Morphological Analyzer for Kannada Inflectional Words Using Hybrid Approach

Prathibha, R J
S J College of Engineering, Mysore
Karnataka
India
rjprathibha@gmail.com

Padma, M C
P E S College of Engineering, Mandya
Karnataka
India
padmapes@gmail.com

ABSTRACT: *Morphological analyzer is a tool which extracts morphological features and grammatical categories associated with the given inflectional word. Kannada is an inflectional and morphologically rich language. This paper presents the design of morphological analyzer for Kannada inflectional words using affix-stripping, rule-based, paradigm-based and questionnaire-based approaches. The proposed model comprises of text normalizer, morphological stemmer, morphological analyzer and a module for semi-automatic insertion of lexical details of unanalyzed words into Kannada monolingual lexicon. The performance of proposed model is tested on 7 different data sets that are manually created by collecting Kannada text from government official circular documents Kannada newspapers, legal documents and news from All India Radio. The experimental result shows that the accuracy of proposed model on these 7 different data sets is around 90%.*

Keywords: Affix-Stripping, Machine Translation System, Morphological Analyzer, Natural Language Processing, Paradigms, Text Normalizer

Received: 1 Sep 2016, Revised 28 Sep 2016, Accepted 9 October 2016

1. Introduction

Natural Language Processing (NLP) is a field of linguistics, computer science and artificial intelligence concerned with the interaction between human (natural) languages and machine. The field of NLP makes computers to perform useful tasks with the natural languages. Kannada is a natural language spoken predominantly by Kannada people in the state of Karnataka. Kannada is an inflectional, derivational and morphologically rich language. Inflectional or declinable words are formed by adding a set of

suffixes to the stem/root word. For example, in English language, some of the different inflectional words that are generated from verb-root "go" are shown below.

go + es = goes go + ing = going go + ne = gone

Here, the word "go" is the stem/root and "es", "ing" and "ne" are the suffixes. Non-inflectional or indeclinable word (*Avyaya*) is a word that cannot be inflected and remains in the same form for all genders, numbers and cases. For example, the words "therefore" (*AAddariNda*), "but" (*AAdare*), "and" (*maththu*) are non-inflectional words.

Morpheme is the smallest unit which carries meaning. For example, the word "studying" comprises of two morphemes, "study" and "ing". The word "study" is an individual unit which carries full meaning and it cannot be broken further into smaller unit of meaning, hence it is a morpheme. Morphology is the field of linguistics concerned with the internal structure of inflectional words. Morphological analyzer extracts the morphological features and grammatical categories associated with the given inflectional word. The morphological features of an inflectional word include root/stem, prefix and suffix. The grammatical categories of an inflectional word include case, gender, number, person, tense and so on.

Computational Morphology is an essential and basic tool required in most of the NLP applications like Machine Translation System (MTS), Named Entity Recognition (NER), spell checker, Information Retrieval System (IRS), Cross Language Information Retrieval (CLIR) system, discourse analysis, natural language generation, sentiment analysis and so on.

Several research works on morphological analysis have been carried out for non-Indian languages such as English, Chinese and European languages using various approaches. However, very less attempts have been made for Indian languages. The standard morphological analyzer developed for English language [18] is not suitable for Indian languages, because Indian languages are morphologically rich having high internal complexities. Hence there is a great demand for the development of morphological analyzer for Indian languages. Developing a well-fledged computational morphological analyzer for highly inflectional language like Kannada is a challenging task.

This paper proposes a model which comprises of text normalizer, morphological stemmer and morphological analyzer for Kannada inflectional words using hybrid approach. Due to unavailability of lexical details of words in the lexicon/dictionary, some input words may not be analyzed by the morphological analyzer. These words are called morphologically unanalyzed words. Lexical details of such unanalyzed words need to be inserted into the lexicon. In this context, this paper also proposes a module "semi-automatic insertion of lexical details of unanalyzed words into Kannada monolingual lexicon", to insert unanalyzed word along with its lexical details into the lexicon. In this paper, for better understanding, Kannada words are represented in English transliterated form.

The rest of paper is organized as follows. Section 2 presents the previous works carried out on design of text normalizer, stemmer and morphological analyzer for different Indian and non-Indian languages. The linguistic background of Kannada language is given in Section 3. The complete description of proposed work is explained in Section 4. The experimental results and discussions are presented in Section 5. Conclusions and future work are given in Section 6.

2. Previous Works

The literature shows that more works on text normalization have been done in the context of text-to-speech conversion systems and automatic speech recognition applications [5] and [21]. These text normalizers identify and transform the non-standard representation of words like abbreviations, numeric data, currency, dates, time, etc., in the given text into their spoken forms. For example, the number 2015 is transformed into "two thousand fifteen". Vishal Gupta developed a normalizer to standardize the variations of spellings in Punjabi language [26]. Gralinski et al., designed a text normalizer in machine translation system from Polish to English and Russian languages [3]. This text normalizer is used to expand the abbreviation in "bilingual" lexicon. An Indic normalizer [4] addresses the multiple representations for the same character of Indian language script. Hence, NLP applications can handle Indian language data in a consistent manner.

In the early 70s, stemmers were designed for English language. Lovins proposed the first stemmer for English language which uses longest-suffix-match approach [6]. An affix comprises of both prefix and suffix. Porter designed an algorithm for stemming, which removes the affixes that are present in inflectional and derivational words [7]. But, later due to growth of

corpus of Indian languages, there was a great demand for the development of stemmer for Indian languages. Most of the stemmers that are designed for Indian languages split the given inflected word into stem and suffixes using suffix stripping approach. These stemmers are suitable only for Information Retrieval System [9, 23, 28, 19, 22, 27, 29, 16].

Literature shows that, Morphological Analyzers (MA) have been developed for Indian languages. Menon et al. have proposed a Tamil Morphological Analyzer and Generator (MAG), which uses Finite State Transducer (FST) and rule- based methodologies [8]. Anand Kumar et al. have designed a sequence labeling MA for Tamil using support vector machine algorithm to handle transitive words, compound words and proper nouns [1]. Implementation of Malayalam MA based on hybrid approach proposed by Vinod et al. uses suffix-stripping and paradigm-based approaches [25]. FST based MA for Hindi, designed by Deepak Kumar et al. uses hand written FST rules to analyze both inflectional and non-inflectional words [2]. MA for Marathi using NLP proposed by Prathiksha Gawade et al. uses paradigm-based FST to analyze Marathi words [14]. The paradigms and rules are language dependents, because each language has its own linguistic rules and morphological structure.

Narayana Murthy proposed a network and process model for Kannada MAG as part of MTS from English to Kannada [10]. Vikram and Shalini have designed a prototype of MA for Kannada based on FST to handle nouns and verbs [24]. Ramasamy Veerappan et al. have identified 15 paradigms for verbs and developed a MAG for Kannada using rule-based FST [15]. Shambhavi et al. developed a Kannada MAG using rule-based with paradigm approach [17]. The *trie* data structure is used to store root words and suffixes. The performance of morphological analyzer depends on the size of root-dictionary and suffix list. The *trie* data structure consumes more memory. Hence, space complexity is high. Prathibha and Padma proposed a Kannada MA for finite verbs which uses hybrid approach [13]. The redundant verb suffixes are present in the verb-suffix table; hence the space complexity is high. Padma and Prathibha developed a MAG for Kannada nouns, which uses paradigm-based, rule-based and suffix stripping approaches [12]. All possible suffixes that are present in the given inflectional noun are provided by this model. From this obtained list of suffixes, it chooses the appropriate one. In this approach, number of iterations is more and hence time complexity is high.

In the existing morphological analyzers for Kannada language, no normalization of input text is carried out. However, it is necessary to design a text normalizer which tokenizes and categorizes the input text into two categories viz., i) Morphologically analyzable words: Tokens that require morphological analysis and ii) Morphologically unanalyzable words: Tokens like numbers, punctuations, abbreviations, symbols, and expressions etc., that do not require morphological analysis. In the literature, such normalizers are not reported. Hence there is a greater need for the design of text normalizer, as it is an important and basic component of morphological analyzer.

The existing stemmers are specifically designed for IRS to retrieve relevant documents based on the input query. Hence, identification and removal of prefixes that are present in the input inflectional words are not considered. These stemmers also do not provide information related to grammatical details such as lexical category, number, case, tense, person etc., of input inflectional word. In MTS, there is a need for extraction of such grammatical details of inflectional word during the stemming process itself; hence overhead on morphological analyzer is reduced.

It is also observed from the survey that majority of work on the design of morphological analyzer has been done for Tamil and Hindi languages; hardly few attempts have been made for Kannada language. Most of the approaches that are used in the design of MA are rule-based, paradigm-based and FST-based. The development of FST-based morphological analyzer is more tedious process because all linguistic rules have to be hard coded. For each individual, inflectional noun and verb form, FST should be written. The FST accepts input word, if and only if the word is contained within the transition. In the literature, no work is reported to address the words that are not analyzed by the morphological analyzer.

This paper presents the "design of morphological analyzer for Kannada inflectional words using hybrid approach" as a part of MTS. This model comprises of text normalizer, morphological stemmer, morphological analyzer and a semi- automatic update of lexical details of unanalyzed words into Kannada monolingual lexicon. The approaches adopted to develop the proposed model are: i) affix-stripping method, which is used in morphological stemmer module to strip off the prefixes and/or suffixes that are present in the given input inflectional word, ii) paradigm-based approach is applied while designing encoded-suffix table, iii) rule-based method is adopted during the creation of Kannada monolingual lexicon, and iv) questionnaire-based approach is used to insert the lexical details of words that are not analyzed by the morphological analyzer into the lexicon.

3. Linguistic Background of Kannada

Kannada is an official and administrative language of Karnataka, a state in South India. Kannada is one of the four major literary languages of the Dravidian family.

3.1 Grammatical categories

The grammatical categories observed in Kannada include nouns, pronouns, verbs, adverbs, adjectives, postpositions, conjunctions and interjections. The nouns, pronouns, adjectives and verbs are inflectional words, and adverbs, postpositions, conjunctions and interjections are non-inflectional words.

3.1.1 Noun and pronoun morphology

In Kannada, nouns and pronouns get inflect for the following grammatical properties:

1. Case: Nominative, accusative, genitive, dative, locative, instrumental/ablative and vocative.
2. Number: Singular and plural.
3. Gender: Masculine, feminine and neuter.

Example: The noun "*huDugarannu*" is an inflected word for accusative case, plural and masculine gender.

List of some suffixes that are added to noun-roots for the generation of inflectional nouns in different cases, numbers and genders are shown in Table 1.

