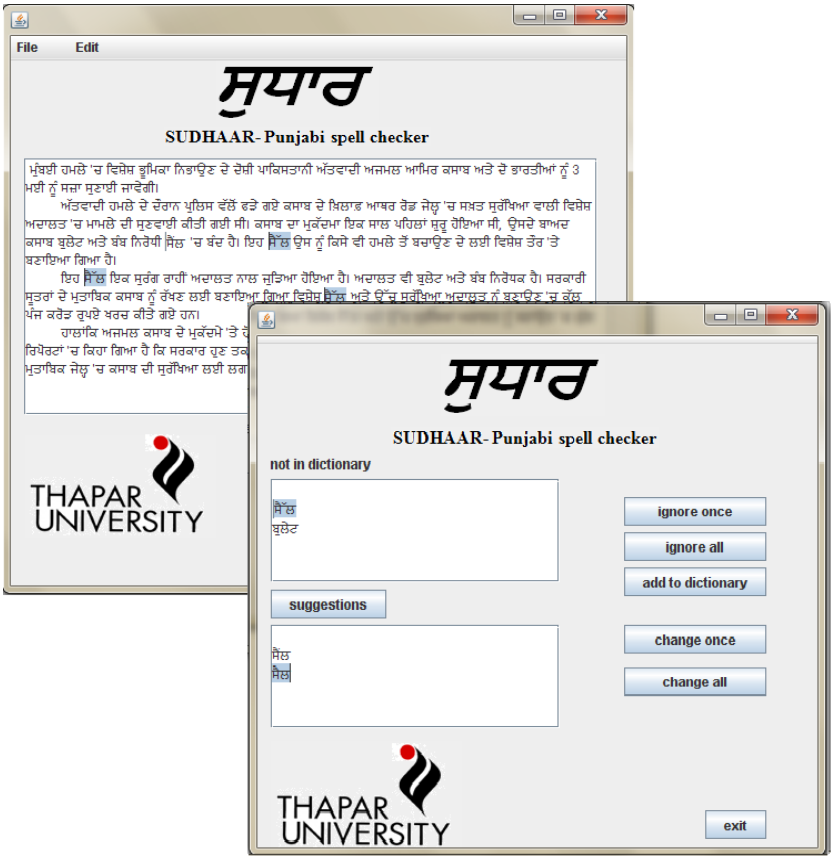

Figure 4.23: Snapshot of 'change all' button

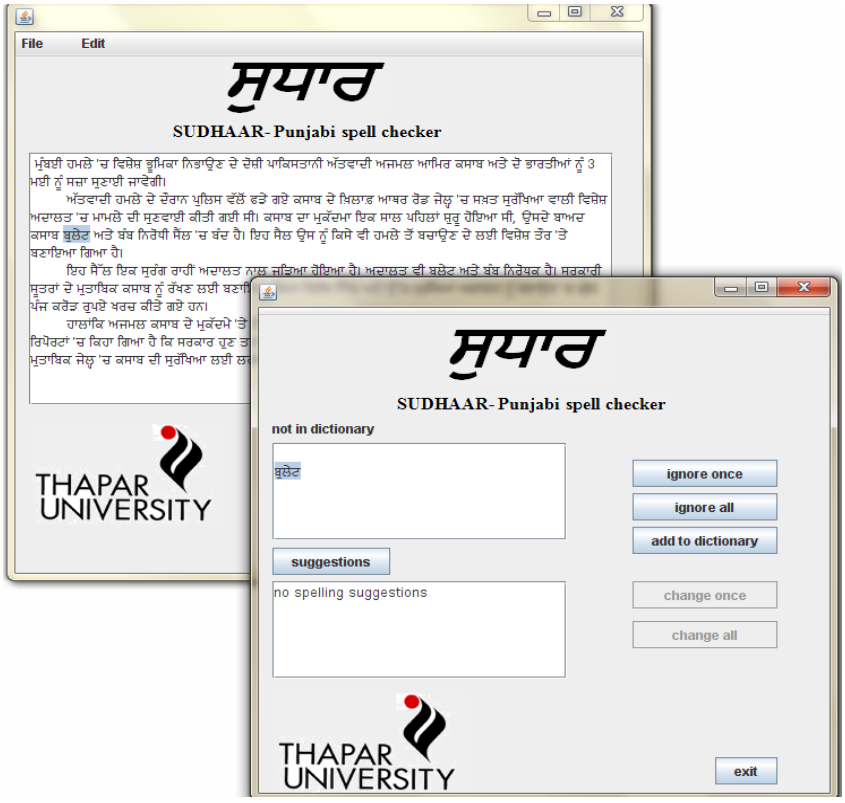

Figure 4.24: Snapshot of "no spelling suggestions"

Figure 4.25: Snapshot of "spell check complete"

4.6.4 Exit

When the 'exit' button is clicked, the message box appears 'do you want to save the changes?' with three options: save, don't save, and cancel.

Cancel:- when 'cancel' button is pressed; it closes only the message box which contains these three buttons.

Don't save: - When 'don't save' button is clicked; whole application gets closed without saving the changes made in the input text. The user can also close the spell checker application from the close menu item of file menu bar.

Save:- When 'save' button is clicked, the save dialog box gets opened from where user can select the location and file name to save the changes made in the input text. The user can also save the changes made in the text from the save menu item of file menu bar.

The snapshots of these three buttons and the file menu items 'save' and 'close' are shown in figures 4.26, 4.27, 4.28.