Case *(Vibhakthi)*	Suffixes	
	Singular	Plural
Nominative *(pratHama)*	*u/nu/Lu/vu/yu*	*ru/vu/gaLu/yaNdiru/vugaLu*
Accusative *(dvithiya)*	*annu/nannu/Lannu/vannu/yannu*	*rannu/aNdirannu/gaLannu/yarannu*
Ablative/ Instrumental *(thruthiya)*	*niNda/diNda/yiNda/viniNda/LiNda*	*gaLiNda/riNda/yariNda*
Dative *(chathurtHi)*	*ge/nige/Lige/vige/yige/kke*	*gaLige/rige/ge/yarige*
Genetive *(shasHti)*	*ya/ina/ara/La/na/da/vina*	*ra/gaLa/nNdira/yara*
Locative *(sapthami)*	*nalli/yalli/Lalli/dalli/vinalli*	*ralli/yaralli/gaLalli/Ndiralli*
Vocative *(saMbhood-Hana)*	*e/vee/a/i*	*gaLe/Ndire/yare/yaNdire/vugaLe*

Table 1. Cases and their Suffixes for Kannada Noun Inflectional Forms

3.1.2 Verb morphology

Verb is much complex than nouns, because all verb roots would not generate the same inflectional forms. Each verb- root concatenates with suffixes to indicate tenses, moods and aspects. Verb-roots are inflected into distinct finite verb forms in past tense, compare to present tense and future tense forms.

Example: The verb-roots "*thinnu*" and "*nagu*" generate inflectional forms for present tense and future tense by adding the same suffixes. But these two verb-roots use distinct suffixes to generate inflectional verbs for past tense. List of suffixes that are

concatenated with these two verb-roots to generate inflectional forms for present tense are shown below.

"*ththeene", "ththeeve", "ththiiye", "ththaane", "ththaaLe", "ththaare", "ththade", "ththave".*

The inflectional forms of verb-root "thinnu" and "nagu" in present tense are given below.

"thinnuththeene", "thinnuththeeve", "thinnuththiiye", "thinnuththaane" ,"thinnuththaaLe", "thinnuththaare", "thinnuththade, "thinnuththave".
"naguhtheene", "naguththeeve", "naguththiiye", "naguththaane", "naguththaaLe", "naguththaare", "naguththade", "naguththave".

List of suffixes that are concatenated with these two verb-roots to generate inflectional forms for future tense are shown below.

"venu", " vevu", " ve", " viri", " vanu", " vaLu", " varu", " vudu", " vuvu".

The inflectional forms of verb-root "thinnu" and "nagu" in future tense are as follows:

"thinnuvenu","thinnuvevu", "thinnuve", "thinnuviri", "thinnuvanu", "thinnuvaLu", "thinnuvaru", "thinnuvudu", "thinnuvuvu".
"naguvenu", " naguvevu", " naguve", " naguviri", " naguvanu", " naguvaLu", " naguvaru", " naguvudu", " naguvuvu".

The inflectional forms of verb-roots "*thinnu*" and "*nagu*" in past tense concatenate with different suffixes. List of suffixes that are concatenated with verb-root "*thinnu*" to generate inflectional forms for past tense are shown below.

"Ndenu"," Ndevu"," Nde"," Ndiri"," Ndanu"," Ndaru"," NdaLu"," Ndithu"," Ndavu".

The inflectional forms of verb-root "thinnu" in past tense are as follows.

"thiNdenu"," thiNdevu"," thiNde"," thiNdiri"," thiNdanu"," thiNdaru"," thiNdaLu"," thiNdithu"," thiNdavu".

List of suffixes that are concatenated with verb-root "*nagu*" to generate inflectional forms for past tense are shown below.
"kkenu"," kkevu"," kke"," kkevu"," kkanu"," kkaru"," kkaLu"," kkithu"," kkavu".

The inflectional forms of verb-root "*nagu*" in past tense are as follows.

"nakkenu"," nakkevu"," nakke"," nakkevu"," nakkanu"," nakkaru"," nakkaLu"," nakkithu"," nakkavu".

It is observed from the above examples that, inflectional forms of verb-roots "*thinnu*" and "*nagu*" in present tense and future tense concatenate with the same suffixes. But for verb inflections in past tense, these verb-roots concatenate with different suffixes. The verb morphology is complex than noun morphology and hence requires more paradigms.

As per Board of Indian Standards (BIS), Kannada verbs are classified into main verb and auxiliary verb. The main verbs are further classified into finite, infinite, non-finite, gerund and participle noun. On the other hand, auxiliary verbs are classified into finite, infinite and non-finite. The main-finite verbs are marked with person-number-gender (PNG) markers. The general syntax of finite verb is - "root + tense marker + PNG marker". Finite verbs inflect for the following grammatical properties:

1. Person: First, second and third
2. Number: Singular and plural
3. Gender: Masculine, feminine and neuter
4. Tense: Past, present and future

Example: The verb "*baredaLu*" is inflected for past tense, third person, singular and feminine gender.

List of some suffixes that are added to verb-roots for formation of inflectional finite-verbs and details of Kannada PNG markers are shown in Table 2.

Person	Number	Gender	Suffixes		
			Present	Past	Future
First					
	Singular	M	thteene, yuththeene,	Ndenu kkenu, ddenu	venu, yuvenu
		F	thteene,	Ndenu,	venu,
		N	Yuththeene,	kkenu, ddenu	yuvenu
			-	-	-
	Plural	M	ththeeve, yuththeeve	Ndevu, kkevu, ddevu	vevu, yuvevu
		F	ththeeve, yuththeeve	Ndevu, kkevu, ddevu	vevu, yuvevu
		N	-	-	-
Second					
	Singular	M	ththiiye, yuththiiye	Nde, kke, dde	ve, yuVe
		F	ththiiye,	Nde,	ve,
		N	-	-	-
	Plural	M	ththiiri, yuththiiri	Ndiri, kkiri, ddiri	viri, yuviri
		F	ththiiri, yuththiiri	Ndiri, kkiri, ddiri	viri, yuviri
		N	-	-	-
Third					
	Singular	M	ththaane, yuththaane	Ndanu, kkanu, ddaru	vanu, yuvanu
		F	ththaaLe, yuththaaLe	NdaLu, kkaLu, ddaLu	vaLu, yuvaLu
		N	ththaDe, yuththaDe,	Ndithu kkithu, ddithu	vudu yuvudu
	Plural	M	ththaare, yuththaare	Ndaru, kkaru, ddaru	varu, yuvaru
		F	ththaare, yuththaare	Ndaru, kkaru, ddaru	varu, yuvaru
		N	ththave,	Ndavu,	vuvu,

Table 2. Kannada Suffixes And Their PNG Markers For Inflectional Finite Verbs

3.1.3 Adjective morphology

In Kannada, adjectives are generated, when genitive case-marker suffixes are attached to nouns. These forms are already covered in noun morphology.

Example: The word *"budhdhivaNthana"* is an adjective inflected from noun-root *"budhdhivaNtha"* for genitive case, singular and masculine gender.

3.1.4 Adverb morphology

The adverbs are non-inflectional words formed by attaching postpositions to nouns, verbs and pronouns.

Example: The words like *"thuMbaa", "bahaLa", "hechchu"* etc.,.

3.1.5 Conjunction and Interjection

All conjunction and interjection words are non-inflectional words. Example: *"AtHavaa", "maththu", "AAdare"* etc.,

4. Proposed Model

The proposed model has four modules viz., i) text normalizer, ii) morphological stemmer, iii) morphological analyzer, and iv) semi-automatic insertion of lexical details of unanalyzed word into Kannada monolingual lexicon. Kannada monolingual lexicon is a dictionary which is created specifically for proposed work by manually collecting data from a well known Kannada dictionary *"Kannada Rathan Kosha"* [11]. This lexicon contains Kannada noun-root/verb- root / indeclinable and their lexical details. The architecture of proposed model is shown in Figure 1.

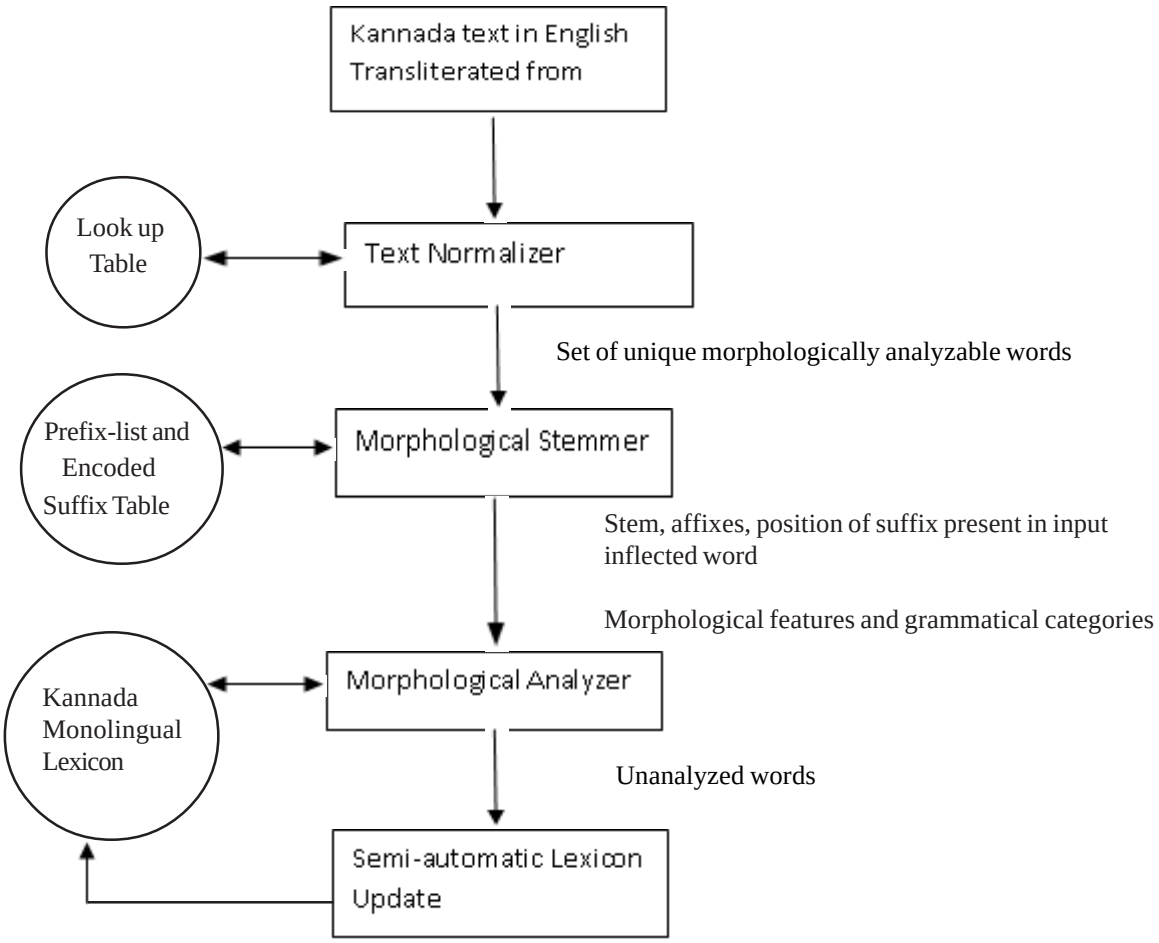

Figure 1. Architecture of The Proposed Model

4.1 Functionality of each module in proposed model

4.1.1 Text normalizer

Text normalization is a complex process, application dependent and often the very first phase in most of NLP applications. In morphological analysis, text normalizer is a pre-processing tool which tokenizes the given text into tokens then identifies and categorizes these tokens into morphologically analyzable (inflectional and non-inflectional) and unanalyzable (numbers,

punctuations, abbreviations, etc.,) words. The words that require no morphological analysis are called morphologically unanalyzable words and are handled by the text normalizer using look-up table and regular expressions. The words that require morphological analysis are handled by morphological stemmer and analyzer. The redundant words that are present in morphologically analyzable category are removed; hence text normalizer is also called a data cleaning module. The output of this module is a set of unique and morphologically analyzable words.

4.1.2 Morphological stemmer

Existing stemmers remove affixes that are present in inflectional word and return the stem/root. These stemmers are suitable for information retrieval systems. However, the proposed morphological stemmer splits input inflectional word into prefix, stem and suffix and also gives details of suffix; later this information is used by morphological analyzer to derive grammatical categories of input inflected word.

4.1.3 Morphological analyzer

The morphological analyzer searches for split stem in Kannada monolingual lexicon. If the given stem is available in lexicon, based on its lexical category-code, grammatical and morphological features of input word are derived. Otherwise, the input word is treated as an unanalyzed word.

4.1.4 Semi-automatic insertion of lexical details of unanalyzed word into Kannada monolingual lexicon

The words that are not analyzed by morphological analyzer due to unavailability of lexical details in lexicon are addressed by this module. The lexical details are like category, gender, paradigm-class to which given word belongs. This module inserts the unanalyzed words along with their relevant lexical details into lexicon.

4.2 Databases created for the proposed model

The proposed model has three databases viz., i) encoded-suffix table, ii) Kannada monolingual lexicon and iii) look- up table. To construct encoded-suffix table, it is necessary to design and classify Kannada noun and verb suffixes/paradigms based on their inflectional forms.

4.2.1 Design and classification of noun and verb suffixes using paradigm-based approach

Paradigms or suffixes are the conjugations of inflectional words. The word-forms are arranged by classifying words according to inflectional categories such as case, number and gender for nouns; tense, person, number and gender for finite verbs. The design of paradigms covers all types of inflections. Biggest challenge is grouping of inflectional words in such a way that members of the same group have similar inflections.

In Kannada, a noun-root can end with letter "*a*" (*a kaaraaNtha*) or "*i*" (*i kaaraaNtha*) or "*u*" (*u kaaraaNtha*) or "*e*" (*e kaaraaNtha*) or "*ai*" (*ai kaaraaNtha*) and it may belong to masculine (M) or feminine (F) or neuter (N) gender. In proposed work, based on these two parameters - ending (*aNtha*) and gender (*liNga*), nouns are categorized into 12 paradigm-classes and are represented from 01 to 12 paradigm-classes. In each paradigm-class, 14 suffix patterns (P1 to P14) are identified and are added to the noun-root to inflect different noun forms with 7 cases and 2 numbers. The details of paradigm-classes and their suffix patterns for nouns are shown in Table 3.

The notations used for noun-suffix patterns (P1-P14) in Table 3 are given below:

P1 - Nominative case and singular.

P2 - Nominative case and plural.

P3 - Accusative case and singular.

P4 - Accusative case and plural.

P5 - Genitive case and singular.

P6 - Genitive case and plural.

P7 - Dative case and singular.

P8 - Dative case and plural.

P9 - Locative case and singular.

P10 - Locative case and plural.

P11 - Instrumental/ablative case and singular.

P12 - Instrumental/ablative case and plural.

P13 - Vocative case and singular.

P14 - Vocative case and plural.

Paradigm Class	Example	ANtha (Ending)	LiNga (Gender)	P1	P2	P3	P4	::	P14
01	huDuga	a	M	nu	ru	nannu	rannu	::	re
02	ANNa	a	M	nu	Ndiru	nannu	Ndirannu	::	Ndire
03	kamala	a	F	Lu	ru	Lannu	rannu	::	re
04	Akka	a	F	Lu	Ndiru	Lannu	Ndirannu	::	Ndire
05	mara	a	N	vu	gaLu	vannu	gaLannu	::	gale
06	chikka	a	N	du	vu	dannu	vugaLannu	::	vugaLe
07	kavi /gudi /Ele /kai	i/e/ai	M/N	yu	gaLu	yannu	gaLannu	::	gale
08	gouri /mahiLe	i/e	F	yu	yaru	yannu	Yarannu	::	yare
09	taayi	i	F	yu	aNdiru	yannu	aNdirannu	::	aNdire
10	guru /vadhu /shathru	u/ru	M/F/N	vu	gaLu	vannu	gaLannu	::	gale
11	heNNu /haNNu	u	F/N	u	ugaLu	annu	ugaLannu	::	ugaLe
12	thaNde	e	M/F	yu	yaNdiru	yannu	yaNdirannu	::	yandire

Table 3. Details of Paradigm-Classes with their Suffix Patterns For Kannada Nouns with Examples

For example, suffix pattern P1 of paradigm-class 01 (*'nu'*) is added to the noun-root (*'huDuga'*) to inflect noun with nominative case and singular form (*'huDuganu'*); suffix pattern P2 of paradigm-class 01 (*'ru'*) is added to the noun-root (*'huDuga'*) to inflect noun with nominative case and plural form (*'huDugaru'*); suffix pattern P3 of paradigm-class 01 ('nannu') is added to the noun-root (*'huDuga'*) to inflect noun with accusative case and singular form (*'huDuganannu'*), and so on.

In Table 3, paradigm-classes 01 and 02 belong to "*a kaaraaNtha*" and masculine-gender, but suffixes in plural forms are different, and thus two separate paradigm-classes (01 and 02) for noun-roots ending with "*a kaaraaNtha*" and masculine-gender. Paradigm-classes 03 and 04 belong to "*a kaaraaNtha*" and feminine-gender, but the suffixes are different in many

cases and numbers, and thus two separate paradigm-classes (03 and 04) for noun-roots ending with "*a kaaraaNtha*" and feminine-gender. However, any given Kannada inflected noun certainly belongs to any one of these 12 paradigm-classes. Nouns ending with same letter and gender need not belong to the same paradigm-class. For example, the noun-roots *'huDuga'* (boy) and *'ANNa'* (brother) are of same gender (masculine) with same ending (*a kaaraaNtha*), belong to paradigm-class 01 and paradigm-class 02 respectively.

The finite verbs are inflected for 3 persons (first, second and third), 2 numbers (singular and plural), 3 genders (masculine, feminine and neuter) and 3 tenses (present, past and future). A verb-root can end with letter *'i' (i kaaraaNtha*) or *'u'* (*u kaaraaNtha*) or *'e'* (*e kaaraaNtha*). For example, the verb-roots "*kaDi*", "*thinnu*" and "*kare*" belong to *'i' kaaraaNtha*, *'u' kaaraaNtha* and *'e' kaaraaNtha* respectively.

The notations that are used to represent tenses are: R for present tense, S for past tense and F for future tense. The finite verbs are identified and categorized into 15 paradigm-classes, out of which 2 paradigm-classes represent present tenses (1R and 2R), 2 paradigm-classes represent future tenses (1F and 2F) and remaining 11 paradigm-classes represent past tenses (1S and 11S). Each paradigm-class has 10 suffix patterns (P1 to P10) that are associated with person-number-gender (PNG) markers. These patterns are conjugated with verb-roots to inflect finite verbs. There are three characters in each pattern, the first character represents person (First, Second and Third), second character signifies number (Singular and Plural) and third character indicates gender (Masculine, Feminine and Neuter). For example, the pattern P1 is associated with TSM (Third person, Singular, Masculine gender) as PNG marker. For example, suffix pattern P1 (*'ththaane'*) is added with verb-root (*'thinnu'*), the inflected finite verb (*'thinnuththaane'*) belongs to TSM (Third person, Singular and Masculine). The details of 15 paradigm-classes and 10 patterns for each paradigm-class of finite verbs are shown in Table 4. The notations used for verb-suffix patterns in Table 4 are given below.

Paradigm class	**Example**	**P1 TSM**	**P2 TPM**	**P3 TSF**	**P4 TPF**	**P5 TSN**	:::	**P10 FPA**
1R	thinnu	ththaane	ththaare	ththaaLe	ththaare	ththade	:::	ththeeve
2R	kuDi/ kare	yuththaane	yuththaare	yuththaaLe	yuththaare	yuththade	:::	yuththeeve
1F	kaaDu	vanu	varu	vaLu	varu	vudu	:::	vevu
2F	hari/ thoLe	yuvanu	yuvaru	yuvaLu	yuvaru	yuvudu	:::	danu
1S	thinnu	Ndanu	Ndaru	NdaLu	Ndaru	Ndithu	:::	yuvevu
2S	kaaNu	NDanu	NDaru	NDaLu	NDaru	NDithu	:::	NDevu
3S	nagu	kkanu	kkaru	kkaLu	kkaru	kkithu	:::	kkevu
4S	ALu	ththanu	ththaru	ththaLu	ththaru	ththithu	:::	ththevu
5S	IDu	ttanu	ttaru	ttaLu	ttaru	ttithu	:::	ttevu
6S	biiLu	ddanu	ddaru	ddaLu	ddaru	ddithu	:::	ddevu
7S	haaDu	idanu	idaru	idaLu	idaru	ithu	:::	idevu
8S	baagu	ggidanu	ggidaru	ggidaLu	ggidaru	ggithu	:::	ggidevu
9S	bare	danu	daru	daLu	daru	yithu	:::	devu
10S	nillu	Nthanu	Ntharu	NthaLu	Ntharu	Nthithu	:::	Nthevu
11S	kooru	thanu	tharu	thaLu	tharu	thithu	:::	thevu

Table 4. Details of Paradigm Classes for Kannada Finite Verbs

P1 - TSM - Third person, Singular, Masculine gender.

P2 - TPM - Third person, Plural, Masculine gender.

P3 - TSF- Third person, Singular, Feminine gender.

P4 - TPF - Third person, Plural, Masculine gender.

P5 - TSN - Third person, Singular, Neutral gender.

P6 - TPN - Third person, Plural, Neutral gender.