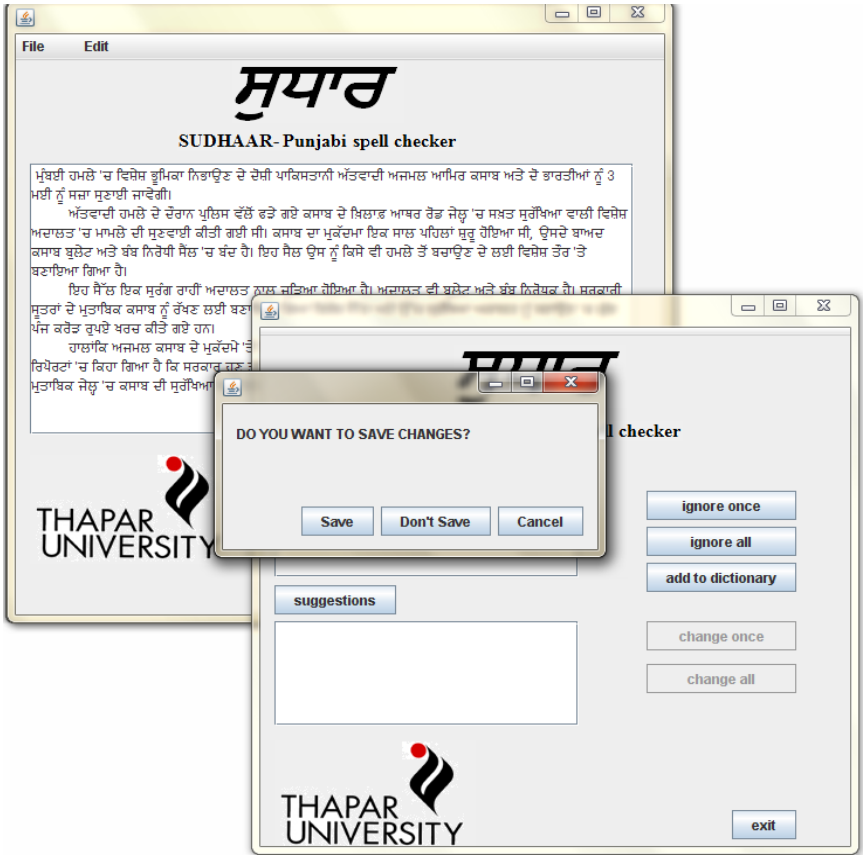

Figure 4.26: Snapshot of 'exit' button

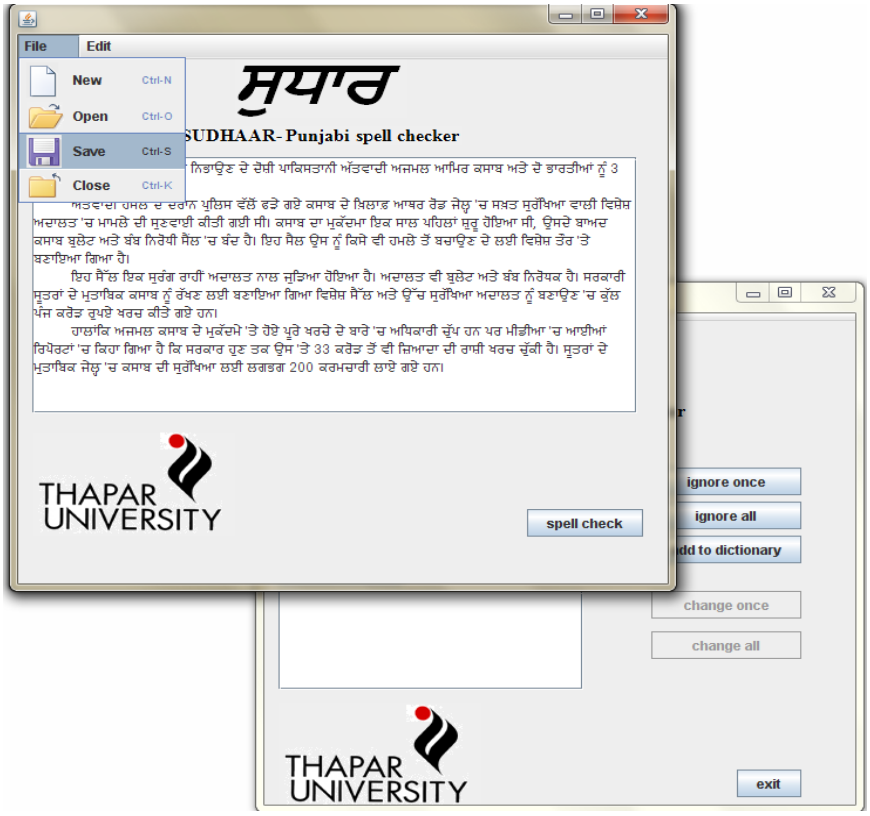

Figure 4.27: Snapshot of 'save' button

Figure 4.28: Snapshot of ‘close’ button

Chapter 5: Testing of the Developed System

We have tested the system by first giving the input from a well known Punjabi newspaper where our system has given zero errors because the words of that text were already entered into the dictionary. Then, we intentionally made hundred errors in the same text and when that text is given as an input, then it showed 75 errors and 15 words were valid words *i.e.* those words exits in the Punjabi dictionary and are not errors even after deletion of some laga or matra from the original word and 10 words were there which were not found by the Spell Checker as errors.