P7 - SSA - Second person, Singular, All gender.

P8 - SPA - Second person, Plural, All gender.

P9 - FSA - First person, Singular, All gender.

P10- FPA - First person, Plural, All gender.

In the proposed work, 12 paradigm-classes for nouns with 14 suffix patterns and 15 paradigm-classes with 10 suffix patterns for finite verbs are identified. From Table 3 and Table 4, it is observed that some suffix patterns are repeated in different cases, numbers and paradigm-classes. For example, in Table 3, noun-suffix pattern P1 ("*yu*") is repeated in paradigm-classes 07, 08, 09 and 12. In Table 4, verb-suffix "*ththaare*" is repeated in the same paradigm-class "1R" as patterns P2 and P4. In order to avoid such occurrence of redundant suffix patterns, an encoded-suffix table is constructed. The construction of encoded-suffix table is explained in next section.

4.2.2 Creation of encoded-suffix table

The constructed encoded-suffix table contains list of suffixes (both nouns and verbs) and their lexical features in encoded form. Lexical features are: set of paradigm-class numbers to which the suffix belongs and position of suffix in noun and verb paradigm classes (Table 3 and Table 4). For example, the suffix "*gaLannu*" is appeared in noun paradigm-classes 05, 07 and 10 at 04th position (Table 3), hence encoded value for suffix "*gaLannu*" is 05 07 10 @ 04. The position of suffix is used to derive case and number for noun, and PNG for finite verb in morphological analyzer module.

The noun paradigm table contains 168 suffixes (12 rows x 14 columns in Table 3) and finite-verb paradigm table contains 150 suffixes (15 rows x 10 columns in Table 4), hence there are 318 suffixes. However, encoded-suffix table contains 112 suffixes for nouns and 148 suffixes for finite verbs, thus 58 redundant suffixes are eliminated. In morphological stemmer module, longest affix removal approach is used in strip off the suffixes that are present in input inflectional words. Hence suffixes in the encoded-suffix table are stored in descending order based on their length. Few entries of the encoded-suffix table are shown in Table 5.

Suffixes	Encoded values
gaLannu	05 07 10 @ 04
gaLiNda	05 07 10 11 @ 06
daralli	06 @ 11
dariNda	06 @ 05
denu	9S @ 9
devu	9S @ 10
Nthe	10S @ 7
yuve	11S @ 7

Table 5. Encoded-Suffix Table

4.2.3 Creation of Kannada monolingual lexicon using rule-based approach

Kannada monolingual lexicon is a dictionary which is specifically created for this proposed work. This dictionary contains root or base form of words and their lexical details in English transliterated form. The lexical details of noun are gender and paradigm-class to which it belongs. For finite verb, the lexical details are paradigm-class and modifier-code. All inflectional forms of both nouns and finite verbs are generated with the help of encoded-suffix table and Kannada monolingual lexicon. Hence, no need to store all inflectional forms of each noun-root (14 forms) and verb-root (54 forms) in the lexicon. This reduces the size of lexicon extensively. Kannada monolingual lexicon contains two columns viz., i) noun/verb root or non-inflectional word and ii) category-code, which specifies the lexical category of noun/verb root or non-inflectional word. For nouns, first character of the category-code represents gender and next two consecutive characters represent paradigm-class to which noun belongs.

Rule-based approach is used to build the lexicon. For nouns, based on ending character of the noun-root, gender and its paradigm-class (Table 3), 23 rules are framed. Few rules are given below.

Rule 1: If (root-word ends with "*a*" (*a kaaraantha*), belongs to masculine-gender and uses the suffixes of paradigm- class 01 to generate its inflectional forms) then its category-code is "M01".

For example, "*huDuga*" is a noun-root ended with "*a*", belongs to masculine gender and uses the suffix patterns of paradigm-class 01 to generate its inflectional forms, and hence its category-code is "M01".

Rule 2: If (root-word ends with "*a*" (*a kaaraantha*), belongs to masculine-gender and uses the suffixes of paradigm- class 02 to generate its inflectional forms) then its category-code is "M02".

For example, "*ANNa*" is a noun-root ended with "*a*", belongs to masculine gender and uses the suffix patterns of paradigm-class 02 to generate its inflectional forms, and hence its category-code is "M02".

Rule 3: If (root-word ends with "*i*" (*i kaaraantha*), belongs to feminine-gender and uses the suffixes of paradigm- class 08 to generate its inflectional forms) then its category-code is "F03".

For example, "*huDugi*" is a noun-root ended with "*i*", belongs to feminine-gender and uses the suffix patterns of paradigm-class 03 to generate its inflectional forms, and hence its category-code is "F08".

Rule 4: If (root-word ends with "*u*" (*u kaaraantha*), belongs to neuter-gender and uses the suffixes of paradigm- class 10 to generate its inflectional forms) then its category-code is "N10".

For example, "*haNNu*" is a noun-root ended with "*i*", belongs to neuter-gender and uses the suffix patterns of paradigm- class 10 to generate its inflectional forms, and hence its category-code is "N10".

For verbs, first two characters of category-code represent paradigm-class to which it belongs and last character is a modifier-number. Modifier-number indicates the number of characters to be removed from verb-root to get stem for past tense verb inflections. For finite verbs, based on paradigm-class and modifier-number (Table 4), 14 paradigm-classes (1S2, 1S3, 2S0, 2S3, 3S2, 4S2, 4S3, 5S2, 6S2, 6S3, 7S1, 8S3, 9S0, 9S1, 9S2) are obtained to generate past tense form of the root. For example, verb-root "*thinnu*" belongs to paradigm-class "1S3". Here 3 is the modifier-number. This number indicates that last three characters of the verb-root to be stripped off. Hence remaining characters "*thi*" is the stem from which verb inflections for past tenses are generated. Past tense forms of the verb-root "*thinnu*" with "*thi*" as stem are "*thiNdanu*", "*thiNdaLu*", "*thiNdithu*", and so on.

Non-inflectional words require no lexical information, and hence their lexical category-code is "A" (*Avyava*). Currently the lexicon consists of 3500 root words and 176 non-inflectional words with their lexical features. The root words are randomly selected from a well known Kannada dictionary called "*Kannada Rathna Kosha*" [11]. Few entries of Kannada monolingual lexicon are given in Table 6. The notations that are used in Kannada monolingual lexicon are; M - Masculine, F - Feminine, N - Neuter, A - Avyaya and S - Past tense.

4.2.4 Creation of look-up table

The punctuation marks, abbreviations and acronyms that are present in input text are handled by text normalizer using look-up table. The contents of look-up table are punctuation marks, abbreviations and acronyms of Kannada language with their relevant tags PUNCT, ABBRV and ACRON respectively. These details are manually entered and stored in look-up table.

Some of entries of look-up table are shown in Table 7.

Root/Word	Lexical Category Code
huDuga	M01
huDugi	F08
haNNu	N10
naNthara	A
thinnu	1S3
kuNi	9S0
kare	9S0

Table 6. Kannada Monolingual Lexicon

Punctuation marks/abbreviations /acronyms	Tag
?	PUNCT
pu. thi. no	ABBRV
ka. raa. ra. saa. ni	ACRON

Table 7. Look-up Table

4.3 Methodologies Used

Methodologies used to develop the four modules of proposed model are explained below.

4.3.1 Text normalizer

In text normalization process, tokenizer plays an important role. In general, tokenizers split given text into tokens by considering space as the delimiter. However, in text document, punctuation marks, symbols and numbers are embedded with text without space. For example, "wow! It's 25th birthday of my son". In such case, a special tokenizer is essential which considers space, numbers, symbols and punctuation marks as delimiters. In this proposed work, an existing Indic tokenizer from Indic NLP library [4] is used to tokenize given input text into tokens by considering spaces, newline, tabs, numbers, symbols and punctuation marks as delimiters.

The output of Indic tokenizer is a set of tokens containing words, numbers, punctuation marks, symbols, expressions and so on. Further these tokens are identified and classified into two categories; i) Tokens having morphemes, morphological features and grammatical categories associated with them. These tokens require morphological analysis. ii) Tokens like punctuation marks, expressions, acronyms, abbreviations, numerical data (numbers, date, time, Internet protocol address etc.,) that do not have morphemes. These tokens require no morphological analysis. In text normalization, these tokens are handled by assigning their relevant tags using look-up table and regular expressions. The redundant words that are present in list of morphologically analyzable category are removed, thus size of input text for morphological stemmer is reduced. Hence, text normalizer is also called a preprocessing and data cleaning module. The output of text normalizer is a set of unique words that require morphological analysis. The block diagram of text normalizer is shown in Figure 2.

Example: for input sentence: *"Aha! 7 suththina mysuru-malligeyu suNdaravaada parimaLavannu niiDuththade"*, the general tokenizer split it into 7 tokens by considering space as delimiter. These tokens are listed below.

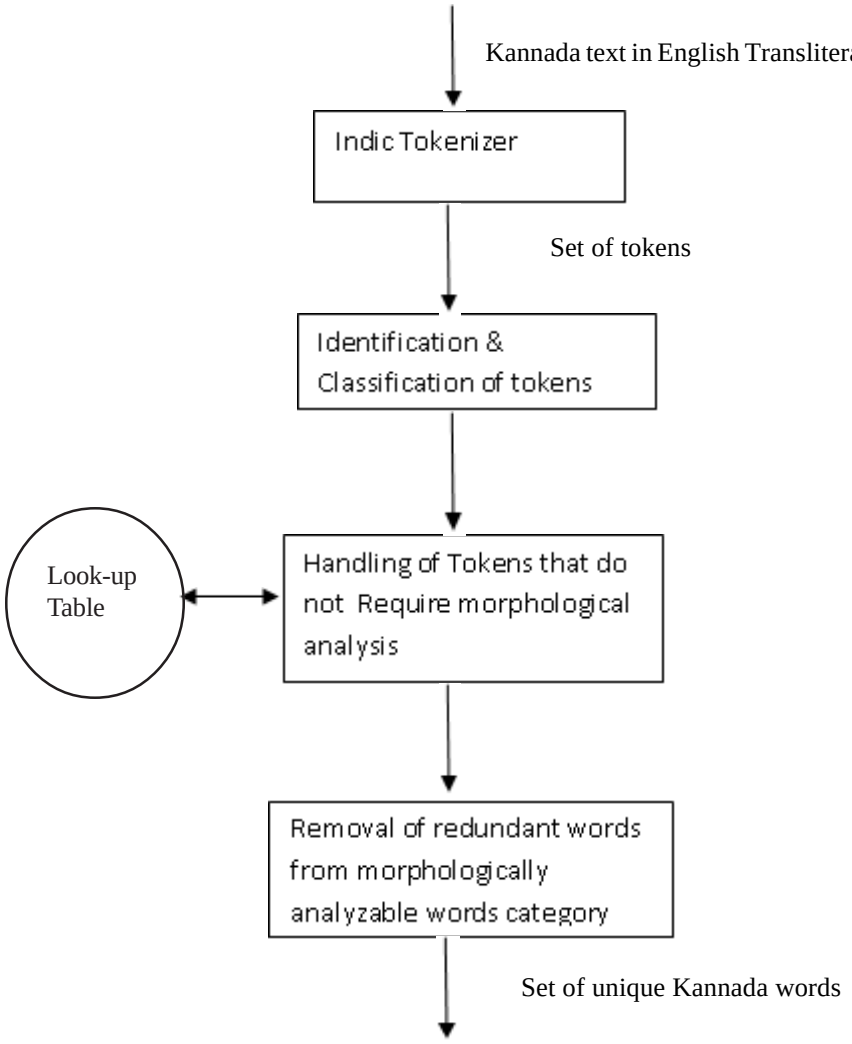

Figure 2. Block Diagram of Text Normalizer

Token 1: "*Aha*!

Token 2: 7

Token 3: *suththina*

Token 4: *mysuru-malligeyu*

Token 5: *suNdaravaada*

Token 6: *parimaLavannu*

Token 7: *niiDuththade*."

However, the Indic tokenizer split given sentence into 13 tokens and these tokens are given below. Token 1: "

Token 2: *Aha*

Token 3: !

Token 4: 7

Token 5: *suththina*

Token 6: *mysuru*

Token 7: -

Token 8: *malligeyu*

Token 9: *suNdaravaada*

Token 10: *parimaLavannu*

Token 11: *niiDuththade*

Token 12: .

Token 13: “

The proposed text normalizer classifies these 13 tokens into two categories; i) Tokens*: Aha, suththina, mysuru, malligeyu, suNdaravaada, parimaLavannu, niiDuththade* as morphologically analyzable words and ii) Tokens: “, !, 7, - , . “ as morphologically unanalyzable words. These morphologically unanalyzable words are handled by assigning their relevant tags using look-up table and regular expressions. Morphologically analyzable words are handled by morphological stemmer and analyzer. These modules are discussed in following sections.

4.3.2 Morphological stemmer

A set of morphologically analyzable words from the output of text normalizer are given as input to Morphological Stemmer (MS) module. This module uses affix-stripping approach to split the given inflectional word into prefix, stem and suffix. Some of the prefixes available in Kannada language are: *“pra, “paraa”, “Apa”, “sam”, “Ava”, “nis”, “nir”, “dus”, “Abhi”, “prathi”, “pari”, “upa”, “A”, “Adhi”, “Athi”, “uth”, “su”, “dur”, “Anu”, “Athi”, “ni”, “ku”*. Initially, morphological stemmer module searches for presence of affixes in the given input word using prefix-list and encoded-suffix table (Table 5). If affixes are present, then it extracts stem from input word by stripping off the affixes. The extracted stem need not be linguistically meaningful. The MS module returns paradigm-classes and position of suffix from encoded- suffix table. If no affixes are present in the input word then it is considered as base/root/indeclinable word. The position of suffix is used to derive grammatical features of input word in morphological analyzer module. Hence this module is called a morphological stemmer rather a stemmer. According to literature survey, the existing stemmers do not provide any information about suffixes that are present in input inflectional word, because, these stemmers are specifically designed for information retrieval system.

4.3.3 Morphological analyzer

The output of morphological stemmer (noun/verb, input word, prefix, stem, noun/verb suffix, paradigm-classes and position of suffix in encoded-suffix table) is given as input to morphological analyzer which extracts remaining morphological and grammatical features of inflected input word. Initially this module searches for input stem in lexicon (Table 6). If the stem is found, then gets its corresponding lexical category-code. If the lexical category-code is “A”, then input word is treated as a non-inflectional word and displays “*Avyaya*” as its lexical category. Otherwise, the input word is treated as an inflectional word and its paradigm-class, gender and/or modifier-number are extracted from lexical category-code. If the input paradigm-class and extracted paradigm-class are same, then input word is considered as a valid inflectional word. If the input word belongs to noun category then number and case of input word are derived from position of suffix and details of input word - “NOUN, prefix, stem, suffix, gender, case and number” are displayed. If the input word belongs to verb category then person, number, gender and tense of input word is derived from position and paradigm-class of suffix. The details of finite verb - “VERB, word, stem, suffix, person, number, gender and tense” are displayed. Otherwise the input word is considered as an unanalyzed word.

4.3.4 Semi-automatic insertion of lexical details of unanalyzed words into Kannada monolingual lexicon

Initially, it is not possible to build a lexicon that can cover all root/base words of a language. Due to unavailability of lexical details of input words in the lexicon, morphological analyzer is not able to analyze such words. These words are called unanalyzed words. However, it is necessary to append lexical details of unanalyzed words into lexicon as and when they encountered in the input text. This issue is handled by the proposed module “semi-automatic insertion of lexical details of unanalyzed words into Kannada monolingual lexicon” using questionnaire-based approach.

For each unanalyzed input word, based on ending character of its stem and paradigm-classes of its suffix, a set of probable and

relevant paradigm-classes are displayed as questionnaire to the user. The user is asked to choose suitable paradigm-class for input stem. The gender of unanalyzed word is determined based on chosen paradigm-class using noun-paradigm-class table (Table 3). The stem along with its gender and paradigm-class is added to lexicon. If the unanalyzed word is an indeclinable/non-inflectional word, it is directly appended into lexicon with 'A' ("*Avyaya*") as its lexical category. Almost all available Kannada verb-roots and their lexical categories are already stored in the lexicon. This module is called a semi-automatic module, because selection of paradigm-class of input word requires user intervention and rest of task is carried out by the module itself.

For example, consider the process of adding lexical details of an unanalyzed word "*shaaleyannu*" into lexicon. Output of proposed morphological stemmer for the word "*shaaleyannu*" is shown below.

"*shaale*" as stem "*yannu*" as suffix
07 08 09 and 12 as its all possible paradigm-classes.

For better understanding, these paradigm-classes are displayed with their relevant examples, based on ending character of the stem ("*e kaaraaNtha*") and paradigm-classes of suffixes as shown below.

7. *Ele*
8. *mahiLe*
12. *Aththe*

Paradigm-class 09 from output of morphological stemmer is not displayed in the list, because paradigm-class 09 is the category-code for stem ending with "i" *(i kaaraaNtha*). The user is asked to select suitable paradigm- class for input stem "*shaale*". With the knowledge of Kannada grammar, user selects category "07. *Ele*", as paradigm-class for root-word "*shaale*". Both root-words "*shaale*" and "*Ele*" are ending with the same letter 'e' and also root-word "*shaale*" generates all inflectional forms using same suffixes that are used by root-word "*Ele*" rather root-words "*mahiLe*" or "*Aththe*".

The gender of selected paradigm-class "07. *Ele*" is neuter gender (Table 3), hence gender of "*shaale*" is also neuter gender. The root-word with its gender and paradigm-class ("*shaale* N07") is added to lexicon. During the next execution of morphological analyzer module, all words that are inflected with stem "*shaale*" like "*shaaleyu*", "*shaalegaLu*", "*shaaleyannu*", "*shaaleyalli*", etc., will be analyzed, because lexical details of root-word "*shaale*" is available in lexicon. This improves the performance of morphological analyzer.

5. Experimental Results and Discussion

Each module of the proposed system is tested individually and experimental results are obtained. Details of data sets used for testing the proposed model, experimental results analysis, performance evaluation of each module and overall system, and comparative study of proposed model with existing Kannada morphological analyzers are discussed in following sections.

5.1 Data sets created for proposed model

Since, no standard Kannada data set is available publicly; 7 different data sets have been manually created by collecting text from different sources such as government official circulars, news papers, legal documents and news from All India Radio (AIR). The details of 7 different data sets are given below.

Data set1: Government official circular documents

Data set2: Kannada News paper (Vijaya Karnataka)

Data set2: Legal documents

Data set4: News from AIR

Data set5: Combination of Data set1 and Data set2

Data set6: Combination of Data set2, Data set3 and Data set4

Data set7: Combination of Data set1, Data set2, Data set3 and Data set4

The characteristics of these 7 different data sets, based on percentage of redundant words, unique words and special type of words like punctuation marks, abbreviations, acronyms, etc., are given in Table 8.

Data Set	No.of words in the input text	% of punctuation marks/numeric data / acronyms/abbreviation	% of redundant words	% of unique words	% of unique base/indeclinable words
1	1700	6.82	11.29	81.88	5.41
2	1200	8.33	14.17	77.50	7.08
3	450	9.56	7.11	83.33	4.89
4	300	8.67	5.00	86.33	4.67
5	2900	7.45	12.48	80.07	6.25
6	1950	8.67	11.13	80.21	5.55
7	3650	7.81	11.21	80.99	5.51

Table 8. Characteristics of Different Data Sets

The sample input text from Kannada news paper (data set2), which is used to test proposed work is given below. The sample text contains all kinds of words such as numbers, punctuation marks, redundant words, acronym, abbreviation, date, etc.,.

raajyada 258 “sarakaari” keeNdragaLalli maththu 196 “Anudaana” keeNdragaLalli vividha vruththigaLa tarabeethiyannu dinaaNka 12/09/2016 riNda 20/09/2016 ravarege Dr. vidyabhushana (C.E.O.) avaru niiDuththaare.

The experimental results obtained by proposed modules on sample input text are discussed in following sections.

5.2 Experimental result of text normalizer module

Token	Tag
258	NUMB
„	PUNCT
„	PUNCT
196	NUMB
„	PUNCT
„	PUNCT
12/10/2015	DATE
20/10/2015	DATE
Dr.	ABBRV
(	PUNCT
C.E.O	ACRON
)	PUNCT
.	PUNCT

Table 9. Output of Normalizer using look-up Table and Regular Expression

The proposed text normalizer tokenizes and categorizes given input text into two categories; i) words that require morphological analysis and ii) words that do not require morphological analysis. In this module, Indic tokenizer [4] is used to split the given input text into set of tokens/words. The given sample input text is split into 28 tokens. Out of which, 13 tokens do not require morphological analysis and are handled by assigning relevant parts of speech tags using look-up table and regular expressions (for numbers). The output of text normalizer for tokens that require no morphological analysis is given in Table 9.

In the given sample input text, remaining 15 words need morphological analysis. But, the word "*keeNdragaLalli*" is repeated twice, so one occurrence of that word is removed and 14 words are considered as unique words. The text normalizer is a preprocessing and data cleaning module. Hence this module reduces the overhead on morphological stemmer and morphological analyzer. The remaining 14 unique words in the given sample input text that require morphological analysis are shown below.

raajyada sarakaari keeNdragaLalli maththu Anudaana vividha vruththigaLa tarabeethiyannu dinaaNka riNda ravarege vidyabhushana avaru niiDuthtaare.