It showed the correct word in suggestion list for 68 words out of 75 that shows the accuracy of 90.6% for error correction and 87.2% for error detection. The table which contains the correct words, words in which errors are inserted, the Spell Checker result for error detection and correction is shown below in table 5.1.

Table 5.1: Testing of Developed System

CORRECT WORDS	INSERTED ERRORS	SPELL CHECKING RESULT	CW GIVEN AS SUGGESTION
ਨਾਨਕ	ਨਾਨ	Found	YES
ਰਿਹਾ	ਰਹਾ	Found	YES
ਬਿੱਟਾ	ਬਟਾ	Found	NO
ਹਮਲਾਵਰਾਂ	ਹਲਾਵਰਾਂ	Found	YES
ਵਿਵਸਥਾ	ਵਿਸਥਾ	Found	YES
ਕੋਠੀਆਂ	ਕਠੀਆਂ	Found	YES
ਲੋਕਾਂ	ਲੋਕ	Valid Word	-----
ਲੈਣਦੇਣ-	ਲੈਣਦੇਣ	Found	YES
ਨੌਜਵਾਨ	ਨੌਵਾਨ	Found	YES
ਸੋਨੂੰ	ਸਨੂੰ	Found	YES
ਨੇੜੇ	ਨੇੜ	Valid Word	-----
ਪ੍ਰਗਟਾਵਾ	ਪ੍ਰਟਾਵਾ	Found	YES
ਸਥਾਨਕ	ਸਥਾਕ	Found	YES
ਰਹੀ	ਰਹ	Not Found	-----
ਰਹੇ	ਹੇ	Not Found	-----
ਗੋਲੀਆਂ	ਗਲੀਆਂ	Valid Word	-----
ਦਿੱਤਾ	ਦਿੱਤ	Not Found	-----
ਦਿਨ-ਦਿਹਾੜੇ	ਦਿਨਦਿਹਾੜੇ	Found	YES