5.3 Experimental result of morphological stemmer module

The morphological stemmer module splits each input word given by text normalizer into stem and affixes. This module returns the type of word (base or "*avyaya*" or noun or verb), prefix, stem, suffix, list of paradigm-classes, and position of suffix in encoded-suffix table. According to the example considered, input words for this module are "*raajyada sarakaari keeNdragaLalli maththu Anudaana vividha vruththigaLa tarabeethiyannu dinaaNka riNda ravarege vidyabhushana avaru niiDuththaare*". This module identifies "*maththu*", "*vividha*", "*riNda*", and "*ravarege*" as non-inflectional words, and "*sarakaari*", "*dinaaNka*" and "*vidyabhushana*" as base form of words. Hence no stemming is required for these kinds of words. The remaining 7 words "*raajyada keeNdragaLalli Anudaana vruththigaLa tarabeethiyannu avaru niiDuththaare*" are inflectional words and require morphological stemming. The experimental result of morphological stemmer for given 14 tokens is shown in Table 10.

SI.No.	Input words	Type	Prefix	Stem	Suffix	List of paradigm class of the suffix	Position of suffix in the encoded-suffix table
1	*raajyada*	Noun	-	*raajya*	da	05	09
2	*sarakaari*	Base	-	-	-	-	-
3	*keeNdragaLalli*	Noun	-	keeNdra	gaLalli	05,07,10	12
4	maththu	Avyaya	-	-	-	-	-
5a.	Anudaana	Base	Anu	daana	-	-	-
5b.	Anudaana	Base	-	Anudaana	-	-	-
6	vividha	Avyaya	-	-	-	-	-
7	vruththigaLa	Noun	-	vruththi	gaLa	05,07,10	10
8	tarabeethiyannu	Noun	-	tarabeethi	yannu	07,08,09,14	03
9	dinaaNka	Base	-	-	-	-	-
10	riNda	Avyaya	-	-	-	-	-
11	ravarege	Avyaya	-	-	-	-	-
12	vidyabhushana	Base	-	-	-	-	-
13	Avaru	Noun	-	Ava	ru	01	02
14	niiDuththaare	Verb	-	niiDu	ththaare	1R	02

Table 10. Out of Moropological Stemmer Module

5.4 Experimental result of morphological analyzer module

Input to morphological analyzer is the result obtained by morphological stemmer. The remaining morphological features and grammatical categories of each input word are determined by this module using position of suffix. The experimental result of morphological analyzer for given input is shown below.

1. *raajyada*: NOUN, *raajya* (root), *da* (suffix), gender (neuter), case (ablative), number (singular)

2. *sarakaari*: NOUN, *sarakaari* (base), gender (neuter)

3. *keeNdragaLalli*: NOUN, *keeNdra* (root), *gaLalli* (suffix), gender (neuter), case (locative), number (plural)

4. *maththu*: AVYAYA

5. a) *Anudaana*: NOUN, *Anu* (prefix), *daana* (base), gender (neuter)

5. b) *Anudaana*: NOUN, *Anudaana* (base) , gender (neuter)

6. *vividha*: AVYAYA

7. *vruththigaLa*: UNANALYZED, NOUN, *vruththi* (stem), *gaLa* (suffix), 05 07 10 (possible paradigm- classes), 10 (position of suffix)

8. *tarabeethiyannu*: NOUN, *tarabeethi* (root), *yannu* (suffix), gender (neuter), case (accusative), number (singular)

9. *dinaaNka*: NOUN, *dinaaNka* (base), gender (neuter)

10. *riNda*: AVYAYA

11. *ravarege*: AVYAYA

12. *vidyaabhuushana*: NOUN, *vidyaabhuushana* (base) , gender (masculine)

13. *Avaru*: NOUN, Ava (root), *ru* (suffix), gender (masculine/feminine), case (nominative), number (plural)

14. *niiDuththaare*: VERB, *niiDu* (root), *ththaare* (suffix), person (third), gender (masculine / feminine), number (plural), tense (present)

It is observed from the output of morphological analyzer that, for input word "*Anudaana*", two forms of morphological features are obtained: i) "*Anu*" as prefix and "*daana*" as base word and ii) "*Anudaana*" as base form of word. This will lead to ambiguity in machine translation system. This ambiguity is resolved using word sense disambiguation system.

The morphological analyzer sometimes fails to analyze given input words due to unavailability of their lexical details in lexicon. These words are treated as unanalyzed words. In the given example, word "*vruththigaLa*" is an unanalyzed word, because root-word "*vruththi*" and its lexical details are not available in lexicon. Hence, it is necessary to add the root-word "*vruththi*" along with its lexical information to lexicon. Insertion of such unanalyzed words and their lexical detail into lexicon is handled by the proposed module "semi-automatic insertion of unanalyzed word into Kannada monolingual lexicon" and is explained in next section.

5.5 Experimental result of semi-automatic insertion of lexical details of unanalyzed words into Kannada monolingual lexicon module

According to literature survey, the existing Kannada morphological analyzers fail to address unanalyzed word. Hence, an extended module is proposed in this paper to handle unanalyzed words to improve the performance of morphological analyzer.

The morphological analyzer is unable to analyze input word "*vruththigaLa*", because its root-word "*vruththi*" and its lexical details are not available in lexicon. The details given by morphological stemmer module for the unanalyzed word "*vruththigaLa*" are given below.

NOUN, *vruththi* (root), *gaLa* (suffix), 05 07 10 (possible paradigm- classes) and 10 (position of suffix).

The proposed module "semi-automatic insertion of lexical details of unanalyzed words into Kannada monolingual lexicon" addresses this problem. Based on the ending character of input root-word and its possible paradigm-classes obtained by morphological stemmer, this module displays paradigm-classes of "*vruththi*" with examples (05 *kavi*, 07. *kurchi*). Here the

paradigm-class 10 is not displayed, because root-word "*vruththi*" ends with 'i', but paradigm-class 10 is category-code for root-words end with character 'u' (Table 3). The user is asked to choose correct paradigm-class for root-word "*vruththi*". Based on chosen paradigm-class (07. *kurchi*), gender (neutral) of the root is determined. The lexical detail of unanalyzed word "*vruththigaLa*" is "*vruththi*" N07. This lexical detail is inserted into the lexicon. During the next execution of morphological analyzer module, all words that are inflected with stem "*vruththi*" like "*vruththiyu*", "*vruththigaLu*", "*vruththiyannu*", "*vruththiyalli*", etc., will be analyzed, because lexical details of root-word "*vruththi*" is available in lexicon. This improves the performance of morphological analyzer.

5.6 Performance evaluation of proposed model

The performance of text normalizer and morphological stemmer are evaluated in terms of accuracy and is given in equation (1) and (2) respectively. The performance of morphological analyzer is computed in terms of precision, recall and Fmeasure, using following equations (3), (4) and (5).

$$\text{Accuracy of Normalizer} = \frac{W_t}{W_i} * 100 \quad (1)$$

$$\text{Accuracy of Morphological Stemmer} = \frac{W_s}{W_u} * 100 \quad (2)$$

$$\text{Precision} = \frac{T_P}{T_P + F_P} * 100 \quad (3)$$

$$\text{Recall} = \frac{T_P}{T_P + F_n} * 100 \quad (4)$$

$$F_{measure} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} * 100 \quad (5)$$

Where:

Wt - Number of correctly tokenized words

Wi - Number of input words

Ws - Number of correctly stemmed words

Wu - Number of unique inflectional words

Tp - True positive (Number of words correctly analyzed)

Fp - False positive (Number of words incorrectly analyzed)

Fn - False negative (Number of words unanalyzed)

The text normalizer, morphological stemmer and analyzer of proposed work are tested on 7 data sets. The analysis of result obtained by these modules is discussed below.

5.6.1 Result analysis of text normalizer

The result analysis of text normalizer is shown in Table 11. The percentage of error in text normalizer module among 7 data sets ranges from 1% to 3%. These errors are due to the incorrect tokenization by Indic tokenizer. Few such incorrect tokenization cases are listed below.

1. The number 24,000 will be tokenized into 3 tokens as i) 24 ii) , iii) 000, instead of single token.

2. The word "*30rashtu*" will be tokenized as "*30rashtu*", instead of 2 tokens as i)30 ii) "*rashtu*" separately.

3. The web site address www.training.org will be tokenized into 5 tokens as i) www ii) . iii) training iv) . v) org, instead of single token.

4. The Internet protocol address 10.49.0.23 will be tokenized into 7 tokens as i)10 ii) . iii) 49 iv) . v)0 vi) . vii) 23, instead of single token.

Data Set	No.of words in the input text(Wi)	No.of punctuation marks, numeric data,acronyms, abbreviation	No.of words Correctly tokenized (Wt)	No.of words incorrectly tokenized	No.of redundant words	No.of unique words	Accuracy (%)
1	1700	104	1688	12	192	1392	99.3
2	1200	70	1170	30	170	930	97.5
3	450	30	437	13	32	375	97.1
4	300	15	289	11	15	259	96.3
5	2900	174	2858	42	362	2322	98.6
6	1950	115	1896	54	217	1564	97.2
7	3650	219	3584	66	409	2956	98.2

Table 11. Result Analysis of Text Normalizer

5.6.2 Result analysis of morphological stemmer

The result analysis of morphological stemmer is shown in Table 12. Figure 3 shows the accuracy of text normalizer and morphological stemmer. The performance of morphological stemmer depends on inflectional suffixes in encoded-suffix table. The reasons for obtaining incorrect stemmed words are listed below.

Data Set	No.of unique input words	No.of base/ indeclinable words	No.of unique inflectional words (Wu)	No.of Correctly stemmed words (Ws)	No.of incorrectly stemmed words	Accuracy (%)
1	1392	92	1300	1268	32	99.3
2	930	85	845	822	23	97.5
3	375	22	353	337	16	95.9
4	259	14	245	228	17	93.5
5	2322	177	2145	2090	55	97.6
6	1564	123	1441	1385	56	96.4
7	2956	213	2743	2655	88	97.0

Table 12. Result Analysis of Morphological Stemmer

1. The longest suffix pattern matching approach is used to split given input word into stem and suffixes. This leads to over stemming error. Due to over stemming, the stem obtained by the stemmer is not a valid root-word. For example, input word "*AvugaLannu*" will be split into "A" and "*vugaLannu*" instead of "*Avu*" and "*gaLannu*". Here, "*vugaLannu*" is longest suffix present in the word "*AvugaLannu*".