CORRECT WORDS	INSERTED ERRORS	SPELL CHECKING RESULT	CW GIVEN AS SUGGESTION
ਚਕਮਾ	ਚਮਾ	Found	YES
ਪਿਸਤੌਲ	ਪਿਤੌਲ	Found	YES
ਹੋਇਆ	ਹਇਆ	Found	YES
ਦੋਵਾਂ	ਦੋਵ	Found	NO
ਦੋਵੇਂ	ਦਵੇਂ	Found	YES
ਤਰ੍ਹਾਂ	ਤਰ੍ਹਾ	Valid Word	-----
ਦਿੰਦਿਆਂ	ਦਿੰਦਆਂ	Found	YES
ਫੇਲ੍ਹ	ਫਲ੍ਹ	Found	YES
ਜਾਣਕਾਰੀ	ਜਾਕਾਰੀ	Found	YES
ਅਤੇ	ਤੇ	Valid Word	-----
ਕਾਮਯਾਬ	ਕਾਯਾਬ	Found	YES
ਅਮਨ	ਅਨ	Found	YES
ਗੋਲੀ	ਗਲੀ	Valid Word	-----
ਇਸੇ	ਸੇ	Valid Word	-----
ਅੱਜ	ਅਜ	Not Found	-----
ਉਠਾ	ਉਠ	Valid Word	-----
ਪ੍ਰੇਸ਼ਾਨ	ਪ੍ਰੇਸ਼ਨ	Found	YES
ਉਕਤ	ਉਤ	Found	YES
ਉਥੇ	ਉਥ	Found	YES
ਉਰਫ	ਉਫ	Found	YES
ਊਧਮ	ਊਮ	Found	YES
ਹਥਿਆਰਬੰਦ	ਹਥਿਬੰਦ	Found	NO
ਗੁੰਡਾਗਰਦੀ	ਗੁੰਡਾਗਦੀ	Found	YES
ਮਨਿੰਦਰ	ਮਨਿੰਰ	Found	YES
ਨਿਕਲਿਆ	ਨਿਲਆ	Found	NO
ਨਾਕਾਰਾਤਮਿਕ	ਨਾਕਰਾਤਮਿਕ	Found	YES
ਮਨਿੰਦਰਜੀਤ	ਮਨਿੰਰਜੀਤ	Valid Word	-----
ਜਿਸ	ਜਿਸਬੰ	Found	NO
ਜਨਮ	ਜਨਮਬੰ	Found	NO
ਛੁਪ	ਛੁਪਫ	Found	YES
ਚਲਾ	ਚਲਾਫ	Found	YES
ਜ਼ਖ਼ਮੀ	ਜ਼ਮੀ	Found	YES
ਘਾਟ	ਘਾਟਖ਼	Found	YES

CORRECT WORDS	INSERTED ERRORS	SPELL CHECKING RESULT	CW GIVEN AS SUGGESTION
ਸ਼ੁਰੂ	ਸ਼ੁਰੂਖ਼	Found	YES
ਗਲੀ	ਗਲ	Not Found	-----
ਗਿਆ	ਗਆ	Not Found	-----
ਖਾਣ	ਖਣ	Found	YES
ਖੁਦ	ਖਦ	Found	YES
ਕਾਰ	ਕਰ	Not Found	-----
ਕੁਝ	ਕੁਝ਼ਖ਼	Found	YES
ਜਦੋਂ	ਦੋਂ	Found	YES
ਗਏ	ਗਏਗ	Found	YES
ਕਰਨ	ਕਰ	Valid Word	-----
ਦਾਖ਼ਲ	ਦਾਖ਼ਗਲ	Found	YES
ਦਹਿਸ਼ਤ	ਦਹਿਤ	Found	YES
ਬਾਹਰੋਂ	ਬਾਹਰ	Valid Word	-----
ਖੜ੍ਹੀ	ਖੜ੍ਹ	Found	YES
ਖੜ੍ਹਾ	ਖੜ੍ਹਾਗ	Not Found	-----
ਮਿੰਟ	ਮਿੰਟਗ	Found	YES
ਦੀਆਂ	ਦੀਆ	Valid Word	-----
ਭੀੜ	ਭੀੜਗ	Found	YES
ਭਾਈ	ਭਾਗ	Valid Word	-----
ਬੜਾ	ਬੜ	Found	YES
ਭਾਲ	ਭਲ	Found	YES
ਮੇਜ਼	ਮਜ਼	Found	YES
ਫਿਰ	ਫਰ	Found	YES
ਦੂਜੇ	ਦੂਜੇਗ	Found	YES
ਦੂਜਾ	ਦੂਜ	Found	YES
ਫੜਨ	ਫਨ	Found	YES
ਕੀਤਾ	ਕਤਾ	Found	YES
ਮਨਾਉਂਦਿਆਂ	ਮਨਾਉਦਿਆਂ	Found	YES
ਛਲਣੀ	ਛਲਣ	Found	YES
ਲਈ	ਲਈਗ	Found	YES
ਦੁਪਹਿਰ	ਦੁਪਹਰ	Found	YES
ਆਏ	ਆਏਗ	Found	YES
ਇਕ	ਇਕਗ	Found	YES

CORRECT WORDS	INSERTED ERRORS	SPELL CHECKING RESULT	CW GIVEN AS SUGGESTION
ਆਲ	ਆਲਗ	Found	YES
ਦੱਸਿਆ	ਦੱਸਆ	Found	YES
ਦਿੱਤਾ	ਦਿਤਾ	Valid Word	-----
ਦਿੱਤੀ	ਦਿਤ	Found	NO
ਬਲਵਿੰਦਰ	ਬਵਿੰਦਰ	Found	YES
ਦਿੱਤਾ	ਦਿੱਤ	Not Found	-----
ਦਿੱਤੀ	ਦਿੱਤਗ	Found	YES
ਨਗਰ	ਨਗ	Found	YES
ਦਾਗ	ਦਾਗਗ	Found	YES
ਦਿਨ	ਦਿਨਗ	Found	YES
ਨਾਚ	ਨਚ	Found	YES
ਤਰਨ	ਤਗਰਨ	Found	YES
ਇੰਡੀਆ	ਇੰਡਆ	Not Found	-----
ਉਤਾਰ	ਉਤਰ	Valid Word	-----

Then, we tested the system again by giving different text, all the words of which were not present in the dictionary, so the system showed approximately 10 words. Then again, we intentionally made hundred errors in the same text and when that text is given as an input, then it showed approximately 80 errors. It showed the correct word in suggestion list for approximately 70 words out of 80, that shows the accuracy of 85% for error correction and 80% for error detection.

Chapter 6: Conclusion and Future Scope

6.1 Conclusion

This Thesis presented a study on Gurmukhi, its origin and symbols used in it. The survey of various Spell Checkers available in different Indian languages and the techniques for error detection and correction was conducted. And various types of errors that can occur in a Punjabi text, with examples, were discussed based on which we have incorporated the following features in the Spell Checker:

- It contains a dictionary of around one million Punjabi words.
- It checks each word of the input text for their presence in dictionary for error detection.
- It gives the suggestions for detected errors and replaces the selected suggestion in input text.
- The user can append word in the dictionary.
- It is a standalone application capable of taking input from any source.
- It is open source software freely available to everyone.

The features, design, implementation, working and testing of the newly developed system 'SUDHAAR' is explained in this thesis. Presently, the suggestion list is for single error only. In this system, we have taken care of only non-real word errors. From the error analysis, it was gathered that the majority of single errors are due to substitution errors, followed by deletion, insertion and transposition errors. The system detects 87.2% of the errors and provides 90.2% of the correct suggestions.