2. Unavailability of inflectional suffix in the encoded-suffix table.

3. Spelling error in the suffix part of given word.

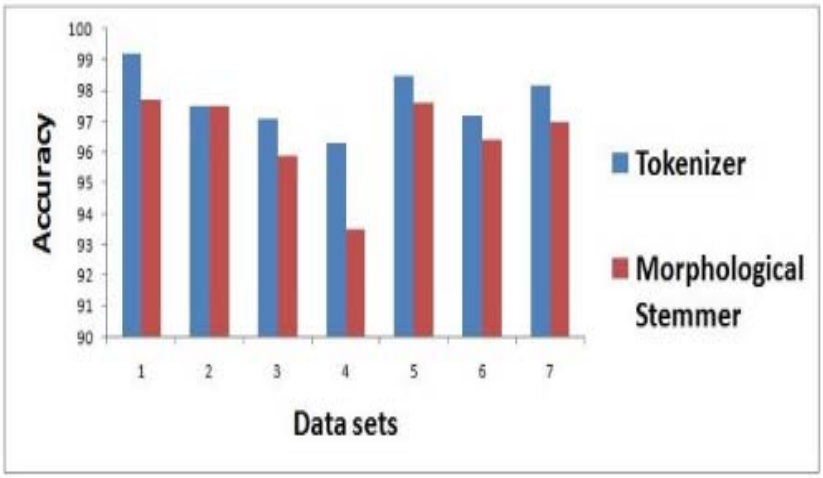

Figure 3. Accuracy of Normalizer and Morphological Stemmer

5.6.3 Result analysis of morphological analyzer

The performance of morphological analyzer depends on availability of root/indeclinable words in lexicon. The result analysis of morphological analyzer is shown in Table 13. The accuracy of morphological analyzer in terms of precision, recall and Fmeasure is shown in Figure 4. The multiple suffixes are not considered in this module. The reasons for incorrect analysis of input words are given below.

Data Set	No.of input words	No.of Correctly analyzed words (Tp)	No.of incorrectly analyzed words (Fp)	No.of unanalyzed words (Fn)	Precision (%)	Recall (%)	Fmeasure (%)
1	1360	1115	222	23	83.40	97.98	90.10
2	907	769	96	42	88.90	94.82	91.77
3	359	294	37	28	88.82	91.30	90.05
4	242	209	10	23	95.43	90.09	92.68
5	2267	1884	318	65	85.56	96.66	90.77
6	1508	1272	143	93	89.89	93.19	91.51
7	2868	2387	365	116	86.74	95.37	90.85

Table 13. Result Analysis of Morphological Analyzer

1. Unavailability of root word or indeclinable in the lexicon.
2. Presence of foreign word. Example: "*semesternalli* = semester + *nalli*".
3. Presence of multiple suffixes in the input word.

   For example: "maathanaadikoLLuththiddanu" = "maathu" +"annu" + "aadi" + "koLLu" + "ththi" + "dda" + "anu".
4. Spelling errors in the root of input word.

The accuracy of text normalizer and morphological stemmer for data set1 is high (99.3% and 99.3%) and low for data set4

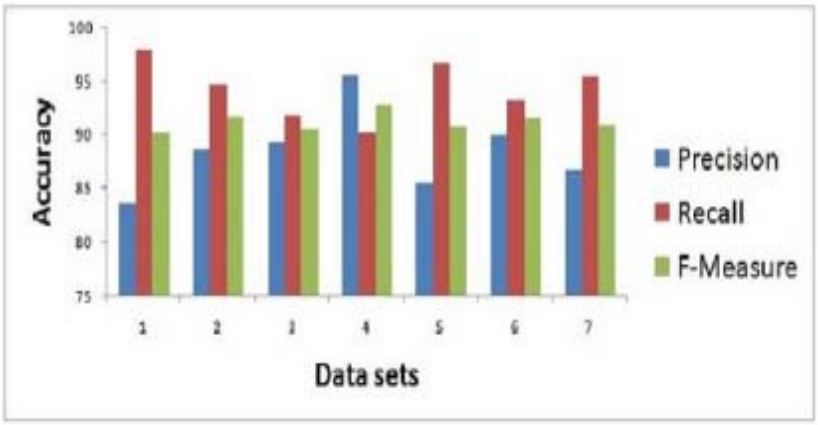

Figure 4. Accuracy of Morphological Analyzer

(96.3% and 93.5%). However, the accuracy of morphological analyzer in terms of Fmeasure is high for data set4 (92.68%) and low for data set3 (90.05%). This shows that, performance of text normalizer depends on the Indic tokenizer. The accuracy of morphological stemmer depends on availability of inflectional suffixes in encoded- suffix table. The precision, recall and Fmeasure of morphological analyzer depend on presence of root/indeclinable words in lexicon. In general, performance of proposed system depends on the content of input text, availability of inflectional suffixes in encoded-suffix table and root/indeclinable words in lexicon.

5.6.4 Performance of semi-automatic insertion of lexical details of unanalyzed words into Kannada monolingual lexicon module

Data Set	No.of unanalyzed words	No.of Inflectional Words	No.of non inflectional (indeclinable) words	No.of non Kannada (foreign) words	No.of unanalyzed words inserted into the lexicon
1	23	20	3	5	18
2	42	29	13	8	34
3	28	24	4	5	23
4	23	16	7	3	20
5	65	49	16	13	52
6	93	69	24	16	77
7	116	89	27	21	95

Table 14. Performance of Semi-automatic Insertion of Lexical Details of Unanalyzed Words into Kannada Monolingual Lexicon Module

The performance of semi-automatic insertion of lexical details of unanalyzed words into Kannada monolingual lexicon module is calculated in terms of number of unanalyzed words from morphological analyzer are inserted into lexicon. The details of unanalyzable words like inflectional, non-inflectional and foreign words are shown in Table 14. After insertion of unanalyzed words into lexicon, performance of morphological analyzer is improved. The details of improvement in morphological analyzer after lexicon update are shown in Table 15. In this table, once again all unanalyzed words are foreign words. The lexicon is a

Kannada monolingual dictionary which contains only Kannada words, hence foreign words are not inserted into lexicon, for which additional special rules are required. It is observed from the Table 13 and Table 15, performance of morphological analyzer has been increased (up to 3%-4%), after insertion of lexical details of unanalyzed Kannada words into lexicon.

Data Set	No.of input words	No.of Correctly analyzed words (Tp)	No.of Incorrectly analyzed words (Fp)	No.of unanalyzed words (Fn)	Precision (%)	Recall (%)	Fmeasure (%)
1	1360	1133	222	5	83.62	99.56	90.89
2	907	803	96	8	89.32	99.01	93.92
3	359	317	37	5	89.55	98.45	93.79
4	242	229	10	3	95.82	98.71	97.24
5	2267	1936	318	13	86.47	99.29	92.41
6	1508	1349	143	16	91.56	98.72	94.98
7	2868	2482	365	21	89.58	98.93	93.96

Table 15. Result Analysis of Morphological Analyzer after Insertion of Unanalysed Word into the Lexicon

5.6.5 Result analysis of overall system

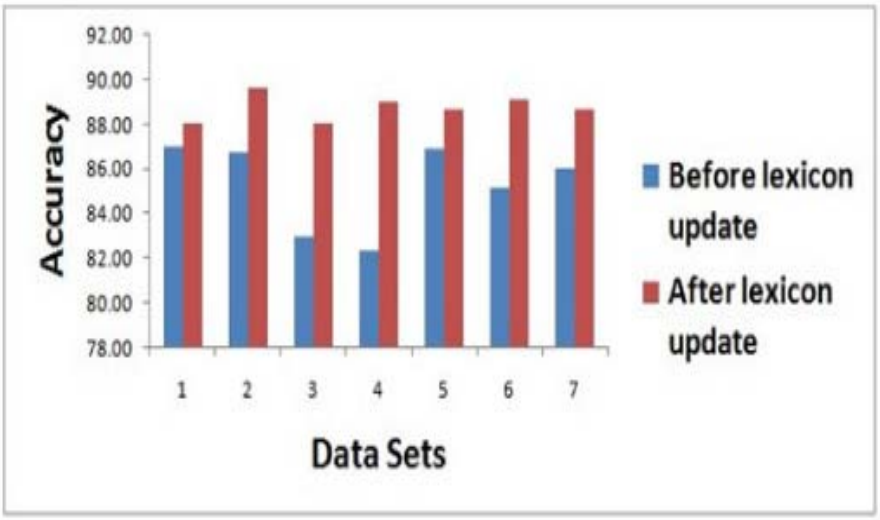

Figure 5. Accuracy of Overall System before and after insertion of Unanalyzed Words into the Lexicon

Overall performance of the proposed system depends on performance of each module. Accuracy of the overall system before and after update of unanalyzed words into lexicon is given in Table 16 and Figure 5. The improvement in performance of proposed system after lexicon update depends on number of unanalyzed words that are updated into lexicon.

Data Set	No.of input words	No.of Correctly analyzed words		Accuracy (%)		%ge of accuracy enhanced
		Before update of un-analyzed words into lexicon	After update of un-analyzed words into lexicon	Before update of unanalyzed words into lexicon	After update of unanalyzed words into lexicon	
1	1700	1479	1497	87.00	88.06	1.06
2	1200	1041	1075	86.75	89.58	2.83
3	450	373	396	82.89	88.00	5.11
4	300	247	267	82.33	89.00	6.67
5	2900	2520	2572	86.90	88.69	1.79
6	1950	1660	1738	85.13	89.13	4.00
7	3650	3140	3235	86.03	88.63	2.60

Table 16. Result Analysis of Proposed Overall System

The percentage of error in text normalizer, morphological stemmer, morphological analyzer and overall system is calculated and shown in Table 17. It is observed that, error in one module has no effect on the other module. However, error in the overall system is cumulative of errors in all three modules. As the number of unanalyzed words updated into lexicon increases, percentage of error in overall system is reduced. The propagation of error in proposed system is shown in Figure 6.