6.2 Future Scope

In future, the developed Spell Checker for Gurmukhi Script can be enhanced by incorporating following change:

- In future it will be enhanced for multiple error word.
- Real word error detection and correction is a subject of Future research.
- This system can be used for other languages also but dictionary for other languages will have to be created for its use.

References

[1] Tanveer Siddiqui, U.S. Tiwary, (2008), "Natural Language Processing and Information Retrieval" Oxford university Press.

[2] Meenu Bhagat, (2007), "Spelling Error Pattern Analysis of Punjabi Typed Text", Thesis report, Thapar University, Patiala.

[3] E.M. Riseman and A. R. Hanson, "A Contextual Post Processing System for Error Correction using Binary N-grams", IEEE Transactions on Computer, pp. 480-493.

[4] P. Kundu and B.B. Chaudhuri (1999), "Error Pattern in Bangla Text", International Journal of Dravidian Linguistics, pp. 49-88.

[5] Robert A. Wagner (1974), "Order and Correction for Regular Languages", Communications of the ACM, pp. 265-268.

[6] F.J. Damerau (1964),"A Technique for Computer Detection and Correction of Spelling Errors". Communication ACM, pp. 171-176.

[7] Knuth, Donald E.(1973), "The Art of Computer Programming: Volume 3 Sorting and Searching" , Addison-Wesley, pp. 107-112.

[8] Monisha Das, S. Borgohain, Juli Gogoi, S. B. Nair (2002), "Design and Implementation of a Spell Checker for Assamese," lec, pp.156, Language Engineering Conference (LEC'02).

[9] B. B. Chaudhuri, "OCR Error Correction of an Inflectional Indian language using Morphological Parsing", TDIL Newsletter.

[10] Dr (Ms) Sanghamitra Mohanty , **"**Analysis and Design of Oriya Morphological Analyser : Some Tests with OriNet", TDIL Newsletter.

[11] Prof. Pushpak Bhattacharyya and Prof. Rushikesh Joshi, "Design and Implementation of a Morphology-based Spellchecker for Marathi", TDIL Newsletter.

[12] Davidson, Leon (1962), "Retrieval of Misspelled Names in an Airlines Passenger record System", Communications of the A.C.M, pp. 169-171.

[13] Dr. T.V. Geetha, Dr. Ranjani Parthasarathi, "Tamil Spell Checker", Resource Centre for Indian Language Technology Solutions, TDIL Newsletter.

[14] R. Ravindra Kumar; K.G.Sulochana, “Malayalam Spell Checker”, Resource Centre for Indian Language Technology Solutions, TDIL Newsletter.

[15] Dr. K. Narayanamoorthy, “Design and Implementation of AKSHARA”, Resource Centre for Indian Language Technology Solutions, TDIL Newsletter.

[16] Dr. R.K. Sharma, ”The Bilingual Punjabi English Spell Checker”, Resource Centre for Indian Languages Technology Solutions – Punjabi (RCILTS - Punjabi), TDIL Newsletter.

[17] V.I. Levenshtein (1966),"Binary codes capable of correcting deletions, insertions and reversals". Sov. Phys. Dokl. , pp. 707-710.

Research Publications

Research Papers Published

- Rupinderdeep Kaur, Parteek Bhatia," Types of Errors in Punjabi Text for the Efficient Development of Spell Checker for Gurmukhi", published in AICTE approved conference NCACCET, Chandigarh.(Jan 29th-30th, 2010).
- Rupinderdeep Kaur, Parteek Bhatia, "Techniques for Error Detection and Correction for the Efficient Development of Spell Checker for Gurmukhi", published in National Conference on Next Generation Computing, Gurgaon. (March 20th, 2010).

Research Paper Communicated

- Rupinderdeep Kaur, Parteek Bhatia," Design and Implementation of SUDHAAR-Punjabi Spell Checker", communicated in International Journal of Information and Telecommunication Technology, India. (May 15th, 2010).