Data Set	Text Normalizer	Morphological stemmer	Morphological analyzer	Overall System before lexicon update	Overall System after lexicon update
1	0.71	2.30	9.90	13.00	11.94
2	2.50	2.47	8.23	13.25	10.42
3	2.89	4.27	9.95	17.11	12.00
4	3.67	6.56	7.32	17.67	11.00
5	1.45	2.37	9.23	13.10	11.31
6	2.77	3.58	8.49	14.87	10.87
7	1.81	2.98	9.15	13.97	11.37

Table 17. Percentage of Errors in the Three Modules and Overall System

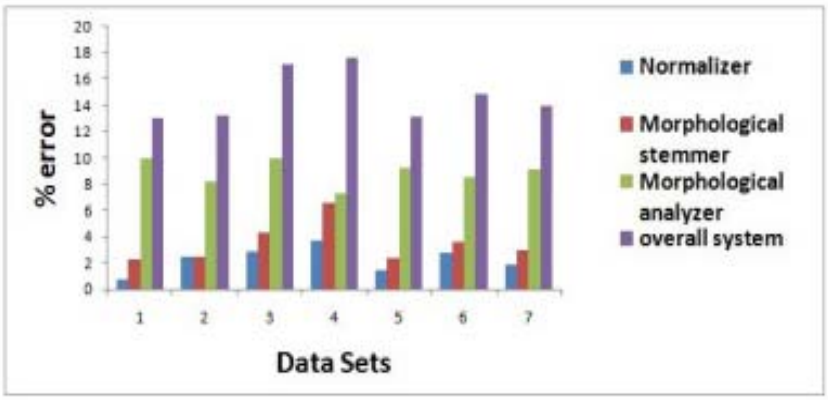

Figure 6. Propagation of Error in the Proposed System

5.6.6 Comparison of proposed model with existing Kannada morphological analyzers

Model	Methodology used	Size of lexicon	Limitations
Model 1 [24]	FST based	500 nouns and verbs	1. Used space as the delimiter in tokenizer. 2. Accepts input word, if and only if it is within the state transition. 3. Addressing of unanalyzed words is not focused. 4. No prefixes are considered. 5. Kannada Derivational words and English Foreign words are not considered
Model 2 [25]	1. Rule and FST based	20,000 nouns,	1. Used space as the delimiter in tokenzier. 2. All rules need to be hard coded. 3. Handling of nonconcatenate words is challenging. 4. Addressing of unanalyzed words is not focused 5. No prefixes are considered 6. Kannada Derivational words and English Foreign words are not considered.
Model 3 [26]	1. Rule and paradigm based	3700 roots	1. Used space as the delimiter in tokenizer 2. Consumes more memory. 3. Kannada Derivational words and English Foreign words are not considered. 4. Addressing of unanalyzed words is not focused. 5. No prefixes are considered.

Table 18a. Comparison of proposed model with the existing Kannada morphological analyzers

In literature, authors of existing Kannada morphological analyzers have created their own data sets. No report has been found regarding type of data, sources of data and number of words in input data considered to test their proposed model. Comparative

study of proposed work with existing Kannada morphological analyzers in terms of methodology used, size of lexicon and their limitations are carried out and these details are given in Table 18.

Model	Methodology used	Size of lexicon	Limitations
Proposed Model	1. Rule-based, paradigm-based, questionnaire-based and affix stripping approaches.	3000 nouns roots,	1. Kannada Derivational words and English Foreign words are not considered.
	2. Identified 12 noun and 15 verb paradigm classes.	500 verb roots	2. Words with multiple suffixes are not considered.

Table 18b. Comparison of proposed model with the existing kannada morphological analyzers

6. Conclusions and Future Work

This paper presents a model to perform morphological analysis of Kannada inflectional nouns and finite verbs using hybrid approach. The proposed model is developed using affix-stripping, rule-based, paradigm-based and questionnaire-based approaches. It handles most of the inflectional nouns and finite verbs that are available in well known Kannada dictionary "*Kannada Rathna Kosha*". The noun-roots and verb-roots are categorized and stored in Kannada monolingual lexicon based on their inflectional forms. In the proposed work, 260 suffixes, 23 prefixes and 3500 root words are stored in encoded-suffix table, prefix-list and Kannada monolingual lexicon respectively. Extensive experimentation is conducted on 7 different data sets that are created for this proposed work. The experimental result shows that, performance of proposed model is around 90% for different types of input text. This model would be used as a prerequisite tool in machine translation system, spell checker, named entity recognition and POS tagger. In future, this model can be enhanced for the analysis Kannada of inflectional nouns and verbs with multiple suffixes and derived words. Generally, most of Kannada text documents contain foreign (English) words. Insertion of these foreign words into lexicon with special rules is essential. Hence, performance of the proposed system can be substantially improved by adding more number of Kannada root-words and foreign words into lexicon.

Author Biographies

M. C. PADMA received B.E. and M. Sc. Tech. by Research degree in Computer Science and Engineering from University of Mysore, India, and Ph.D. degree from Visvesvaraya Technological University, Belgaum, India. Currently, she is Professor and Head, Department of Computer Science and Engineering, PES College of Engineering, Mandya, India. She is a member of IEEE, MISTE, CSI, IEI professional societies. Her research areas are Pattern Recognition, Natural Language Processing, Document Image Analysis and Recognition. She has published 45 papers in National/International Conferences/Journals and conducted National/International Conferences and Editor for Proceedings of International Conference Emerging Research in Electronics, Computer Science and Technology.

R. J. PRATHIBHA received her B.E. in Computer Science and M. Tech. in Software Engineering from Visvesvaraya Technological University (VTU), Belgaum, India. Currently, she is working as Assistant Professor in the department of Information Science and Engineering, S J College of Engineering, Mysore, India. Her research areas are Natural Language Processing, Machine Translation,Artificial Intelligence, Machine Learning, Data Mining and Big Data Analytics. She has published 8 papers in International Conferences/Journals.

References

[1] Anand Kumar, M., Dhanalakshmi, V.V., Soman, K.P., Rajendran S. (2010). A Sequence Labeling Approach to Morphological Analyzer for Tamil Language, *International Journal on Computer Science and Engineering*, 02 (6) 1944–1951.

[2] Deepak Kumar., Manjeeth Singh., Seema Shukla. (2012). FST Based Morphological Analyzer for Hindi Language, *International Journal of Computer Science and Issues*, 9, p. 349–353.

[3] Filip, Gralinski., Krzysztof, Jassem., Agnieszka, Wagner., Mikolaj, Wypych (2006). Text Normalization as a Special Case of Machine Translation, *In:* Proceedings of the International Multi conference on Computer Science and Information Technology, p. 51–56.

[4] Indic NLP library, http://anoopkunchukuttan.github.io/indic_nlp_library/

[5] Kallimani, Jagadish S., Srinivasa, K G., Eswara Reddy, B. (2012) . Normalization of Non Standard Words for Kannada Speech Synthesis, *International Journal of Advances in Computer Science and Technology*, 1 (1) 21–26.

[6] Lovins, J.B. (1968). Development of a stemming algorithm, *Mechanical Translations and Computational Linguistics*, 11 (1) 22–31.

[7] Porter, M. F. (1980). An algorithm for suffix stripping, *Program*, 14 (3) 130–137.

[8] Menon, A.G., Saravanan S., Loganathan R., Soman, K. (2010). Amrita Morph Analyzer and Generator for Tamil: A Rule Based Approach, *In:* Tamil International Conference, Coimbatore, India.

[9] Mohd. Shahid Hussain (2012) . An Unsupervised Approach to Develop Stemmer, *International Journal on Natural Language computing*, 1 (2) 15–23.

[10] Narayana Murthy, K. (2003). A Network and Process Model for Kannada morphological Analysis/Generation, Department of Computer and Information Sciences, University of Hyderabad, Hyderabad, INDIA.

[11] Nayak, H. M. (1994). Kannada Rathna Kosha, Kannada Sahithya Parishath, Kannada Abhivruddhi Pradhikara, Bengaluru.

[12] Padma, M.C., Prathibha, R.J. (2014) Development of Morphological Stemmer, Analyzer and Generator for Kannada nouns, Emerging Research in Electronics, Computer Science and Technology, Springer, Vol. 248, p. 713–723.

[13] Prathibha, R.J., Padma, M.C. (2013). Development of Morphological Analyzer for Kannada Verbs, *In*: Fifth International Conference on Advances in Recent Technologies in Communication and Computing. p. 22–27.

[14] Gawade, Prathiksha., Madhavi, Deepika., Gaikwad, Jayshree., Jadhav, Sharvari., Ambekar, Raghul (2013). Morphological Analyzer for Marathi using NLP, *International Journal of Engineering Research and Applications*, 3, p. 322–326.

[15] Veerappan, Ramasamy., Antony P.J., Saravanan S. Soman. K.P. (2011). A Rule Based Kannada Morphological Analyzer and Generator using Finite State Transducer, *International Journal of Computer Applications* . 10, p. 45–52.

[16] Ayadi, Rami Maraoui, Mohsen Zrigui, Mounir (2011). SCAT: a system of classification for Arabic texts, *International Journal of Internet Technology and Secured Transactions*, 3 (1).

[17] Shambhavi, B.R., Ramakanth Kumar, P., Srividya, K., Jyothi, B.J., Kundargi, Spoorti., Varsha Shastri, G. (2011). Kannada Morphological Analyzer and Generator Using Trie, *International Journal of Computer Science and Network Security*, 11 (1) 112-116.

[18] Stanford natural language tool kit, http://nlp.stanford.edu

[19] Suma Bhat. (2013). Statistical stemmer for Kannada, *In*: International Conference on Natural Language Processing, p. 25-32.

[20] Sunitha, K.V.N., Kalyani, N. (2009). A Novel Approach to Improve Rule Based Telugu Morphological Analyzer. World Congress on Nature and Biologically Inspired Computing, p. 1649–1652.

[21] Sunitha, K.V.N., Sunitha Devi, P. (2013). Text Normalization for Telugu Text-to-Speech Synthesis, *International Journal of Computers and Technology*, 11 (2) 2240– 2249.

[22] Thangarasu, M., Manavalan, R. (2013). Design and development of Stemmer for Tamil language: Cluster Analysis, *International Journal of Advanced Research in Computer Science and Software Engineering*, 3, p. 812–818.

[23] Mishra, Upendra., Prakash, Chandra (2012). MOULIK: An effective stemmer for Hindi language, *International Journal on Computer Science and Engineering*, 4 (5) 711–717.

[24] Vikram, T.N., Shalini, R. Urs. (2007) . Development of Prototype Morphological Analyzer for South Indian Language of

Kannada, *Asian Digital Libraries*, p. 109–116.

[25] Vinod P. M., Javan V, Bhadran V. K. (2012). Implementation of Malayalam Morphological Analyzer Based on Hybrid approach, *In*: Proceedings of the 24th Conference on Computational Linguistics and Speech Processing, p. 307–317.

[26] Gupta, Vishal. (2013) Automatic Normalization of Punjabi Words, *International Journal of Engineering Trends and Technology*, 6 (1) 353–357.

[27] Gupta, Vishal. (2014). Hindi Rule Based Stemmer for Nouns, *International Journal of Advanced Research in Computer Science and Software Engineering*, 4, p. 62–65.

[28] Ramachandran, Vivek Anandan., Krishnamurthi, Ilango. (2012). An Iterative Stemmer for Tamil Language, *Intelligence Information and Database Systems*, *Springer,* p. 197–205.

[29] Cherif, Walid., Madani, Abdellah., Kissi, Mohamed. (2015). New rules-based algorithm to improve Arabic stemming accuracy', *International Journal on Knowledge Engineering and Data Mining*, 3 (3/4) 315-336.